MILCH SHAKE: An efficient method for constraint dynamics applied to alkanes

A.G. Bailey
Department of Physics, Imperial College London
South Kensington Campus, Prince Consort Road
London SW7 2AZ, UK

C.P. Lowe
van 't Hoff Institute for Molecular Science, Universiteit van Amsterdam
Nieuwe Achtergracht 188
1018 WV Amsterdam, The Netherlands

December 3, 2008

Summary of contribution

We describe a method to impose constraints in a molecular dynamics simulation. A technique developed to solve the special case of a linearly topology (MILC SHAKE) is hybridized with the SHAKE algorithm. The methodology, which we term MILC-hybridized SHAKE (or MILCH SHAKE) applies to more complex topologies. Here we consider the important case of all atom models of alkanes. Exploiting the mass difference between carbon and hydrogen we show that for higher alkanes MILCH SHAKE can be an order of magnitude faster than SHAKE.

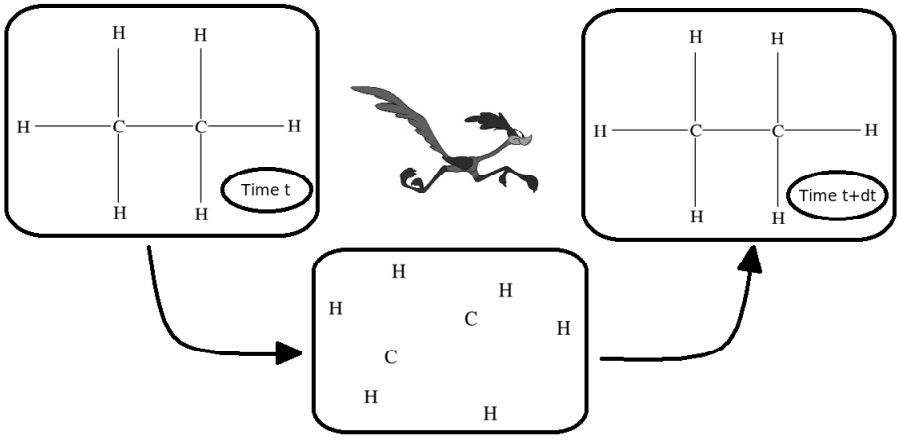
Keywords: Constraints, SHAKE, Molecular dynamics, Simulations, Alkanes

Graphical Abstract

MILCH SHAKE: An efficient method for constraint dynamics applied to alkanes

A.G. Bailey and C.P. Lowe
Department of Physics, Imperial College London, Prince Consort Road, London SW7 2AZ, UK
van 't Hoff Institute of Molecular Science, Universiteit van Amsterdam, Nieuwe Achtergracht 188, 1018 WV Amsterdam, The Netherlands

We describe a method to impose constraints in a molecular dynamics simulation. A technique developed to solve the special case of a linearly topology (MILC SHAKE) is hybridized with the SHAKE algorithm. The methodology, which we term MILC-hybridized SHAKE (or MILCH SHAKE) applies to more complex topologies. Here we consider the important case of all atom models of alkanes. Exploiting the mass difference between carbon and hydrogen we show that for higher alkanes MILCH SHAKE can be an order of magnitude faster than SHAKE.



Abstract

We describe a method to impose constraints in a molecular dynamics simulation. A technique developed to solve the special case of a linearly topology (MILC SHAKE) is hybridized with the SHAKE algorithm. The methodology, which we term MILC-hybridized SHAKE (or MILCH SHAKE) applies to more complex topologies. Here we consider the important case of all atom models of alkanes. Exploiting the mass difference between carbon and hydrogen we show that for higher alkanes MILCH SHAKE can be an order of magnitude faster than SHAKE. Keywords: Constraints, SHAKE, Molecular dynamics, Simulations, Alkanes

1 Introduction

Imposing geometric constraints is a useful technique in molecular dynamics (MD) simulations. For example, entire molecules can be treated as rigid bodies [1, 2, 3]. If more detail is necessary, then fixing the distance between atoms is a reasonable way of modeling chemical bonds. Further, it integrates out the fast vibrational degrees of freedom associated with the stiff inter-atomic bonding potential. Consequently, much longer time steps are possible, allowing a faster sampling of phase space. Here we address the question of efficiently imposing such constraints. The methodology itself is appropriate for problems where constraints are appropriate. We say this because using constraints is not a unique solution for solving this problem. Multiple time step methods use a shorter time step for the fast degrees of freedom and a longer one for the remainder. The class of algorithms introduced by Tuckerman *et al.* are also symplectic [4]. Subsequent developments have also cured some of the numerical problems associated with the original method [5, 6, 7, 8, 9]. Multiple time stepping does not have the well known drawback of constraints, that averages in the constrained system differ from those of the unconstrained system [10]. However, for complex molecules this is a small effect and constraints might be the preferred option. In addition, there are arguments that treating bonds as constraints is actually more realistic than treating them as classical harmonic oscillators [11]. The two approaches are not necessarily mutually exclusive either. A hybrid approach combining multiple time stepping and constraints can also be efficient [12].

A formalism for tackling the constraint problem numerically was derived by Ryckaert *et al.* [13]. The idea is to apply a constraint force at time t such that the constraints are satisfied at $t + \Delta t$, where Δt is the time step. This means that constraint-related numerical errors from integrating the equations of motion do not propagate. Their methodology is not unique. For example de Leeuw *et al.* [14] derived the conjugate momenta and Hamiltonian for a system subject to such constraints, from which the equations of motion directly follow. However, using this method constraint-related numerical errors do propagate and an additional procedure is necessary to correct for this.

Along with their methodology for imposing constraints using Lagrange's method of undetermined multipliers, Ryckaert *et al.* [13] concurrently developed a simple iterative algorithm to calculate the magnitudes of the Lagrange multipliers associated with the constraint forces. This combination of the method for stopping the propagation of constraint-related numerical errors and the iterative procedure for calculating the Lagrange multipliers is referred to as SHAKE. Subsequently, this procedure was extended to also impose constraints on the relative velocity (the RATTLE algorithm [15]) and acceleration (the WIGGLE algorithm [16]). Further, it is not actually limited to bond length constraints. For example the Q-SHAKE algorithm [17] uses SHAKE to constrain a set of linked rigid bodies representing a molecule. However, here we only consider its application to bond length constraints.

Within the framework of the SHAKE procedure, the iterative scheme for calculating the multipliers can be replaced with several other methods. For example, Ciccotti *et al.* developed a method based on matrix inversion [2]. The LINCS algorithm solves for the multipliers using a series expansion approximation of the inverse of the linearized constraint equations. This has an advantage in that it is easy to efficiently parallelize. This was not true of the original iterative scheme. However, by rephrasing the problem Elber [18] developed an iterative method that does work efficiently in parallel.

In this paper, we always use the SHAKE methodology. Where we use the term 'SHAKE', from now on, it refers specifically to the iterative part of the full SHAKE procedure, the calculation of the Lagrange

multipliers. For a system with n constraints, a set of n non-linear equations must be solved. Using the SHAKE iteration, these equations are decoupled and solved sequentially. This procedure violates constraints previously satisfied, but, applied iteratively, the error generally shrinks and the scheme usually converges. The SHAKE iteration is generally applicable, but the iterative scheme can be slowly convergent for some topologies. For problems involving small numbers of constraints, in MD terms small molecules, alternative methods are more efficient. For example, for water molecules the SETTLE algorithm [19] essentially uses an analytic solution for the constraints problem. Kräutler *et al.* [11] also showed with their M-SHAKE algorithm that approximating the solution of the constraint equations using Newton's method, at a cost of order n^3 per iteration, can be more efficient than SHAKE for small molecules (small n). This is because the convergence is quadratic, whereas for SHAKE it is linear. More recently, Gonnet developed the P-SHAKE method [20] that uses a pre-conditioner to reduce the computational cost per iteration to order n^2 while maintaining quadratic convergence. For particles constrained to lie on a straight line, Tapia-McClung and Gronbech-Jensen [21] showed that an iterative procedure is actually unnecessary.

The set of constraint equations that are solved for the Lagrange multipliers can be written in matrix form. The recently developed algorithm "Matrix Inverted Linearized Constraints" SHAKE (MILC SHAKE) takes into account that under certain circumstances the Jacobian of the constraint matrix is tridiagonal [22], and trivially inverted in order n operations [23]. This being the case, the cost per iteration, using a simplified Newton iteration to solve for the constraint force, is order n. The condition required is that we have a linear architecture. That is, a chain in which only successive sites are connected. The linear chain is encountered in several situations. An example in the field of MD is the simulation of alkanes using a united atom model (UAM) [24]. Similar methodology is also used for mesoscopic simulation of inextensible filaments of the type frequently encountered in biological systems [25, 26]. Bailey *et al.* [22] demonstrated that direct matrix inversion using MILC SHAKE can speed up constraint calculations by orders of magnitude compared to SHAKE.

Although MILC SHAKE is a powerful technique for linear architectures, one immediately asks the question, can it be generalized? After all, real molecules are not usually linear. Let us focus on a particularly important class of chemicals, the alkanes. While MILC SHAKE can be used to calculate constraints of linear alkanes using a UAM, this is frequently an inadequate description of dense liquid and solid phases. It is well established that not treating hydrogen atoms explicitly will yield an incorrect scaling of system pressure with temperature and density [27, 28]. Greater chemical detail is also necessary when looking at the molecular stacking behavior near interfaces [29, 30]. Furthermore, Chen *et al.* pointed out that the explicit-hydrogen models have the added advantage of allowing one to assign partial charges. This can be relevant when determining the interaction of alkanes in a polar environment [31]. In an attempt to overcome the deficiencies of the united atom model, yet retain computational viability, an anisotropic UAM [31] and the TraPPE-EH potential [32] have been proposed. However, if there were a sufficiently efficient method to incorporate the hydrogens using constraints, this would clearly be preferable.

Here we address the question of calculating constraints of non-linear architectures by developing and testing an extension of the plain vanilla MILC SHAKE algorithm. It is "Matrix Inverted Linearized Constraints" hybridized with SHAKE iteration, so subsequently we refer to it as MILCH SHAKE. We test our method on n-alkanes ranging from ethane to dodecane (twelve carbons). The results show that in all the cases considered MILCH SHAKE outperforms SHAKE. The improvement in efficiency relative to SHAKE is a significant factor for ethane and up to an order of magnitude for higher alkanes. Furthermore, the CPU overhead per constraint decreases with increasing chain length (that is, increasing number of constraints). This makes the method particularly advantageous for long molecules. In principle it also generalizes to other topologies and in Section 4 we elaborate on how one would proceed more generally to treat cyclic, substituted and branched structures.

2 Description of the Algorithm

Let us start with a single molecule. The position of atom i at time t is denoted by $\mathbf{r}_i(t)$. The aim is to constrain bonded pairs of atoms to be a specified distance apart. The constraints can be written in the form

of n equations, where n is the number of bonds (constraints). Defining $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ as the bond vector and l_{ij} as the bond length, they are

$$\sigma_{ij}(\{\mathbf{r}(t)\}) = \mathbf{r}_{ij}^2(t) - l_{ij}^2 = 0. \tag{1}$$

Eq. (1) only holds when the magnitude of the bond vector equals the specified bond length. Using Lagrange's method of undetermined multipliers [33], we can enforce the constraint equations while integrating Newton's laws. After the introduction of a zero potential term, the equation of motion in Cartesian coordinates is

$$m_i \frac{d^2\mathbf{r}_i(t)}{dt^2} = -\frac{\partial}{\partial \mathbf{r}_i}\left[U(\{\mathbf{r}(t)\}) + \sum_p \lambda_{ip}\sigma_{ip}(\{\mathbf{r}(t)\})\right], \tag{2}$$

where the summation is over all atoms, indexed by p, connected to atom i. $U(\{\mathbf{r}(t)\})$ is the potential energy of the system, and m_i is the mass of atom i. The λ_{ip} terms represent the undetermined Lagrange multipliers between the two indexed atoms.

We can write the position after a time step Δt as the sum of the unconstrained positions after applying all forces except constraints ($\{\tilde{\mathbf{r}}\}$) plus the correction due to the constraint forces ($\{\mathbf{F}^C\}$). The updated positions using velocity Verlet [34] are

$$\mathbf{r}_i(t+\Delta t) = \tilde{\mathbf{r}}_i(t+\Delta t) + \frac{\Delta t^2}{2m_i}\mathbf{F}_i^C(t). \tag{3}$$

These forces result from pair constraint forces, each acting along its respective bond vector. For brevity, the position-dependence of the force has not been written explicitly. Following from Eq. (2), a constraint force takes the form

$$\mathbf{F}_i^C(t) = \frac{1}{\Delta t^2}\sum_p \lambda_{ip}\frac{\partial\sigma_{ip}(\{\mathbf{r}(t)\})}{\partial\mathbf{r}_i} = \frac{2}{\Delta t^2}\sum_p \lambda_{ip}\mathbf{r}_{ip}(t). \tag{4}$$

The factors of Δt are absorbed in the multiplier for convenience. Now the updated positions are

$$\mathbf{r}_i(t+\Delta t) = \tilde{\mathbf{r}}_i(t+\Delta t) + \frac{1}{m_i}\sum_p \lambda_{ip}\mathbf{r}_{ip}(t). \tag{5}$$

Substituting Eq. (5) into Eq. (1), we arrive at the system of n equations that we need to solve:

$$\sigma_{ij}(\{\mathbf{r}(t+\Delta t)\}) = \left[\tilde{\mathbf{r}}_{ij}(t+\Delta t) + \frac{1}{m_i}\sum_p \lambda_{ip}\mathbf{r}_{ip}(t) - \frac{1}{m_j}\sum_q \lambda_{jq}\mathbf{r}_{jq}(t)\right]^2 - l_{ij}^2 = 0. \tag{6}$$

The indices p and q cycle through all atoms bonded to atoms i and j, respectively.

2.1 SHAKE

The best known method for solving this set of equations is the algorithm SHAKE [13]. To calculate a solution, we focus on one bond and calculate

$$\lambda_{ij} = \frac{\tilde{\mathbf{r}}_{ij}^2(t+\Delta t) - l_{ij}^2}{2\left(m_i^{-1} + m_j^{-1}\right)\mathbf{r}_{ij}(t)\cdot\tilde{\mathbf{r}}_{ij}(t+\Delta t)}. \tag{7}$$

This is an approximation to the solution for the Lagrange multiplier that satisfies the constraint between atoms i and j. It results from setting all terms of order λ^2 to zero and eliminating the explicit dependence on all other Lagrange multipliers. The two atomic positions are updated according to the following equations.

$$\begin{aligned}\tilde{\mathbf{r}}_i(t+\Delta t) &= \tilde{\mathbf{r}}_i(t+\Delta t) + \frac{1}{m_i}\lambda_{ij}\mathbf{r}_{ij}(t)\\ \tilde{\mathbf{r}}_j(t+\Delta t) &= \tilde{\mathbf{r}}_j(t+\Delta t) - \frac{1}{m_j}\lambda_{ij}\mathbf{r}_{ij}(t)\end{aligned} \tag{8}$$

The next bond is then adjusted in a similar fashion, regardless of whether the preceding constraint has been violated in the process. All of the constraints of the system are cycled through in this manner until each has been approximated once to linear order. The relative error of every bond is then calculated, and if all constraints are satisfied to within a predefined error, the procedure is complete. If not, the bonds are cycled through repeatedly until the solution converges.

2.2 MILC SHAKE

An efficient algorithm for imposing constraints (termed MILC SHAKE) was developed by Bailey *et al.* [22]. This method only applies for simple linear and ring architectures, so it is not applicable for the problem we address here. However, the MILC SHAKE method does form an important part of the hybrid scheme we describe. As such, we begin with a brief recapitulation of the algorithm.

To solve Eq. (6), we use a Newton-like method [23]. The procedure involves first calculating the $n \times n$ Jacobian. Restricting our attention to a linear architecture, it is

$$\begin{aligned}
\mathbf{J}_{i,i-1} &= \frac{-2}{m_i}\ \tilde{\mathbf{r}}_{i,i+1}(t+\Delta t)\cdot\mathbf{r}_{i-1,i}(t),\\
\mathbf{J}_{i,i} &= \frac{2}{\mu_{i,i+1}}\ \tilde{\mathbf{r}}_{i,i+1}(t+\Delta t)\cdot\mathbf{r}_{i,i+1}(t),\\
\mathbf{J}_{i,i+1} &= \frac{-2}{m_{i+1}}\ \tilde{\mathbf{r}}_{i,i+1}(t+\Delta t)\cdot\mathbf{r}_{i+1,i+2}(t),
\end{aligned} \tag{9}$$

where μ_{ij} is the reduced mass of two atoms i and j defined as $\mu_{ij} = m_i m_j / (m_i + m_j)$. The first and last equations, representing the end bonds, only have a diagonal and one off-diagonal component. Explicitly, they are

$$\begin{aligned}
\mathbf{J}_{1,1} &= \frac{2}{\mu_{1,2}}\ \tilde{\mathbf{r}}_{1,2}(t+\Delta t)\cdot\mathbf{r}_{1,2}(t),\\
\mathbf{J}_{1,2} &= \frac{-2}{m_2}\ \tilde{\mathbf{r}}_{1,2}(t+\Delta t)\cdot\mathbf{r}_{2,3}(t),\\
\mathbf{J}_{n,n-1} &= \frac{-2}{m_n}\ \tilde{\mathbf{r}}_{n,n+1}(t+\Delta t)\cdot\mathbf{r}_{n-1,n}(t),\\
\mathbf{J}_{n,n} &= \frac{2}{\mu_{n,n+1}}\ \tilde{\mathbf{r}}_{n,n+1}(t+\Delta t)\cdot\mathbf{r}_{n,n+1}(t).
\end{aligned} \tag{10}$$

All other matrix elements are zero. The Jacobian is then used to solve the system of linear equations

$$\tilde{\sigma}(\{\mathbf{r}\}) = \mathbf{J}\lambda. \tag{11}$$

The term $\tilde{\sigma}(\{\mathbf{r}\})$ is a vector of n constraint residues.

An iterative procedure is required to converge to the correct values of the vector λ. We use a simplified Newton iteration, the Chord Method [20, 35], to do so. On the first iteration, Eq. (11) is solved with the left-hand side equal to the vector of constraint residues calculated from Eq. (1) using the unconstrained positions, $\tilde{\sigma}^0(\{\tilde{\mathbf{r}}(t+\Delta t)\})$. The solution, λ^0, is a first approximation to λ. We apply our current estimation of the constraint forces given λ^0 to the unconstrained positions to arrive at a set of intermediate coordinates $(\{\mathbf{r}^1(t+\Delta t)\})$, where the superscript indexes the iteration. These coordinates are then used to calculate the instantaneous residues $\sigma(\{\mathbf{r}^1(t+\Delta t)\})$, which we call δ^1. The term δ^1 is added to σ^0 to form the sum of the constraint residues leading up to the current iteration, denoted as $\tilde{\sigma}^1$. Eq. (11) is re-solved using $\tilde{\sigma}^1$ as the left-hand side to get λ^1, a more accurate approximation to λ. The method iterates until the intermediate coordinates satisfy all constraints to a pre-defined degree of accuracy. Following the Chord Method, the Jacobian is calculated only once throughout this procedure.

The iterative procedure is therefore

$$\tilde{\sigma}^k = \tilde{\sigma}^{k-1} + \delta^k = \mathbf{J}\ \lambda^k \tag{12}$$

$$\delta^k_{i,i+1} = \mathbf{r}^k_{i,i+1}(t+\Delta t)^2 - l^2_{i,i+1}, \tag{13}$$

where the set of vectors ($\{\ \mathbf{r}^k_{i,i+1}(t+\Delta t)\ \}$) are the bond vectors calculated at iteration k using the intermediate coordinates.

The cost of a simplified Newton iteration is generally order n^3, corresponding to the cost of solving Eq. (12) every iteration. However, the Jacobian of this set of equations – when they are ordered sequentially, matching their relative position along the contour of the chain – is of a very special form: it is tridiagonal. Inverting a tridiagonal matrix can be done easily and efficiently in order n operations [23]. This being the case, the cost of the algorithm is order n.

2.3 MILCH SHAKE

The MILC SHAKE algorithm is orders of magnitude faster than SHAKE for the case of a linear architecture, encountered when, for example, linear alkanes are modeled with a UAM. Nonetheless, one may often wish to simulate an alkane with explicit hydrogen, for reasons stated in the introduction. For the case of n-alkanes, the introduction of explicit hydrogens will lead to a Jacobian of the constraints that contains many non-tridiagonal elements.

However, with an astute observation, one can still calculate the Lagrange multipliers in a computationally efficient manner. The key point is that a *subset* of the constraints in an alkane can be approximated as "linear", in the sense that we can define a contiguous chain in which each atom is connected to at most two other atoms. These constraints, which we term the "backbone", can consequently be treated using MILC SHAKE. The remaining constraints we refer to as "substituent" constraints, which can be treated using SHAKE. The result is a hybrid method alternating between one iteration of MILC SHAKE and one iteration of SHAKE.

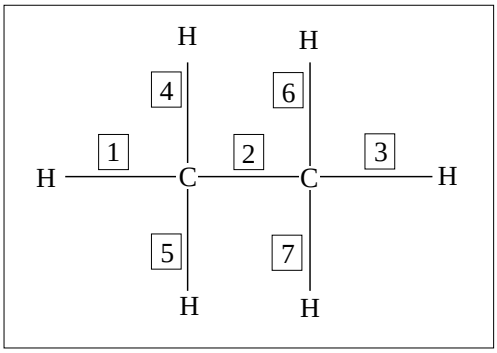


Figure 1: A schematic of ethane, with bonds labeled by the boxed numbers.

Let us look at ethane for illustration. Figure 1 depicts the chemical structure, where bonds (and constraint equations) are labeled by the boxed numbers. The Jacobian of the constraints ordered in sequence are listed below.

$$\mathbf{J} = 2 \left(\begin{array}{ccccccc} \frac{\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_1}{\mu_1} & \frac{-\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_2}{m_c} & 0 & \frac{-\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_4}{m_c} & \frac{-\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_5}{m_c} & 0 & 0 \\ \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_1}{m_c} & \frac{\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_2}{\mu_2} & \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_3}{m_c} & \frac{\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_4}{m_c} & \frac{\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_5}{m_c} & \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_6}{m_c} & \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_7}{m_c} \\ 0 & \frac{-\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_2}{m_c} & \frac{\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_3}{\mu_3} & 0 & 0 & \frac{\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_6}{m_c} & \frac{\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_7}{m_c} \\ \hline \frac{-\tilde{\mathbf{r}}_4 \cdot \mathbf{r}_1}{m_c} & \frac{-\tilde{\mathbf{r}}_4 \cdot \mathbf{r}_2}{m_c} & 0 & \frac{\tilde{\mathbf{r}}_4 \cdot \mathbf{r}_4}{\mu_4} & \frac{\tilde{\mathbf{r}}_4 \cdot \mathbf{r}_5}{m_c} & 0 & 0 \\ \frac{-\tilde{\mathbf{r}}_5 \cdot \mathbf{r}_1}{m_c} & \frac{\tilde{\mathbf{r}}_5 \cdot \mathbf{r}_2}{m_c} & 0 & \frac{\tilde{\mathbf{r}}_5 \cdot \mathbf{r}_4}{m_c} & \frac{\tilde{\mathbf{r}}_5 \cdot \mathbf{r}_5}{\mu_5} & 0 & 0 \\ 0 & \frac{-\tilde{\mathbf{r}}_6 \cdot \mathbf{r}_2}{m_c} & \frac{\tilde{\mathbf{r}}_6 \cdot \mathbf{r}_3}{m_c} & 0 & 0 & \frac{\tilde{\mathbf{r}}_6 \cdot \mathbf{r}_6}{\mu_6} & \frac{\tilde{\mathbf{r}}_6 \cdot \mathbf{r}_7}{m_c} \\ 0 & \frac{-\tilde{\mathbf{r}}_7 \cdot \mathbf{r}_2}{m_c} & \frac{\tilde{\mathbf{r}}_7 \cdot \mathbf{r}_3}{m_c} & 0 & 0 & \frac{\tilde{\mathbf{r}}_7 \cdot \mathbf{r}_6}{m_c} & \frac{\tilde{\mathbf{r}}_7 \cdot \mathbf{r}_7}{\mu_7} \end{array} \right) \tag{14}$$

The quantity m_c is the atomic mass of carbon. In this section, constrained ($\{\mathbf{r}(t)\}$) and unconstrained ($\{\tilde{\mathbf{r}}(t+\Delta t)\}$) bond vectors are labeled with the matching index of the bond from Figure 1. The explicit time dependence has also been omitted for clarity. The reduced mass of the two atomic species associated with constraint i is given by μ_i. The horizontal line has been added to visually separate the constraints along the "backbone" (λ_1 - λ_3) from the constraints corresponding to the side hydrogen bonds (λ_4 - λ_7). The "backbone" from here onward is defined as all carbon-carbon bonds plus a carbon-hydrogen bond at each terminal carbon atom.

Why do we take this definition of the backbone? If we were to follow the procedure for the simplified Newton iteration as we used for MILC SHAKE, the cost of solving Eq. (12) for ethane is order n^3 since the Jacobian of the constraints, Eq. (14), is no longer tridiagonal. However, consider the relative magnitude of the elements of the Jacobian. First, we look at the mass dependence. Conveniently, for the case of ethane, and all hydrogenated alkanes, the mass difference between the backbone carbon atoms and the hydrogen substituents is significant: $\sim$12:1. All of the non-diagonal components of Eq. (14) are proportional to $1/m_c \approx 1/12$. Furthermore, in an sp^3 hybridized structure, the magnitude of $\tilde{\mathbf{r}}_i \cdot \mathbf{r}_j$ where $i \neq j$ is dependent upon the size of the time step, but it will be $\sim 1/3$ for adjacent bonds of a tetrahedral structure with bond angles of 109.5^o. The non-diagonal components are therefore near $\sim 1/36$. The diagonal components, on the other hand, are significantly larger. Given the reduced masses of a carbon-carbon and carbon-hydrogen bond are, respectively, ~ 6 and ~ 1, the corresponding elements of $\mathbf{J}_{i,i}$ are either $\sim 1/6$ or ~ 1. The $\tilde{\mathbf{r}}_i \cdot \mathbf{r}_i$ terms are close to unity. What we have is a diagonally dominant matrix. An approximation to Eq. (14) can be written as

$$\mathbf{J} \approx 2 \left(\begin{array}{ccccccc} \frac{\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_1}{\mu_1} & \frac{-\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_2}{m_c} & 0 & 0 & 0 & 0 & 0 \\ \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_1}{m_c} & \frac{\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_2}{\mu_2} & \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_3}{m_c} & 0 & 0 & 0 & 0 \\ 0 & \frac{-\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_2}{m_c} & \frac{\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_3}{\mu_3} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \frac{\tilde{\mathbf{r}}_4 \cdot \mathbf{r}_4}{\mu_4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\tilde{\mathbf{r}}_5 \cdot \mathbf{r}_5}{\mu_5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\tilde{\mathbf{r}}_6 \cdot \mathbf{r}_6}{\mu_6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\tilde{\mathbf{r}}_7 \cdot \mathbf{r}_7}{\mu_7} \end{array} \right). \tag{15}$$

This suggests a convenient partition of the solution into the backbone constraints and the remaining substituent constraints. Specifically, the tridiagonal component of the Jacobian, containing the backbone constraints, can be solved using MILC SHAKE iteration (Section 2.2), while SHAKE iteration (Section 2.1) can be applied to the remaining carbon-hydrogen bonds.

Our algorithm MILCH SHAKE does exactly this. The procedure entails first calculating both the initial

residue, $\tilde{\sigma}^0$, and the Jacobian of only the backbone constraints, defined for our example as

$$\tilde{\mathbf{J}} = 2 \begin{pmatrix} \frac{\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_1}{\mu_1} & \frac{-\tilde{\mathbf{r}}_1 \cdot \mathbf{r}_2}{m_c} & 0 \\ \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_1}{m_c} & \frac{\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_2}{\mu_2} & \frac{-\tilde{\mathbf{r}}_2 \cdot \mathbf{r}_3}{m_c} \\ 0 & \frac{-\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_2}{m_c} & \frac{\tilde{\mathbf{r}}_3 \cdot \mathbf{r}_3}{\mu_3} \end{pmatrix}. \tag{16}$$

The iterative procedure then begins. First, substituent bonds are cycled through once using SHAKE iteration (Eq. (7)). Next, the backbone constraints are approximated by solving $\tilde{\mathbf{J}}\lambda^k = \tilde{\sigma}^k$, as in the MILC SHAKE scheme (Section 2.2). The current approximation to the positions, $\mathbf{r}^k(t + \Delta t)$, given the estimate of λ^k, is calculated, followed by the instantaneous residue δ^k. If the relative error of every constraint is within a predefined tolerance, the procedure is complete. If it is not, δ^k is added to the cumulative residue to give $\tilde{\sigma}^{k+1}$ and the iteration is repeated.

All explicit dependence of of backbone constraints on the substituent constraints has been approximated as zero to arrive at Eq. (16), but this approximation does not affect the accuracy of the solution of the Lagrange multipliers. This is because an iterative procedure is applied to converge to the solution, one that takes into account the coupling of the constraints through the residue, δ^k. The residue is calculated from a current estimate to the positions that is based on all values of the constraint forces, those from the backbone and the substituents alike. Therefore, the interdependence of the two sets of constraints is maintained through this correction term that is added to the cumulative residue prior to re-solving for the backbone constraints each iteration.

In the example above we have specifically listed the equations for ethane, in the interests of clarity. However, the same principle applies to all n-alkanes. The Jacobian $\tilde{\mathbf{J}}$ has a size of the number of carbons plus one, and it can be calculated using Eq. (9). The algorithm for MILCH SHAKE is summarized in the form of pseudo code below.

Algorithm 1 Pseudo code of MILCH SHAKE, given constraint error τ_{ij} and a predefined tolerance τ_0.

```
: Calculate J̃, σ̃^0
: FOR all molecules
:    LOOP
:       FOR all constraints
:          Calculate τ_ij using {r^k}
:       END FOR
:       IF max(τ_ij) ≤ τ_0
:          SAVE current positions, {r^k}
:          EXIT LOOP
:       END IF
:       FOR all substituent constraints
:          Calculate SHAKE approximation to λ_ij
:          Update r̃_i and r̃_j
:       END FOR
:       Solve J̃λ^k = σ̃^k
:       Calculate the current positions, {r^k}, using λ^k
:       Calculate δ^k({r^k})
:       Add δ^k to σ̃^k to get σ̃^{k+1}
:    END LOOP
: END FOR
```

3 Numerical Tests

To compare the efficiency of MILCH SHAKE compared to SHAKE, a series of simulations was carried out. All calculations were performed on a desktop workstation using an Intel Pentium D processor (3.00 GHz) running Fedora Linux. Code was compiled with the Intel Fortran Compiler 10.1.015 with the optimization flag set to *3*, using the BLAS and LAPACK routines. The SHAKE code used for comparative purposes was written in house.

We define the relative error between atoms i and j as

$$\frac{|r_{ij}(t) - l_{ij}|}{l_{ij}} = \tau_{ij}. \tag{17}$$

The predefined value of τ_0, also referred to here as the *accuracy*, is the maximum allowable value of τ_{ij} for all i, j. The accuracy of the constraint forces in constrained systems is important because the degree of permissible violation of the constraints given by τ_0 determines the energy drift of the system [11]. We have empirically found that the maximum tolerable error is approximately 10^{-8} for a dynamic system of a filament evolving under the influence of an energy penalty for bending, in order for energy to be conserved to within one part in 10^{-4}. This may serve as a guideline.

Bond lengths were taken to be 1.54 $\mathring{A}$ for a carbon-carbon bond and 1.10 $\mathring{A}$ for a carbon-hydrogen bond [36]. The system was initialized in the lowest energy, zig-zag conformation. A realistic, relative atomic mass of 12:1 for carbon to hydrogen was specified. The starting geometry, for which the constraints were satisfied, was then perturbed. We did this by taking a velocity from the Maxwellian for the velocities of the atoms and calculating the positions a time Δt later. This realistically accounts for the larger displacement of the hydrogens due to their smaller mass. The magnitude of the time step was chosen such that the error in the position constraints prior to applying constraint forces was approximately 0.1%. This is comparable to the starting error typically encountered in a molecular dynamics simulation.

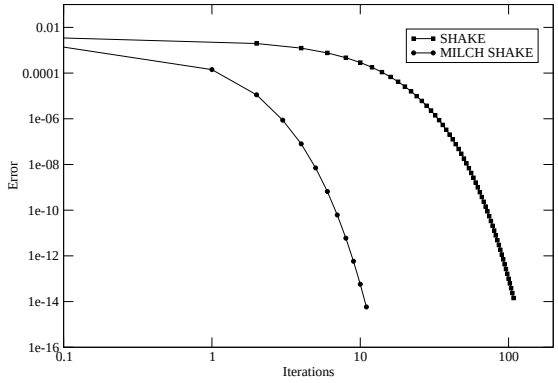


Figure 2: Relative error as a function of the number of iterations to calculate the position constraints using SHAKE and MILCH SHAKE for hexane.

We first determine how the number of iterations scale with system accuracy throughout the course of the constraint force calculation. Results for hexane are displayed and are qualitatively representative of all alkane sizes investigated. Figure 2 shows the relative error as a function of the number of iterations required to calculate the position constraints using SHAKE and MILCH SHAKE. Nearly ten times the number of iterations are needed when using SHAKE to achieve an acceptable error of less than 10^{-8}. This would be inconsequential if the CPU time per iteration were significantly different. However, this is not the case. Approximately the same computational overhead is required for a single iteration using either method. Therefore, the relative number of required iterations is indicative of the relative computational cost. This trend is exhibited in Figure 3, a plot of the CPU time as a function of system accuracy. MILCH SHAKE is

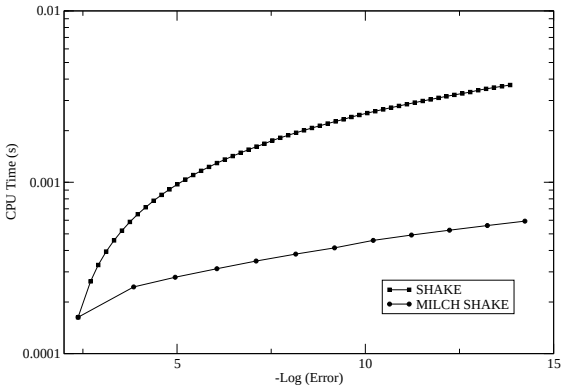


Figure 3: CPU time as a function of relative error required to calculate the position constraints using SHAKE and MILCH SHAKE for hexane.

nearly an order of magnitude more efficient than SHAKE in the error range that is acceptable for MD for hexane.

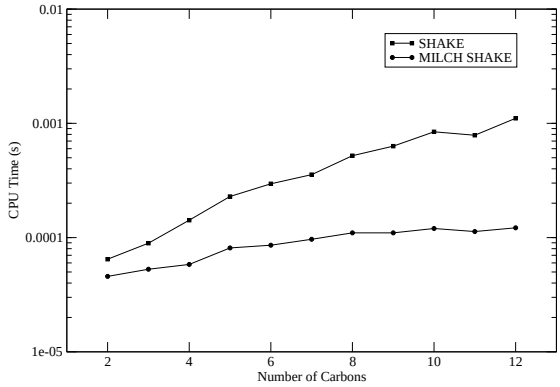


Figure 4: Total CPU time required to calculate the position constraints to an accuracy of 10^{-14} using SHAKE and MILCH SHAKE as a function of chain length.

We next investigate how these results differ with chain length. Alkanes ranging from two to twelve carbons were analyzed. For each size, the CPU time required to achieve an error of 10^{-14} was calculated and then averaged over 100 independent random violations. Results for the total CPU time are shown in Figure 4. MILCH SHAKE is a factor of three more efficient for the smallest alkane, ethane. However, as the chain length increases, the improvement becomes more significant. MILCH SHAKE is an order of magnitude more efficient than SHAKE for alkanes larger than octane.

One can observe from Figure 4 that the slope of SHAKE is greater than that of MILCH SHAKE, meaning that the CPU time per constraint is changing as a function of chain length. Figure 5 explicitly plots the expense normalized by the number of constraints. The overhead per constraint using MILCH SHAKE is actually reduced as the backbone of the chain increases. This can be explained by recognizing that the average number of iterations required for convergence is roughly the same (11-13 iterations) for every chain length investigated. The decrease in required CPU time per constraint is a reflection of the initial time investment leading to the calculation of the Jacobian. The time it takes to calculate $\mathbf{J}$ becomes comparable to the time of the remaining iterative procedure. In general one can explain the reverse trend of SHAKE by noting that with increasing n, the neglect of coupling becomes increasingly significant with greater system

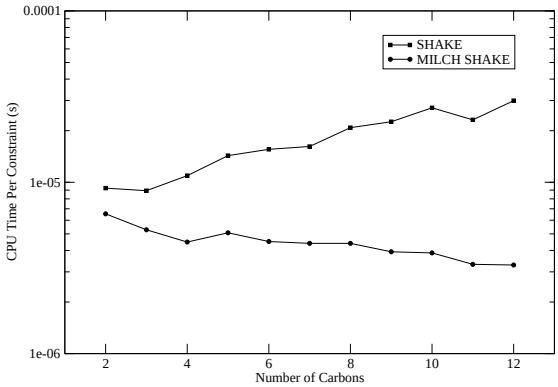


Figure 5: CPU time per constraint required to calculate the position constraints to an accuracy of 10^{-14} using SHAKE and MILCH SHAKE as a function of chain length.

size when there are more degrees of freedom to coordinate.

4 Discussion and Conclusions

We have developed a computational technique for constraining non-linear architectures and tested it for hydrogenated n-alkanes. It is over an order of magnitude more efficient than SHAKE for large molecules, and a significant factor more efficient for ethane, the smallest molecule investigated. A pseudo code implementing MILCH SHAKE is included as an appendix. Code in FORTRAN 90 is also available for download [37]. The results suggest that the convergence of MILCH SHAKE is linear, the same as SHAKE. A possible improvement would be to use a more complex iterative procedure. This would involve the overhead of recalculating the Jacobian for each iteration but might improve the convergence and reduce the computational time required. However, because the linearly convergent SHAKE iteration is one component of MILCH SHAKE we believe that the actual order of convergence would not change. Nonetheless, our results demonstrate that, compared to SHAKE, MILCH SHAKE is advantageous for all alkanes, particularly when high accuracy is required. With MILCH SHAKE it is practical to reach machine accuracy at a modest computational cost. In practice this would minimize any problem with energy drift as a result of constraint violation.

As noted above, MILCH SHAKE significantly reduces the computational overhead of constraint calculations for alkanes. Given the significant reduction in computational expense, it is possible that one could reconsider using constraints where it was considered too computationally expensive before. Using our method, the CPU time per constraint actually decreases with increasing chain length, making the method even more advantageous for large systems. Strictly, we can only reach this conclusion for the test case we have studied here. That is, random violations of the constraints. We do not expect that the correlation in the violation of the constraints that occurs in a full dynamic simulation will influence this conclusion (it does not for MILC SHAKE [22]). Nonetheless, this needs confirming by implementing and testing the algorithm in such simulations.

The matrix inversion methods M-SHAKE and P-SHAKE are faster than SHAKE for a relatively small number of constraints. For larger molecules, and consequently more constraints, they are increasingly less efficient because the computational time required per iteration increases at a faster rate with increasing number of constraints than it does for SHAKE (as n^3 and n^2, respectively). For the problem of alkanes, that we consider here, we can estimate the point where MILCH SHAKE outperforms these methods. For a comparable accuracy, P-SHAKE requires approximately one-third the CPU time required by SHAKE for ethane. For MILCH SHAKE it is approximately half the time. So for ethane P-SHAKE, but not M-SHAKE, is probably more efficient than our method. Based on the scaling with the number of constraints,

we estimate that for propane M-SHAKE will still outperform SHAKE, requiring a factor of approximately two thirds the CPU time. However, for propane MILCH SHAKE requires half the CPU time. Based on this estimate, MILCH SHAKE is more efficient for propane and higher alkanes. We should point out that there are also other methods for solving the problem as well as matrix inversion and SHAKE. Notably, the LINCS algorithm [38] solves for the multipliers by using a series expansion to approximate the inverse of the Jacobian. For moderate accuracy it is reported to be up to four times faster than SHAKE. Possibly, replacing the SHAKE part of the MILCH SHAKE algorithm with LINCS will result in further improvement of the scheme.

One may also wish to constrain the relative velocities along the bond vectors, following in the footsteps of RATTLE [15]. The same method to constrain the positions described above can be implemented to constrain the velocities, which will certainly be faster than the iterative procedure described in RATTLE. Alternatively, when arranged in a smart way, the Jacobian for the velocity constraints is a band diagonal matrix with three super- and three sub-diagonal entries at its widest point. Algorithms to invert band diagonal systems can be applied to efficiently solve this system of equations, without having to revert to full matrix methods such as LU-factorization [23]. Furthermore, the velocity constraint equations are linear, so one band-diagonal matrix inversion solves for the Lagrange multipliers exactly eliminating the need for iteration [22].

Finally, it is important to note that although here we focus on alkanes, the methodology also applies to other molecules of interest. For instance, using the same procedure to calculate the constraint forces of the simplest alkene, ethene, we find a factor of three improvement using MILCH SHAKE. The results we report are restricted to n-alkanes, but cyclic alkanes are a trivial extension. All one has to do is replace the tridiagonal solver with a cyclic tridiagonal matrix solver [23] or manipulate the matrix to tridiagonal form [22]. The latter is easily done in order n operations. Although we have exploited the relative mass of the two constituents of alkanes, carbon and hydrogen, to approximate the non-tridiagonal terms as zero (Eq. (9)), this condition is not necessary. We tested the algorithm with the mass of hydrogen set equal to the mass of carbon and MILCH SHAKE was still significantly faster than SHAKE (although to a lesser extent than with the correct masses). If the molecule is predominantly hydrogenated but substituted with one heavier species, such as a halogen, SHAKE can be used to constrain the carbon-halogen bond and the specific coefficients in the Jacobian dependent upon this bond can be recalculated every iteration using the new values of $\tilde{\mathbf{r}}(t + \Delta t)$ for the two atoms involved in the constraint. Only a few elements will require re-calculation, but it will be an additional computational expense. The resulting algorithm will still be faster than SHAKE if the molecule is predominantly hydrogenated. Only in the extreme case where most substituents are heavy, is one effectively recalculating the Jacobian every iteration. Then it is no longer a simplified Newton iteration but rather Newton's Method, with order n^3 cost and quadratic convergence [23]. We should also point out that the atomic masses only need to take realistic values if one is interested in the dynamic properties of the system. For static properties the atomic mass can be chosen freely to optimize the performance of the algorithm [39, 40]. At a higher level of complexity, bifurcations in the backbone can be treated by the incipient stages of Gaussian elimination to tri-diagonalize the matrix [22]. However, the relative efficiency of the solution is dependent upon the complexity of the molecule. In the interest of clarity we leave it beyond the scope of this article. Here we simply conclude that for n-alkanes MILCH SHAKE is a significant improvement on SHAKE and that, possibly, extensions of the approach will be efficient for other classes of problem.

A.G.B thanks the Thouron Award and the National Science Foundation Graduate Research Fellowship Program for funding. She also thanks A.P.S. for agreeing to let her go on a sabbatical before completing her Ph.D.

References

[1] Evans, D.J.; Murad, S. *Mol Phys* 1977, **34**, 327.

[2] Ciccotti, G.; Ferrario, M.; Ryckaert, J.-P. *Mol Phys* 1982, **47**, 1253.

[3] Dullweber, A.; Leimkuhler, B.; McLachlan, R. *J Chem Phys* 1997, **107**, 5840.

[4] Tuckerman, M.; Berne, B.J.; Martyna, G.J. *J Chem Phys* 1992, **97**, 1990.

[5] Schlick, T.; Mandziuk, M.; Skeel, R.D.; Srinivas, K. *J Comput Phys* 1998, **140**, 1.

[6] Ma, Q.; Izaguirre, J.A.; Skeel R.D. *SIAM J Sci Comput* 2003, **24**, 1951.

[7] Sandu, A.; Schlick, T. *J Comput Phys* 2003, **151**, R45.

[8] Chin, S. *J Chem Phys* 2004, **120**, 8.

[9] Abrams, J.B.; Tuckerman, M.E.; Martyna, G.J. *Lect Notes Phys* 2006, **703**, 139.

[10] Frenkel, D.; Smit, B. *Understanding Molecular Simulation*; Academic Press: London, UK, 1996.

[11] Kräutler, V.; van Gunsteren, W.F.; Hünenberger, P.H. *J Comput Chem* 2001, **22**, 501.

[12] Zhu, Z.; Schuster, D.I.; Tuckerman, M. *Biochemistry-US* 2003, **42**, 1326.

[13] Ryckaert, J.-P.; Ciccotti, G.; Berendsen, H.J.C. *J Comput Phys* 1977, **23**, 327.

[14] de Leeuw, S.W.; Perram, J.W.; Petersen, H.G. *J Stat Phys* 1990, **61**, 1203.

[15] Andersen, H.C. *J Comput Phys* 1983, **52**, 24.

[16] Lee, S.-H.; Palmo, K.; Krimm, S. *J Comp Phys* 2005, **210**, 171.

[17] Forester, T.R.; Smith, W. *J Comp Chem* 1998, **19**, 102.

[18] Weinbach, Y.; Elber, R. *J Comput Phys* 2005, **209**, 193.

[19] Miyamoto, S.; Kollman, P.A. *J Comput Chem* 1992, **13**, 952.

[20] Gonnet, P. *J Comput Phys* 2007, **220**, 740.

[21] Tapia-McClung, H.; Gronbech-Jensen, N. *J Polym Sci Pol Phys* 2005, **43**, 911.

[22] Bailey, A.G.; Lowe, C.P.; Sutton, A.P. *J Comput Phys* 2008, **227**, 8949.

[23] Press, W.H.; Teukolsky, S.A.; Vetterling, W.T.; Flannery, B.P. *Numerical recipes in FORTRAN (2nd ed.): the art of scientific computing*; Cambridge University Press: New York, NY, 1992.

[24] Siepmann, J.I.; Karaborni, S.; Smit, B. *Nature* 1993, **365**, 330.

[25] Lowe, C.P. *Philos T Roy Soc B* 2002, **358** 1543.

[26] Lagomarsino, M.C.; Pagonabarraga, I.; Lowe, C.P. *Phys Rev Lett* 2005, **94**, 148104.

[27] Ryckaert, J.-P.; Klein, M.L. *J Chem Phys* 1986, **85**, 1613.

[28] Ryckaert, J.-P.; Klein, M.L.; McDonald, I.R. *Phys Rev Lett* 1987, **58**, 698.

[29] Moller, M.A.; Tildesley, D.J.; Kim, K.S.; Quirke, N. *J Chem Phys* 1991, **94**, 8390.

[30] Connolly, M.J.; Roth, M.W.; Gray, P.A.; Wexler, C. *Langmuir* 2008, **24**, 3228.

[31] Chen, B; Siepmann, J.I. *J Phys Chem* 1999, **103**, 5370.

[32] Toxvaerd, S. *J Chem Phys* 1990, **93**, 4290.

[33] Goldstein, H. *Classical Mechanics (2nd Edition)*; Addison-Wesley: Reading, MA, 1980.

[34] Swope, W.C.; Andersen, H.C.; Berens, P.H.; Wilson, K.R. *J Chem Phys* 1982, **76**, 6548.

[35] Gill, P.E.; Murray, W.; Wright, M.H. *Numerical linear algebra and optimization*; Addison-Wesley, Redwood City, California, 1991.

[36] Weast, R.C.; Beyer, W.H. *Handbook of Chemistry & Physics (65th Edition)*; Chemical Rubber Company Press, 1984.

[37] [http://www.cmth.ph.ic.ac.uk/people/aimee.bailey/milchshake.html] Demonstrative MILCH SHAKE code available for download. If the web page no longer exists at the time of attempted access, please contact the authors for code. (Uploaded 07 04 2008)

[38] Hess, B; Bekker, H.; Berendsen, H.J.C.; Fraaije, J.G.E.M. *J Comput Chem* 1997, **18**, 1463.

[39] Bennet, C.H. *J Comp Phys* 1975, **19**, 267.

[40] Feenstra, K.A.; Hess, B.; Berendsen, H.J.C. *J Comput Chem* 1999, **20**, 786.

Figure Captions

1. Figure 1. A schematic of ethane, with bonds labeled by the boxed numbers.
2. Figure 2. Relative error as a function of the number of iterations to calculate the position constraints using SHAKE and MILCH SHAKE for hexane.
3. Figure 3. CPU time as a function of relative error required to calculate the position constraints using SHAKE and MILCH SHAKE for hexane.
4. Figure 4. Total CPU time required to calculate the position constraints to an accuracy of 10^{-14} using SHAKE and MILCH SHAKE as a function of chain length.
5. Figure 5. CPU time per constraint required to calculate the position constraints to an accuracy of 10^{-14} using SHAKE and MILCH SHAKE as a function of chain length.