Overcoming instabilities in Verlet-I/r-RESPA with the mollified impulse method

Jesús A. Izaguirre[1], Qun Ma[1], Thierry Matthey[2], Jeremiah Willcock[1], Thomas Slabach[1], Branden Moore[1], and George Viamontes[1]

[1] Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, Indiana 46556-0309, USA
[2] Department of Informatics, University of Bergen, N-5020 Bergen, Norway

Abstract. The primary objective of this paper is to explain the derivation of symplectic mollified Verlet-I/r-RESPA (MOLLY) methods that overcome linear and nonlinear instabilities that arise as numerical artifacts in Verlet-I/r-RESPA. These methods allow for lengthening of the longest time step used in molecular dynamics (MD). We provide evidence that MOLLY methods can take a longest time step that is 50% greater than that of Verlet-I/r-RESPA, for a given drift, including no drift. A 350% increase in the timestep is possible using MOLLY with mild Langevin damping while still computing dynamic properties accurately. Furthermore, longer time steps also enhance the scalability of multiple time stepping integrators that use the popular Particle Mesh Ewald method for computing full electrostatics, since the parallel bottleneck of the fast Fourier transform associated with PME is invoked less often. An additional objective of this paper is to give sufficient implementation details for these mollified integrators, so that interested users may implement them into their MD codes, or use the program ProtoMol in which we have implemented these methods.

Using simple analysis of a 1-d model problem we show the linear instability present in Verlet-I/r-RESPA at approximately half the period of the fastest motion, and more interestingly, how the mollified methods can be designed to overcome them. The paper also includes an experimental component that shows how these methods overcome instability barriers in practice.

We also present evidence that more complicated instabilities are present in Verlet-I/r-RESPA than linear analysis reveals. In particular, we postulate nonlinear resonance mechanisms hereto ignored, although these mechanisms are known for leapfrog. *This means that Verlet-I/r-RESPA is no better than leapfrog if one wants a simulation with no drift.* Currently, we use mild Langevin damping to overcome these nonlinear instabilities, but it is possible to design symplectic MOLLY integrators that are nonlinearly stable as well.

1 Introduction

The primary objective of this paper is to explain the derivation of symplectic mollified Verlet-I/r-RESPA methods (MOLLY) that overcome linear and nonlinear instabilities that arise as numerical artifacts in Verlet-I/r-RESPA. These methods allow for lengthening of the longest time step used in molecular dynamics (MD). We provide evidence that MOLLY methods can take a longest time step that is 50% greater than that of Verlet-I/r-RESPA, for a given drift, including no drift. Results presented elsewhere show that a 350% increase in the timestep is possible using MOLLY with mild Langevin damping while still computing dynamic properties accurately [29]. Furthermore, longer time steps also enhance the scalability of multiple time stepping (MTS) integrators that use the popular Particle Mesh Ewald (PME) [14] method for computing full electrostatics, since the parallel bottleneck of the fast Fourier transform (FFT) associated with PME is invoked less often. An additional objective of this paper is to give sufficient implementation details for these mollified integrators, so that interested users may implement them into their MD codes, or use the codes that we have made available. Both mathematical and software aspects are covered.

This paper presents simple analysis of a 1-d model problem, following Schlick and co-workers [4,44], to show the linear instability present in Verlet-I/r-RESPA at approximately half the period of the fastest motion, and more interestingly, how the mollified methods can be designed to overcome them. The paper also includes an experimental component that shows how these methods overcome instability barriers in practice. In the interest of presenting reproducible results, extensive details of the experimental protocol are provided. This includes algorithmic and implementation details of the methods, as well as details of the test systems.

We also present evidence that more complicated instabilities are present in the Verlet-I/r-RESPA family of methods than linear analysis reveals. In particular, we postulate nonlinear resonance mechanisms hereto ignored, although these mechanisms are known for leapfrog [47]. Nonlinear stability analysis confirming these results is presented elsewhere [38]. Currently, we use mild Langevin damping to overcome these nonlinear instabilities, but it is possible to design symplectic MOLLY integrators that are nonlinearly stable as well.

2 Background

Molecular dynamics (MD) solves Newton's equations of motion by evaluating pairwise interactions between particles (force evaluation), marching the system in time (numerical integration), and imposing boundary conditions. Some tutorials on multiple time stepping integration are in [27,43,46,47,51]. We consider the requirements of MD software that incorporates state-of-the-art MTS integrators of an arbitrary number of levels. Examples of these integrators are the extrapolative method LN [1,2,44] and the mollified impulse method, or MOLLY [17,18,28,29,31,53], which is in turn a more stable variant of Verlet-I/r-RESPA [20,56]. We present linear analysis of a simple model problem discretized using Verlet-I/r-RESPA and different MOLLY methods to show how MOLLY can overcome the instability barrier due to linear resonance. We then show evidence of nonlinear instabilities in Verlet-I/r-RESPA.

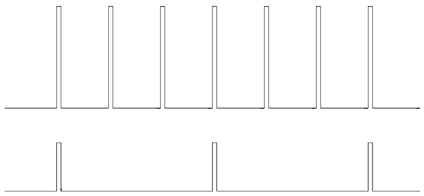

Fig. 1. Schematic for the *Impulse* multiple time stepping method.

Finally, we present experimental evaluation of the MOLLY methods. We introduce a particular algorithm-development platform for MD called ProtoMol [32,30,41]. We describe how the MOLLY methods are implemented seamlessly there, and provide full derivations of MOLLY in the appendix. The experiments confirm the results of the linear and nonlinear analysis.

2.1 Multiple Time Stepping

Here we present a review of multiple time stepping for molecular dynamics. For more details, we recommend the comprehensive tutorials [7,35] and the books [15,22,36,45].

The numerical integration of Newton's equations of motion is limited by stability: the length of time steps one can take to integrate the equations of motion is fairly short relative to the total length needed for simulations — time steps are in the order of femtoseconds (10^{-15} seconds) whereas simulations of a few microseconds (10^{-6} seconds) up to one second are most desired.

Multiple time stepping integrators have been used to lengthen the time step for most of the interactions in the equations of motion. These methods evaluate different parts of the force at different frequencies. Limitations on the step size in MTS integrators are still severe, and these are mostly due to stability rather than accuracy.

A typical MTS integrator is the Verlet-I/r-RESPA multiple time stepping impulse method. In this method the force is split into different components whose dynamics correspond to different time scales, which are then represented as appropriately weighted impulses (with weights determined by consistency). The impulse method is

$$M\frac{\mathrm{d}^2}{\mathrm{d}t^2}X = -\sum_{n'=-\infty}^{\infty} \delta t\, \boldsymbol{\delta}(t-n'\Delta t)\nabla U^{\mathrm{fast}}(X) - \sum_{n=-\infty}^{\infty} \Delta t\, \boldsymbol{\delta}(t-n\Delta t)\nabla U^{\mathrm{slow}}(X) \tag{1}$$

where the partitioning of U into U^{fast} and U^{slow} is chosen so that an appropriate time step Δt for the slow part of the force is larger than a time step δt for the fast part. In the formula, $\boldsymbol{\delta}$ is the Dirac delta function. This is illustrated schematically in Figure 1. Verlet-I/r-RESPA can be written as Algorithm 1. MTS integrators may use more than two levels. An elegant way to consider the generalization of MTS integrators to arbitrary numbers of levels is the use of the Trotter factorization, cf. [15,56].

For systems with flexible water, where there are bond vibrations and angle torsions, this method permits an increase from 1 to 3 fs in the length of the longest time step Δt, with no drift, and to 4 fs with little drift. It is completely unstable at 5 fs.

half a kick

$$P^{n-1+\epsilon} = P^{n-1} + \frac{\Delta t}{2} F^{\text{slow},n-1}. \tag{2}$$

a vibration Propagate X^{n-1}, $P^{n-1+\epsilon}$ by integrating

$$\frac{\mathrm{d}}{\mathrm{d}t} X = M^{-1} P, \quad \frac{\mathrm{d}}{\mathrm{d}t} P = F^{\text{fast}}(X) \tag{3}$$

for an interval Δt to get X^n, $P^{n-\epsilon}$.

half a kick

$$F^{\text{slow},n} = F^{\text{slow}}(X^n), \tag{4}$$

$$P^n = P^{n-\epsilon} + \frac{\Delta t}{2} F^{\text{slow},n}. \tag{5}$$

Algorithm 1: Verlet-I/r-RESPA method. The symbols $P^{n-1+\epsilon}$ and $P^{n-\epsilon}$ represent momenta just after the $(n-1)$th kick, and just before the nth kick, respectively.

When Verlet-I/r-RESPA was introduced, it was predicted that there would occur resonances that might induce instability if the frequency of the slow force impulse coincides with a normal mode frequency of the system. Resonance produces an oscillation in the positions whose amplitude increases with time. There is empirical evidence that time steps of approximately half of the fastest period present on the system or greater are not possible with this method. Thus, when flexible water molecules are present in the system, for example as solvent, the linear stability limit is around 5 fs.

2.2 The Mollified Impulse Method

We have worked on the mollified impulse method (MOLLY), a family of integrators [17] that counteracts the instabilities present in the MTS Verlet-I/r-RESPA integrator. This is accomplished by perturbing the potential using time averaged positions. The time average is obtained by doing dynamics over vibrations using forces that produce those vibrations. Thus,

$$U^{\text{slow}}(X) \rightarrow U^{\text{slow}}(\mathcal{A}(X)), \tag{6}$$

with the force defined as a gradient of this averaged potential,

$$-\nabla U^{\text{slow}}(X) \to -\mathcal{A}_X(X)^{\mathrm{T}}\nabla U^{\text{slow}}((X)), \tag{7}$$

where $\mathcal{A}_X(X)$ is a Jacobian matrix. MOLLY can be written as Algorithm 2.

half a mollified kick

$$P^{n-1+\epsilon} = P^{n-1} + \frac{\Delta t}{2}F^{\text{slow},n-1}. \tag{8}$$

a fluctuation Propagate X^{n-1}, $P^{n-1+\epsilon}$ by integrating

$$\frac{\mathrm{d}}{\mathrm{d}t}X = M^{-1}P, \quad \frac{\mathrm{d}}{\mathrm{d}t}P = F^{\text{fast}}(X) \tag{9}$$

(e.g., Verlet/leapfrog with time step δt) for an interval Δt to get X^n and $P^{n-\epsilon}$.

a time averaging Calculate a temporary vector of time-averaged positions $\bar{X}^n = \mathcal{A}(X^n)$ and a Jacobian matrix $J^n = \mathcal{A}_x(X^n)^{\mathrm{T}}$. The time averaging function $\mathcal{A}(x)$ uses only the fastest forces $F^{\text{reduced}}(x)$.

half a mollified kick

$$P^n = P^{n-\epsilon} + \frac{\Delta t}{2}F^{\text{slow},n} \tag{10}$$

Algorithm 2: Mollified impulse method. The symbols $P^{n-1+\epsilon}$ and $P^{n-\epsilon}$ represent momenta just after the $(n-1)$th kick, and just before the nth kick, respectively. Note that $\bar{X}^n$ is used only for the purpose of evaluating F^{slow}, it does not replace the value of X^n.

This perturbation compensates for finite Δt artifacts. Intuitively, averaged positions are better than instantaneous values for a rapidly changing trajectory $X(t)$. Perturbing the potential rather than the force ensures that the numerical integrator remains symplectic [45]. The force used by MOLLY is the gradient of the perturbed potential. The pre-factor $\mathcal{A}_X(X)^{\mathrm{T}}$ can be seen as a filter that eliminates components of the slow force impulse in the directions of the fast forces, and thus improves the stability of Verlet-I/r-RESPA. Different averaging functions give rise to MOLLY integrators with different stability and accuracy properties. We have used two different averaging methods, one based on explicit time averaging, which is reported in [53], and another based on complete elimination of linear instabilities, reported in [31]. These two methods overcome the half period barrier and achieve a 50% speedup over Verlet-I/r-RESPA. A stochastic variant of MOLLY has recently been shown to allow time steps 350% larger [29].

These filters currently do not filter out all possible linear resonances, primarily for efficiency purposes. With better filters, further improvements in the time step should be possible. However, in practice, even a perfect linear filter is not good enough. In this paper we show empirically that there is another instability that is reported here for the first time: there is a nonlinear instability, namely a 3:1 unconditionally unstable resonance, and a 4:1 conditionally stable resonance in Verlet-I/r-RESPA. The nonlinear stability analysis is reported elsewhere.

BSpline MOLLY It is possible to use time averagings that consist of numerically integrating an auxiliary, reduced problem:

$$\mathcal{A}(x) = \frac{1}{\Delta t}\int_0^\infty \phi\left(\frac{t}{\Delta t}\right)\tilde{X}(t)dt \tag{11}$$

where $\phi\left(\frac{t}{\Delta t}\right)$ is a weight function, and $\tilde{X}(t)$ solves an *auxiliary* problem

$$M\frac{\mathrm{d}^2}{\mathrm{d}t^2}\tilde{X} = F^{\text{fastest}}(\tilde{X}), \quad \tilde{X}(0) = x, \quad \frac{\mathrm{d}}{\mathrm{d}t}\tilde{X}(0) = 0. \tag{12}$$

This approach is computationally feasible if the weight functions ϕ have compact support in time. The paper [17] suggests using B-spline weight functions, which are non-zero over a short interval. The effectiveness of the averagings induced by these weight functions is directly related to the extensiveness of the time averaging. One such B-spline weight function that has been tested is called ShortAverage:

$$\phi(s) = \begin{cases} 0, & s < 0, \\ 2, & 0 \le s < \frac{1}{2}, \\ 1, & s = \frac{1}{2}, \\ 0, & s > \frac{1}{2}. \end{cases} \tag{13}$$

A longer averaging that is a scaling of ShortAverage is called LongAverage:

$$\phi(s) = \begin{cases} 0, & s < 0, \\ 1, & 0 \le s < 1, \\ \frac{1}{2}, & s = 1 \\ 0, & s > 1. \end{cases} \tag{14}$$

The coding of $\mathcal{A}(x)$ and $\mathcal{A}_x(x)$ can be done by hand in a systematic manner. First the calculation of $\mathcal{A}(x)$ is coded, and then the differentiation, applying the chain rule with respect to each of the components of x to yield code for $\mathcal{A}_x(x)$. As an example suppose that the leapfrog method with time step δt is coded for the calculation of $\mathcal{A}(x)$. This is then differentiated to obtain $\mathcal{A}_x(x)$. The result is the following code for calculating $\mathcal{A}(x)$ and $\mathcal{A}_x(x)$:

Initialization is given by

$$\begin{aligned} &X := x,\, X_x := I, \\ &P := 0,\; P_x := 0, \\ &B := 0,\; B_x := 0, \end{aligned} \tag{15}$$

and step by step integration by

$$\begin{array}{ll} P := P + \frac{1}{2}\delta t F^{\mathrm{fastest}}(X), & P_x := P_x + \frac{1}{2}\delta t F_x^{\mathrm{fastest}}(X) X_x, \\ B := B + \frac{1}{2}\delta t X \phi(t/\Delta t), & B_x := B_x + \frac{1}{2}\delta t X_x \phi(t/\Delta t), \\ X := X + \delta t M^{-1} P, & X_x := X_x + \delta t M^{-1} P_x, \\ t := t + \delta t, & \\ B := B + \frac{1}{2}\delta t X \phi(t/\Delta t), & B_x := B_x + \frac{1}{2}\delta t X_x \phi(t/\Delta t), \\ P := P + \frac{1}{2}\delta t F^{\mathrm{fastest}}(X), & P_x := P_x + \frac{1}{2}\delta t F_x^{\mathrm{fastest}}(X) X_x. \end{array} \tag{16}$$

The value $(1/\Delta t)B$ is used for $\mathcal{A}(x)$ and $(1/\Delta t)B_x$ for $\mathcal{A}_x(x)$. We continue the above integration until we reach a value of t such that $\phi(t/\Delta t)$ is zero at this value and remains zero for larger values of t. In practice, one can choose δt equal to the stepsize of the lowest level integrator. In the above loop, $F_x = -U_{xx}^{\mathrm{fastest}}(x)$ must be computed efficiently. We have included the derivation of the analytical form of the Hessian matrices for the CHARMm force field used in our implementation of MOLLY, $U_{xx}^{\mathrm{fastest}}(x)$, in the Appendix.

If the fastest forces included are bonded terms, such as bond and angle interactions, one may easily compute the Hessians F_x. Once individual Hessian matrices for angle energies and bond energies are computed, one has to somehow assemble them to form a complete Hessian. To illustrate the procedure, consider a water system, which is easy since all the water molecules are decoupled from each other. Assume the O is numbered as atom j, and the H as i and k, and only bond and angle energies are included in the reduced system. Suppose the we have computed the Hessian matrices of bond energy for atoms i and j, and j and k, *i.e.*, H_{ij}^{bd} and H_{jk}^{bd}, and of angle energy for atoms i, j and k, *i.e.*, H_{ijk}^{a}. The entries of the assembled Hessian matrix for this whole molecule should be as follows:

$$\begin{aligned}
H^{\mathrm{total}}[0][0] &= H_{ijk}^{\mathrm{a}}[0][0] + H_{ij}^{\mathrm{bd}}[0][0], \\
H^{\mathrm{total}}[0][1] &= H_{ijk}^{\mathrm{a}}[0][1] + H_{ij}^{\mathrm{bd}}[0][1], \\
H^{\mathrm{total}}[0][2] &= H_{ijk}^{\mathrm{a}}[0][2], \\
H^{\mathrm{total}}[1][0] &= H_{ijk}^{\mathrm{a}}[1][0] + H_{ij}^{\mathrm{bd}}[1][0], \\
H^{\mathrm{total}}[1][1] &= H_{ijk}^{\mathrm{a}}[1][1] + H_{ij}^{\mathrm{bd}}[1][1] + H_{jk}^{\mathrm{bd}}[0][0], \\
H^{\mathrm{total}}[1][2] &= H_{ijk}^{\mathrm{a}}[1][2] + H_{jk}^{\mathrm{bd}}[0][1], \\
H^{\mathrm{total}}[2][0] &= H_{ijk}^{\mathrm{a}}[2][0], \\
H^{\mathrm{total}}[2][1] &= H_{ijk}^{\mathrm{a}}[2][0] + H_{jk}^{\mathrm{bd}}[1][0], \\
H^{\mathrm{total}}[2][2] &= H_{ijk}^{\mathrm{a}}[2][2] + H_{jk}^{\mathrm{bd}}[1][1].
\end{aligned}$$

For systems other than water, the above simple method does not work because one or two of the atoms in one angle may also be involved in some other angles. For other systems, our solution is to assemble the individual Hessian matrices into a sparse matrix structure representing the total Hessian for the system, with each entry being a 3×3 matrix, i.e., a tensor. Cf. [37].

3 Linear Stability Analysis

Following Schlick and coworkers [4,44], we analyze Verlet-I/r-RESPA and MOLLY for the simple 1-d model problem

$$x' = p; \quad p' = -(\lambda_1 + \lambda_2)x, \tag{17}$$

that models a particle of unit mass at position x, connected to two springs with force constants $\lambda_1 = \Omega^2$ and $\lambda_2 = \omega^2$, where $\Omega^2 \geq \omega^2$.

3.1 Discretization using Verlet-I/r-RESPA and MOLLY

The discretization of Equation (17) using Verlet-I/r-RESPA of Algorithm 1, and MOLLY of Algorithm 2, with a longest time step Δ, and assuming analytical integration of the fast spring force associated with Ω, can be written in a generic matrix form as

$$\begin{bmatrix} x_{n+1} \\ p_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\omega^2 \frac{\Delta}{2} G^2 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \frac{\sin\theta}{\Omega} \\ -\Omega\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\omega^2 \frac{\Delta}{2} G^2 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ p_n \end{bmatrix}, \tag{18}$$

where $\theta = \Omega\Delta$, and G is the Fourier transform of the averaging and mollification functions, with the special case $G = 1$ for Verlet-I/r-RESPA, cf. [17,18,27].

Stability depends on the magnitude eigenvalues of the above propagation matrix. Because the scheme is symplectic, the determinant is one, and instability occurs when the eigenvalues become real and reciprocal of each other. Stability is insured if the value of the trace of this matrix

$$t(\theta) = -2\omega^2 \Delta G^2 \frac{\sin\theta}{\Omega} + 2\cos\theta > -2,$$

or equivalently, the magnitude of the eigenvalues must be 1 for stability.

The main result of the analysis is that *Verlet-I/r-RESPA and* ShortAverage *are unstable around $\theta = \pi$, although the former significantly more so, whereas* LongAverage *is stable.* This conclusion is validated by the experiments in this paper.

3.2 Assumptions and simplifications

The limit of interest is $\Omega\delta \ll 1$, which holds when assuming analytical integration of the fast forces ($\delta = 0$). We also want $\omega\Delta \leq 1$ to observe the stability condition of leapfrog, $\omega\Delta < 2$. Thus, we may assume that, for example, $\omega = \Omega/4$. This is nearly the worst case corresponding to poor separation of time scales in MD. Other choices are possible. Also, since the matrix BAB is similar to AB^2, we may write the propagation matrix of Equation (18) as:

$$\begin{bmatrix} \cos\theta & \frac{\sin\theta}{\Omega} \\ -\Omega\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{16}\Omega\theta G^2 & 1 \end{bmatrix}, \tag{19}$$

with eigenvalues

$$
\begin{aligned}
&\cos\theta - \frac{1}{32}(\sin\theta)\,\theta G^2 \\
&\quad \pm \frac{1}{32}\sqrt{\left(1024\cos^2\theta - 64\,(\cos\theta\sin\theta)\,\theta G^2 + \left(\sin^2\theta\right)\theta^2 G^4 - 1024\right)}.
\end{aligned} \tag{20}
$$

3.3 Verlet-I/r-RESPA

For the impulse method, $G = 1$, and the magnitude of an eigenvalue of the propagation matrix is plotted in Figure 2(a). It is clearly seen that it is unstable for $\Theta \approx \pi$. This corresponds to half the fastest period of motion. This is validated in the experimental section below, and has been shown before with analytical and empirical evidence [9,53]. It is presented here for completeness.

3.4 ShortAverage

For ShortAverage we compute the Fourier transform of Equation (13). This is given by

$$
\begin{aligned}
G &= \int_{-0.5}^{0.5} \exp(-i\theta x) \\
&= \frac{2}{\theta}\sin .5\theta.
\end{aligned} \tag{21}
$$

Substituting G in Equation (20), we get the plot of Figure 2(b). Note that it is unstable, but less than Verlet-I/r-RESPA.

3.5 LongAverage

For LongAverage the Fourier transform of Equation 14 is given by

$$
\begin{aligned}
G &= \int_{-1}^{1} \frac{1}{2}\exp(-i\theta x) \\
&= \frac{1}{\theta}\sin\theta.
\end{aligned} \tag{22}
$$

Substituting G in Equation (20), we get the plot of Figure 2(c). This is stable for $\Theta \approx= \pi$. These results are confirmed in Figure 7 below.

4 Nonlinear Stability

Nonlinear resonances that lead to instabilities are present in the Verlet-I/r-RESPA method. Unstable resonances usually manifest themselves in the neighborhood of a certain time step: There is a definite range of time steps

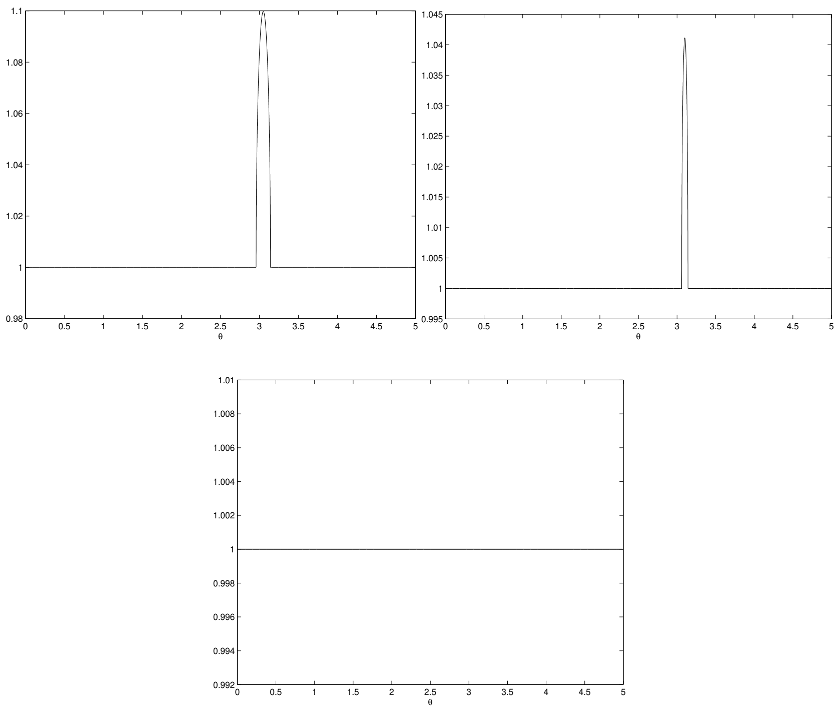

Fig. 2. **(a)** Plot of eigenvalue of propagation matrix for **Verlet-I/r-RESPA** discretization for 1-d model problem $x' = p; \quad p' = -(\lambda_1 + \lambda_2)x$. The function is $\cos\theta - \frac{1}{32}(\sin\theta)\,\theta - \frac{1}{32}\sqrt{\left(1024\cos^2\theta - 64(\cos\theta\sin\theta)\,\theta + \left(\sin^2\theta\right)\theta^2 - 1024\right)}$ **(b)** The same for **ShortAverage** discretization. The function is $\cos\theta - \frac{1}{8}\frac{\sin\theta}{\theta}\sin^2 0.5\theta - \frac{1}{8}\sqrt{\left(64\cos^2\theta - 16(\cos\theta)\frac{\sin\theta}{\theta}\sin^2 0.5\theta + \frac{\sin^2\theta}{\theta^2}\sin^4 0.5\theta - 64\right)}$ **(c)** The same for **LongAverage**. The function is $\cos\theta - \frac{1}{8}\frac{\sin^3\theta}{\theta} - \frac{1}{8}\sqrt{\left(-16\frac{\sin^3\theta}{\theta}\cos\theta + \frac{\sin^6\theta}{\theta^2} - 64\sin^2\theta\right)}$.

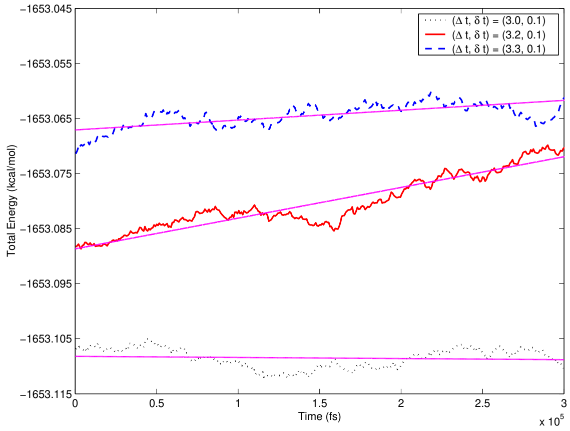

Fig. 3. Results for 300 ps of simulation for a 20 Ådiameter sphere of flexible TIP3P water at about 3.8 K using Verlet-I/r-RESPA. The fluctuation of the total energy is averaged out by showing only the average value of every 250 data points (spanning approximately 750 fs). The block-averaged total energy is then shifted to distinguish better among them. The integration is seen to be unstable at time steps of 3.2 fs. This shows evidence of nonlinear instability at around a third the fastest period.

that causes unbounded energy drift, even if the neighboring time steps are stable.

KAM theory permits the analysis of nonlinear instabilities near the equilibrium point of an integrator [51, p. 132–133]. For MTS integrators, the equilibrium point is close to, but not exactly, the state at zero temperature. We perform simulations of a flexible TIP3P water system of 20 Å of diameter with fastest period around 10 fs at 3.8 K.

The results show that in the neighborhood of $\Delta t = 3$ fs, there is an unstable resonance that manifests itself in a more pronounced drift at that time step than at neighboring time steps.

Figure 3 shows results near the equilibrium of the method. Signs of instability are evident around 3.2 fs, or about a third of the fastest period. Figure 4 shows results at 300 K that also display a slightly shifted resonance at 2.9 fs. The nonlinear stability analysis of the Verlet-I/r-RESPA method that explains these results is somewhat involved and beyond the scope of this paper. It is presented in [38].

5 Experimental Evaluation

To evaluate the performance of the MOLLY, we introduce the testing platform first, with sufficient implementation details for those readers interested in implementing the methods or reproducing results. Then we describe the test system and results.

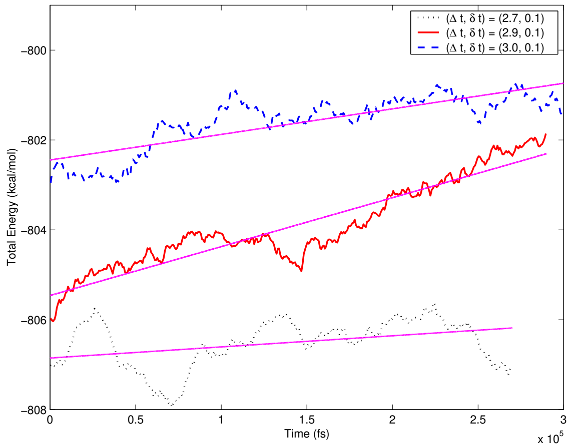

Fig. 4. Results for 300 ps of simulation for water at about 300 K using Verlet-I/r-RESPA. The fluctuation of the total energy is averaged out by showing only the average value of every 250 data points (spanning approximately 750 fs). The block-averaged total energy is then shifted to let them distinguish from each other. The integration is seen to be unstable at 2.9 fs of cyclelength. This shows evidence of nonlinear instability.

5.1 Implementation in ProtoMol

ProtoMol is a parallel, object-oriented, component-based framework for MD simulations [32,30,41]. The framework is designed for non-bonded, bonded, short-range and long-range forces for applications with tens of thousands of atoms representing biomolecules and solvents. It has a modular design that allows for easy prototyping of complex methods. For readers interested in software techniques that are helpful in implementing scientific software, we recommend to read about the object oriented framework POOMA [23] and the object oriented library for molecular simulation OOMPAA [24], both of which offer examples of powerful abstractions for scientific codes. Generic libraries such as the STL [48], Blitz++ [58], and MTL [49] are good references on how to write high performance software in C++ [55]. Excellent references to scientific computing in C++ are [3,59].

There are excellent MD programs such as AMBER [60], CHARMm [10], NAMD [34], SPASM [6], X-PLOR [12], PINY_MD [57], and many others [5,11,13,26,40,50]. Indeed, one has to answer the question, why another MD program? Our answer is that many design decisions of programs intended for production simulations render them inappropriate to serve as an algorithm development and academic research platform. In particular, to implement MOLLY methods, we identified the following requirements:

1. Support an arbitrary number of levels in MTS integrators. It should be easy for the user to compose different integrators in an MTS chain.
2. The algorithm developer should be able to easily associate a subset of forces with each integration level.
3. To accommodate MOLLY integrators, there should be a user defined pre-processing of coordinates (averaging) and a post-processing of forces (mollification).

In order to allow for user composition of MTS integrators with good performance we have implemented a twofold solution: an integrator definition language that allows the user to compose new MTS integrators at run time, and an integrator hierarchy that efficiently supports the integrator definition language. Object composition and inheritance are the main techniques for code reuse and design [16, p. 18].

```
Integrator {
  level N-1 integrator_name {
    cyclelength||timestep value
    force forcename forceoptions, [forcename forceoptions]
    ···   }
  ···
  level 0 integrator_name {
    ···   }
}
```

Program 1: Grammar for PROTOMOL's integrator definition language.

Integrator definition language. At the highest level, our solution is to provide users with an integrator definition language, where he or she can select the following: integrator to be used at each level and forces associated with that level. An integrator can be MTS or single time stepping (STS). Different STS integrators may be used to define different equations of motion. Associated with each level is a set of forces to be evaluated at that level, called a force group in our framework. This provides great flexibility to the user in partitioning the forces across the multiple levels. An abstract user definition of a new MTS integrator with N levels is given in Program 1. A MOLLY method is defined in Program 2. We use this language to describe the experiments that were performed below.

Integrator hierarchy and inheritance. Now we will discuss how we support this integrator definition language in PROTOMOL. We assume throughout this presentation that one uses the velocity or endpoint form of the integrators. This form can be more easily extended to multiple levels than the

```
Integrator {
  level 1 MOLLY {
    cyclelength 6
    force Coulomb -algorithm Full -switchingFunction ComplementSWC1}
  level 0 Leapfrog {
    timestep 1 fs
    force Improper, Dihedral, Bond, Angle
    force Coulomb -algorithm Cutoff -switchingFunction SWC1
    force LennardJones -algorithm Cutoff -switchingFunction SWC2}
}
```

Program 2: Two level MOLLY MTS integrator.

position or midpoint form of the integrators, even though the latter is more stable [4,8,21,51,54,56].

One can abstract the behavior of Verlet, Verlet-I/r-RESPA, and MOLLY in an algorithmic fashion as follows:

1. *halfkick();*
2. *doDriftOrVibration();*
3. *calculateForces();*
4. *halfkick();*

The function *doDriftOrVibration()* is the key to the abstraction. For an MTS integrator, it executes the next level of integration, whereas for an STS integrator, it executes the drift routine. The function *calculateForces()* evaluates each force in the *force group*. MOLLY also defines a pre-processing of the positions and a post-processing of the forces. The integrator class hierarchy is designed using inheritance (Fig. 5).

Relationship between integrator definition language and integrator hierarchy At run time, an integrator definition is interpreted and the correct integrator hierarchy is set up by PROTOMOL. This works because the integration methods are virtual, and are dynamically associated with the specific type of the integrator object that calls it. A virtual function allows for a common interface but specialized behavior. The beauty of virtual functions is that an existing code can be extended without modification. This does not hurt performance since integration is a relatively infrequent operation; most of the computing time is spent in force evaluation. An example of a 3-level chain of integrators set by PROTOMOL at run time is shown in Fig. 6.

5.2 Numerical tests

Numerical experiments were done using flexible water, based upon the TIP3P model [33], with flexibility incorporated by adding bond stretching and angle

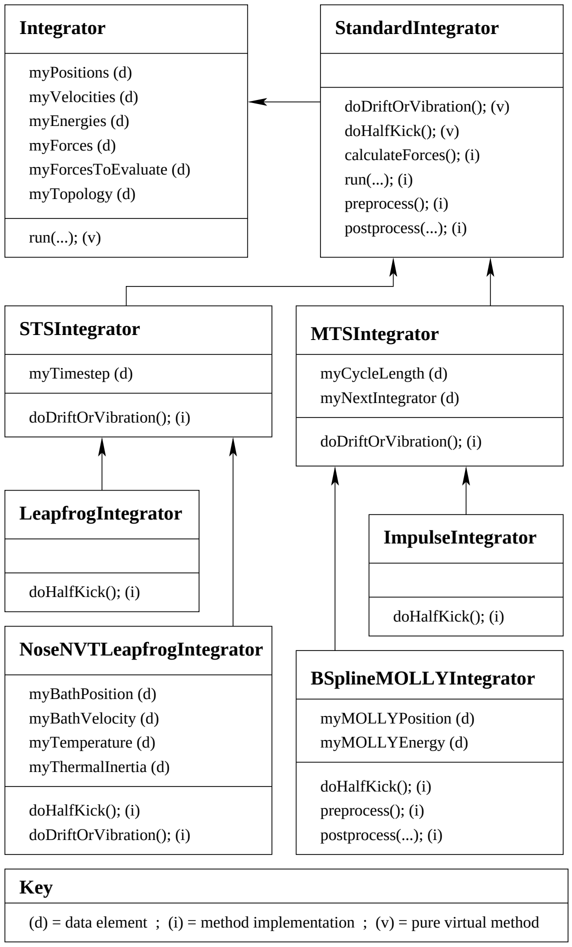

Fig. 5. Integrator hierarchy in ProtoMol.

bending harmonic terms (cf. [36, p. 184]). Experiments such at those in [9] suggest that flexible water models are particularly sensitive to destabilizing artifacts in numerical integrators. This is a system that has fastest motions with periods of around 10 fs. For each simulation a trace of the following information was generated: all of the components of the energy, positions (trajectories), velocities, and forces.

A small problem was used for the tests, consisting of a 10 Å radius sphere with 423 atoms equilibrated during 100 ps of simulation time by minimization followed by temperature rescaling to 300 K. By equilibrating we avoid highly improbable values of different contributions to energies. The potential energy

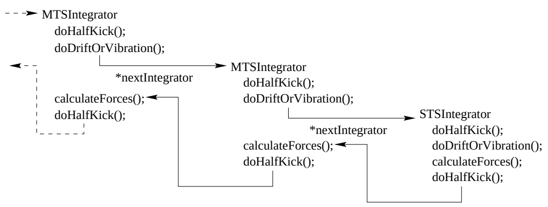

Fig. 6. Multiple time stepping integrator.

function for an electrostatic interaction is given by

$$U_{ij}^{\text{electrostatic}} = C\frac{q_i q_j}{x_{ij}}, \tag{23}$$

where $x_{ij} = \|x_j - x_i\|$ is the distance between atoms i and j, q_i is the charge for atom i, and $C = 332.0636\,\text{kcal}\,\text{mol}^{-1}\,\text{K}^{-1}$. The energy for a Lennard-Jones interaction is

$$U_{ij}^{\text{Lennard-Jones}} = 4\epsilon_{ij}\left(\left(\frac{\sigma_{ij}}{x_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{x_{ij}}\right)^{6}\right) SW(x_{ij}), \tag{24}$$

where ϵ_{ij} and σ_{ij} are the Lennard-Jones energy minimum and cross over point (where the LJ function is zero) and SW is a switching function defined below. The energy for a bond interaction is

$$U_k^{\text{bond}} = \frac{1}{2}K_{\text{B}}(x_{ij} - l_k)^2, \tag{25}$$

where K_{B} is a bond force constant and l_k is a reference bond length between atoms i and j for constraint k. Finally, the energy for an angle interaction is

$$U^{\text{angle}} = \frac{1}{2}K_{\text{A}}(\theta_k - \theta_0)^2, \tag{26}$$

where K_{A} is an angle force constant, and θ_k and θ_0 are the current value of the angle and the reference angle for angle constraint k.

For flexible water, $K_{\text{A}} = 55\,\text{kcal}\,\text{mol}^{-1}\,\text{degrees}^2$, $K_{\text{B}} = 450\,\text{kcal}\,\text{mol}^{-1}\,\text{Å}^2$, $q_{\text{O}} = 0.417\,\text{e}$, $q_{\text{H}} = -0.834\,\text{e}$, $l_{\text{O-H}} = 0.957\,\text{Å}$, and $\theta_0 = 104.52\,\text{degrees}$. The Lennard-Jones parameters are $\sigma_{\text{H-H}} = 0.4\,\text{Å}$, $\sigma_{\text{O-O}} = 3.1506$ Å, $\sigma_{\text{O-H}} = 1.75253\,\text{Å}$, $\epsilon_{\text{H-H}} = 0.046\,\text{kcal}\,\text{mol}^{-1}$, $\epsilon_{\text{O-O}} = 0.1521$ $\text{kcal}\,\text{mol}^{-1}$, $\epsilon_{\text{O-H}} = 0.08365$ $\text{kcal}\,\text{mol}^{-1}$.

Figure 7 illustrates these three methods at 5 fs The numerical results confirm the predictions of the linear stability analysis: the Bspline method called LongAverage overcomes the linear instability at half the fastest period or 5 fs better than ShortAverage. Unsurprisingly, the Verlet-I/rRESPA method is absolutely unstable there.

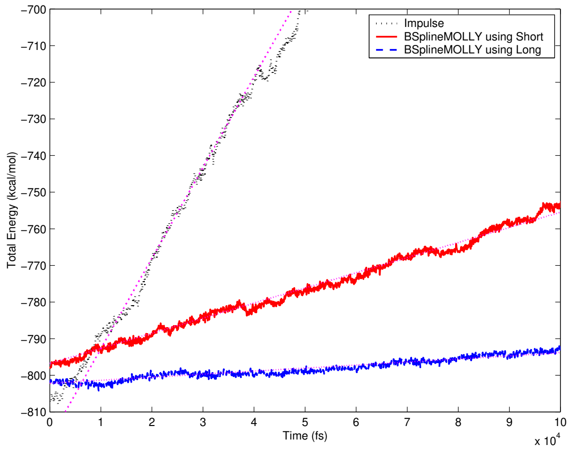

Fig. 7. Results for 100 ps of simulation for water at about 300 K using Impulse-Verlet and BSpline mollified integrators using ShortAverage and LongAverage as averaging B-Splines, all using $(\Delta t, \delta t) = (5.0, 1.0)$. The fluctuation of the total energy is averaged out by showing only the average value of every 10 data points (spanning approximately 50 fs). The block-averaged total energy is then shifted to let them distinguish from each other. The integration is seen to be for both Verlet-I/r-RESPA and ShortAverage MOLLY are unstable at $\Delta t = 5\,\mathrm{fs}$, but LongAverage MOLLY is stable at 5 fs.

Method \ Δt (fs)	3	4	5	6
Impulse	0.797 ± 0.006	1.626 ± 0.008	$--$	$--$
BMI-S	$--$	0.974 ± 0.008	21.684 ± 0.014	17.263 ± 0.024
BMI-L	-0.065 ± 0.006	0.711 ± 0.006	1.500 ± 0.010	8.139 ± 0.020

Table 1. Percent Relative Drift of total energy using B-spline Mollified Impulse integrators on the flexible TIP3P water model at $300K$ for 200 ps, where all methods use timestep $\delta t = 1.0fs$ for leapfrog integrator, and where *BMI-S* stands for **B**-spline **M**ollified **I**mpulse method using ShortAverage, and *BMI-L* stands for **B**-spline **M**ollified **I**mpulse method using LongAverage. The *Percent Relative Drift*, D, is computed using $D = (m \pm 2.0\sigma_m)\, t\, / \bar{K} * 100$ where m is the coefficient of the linear regression on the energy, and σ_m the standard deviation of m, t the simulation time, and $\bar{K}$ the average kinetic energy, respectively.

Table 1 shows the drift for a fixed simulation time of 200 ps relative to the average kinetic energy of the system. It is interesting that one observes a drift around 3 fs for Verlet-I/r-RESPA but not for the MOLLY methods. This drift corresponds to the 3:1 nonlinear instability that we postulate exists in Verlet-I/r-RESPA. *This means that the impulse method is not much better than leapfrog if one wants a simulation with no drift.*

In other papers we show how another MOLLY method achieves a computing time speedup of 38% over Verlet-I/r-RESPA using a longest time step that is 50% longer than the longest possible for Verlet-I/r-RESPA [31,27]. This method is implemented in NAMD 2 [34]. Using mild Langevin damping, speedups of 350% over Verlet-I/r-RESPA are possible, while preserving the dynamical properties [29].

6 Conclusions

We have shown how one can design mollified versions of Verlet-I/r-RESPA to overcome instabilities present in the method. MOLLY is a practical method that produces real speedups. In this paper we have shown simple analysis of the method to show how to overcome the linear instability at half the fastest period. The predictions of our analysis are confirmed by numerical experiments. We also pinpoint the fact that one needs to take care of nonlinear instabilities in Verlet-I/r-RESPA to take full advantage of the promise of MTS integrators. These instabilities have not been reported before for Verlet-I/r-RESPA, although they have been known to exist in Verlet or leapfrog.

We have also given extensive details on how to implement these and other MTS integrators, describing our particular implementation in the program PROTOMOL. We have provided the derivation of the Hessians of the CHARMM field for those interested in implementing the MOLLY methods, or who may be interested in using them for normal mode analysis.

Acknowledgments

We acknowledge the inspiration provided by Dr. Robert Skeel's work on MTS integrators in many aspects of this work. We are grateful to Dr. Atul Bahel for his implementation of some integrators into PROTOMOL. The following students have collaborated in the implementation of PROTOMOL: Trevor Cickovski, Thomas Steinbach, Scott Stender, Jeffrey Stine, and Joseph Taylor. This research was partially supported under NSF biocomplexity grant MPS-0083653. Thomas Slabach was suppported by a fellowship from the Center for Applied Mathematics at the University of Notre Dame. Thierry Matthey tested MOLLY in PROTOMOL in the Norwegian super-computing facilities in Bergen through a Norges Forskningsråd grant.

References

1. E. Barth and T. Schlick. Extrapolation versus impulse in multiple-timestepping schemes. II. Linear analysis and applications to Newtonian and Langevin dynamics. *J. Chem. Phys.*, 109(5):1633–1642, Aug 1998.
2. E. Barth and T. Schlick. Overcoming stability limitations in biomolecular dynamics. I. Combining force splitting via extrapolation with Langevin dynamics in LN. *J. Chem. Phys.*, 109(5):1617–1632, August 1998.
3. J. J. Barton and L. R. Nackman. *Scientific and Engineering C++: an introduction with advanced techniques and examples.* Addison-Wesley, Reading, Massachusetts, 1994.
4. P. F. Batcho and T. Schlick. Special stability advantages of position lVerlet over velocity Verlet in multiple-timestep integratio. *J. Chem. Phys.*, 2001.
5. D. M. Beazley and P. S. Lomdahl. Message-passing multi-cell molecular dynamics on the connection machine 5. *Parallel Computing*, 20:173–195, 1994.
6. D. M. Beazley and P. S. Lomdahl. Lightweight computational steering of very large scale molecular dynamics simulations. In *Proceedings of Supercomputing '96*, 1996.
7. H. J. C. Berendsen. Molecular dynamics simulations: The limits and beyond. In P. Deuflhard, J. Hermans, B. Leimkuhler, A. Mark, S. Reich, and R. D. Skeel, editors, *Computational Molecular Dynamics: Challenges, Methods, Ideas*, volume 4 of *Lecture Notes in Computational Science and Engineering*, pages 3–36. Springer-Verlag, Nov. 1998.
8. J. J. Biesiadecki and R. D. Skeel. Dangers of multiple-time-step methods. *J. Comput. Phys.*, 109(2):318–328, Dec. 1993.
9. T. Bishop, R. D. Skeel, and K. Schulten. Difficulties with multiple timestepping and the fast multipole algorithm in molecular dynamics. *J. Comput. Chem.*, 18(14):1785–1791, Nov. 15, 1997.
10. B. R. Brooks and M. Hodošček. Parallelization of CHARMm for MIMD machines. *CDA*, 7:16–22, Dec. 1992.
11. D. Brown, H. Minoux, and B. Maigret. A domain decomposition parallel processing algorithm for molecular dynamics simulations of systems of arbitrary connectivity. *Computer Physics Communications*, 103:170–186, 1997.
12. A. T. Brünger. *X-PLOR, Version 3.1: A System for X-ray Crystallography and NMR.* Yale University Press, New Haven and London, 1992.
13. T. W. Clark, R. v. Hanxleden, J. A. McCammon, and L. R. Scott. Parallelizing molecular dynamics using spatial decomposition. In *Proceedings of the Scalable High-Performance Computing Conference*, pages 95–102, Los Alamitos, Calif., 1994. IEEE Computer Society Press.
14. T. Darden, D. York, and L. Pedersen. Particle mesh Ewald. An N·log(N) method for Ewald sums in large systems. *J. Chem. Phys.*, 98:10089–10092, 1993.
15. D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications.* Academic Press, 1996.
16. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns. Elements of Reusable Object-Oriented Software.* Addison-Wesley, Reading, Massachusetts, 1995.
17. B. García-Archilla, J. M. Sanz-Serna, and R. D. Skeel. Long-time-step methods for oscillatory differential equations. *SIAM J. Sci. Comput.*, 20(3):930–963, Oct. 20, 1998.

18. B. García-Archilla, J. M. Sanz-Serna, and R. D. Skeel. The mollified impulse method for oscillatory differential equations. In D. F. Griffiths and G. A. Watson, editors, *Numerical Analysis 1997*, pages 111–123, London, 1998. Pitman.
19. A. Griewank, D. Juedes, and J. Utke. ADOL-C, a package for the automatic differentiation of algorithms written in C/C++. *ACM Trans. Math. Softw.*, 22(2):131–167, 1996.
20. H. Grubmüller. Dynamiksimulation sehr großer Makromoleküle auf einem Parallelrechner. Master's thesis, Physik-Dept. der Tech. Univ. München, Munich, 1989.
21. H. Grubmüller, H. Heller, A. Windemuth, and K. Schulten. Generalized Verlet algorithm for efficient molecular dynamics simulations with long-range interactions. *Molecular Simulation*, 6:121–142, 1991.
22. J. M. Haile. *Molecular Dynamics Simulation*. John Wiley and Sons, 1992.
23. S. Haney and J. Crotinger. How templates enable high-performance scientific computing in C++. *Computing In Science & Engineering*, 1(4):66–72, Jul-Aug 1999. POOMA reference.
24. G. A. Huber and J. A. McCammon. OOMPAA—Object-oriented model for probing assemblages of atoms. *J. Comput. Phys*, 151(1):264–282, May 1, 1999.
25. D. D. Humphreys, R. A. Friesner, and B. J. Berne. A multiple-time-step molecular dynamics algorithm for macromolecules. *J. Phys. Chem.*, 98(27):6885–6892, July 7, 1994.
26. Y.-S. Hwang, R. Das, J. H. Saltz, M. Hodošček, and B. R. Brooks. Parallelizing molecular dynamics programs for distributed-memory machines. *IEEE Computational Science & Engineering*, 2(2):18–29, Summer 1995.
27. J. A. Izaguirre. *Longer Time Steps for Molecular Dynamics*. PhD thesis, University of Illinois at Urbana-Champaign, 1999. Also UIUC Technical Report UIUCDCS-R-99-2107. Available online via `http://www.cs.uiuc.edu/research/tech-reports.html`.
28. J. A. Izaguirre. Generalized mollified multiple time stepping methods for molecular dynamics. In A. Brandt, J. Bernholc, and K. Binder, editors, *Multiscale Computational Methods in Chemistry and Physics*, volume 177 of *NATO Science Series: Series III Computer and Systems Sciences*, pages 34–47. IOS Press, Amsterdam, Netherlands, Jan 2001.
29. J. A. Izaguirre, D. P. Catarello, J. M. Wozniak, and R. D. Skeel. Langevin stabilization of molecular dynamics. *J. Chem. Phys.*, 114(5):2090–2098, Feb. 1, 2001.
30. J. A. Izaguirre, T. Matthey, J. Willcock, Q. Ma, B. Moore, T. Slabach, and G. Viamontes. A tutorial on the prototyping of multiple time stepping integrators for molecular dynamics. Available from `http://www.cse.nd.edu/~lcls/`Protomol.html, 2001.
31. J. A. Izaguirre, S. Reich, and R. D. Skeel. Longer time steps for molecular dynamics. *J. Chem. Phys.*, 110(19):9853–9864, May 15, 1999.
32. J. A. Izaguirre, J. Willcock, T. Matthey, T. B. Slabach, T. Steinbach, S. Stender, G. F. Viamontes, and J. Mohnke. ProtoMol: An object oriented framework for molecular dynamics. `http://www.cse.nd.edu/~lcls`, 2000.
33. W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.*, 79:926–935, 1983.

34. L. Kalé, R. Skeel, M. Bhandarkar, R. Brunner, A. Gursoy, N. Krawetz, J. Phillips, A. Shinozaki, K. Varadarajan, and K. Schulten. NAMD2: Greater scalability for parallel molecular dynamics. *J. Comp. Phys.*, 151:283–312, 1999.
35. M. Karplus. Molecular dynamics: Applications to proteins. In J.-L. Rivail, editor, *Modelling of Molecular Structures and Properties*, volume 71 of *Studies in Physical and Theoretical Chemistry*, pages 427–461, Amsterdam, 1990. Elsevier Science Publishers. Proceedings of an International Meeting.
36. A. R. Leach. *Molecular Modelling, Principles and Applications.* Addison Wesley Longman Limited, Essex, 1996.
37. M. López-Marcos, J. M. Sanz-Serna, and R. D. Skeel. Explicit symplectic integrators using Hessian–vector products. *SIAM J. Sci. Comput.*, 18:223–238, Jan. 1997.
38. Q. Ma, R. D. Skeel, and J. A. Izaguirre. Verlet-i/r-respa is nonlinearly unstable. In preparation, 2001.
39. A. D. MacKerell Jr., D. Bashford, M. Bellott, R. L. Dunbrack Jr., J. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, I. W. E. Reiher, B. Roux, M. Schlenkrich, J. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, and M. Karplus. All-hydrogen empirical potential for molecular modeling and dynamics studies of proteins using the CHARMM22 force field. *J. Phys. Chem. B*, 102:3586–3616, 1998.
40. T. Matthey and J. P. Hansen. Evaluation of MPI's one-sided communication mechanism for short-range molecular dynamics on the Origin2000. In *PARA2000 and Workshop on Applied Parallel Computing*, 2000.
41. T. Matthey and J. A. Izaguirre. ProtoMol: A molecular dynamics framework with incremental parallelization. In *Proc. of the Tenth SIAM Conf. on Parallel Processing for Scientific Computing (PP01)*, Proceedings in Applied Mathematics, Philadelphia, March 2001. Society for Industrial and Applied Mathematics.
42. M. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. Kalé, R. D. Skeel, and K. Schulten. NAMD – A parallel, object-oriented molecular dynamics program. *Int. J. Supercomput. Appl. High Perform. Comput.*, 10:251–268, 1996.
43. S. Reich. Dynamical systems, numerical integration, and exponentially small estimates, 1998. Habilitation Thesis.
44. A. Sandu and T. Schlick. Masking resonance artifacts in force-splitting methods for biomolecular simulations by extrapolative Langevin dynamics. *J. Comput. Phys*, 151(1):74–113, May 1, 1999.
45. J. Sanz-Serna and M. Calvo. *Numerical Hamiltonian Problems.* Chapman and Hall, London, 1994.
46. T. Schlick. Some failures and successes of long-timestep approaches to biomolecular simulations. In P. Deuflhard, J. Hermans, B. Leimkuhler, A. Mark, S. Reich, and R. D. Skeel, editors, *Algorithms for Macromolecular Modelling*, volume 4 of *Lecture Notes in Computational Science and Engineering*, pages 221–250. Springer-Verlag, 1998.
47. T. Schlick, M. Mandziuk, R. D. Skeel, and K. Srinivas. Nonlinear resonance artifacts in molecular dynamics simulations. *J. Comput. Phys.*, 139:1–29, 1998.
48. SGI. The Standard Template Library: Introduction. `http://www.sgi.com/Technology/STL/stl_introduction.html`.

49. J. G. Siek and A. Lumsdaine. The Matrix Template Library: A unifying framework for numerical linear algebra. In *International Symposium on Computing in Object-Oriented Parallel*, 1998. Also available from http://www.lsc.nd.edu/downloads/research/mtl/papers/mtl_poosc.pdf.
50. R. D. Skeel. Macromolecular dynamics on a shared-memory multiprocessor. *J. Comp. Chem.*, 12(2):175–179, January 1991.
51. R. D. Skeel. Integration schemes for molecular dynamics and related applications. In M. Ainsworth, J. Levesley, and M. Marletta, editors, *The Graduate Student's Guide to Numerical Analysis*, SSCM, pages 119–176. Springer-Verlag, Berlin, 1999.
52. R. D. Skeel and J. J. Biesiadecki. Symplectic integration with variable stepsize. *Annals of Numer. Math.*, 1:191–198, 1994.
53. R. D. Skeel and J. Izaguirre. The five femtosecond time step barrier. In P. Deuflhard, J. Hermans, B. Leimkuhler, A. Mark, S. Reich, and R. D. Skeel, editors, *Computational Molecular Dynamics: Challenges, Methods, Ideas*, volume 4 of *Lecture Notes in Computational Science and Engineering*, pages 303–318. Springer-Verlag, Berlin Heidelberg New York, Nov. 1998.
54. R. D. Skeel, G. Zhang, and T. Schlick. A family of symplectic integrators: stability, accuracy, and molecular dynamics applications. *SIAM J. Sci. Comput.*, 18(1):203–222, Jan. 1997.
55. B. Stroustrup. *The C++ Programming Language*. Addison-Wesley, third edition, 1997.
56. M. Tuckerman, B. J. Berne, and G. J. Martyna. Reversible multiple time scale molecular dynamics. *J. Chem. Phys*, 97(3):1990–2001, 1992.
57. M. E. Tuckerman, D. Yarne, S. O. Samuelson, A. L. Hughes, and G. J. Martyna. Exploiting multiple levels of parallelism in Molecular Dynamics based calculations via modern techniques and software paradigms on distributed memory computers. *CPC*, 128:333–376, 2000.
58. T. Veldhuizen. Blitz++: The library that thinks it is a compiler. Conference presentation, Extreme! Computing Laboratory, Indiana University Computer Science Department, Sep. 1998. http://oonumerics.org/blitz/blitztalk.ps.gz.
59. T. Veldhuizen. Techniques for scientific c++. Technical Report 542, Indiana University Computer Science Department, 2000. http://extreme.indiana.edu/~tveldhui/papers/techniques/.
60. J. Vincent and K. M. Merz. A highly portable parallel implementation of Amber using the message passing interface standard. *J. Comp. Chem.*, 11:1420–1427, 1995.

Appendix Derivation of the Hessians for MOLLY

Here we derive the Hessians for most forces used in the CHARMm force field [39]. These have been validated using automatic differentiation tools [19] and symbolic differentiation tools such as Mathematica.

Hessian of Bond Energy

Bonds [42] describe a linear bond between two atoms. These bonds are described by a simple harmonic springs. The energy of a bond between atoms

i and j is given by:

$$E_{bond} = k\left(|\boldsymbol{r}_{ij}| - r_0\right)^2, \tag{27}$$

where k is the spring constant, $\boldsymbol{r}_{ij} = \boldsymbol{r}_j - \boldsymbol{r}_i$, and r_0 is the rest distance of the bond. Its derivatives are:

$$E_r^{\text{bond}} = 2k(|\boldsymbol{r}_{ij}| - r_0)[-\hat{r_{ij}}, \hat{r_{ij}}]^T, \tag{28}$$

$$E_{rr}^{\text{bond}} = 2k\frac{|\boldsymbol{r}_{ij}| - r_0}{|\boldsymbol{r}_{ij}|}\begin{bmatrix} I & -I \\ -I & I \end{bmatrix} + \frac{2\,k\,r_0}{|\boldsymbol{r}_{ij}|}\begin{bmatrix} \hat{v_{ij}}\hat{v_{ij}}^T & -\hat{r_{ij}}\hat{r_{ij}}^T \\ -\hat{v_{ij}}\hat{v_{ij}}^T & \hat{r_{ij}}\hat{r_{ij}}^T \end{bmatrix}. \tag{29}$$

6.1 Hessian of the Angle Energy

Angle interactions describe angular bonds between three atoms. These bonds are modeled as harmonic angular springs. The energy of such a bond between atoms i, j, and k is given by:

$$E_{angle} = E_\theta + E_{\text{ub}}, \tag{30}$$

$$E_\theta = k_\theta\left(\theta - \theta_0\right)^2, \tag{31}$$

$$E_{\text{ub}} = k_{\text{ub}}\left(|\boldsymbol{r}_{ik}| - r_{\text{ub}}\right)^2, \tag{32}$$

where k_θ is the force constant, $\theta = \cos^{-1}\left(\frac{\boldsymbol{r}_{ij}\cdot\boldsymbol{r}_{kj}}{|\boldsymbol{r}_{ij}||\boldsymbol{r}_{kj}|}\right)$, θ_0 is the rest angle of this bond, k_{ub} is Urey-Bradley constant, $\boldsymbol{r}_{ik} = \boldsymbol{r}_k - \boldsymbol{r}_i$, $|\boldsymbol{r}_{ik}|$ is the calculated distance between atoms i and k and r_{ub} is rest distance for the Urey-Bradley term.

The Hessian of the Urey-Bradley energy of Equation (30) can be obtained easily:

$$E_{rr}^{\text{ub}} = 2k_{\text{ub}}\begin{bmatrix} I & 0 & -I \\ 0 & 0 & 0 \\ -I & 0 & I \end{bmatrix} + \frac{2\,k_{\text{ub}}\,r_{\text{ub}}}{|\boldsymbol{r}_{ik}|}\begin{bmatrix} \hat{r_{ik}}\hat{r_{ik}}^T - I & 0 & -\hat{r_{ik}}\hat{r_{ik}}^T + I \\ 0 & 0 & 0 \\ -\hat{r_{ik}}\hat{r_{ik}}^T + I & 0 & \hat{r_{ik}}\hat{r_{ik}}^T - I \end{bmatrix}. \tag{33}$$

For E_θ, let $C(\alpha,\beta,\gamma) = \frac{\alpha+\beta-\gamma}{2\sqrt{\alpha}\sqrt{\beta}}$ where α, β and γ are scalars, and $\alpha = |\boldsymbol{r}_{ij}|^2$, $\beta = |\boldsymbol{r}_{kj}|^2$, and $\gamma = |\boldsymbol{r}_{ki}|^2$.

Now the angle energy can be expressed as

$$E_\theta(C(\alpha,\beta,\gamma)) = k_\theta(\cos^{-1}(C(\alpha,\beta,\gamma)) - \theta_0)^2. \tag{34}$$

The second derivative of the E_θ part is then:

$$E_{rr}^\theta = \overbrace{E_C\,C_{rr}}^{E_{rr}^a} + \underbrace{\left(\frac{2\,k_\theta[\sin\theta - (\theta-\theta_0)\cos\theta]}{\sin^3\theta}\right)C_r C_r^T}_{E_{rr}^b}, \tag{35}$$

where $E_C = -\frac{2k_\theta(\theta-\theta_0)}{\sin\theta}$, $C_r = f\,\alpha_r + g\,\beta_r + h\,\gamma_r$ in which $f = \frac{\alpha-\beta+\gamma}{4\,\alpha^{3/2}\sqrt{\beta}}$, $g = \frac{-\alpha+\beta+\gamma}{4\sqrt{\alpha}\,\beta^{3/2}}$, $h = -\frac{1}{2\sqrt{\alpha}\sqrt{\beta}}$, $\alpha_r = (-2,2,0)^T\sqrt{\alpha}\;\hat{r_{ij}}$, $\beta_r = (0,2,-2)^T\sqrt{\beta}\;\hat{r_{kj}}$, and $\gamma_r = (2,0,-2)^T\sqrt{\gamma}\;\hat{r_{ki}}$. The C_{rr} is computed as follows:

$$C_{rr} = \overbrace{(f\,\alpha_{rr} + g\,\beta_{rr} + h\,\gamma_{rr})}^{C^a_{rr}} + \underbrace{(\alpha_r\,f_r^T + \beta_r\,g_r^T + \gamma_r\,h_r^T)}_{C^b_{rr}}, \tag{36}$$

where

$$\alpha_{rr} = \begin{bmatrix} 2I & -2I & 0 \\ -2I & 2I & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \beta_{rr} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2I & -2I \\ 0 & -2I & 2I \end{bmatrix}, \quad \gamma_{rr} = \begin{bmatrix} 2I & 0 & -2I \\ 0 & 0 & 0 \\ -2I & 0 & 2I \end{bmatrix}, \tag{37}$$

and

$$\begin{aligned} &f_r = f_\alpha\,\alpha_r + f_\beta\,\beta_r + f_\gamma\,\gamma_r, \quad g_r = g_\alpha\,\alpha_r + g_\beta\,\beta_r + g_\gamma\,\gamma_r, \\ &h_r = h_\alpha\,\alpha_r + h_\beta\,\beta_r + h_\gamma\,\gamma_r, \end{aligned} \tag{38}$$

where

$$f_\alpha = \frac{-\alpha+3\beta-3\gamma}{8\,\alpha^{5/2}\sqrt{\beta}}, \quad f_\beta = \frac{-\alpha-\beta-\gamma}{8\,\alpha^{3/2}\,\beta^{3/2}}, \quad f_\gamma = \frac{1}{4\alpha^{3/2}\sqrt{\beta}}, \tag{39}$$

$$g_\alpha = f_\beta, \quad g_\beta = \frac{3\alpha-\beta-3\gamma}{8\sqrt{\alpha}\,\beta^{5/2}}, \quad g_\gamma = \frac{1}{4\sqrt{\alpha}\,\beta^{3/2}}, \tag{40}$$

$$h_\alpha = f_\gamma, \quad h_\beta = g_\gamma, \quad h_\gamma = 0. \tag{41}$$

In Equation (36), the first part, C^a_{rr}, becomes

$$C^a_{rr} = \begin{bmatrix} 2(f+h)I & -2fI & -2hI \\ -2fI & 2(f+g)I & -2gI \\ -2hI & -2gI & 2(g+h)I \end{bmatrix}, \tag{42}$$

whereas the second part, C^b_{rr}, becomes

$$\begin{aligned} C^b_{rr} = {} & f_\alpha\,\alpha_r\alpha_r^T + f_\beta\,\alpha_r\beta_r^T + f_\gamma\,\alpha_r\gamma_r^T + g_\alpha\,\beta_r\alpha_r^T + g_\beta\,\beta_r\beta_r^T + \\ & g_\gamma\,\beta_r\gamma_r^T + h_\alpha\,\gamma_r\alpha_r^T + h_\beta\,\gamma_r\beta_r^T + h_\gamma\,\gamma_r\gamma_r^T, \end{aligned} \tag{43}$$

where

$$\alpha_r \alpha_r^T = \begin{bmatrix} 4\,\hat{r_{ij}}\hat{r_{ij}}^T & -4\,\hat{r_{ij}}\hat{r_{ij}}^T & 0 \\ -4\,\hat{r_{ij}}\hat{r_{ij}}^T & 4\,\hat{r_{ij}}\hat{r_{ij}}^T & 0 \\ 0 & 0 & 0 \end{bmatrix} \alpha \tag{44}$$

$$\beta_r \beta_r^T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4\,\hat{r_{jk}}\hat{r_{jk}}^T & -4\,\hat{r_{jk}}\hat{r_{jk}}^T \\ 0 & -4\,\hat{r_{jk}}\hat{r_{jk}}^T & 4\,\hat{r_{jk}}\hat{r_{jk}}^T \end{bmatrix} \beta \tag{45}$$

$$\gamma_r \gamma_r^T = \begin{bmatrix} 4\,\hat{r_{ik}}\hat{r_{ik}}^T & 0 & -4\,\hat{r_{ik}}\hat{r_{ik}}^T \\ 0 & 0 & 0 \\ -4\,\hat{r_{ik}}\hat{r_{ik}}^T & 0 & 4\,\hat{r_{ik}}\hat{r_{ik}}^T \end{bmatrix} \gamma \tag{46}$$

$$\beta_r \alpha_r^T = \begin{bmatrix} 0 & 0 & 0 \\ -4\,\hat{r_{kj}}\hat{r_{ij}}^T & 4\,\hat{r_{kj}}\hat{r_{ij}}^T & 0 \\ 4\,\hat{r_{kj}}\hat{r_{ij}}^T & -4\,\hat{r_{kj}}\hat{r_{ij}}^T & 0 \end{bmatrix} \sqrt{\alpha}\sqrt{\beta} \tag{47}$$

$$\gamma_r \alpha_r^T = \begin{bmatrix} -4\,\hat{r_{ki}}\hat{r_{ij}}^T & 4\,\hat{r_{ki}}\hat{r_{ij}}^T & 0 \\ 0 & 0 & 0 \\ 4\,\hat{r_{ki}}\hat{r_{ij}}^T & -4\,\hat{r_{ki}}\hat{r_{ij}}^T & 0 \end{bmatrix} \sqrt{\alpha}\sqrt{\gamma} \tag{48}$$

$$\gamma_r \beta_r^T = \begin{bmatrix} 0 & 4\,\hat{r_{ki}}\hat{r_{kj}}^T & -4\,\hat{r_{ki}}\hat{r_{kj}}^T \\ 0 & 0 & 0 \\ 0 & -4\,\hat{r_{ki}}\hat{r_{kj}}^T & 4\,\hat{r_{ki}}\hat{r_{kj}}^T \end{bmatrix} \sqrt{\beta}\sqrt{\gamma} \tag{49}$$

$$\alpha_r \beta_r^T = (\beta_r \alpha_r^T)^T, \quad \alpha_r \gamma_r^T = (\gamma_r \alpha_r^T)^T, \quad \beta_r \gamma_r^T = (\gamma_r \beta_r^T)^T. \tag{50}$$

The second part of Equation (35) becomes

$$\begin{aligned} E_{rr}^b = & \left(\frac{2\,k_\theta [\sin\theta - (\theta - \theta_0)\cos\theta]}{\sin^3\theta} \right) (f^2\,\alpha_r \alpha_r^T + \\ & g\,f\,\beta_r \alpha_r^T + h\,f\,\gamma_r \alpha_r^T + \\ & g\,f\,\alpha_r \beta_r^T + g^2\,\beta_r \beta_r^T + h\,g\,\gamma_r \beta_r^T + \\ & h\,f\,\alpha_r \gamma_r^T + g\,h\,\beta_r \gamma_r^T + h^2\,\gamma_r \gamma_r^T). \end{aligned} \tag{51}$$

Hessian of van der Waals Energy

The van der Waals interactions describe the forces resulting from local interactions of atoms. The van der Walls energy between two atoms i and j is described by

$$E_{\mathrm{vdw}} = \frac{A}{|\boldsymbol{r}_{ij}|^{12}} - \frac{B}{|\boldsymbol{r}_{ij}|^{6}} \tag{52}$$

where A and B are constants specified for a pair of atom types explicitly in the parameter file, $\boldsymbol{r}_{ij} = \boldsymbol{r}_j - \boldsymbol{r}_i$, the vector from atom i to atom j, $|\boldsymbol{r}_{ij}|$ is the length of vector $\boldsymbol{r}_{ij}$.

The first derivative of the van der Waals energy is as follows.

$$E_r = \left(\frac{6\,B}{|\boldsymbol{r}_{ij}|^8} - \frac{12\,A}{|\boldsymbol{r}_{ij}|^{14}} \right) [-\boldsymbol{r_{ij}}, \boldsymbol{r_{ij}}]^T. \tag{53}$$

The Hessian of van der Waals energy is as follows.

$$E_{rr} = C_1 \begin{bmatrix} I & -I \\ -I & I \end{bmatrix} + C_2 \begin{bmatrix} \hat{r_{ij}}\,\hat{r_{ij}}^T & -\hat{r_{ij}}\,\hat{r_{ij}}^T \\ -\hat{r_{ij}}\,\hat{r_{ij}}^T & \hat{r_{ij}}\,\hat{r_{ij}}^T \end{bmatrix}. \tag{54}$$

where $C_1 = \frac{6\,B}{|\boldsymbol{r}_{ij}|^8} - \frac{12\,A}{|\boldsymbol{r}_{ij}|^{14}}$, and $C_2 = \frac{-48\,B}{|\boldsymbol{r}_{ij}|^8} + \frac{168\,A}{|\boldsymbol{r}_{ij}|^{14}}$.

Hessian of Electrostatic Energy

Electrostatics describes the force resulting from the interaction between two charged particles. The electrostatic energy between two atoms i and j is described by the Coulomb's Law as:

$$E_{\text{elect}} = \frac{\epsilon_{14}\,C\,q_i\,q_j}{\epsilon_0\,|\boldsymbol{r}_{ij}|} \tag{55}$$

where ϵ_{14} is scaling factor for $1-4$ interactions, $C = 2.31 \times 10^{-19}\,\mathrm{J\,nm}$, q_i, q_j are charges for atom i and j, ϵ_0 is dielectric constant, $\boldsymbol{r}_{ij}$ and $|\boldsymbol{r}_{ij}|$ are same as defined before.

The first derivative of the electrostatic energy is as follows.

$$E_r = -\frac{\epsilon_{14}\,C\,q_i\,q_j}{\epsilon_0\,|\boldsymbol{r}_{ij}|^3}\,[-\boldsymbol{r_{ij}},\,\boldsymbol{r_{ij}}]. \tag{56}$$

The Hessian of electrostatic energy is as follows.

$$E_{rr} = -\frac{\epsilon_{14}\,C\,q_i\,q_j}{\epsilon_0\,|\boldsymbol{r}_{ij}|^3} \begin{bmatrix} I - 3\hat{r_{ij}}\,\hat{r_{ij}}^T & -I + 3\hat{r_{ij}}\,\hat{r_{ij}}^T \\ -I + 3\hat{r_{ij}}\,\hat{r_{ij}}^T & I - 3\hat{r_{ij}}\,\hat{r_{ij}}^T \end{bmatrix}. \tag{57}$$

Hessian of Switching Functions

A switching function is often applied to a nonbonded energy computation to suppress the destablizing factor introduced by the cutoff approximations. There are two popular switching functions, *SWC1* [52] and *SWC2* [25]:

$$SWC1\,(\boldsymbol{r}_{ij}) = \begin{cases} 1 - (\frac{3}{2}|\boldsymbol{r}_{ij}|{r_1}^2 - \frac{1}{2}|\boldsymbol{r}_{ij}|^3){r_1}^{-3} & \text{if} |\boldsymbol{r}_{ij}| \le r_1, \\ 0 & \text{if} |\boldsymbol{r}_{ij}| > r_1, \end{cases} \tag{58}$$

where r_1 is the distance where the function value becomes zero. The first and second derivatives of this switching function is given as follows. Let Y be the

SWC1 function, then

$$Y_r = -\frac{3}{2r_0}(-\hat{r_{ij}}, \hat{r_{ij}})^T + \frac{3|\boldsymbol{r}_{ij}|^2}{2r_0^3}(-\hat{r_{ij}}, \hat{r_{ij}})^T, \tag{59}$$

$$\begin{aligned} Y_{rr} &= (-\frac{3}{2r_1|\boldsymbol{r}_{ij}|} + \frac{3|\boldsymbol{r}_{ij}|}{2r_1^3}) \begin{bmatrix} I & -I \\ -I & I \end{bmatrix} \\ &+ (\frac{3}{2r_1|\boldsymbol{r}_{ij}|} + \frac{3|\boldsymbol{r}_{ij}|}{2r_1^3}) \begin{bmatrix} \hat{r_{ij}}\,\hat{r_{ij}}^T & -\hat{r_{ij}}\,\hat{r_{ij}}^T \\ -\hat{r_{ij}}\,\hat{r_{ij}}^T & \hat{r_{ij}}\,\hat{r_{ij}}^T \end{bmatrix}. \end{aligned} \tag{60}$$

The other switching function, *SWC2*, is more expensive to evaluate, and is given by:

$$SWC2(\boldsymbol{r}_{ij}) = \begin{cases} 1 & \text{if} |\boldsymbol{r}_{ij}| \le r_0, \\ \frac{(|\boldsymbol{r}_{ij}|^2 - {r_1}^2)^2({r_1}^2 + 2|\boldsymbol{r}_{ij}|^2 - 3{r_0}^2)}{({r_1}^2 - {r_0}^2)^3} & \text{if} r_0 \le |\boldsymbol{r}_{ij}| < r_1, \\ 0 & \text{if} |\boldsymbol{r}_{ij}| > r_1, \end{cases} \tag{61}$$

where r_1 is the distance where the function value becomes zero, and r_0 that where it becomes active. Let Y be the *SWC2* function which is active, the first derivative is given by

$$Y_r = \frac{12(|\boldsymbol{r}_{ij}|^2 - {r_1}^2)(|\boldsymbol{r}_{ij}|^2 - {r_0}^2)}{({r_1}^2 - {r_0}^2)^3}(-\boldsymbol{r}_{ij}, \boldsymbol{r}_{ij})^T \tag{62}$$

The Hessian matrix is given by two parts:

$$Y_{rr} = Y_{rr}^a + Y_{rr}^b \tag{63}$$

where

$$Y_{rr}^a = \frac{12(|\boldsymbol{r}_{ij}|^2 - {r_1}^2)(|\boldsymbol{r}_{ij}|^2 - {r_0}^2)}{({r_1}^2 - {r_0}^2)^3} \begin{bmatrix} I & -I \\ -I & I \end{bmatrix}, \tag{64}$$

$$Y_{rr}^b = \frac{24|\boldsymbol{r}_{ij}|^2(2|\boldsymbol{r}_{ij}|^2 - {r_0}^2 - {r_1}^2)}{({r_1}^2 - {r_0}^2)^3} \begin{bmatrix} \hat{r_{ij}}\,\hat{r_{ij}}^T & -\hat{r_{ij}}\,\hat{r_{ij}}^T \\ -\hat{r_{ij}}\,\hat{r_{ij}}^T & \hat{r_{ij}}\,\hat{r_{ij}}^T \end{bmatrix}. \tag{65}$$

Hessian of Nonbonded Energy With Switching Functions

When the effective energy (E_e) is taken as the raw energy (E) multiplied by a switching function (Y), *i.e.*,

$$E_e = EY,$$

the Hessian matrix of the effective energy is given as the following using chain rule:

$$\frac{\partial^2 E_e}{\partial r^2} = \frac{\partial^2 E}{\partial r^2} Y + 2\frac{\partial Y}{\partial r}\left(\frac{\partial E}{\partial r}\right)^T + \frac{\partial^2 Y}{\partial r^2} E. \tag{66}$$

These are nearly all the formulas needed to implement BSpline MOLLY methods that compute Hessians. Dihedral and improper Hessians may be derived in a way analogous to the angle Hessian.