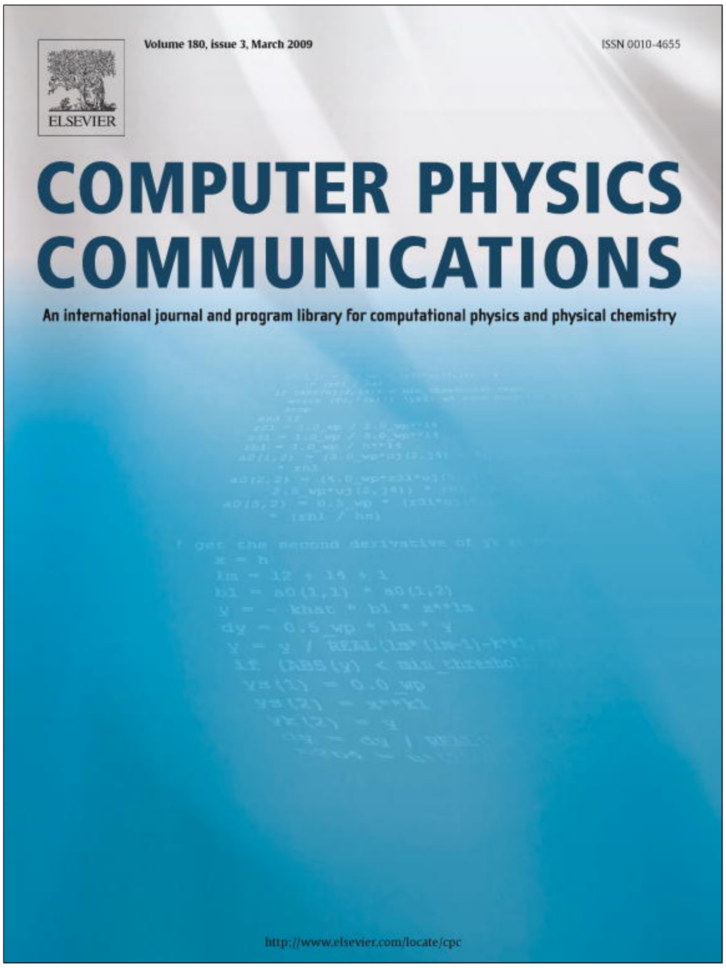

Volume 180, issue 3, March 2009

ISSN 0010-4655

ELSEVIER

COMPUTER PHYSICS COMMUNICATIONS

An international journal and program library for computational physics and physical chemistry

http://www.elsevier.com/locate/cpc

Computer Physics Communications 180 (2009) 360–364

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

ϑ-SHAKE: An extension to SHAKE for the explicit treatment of angular constraints

Pedro Gonnet [a], Jens H. Walther [b,c], Petros Koumoutsakos [b,*]

[a] *Department of Computer Science, ETH Zürich, 8092 Zürich, Switzerland*
[b] *Chair of Computational Science, ETH Zürich, 8092 Zürich, Switzerland*
[c] *Department of Mechanical Engineering, Fluid Section, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark*

ARTICLE INFO

Article history:
Received 19 June 2008
Received in revised form 12 September 2008
Accepted 21 October 2008
Available online 14 November 2008

PACS:
02.70.Ns
02.60.Cb
31.15.xv

Keywords:
Molecular dynamics
Simulations
Constraints
Angular constraints
SHAKE
M-SHAKE
P-SHAKE
Lagrange multipliers

ABSTRACT

This paper presents ϑ-SHAKE, an extension to SHAKE, an algorithm for the resolution of holonomic constraints in molecular dynamics simulations, which allows for the explicit treatment of angular constraints. We show that this treatment is more efficient than the use of fictitious bonds, significantly reducing the overlap between the individual constraints and thus accelerating convergence. The new algorithm is compared with SHAKE, M-SHAKE, the matrix-based approach described by Ciccotti and Ryckaert and P-SHAKE for rigid water and octane.

1. Introduction

Molecular Dynamics (MD) simulations are restricted by the small time steps required for the numerical integration of the fast motions of the molecules in the system. In polyatomic molecules, these fast motions are often associated with intramolecular vibrations that usually occur on a much smaller time scale than the rotational and translational motion of the molecule. In cases where the internal degrees of freedom can be neglected, constraining the fast internal motions (*i.e.* internal bonds and angles) allows an increase of the simulation time step, thus making simulations more efficient.

The numerical integration of the MD equations of motion subject to non-linear constraints was formulated using Lagrange multipliers by Ryckaert et al. [1] who introduced the well known SHAKE algorithm. In this algorithm the intramolecular atomic coordinates are iteratively and independently adjusted until all constraints are satisfied up to a prescribed tolerance. This iterative solution of the constraint equations does not take into account the coupling between individual constraints and the algorithm converges slowly for molecules with highly interconnected constraints, such as rigid molecules or molecules with rigid sub-groups. In [2], Ciccotti and Ryckaert presented a matrix-based approach (referred to in the following as Ciccotti's Matrix Method, CMM), where the constraints are solved for simultaneously as a system of non-linear equations using Newton's Method. The method was extended to account for geometric constraints suitable for handling rigid molecules with many atoms [3,4]. This matrix-based approach was later presented in different variants in [5], LINCS [6] and M-SHAKE [7]. These methods converge faster, yet at a significantly higher cost per iteration.

Further efforts include SETTLE [8], which solves the constraint equations for water or any other three-body rigid molecule analytically, and RATTLE [9], MSHAKE [10] and WIGGLE [11], which solve the constraint equations using derived constraint velocities and forces, as opposed to positions in SHAKE. Finally, in P-SHAKE [12], the constraints are linearly re-combined so that they are effectively decoupled, allowing them to be solved efficiently using the origi-

[*] Corresponding author.
E-mail addresses: gonnetp@inf.ethz.ch (P. Gonnet), walther@inf.ethz.ch (J.H. Walther), petros@ethz.ch (P. Koumoutsakos).

0010-4655/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.cpc.2008.10.020

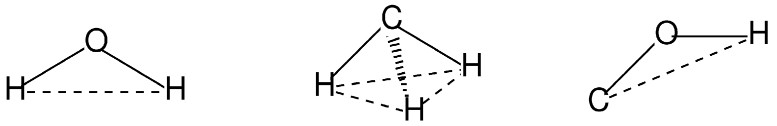

Fig. 1. Fictitious-bonds in rigid H_2O, CH_3 and –COH, shown as dashed lines.

nal SHAKE algorithm. This approach, however, is only applicable to rigid or semi-rigid molecules.

We note that in all these methods, the constraints are formulated as distance constraints, *i.e.* the distance between two atoms is kept at a prescribed value. Therefore, replacing tight, high-frequency bonds such as C–H or O–H is straight forward. However, once these bonds have been replaced, tight, high-frequency angular motions, *e.g.*, in H_2O, CH_3 or –COH groups, quickly become the source of the highest frequencies in the system. These angles are usually constrained using *fictitious bonds*, as shown in Fig. 1.

For methods that solve the constraint equations simultaneously (*e.g.*, M-SHAKE) or that de-couple the constraints (*e.g.*, P-SHAKE), the addition of fictitious bonds does not cause any problems. However, for the faster algorithms such as SHAKE, which treat the constraints as if they were decoupled, the increase in connectivity caused by the addition of a fictitious bond drastically reduces their convergence rate. The resulting increase in the number of iterations practically eliminates their advantage over the slower, simultaneous solvers.

In this paper we present ϑ-SHAKE, an extension to SHAKE in which angular constraints are treated explicitly. This special treatment, by reducing the connectivity of the system, allows for faster convergence through less iterations, yet at the same cost per iteration.

The paper is organized as follows: In Section 2 we outline the SHAKE algorithm, and in Section 3 we present the proposed ϑ-SHAKE algorithm. In Section 4, we present simulations of water and octane using ϑ-SHAKE and compare its efficiency with the SHAKE, M-SHAKE, CMM and P-SHAKE algorithms.

2. The SHAKE algorithm revisited

The n_c linear distance constraints within a molecule can be written in the form of n_c constraint equations

$$\sigma_l^{(t)} = \|\mathbf{x}_{l\alpha}^{(t)} - \mathbf{x}_{l\beta}^{(t)}\|_2^2 - d_l^2 = 0, \quad l = 1 \ldots n_c, \tag{1}$$

where $\mathbf{x}_{l\alpha}$ and $\mathbf{x}_{l\beta}$ are the positions of the two particles involved in the constraint σ_l at time t and d_l is the prescribed inter-atomic distance. In the following we will always assume the 2-norm when the notation $\|\cdot\|$ is used.

The constraints can be enforced while integrating the equations of motion by using Lagrange's method of undetermined multipliers [1]. Using the velocity Verlet algorithm for time integration, this is equivalent to finding the Lagrange multipliers λ_k, $k = 1 \ldots n_c$, for each time step, such that the constrained particle positions

$$\mathbf{x}_i^{(t+\Delta t)} = \tilde{\mathbf{x}}_i^{(t+\Delta t)} + \sum_{k=1}^{n_c} \lambda_k \mathbf{v}_{i,k}, \tag{2}$$

where $\mathbf{v}_{i,k} = \frac{\partial \sigma_k^{(t)}}{\partial \mathbf{x}_i^{(t)}} (\Delta t)^2 m_i^{-1}$ is the constraint gradient of σ_k on $\mathbf{x}_i$, $\tilde{\mathbf{x}}_i^{(t+\Delta t)}$ is the unconstrained particle position at time $t + \Delta t$ and m_i the mass of the particle i, satisfy all of the n_c constraints.

Inserting the constrained positions into Eq. (1), we obtain

$$\sigma_l^{(t)} = \left\| \tilde{\mathbf{x}}_{l\alpha}^{(t)} + \sum_{k=1}^{n_c} \lambda_k \mathbf{v}_{l\alpha,k} - \tilde{\mathbf{x}}_{l\beta}^{(t)} - \sum_{k=1}^{n_c} \lambda_k \mathbf{v}_{l\beta,k} \right\|^2 - d_l^2. \tag{3}$$

```
 1. for $i = 1 \ldots n_c$ do
 2.   $\mathbf{v}_{i\alpha,i} \leftarrow \frac{\partial \sigma_i^{(t)}}{\partial \mathbf{x}_{i\alpha}} (\Delta t)^2 m_{i\alpha}^{-1}$
 3.   $\mathbf{v}_{i\beta,i} \leftarrow \frac{\partial \sigma_i^{(t)}}{\partial \mathbf{x}_{i\beta}} (\Delta t)^2 m_{i\beta}^{-1}$
 4. end for
 5. repeat
 6.   for $i = 1 \ldots n_c$ do
 7.     $\lambda_i \leftarrow -\frac{\sigma_i}{\partial \sigma_i / \partial \lambda_i}\Big|_{\lambda_k = 0, k = 1 \ldots n_c}$
 8.     $\mathbf{x}_{i\alpha} \leftarrow \mathbf{x}_{i\alpha} + \lambda_i \mathbf{v}_{i\alpha,i}$
 9.     $\mathbf{x}_{i\beta} \leftarrow \mathbf{x}_{i\beta} + \lambda_i \mathbf{v}_{i\beta,i}$
10.   end for
11. until $\max_i |\sigma_i| \leqslant \tau$
```

Algorithm 1. SHAKE.

The constrained positions can then be computed by solving the system of n_c non-linear constraint equations $\sigma_k = 0$ for the n_c Lagrange multipliers λ_k.

The most straightforward way of solving any system of non-linear equations is Newton's Method or any Quasi-Newton Method. The solutions are computed iteratively using

$$\underline{\lambda}^{(n+1)} \leftarrow \underline{\lambda}^{(n)} - (\mathbf{J}_\sigma)^{-1} \underline{\sigma}^{(n)}, \tag{4}$$

where $\underline{\lambda}$ is the vector containing the n_c Lagrange multipliers λ_k and $\underline{\sigma}$ the vector of the n_c constraint residuals computed as in Eq. (3) with the current values of the λ_k. $\mathbf{J}_\sigma$ is the Jacobian of the equations σ_l:

$$\mathbf{J}_\sigma = \begin{pmatrix} \frac{\partial \sigma_1}{\partial \lambda_1} & \frac{\partial \sigma_1}{\partial \lambda_2} & \cdots & \frac{\partial \sigma_1}{\partial \lambda_{n_c}} \\ \frac{\partial \sigma_2}{\partial \lambda_1} & \frac{\partial \sigma_2}{\partial \lambda_2} & \cdots & \frac{\partial \sigma_2}{\partial \lambda_{n_c}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \sigma_{n_c}}{\partial \lambda_1} & \frac{\partial \sigma_{n_c}}{\partial \lambda_2} & \cdots & \frac{\partial \sigma_{n_c}}{\partial \lambda_{n_c}} \end{pmatrix}. \tag{5}$$

This iteration can be simplified if, in every iteration, the particle positions $\mathbf{x}_i$ are updated according to Eq. (2). The λ_k can then be reset to zero and the derivatives in $\mathbf{J}_\sigma$ are greatly simplified. In every iteration we then only need to solve

$$\mathbf{J}_\sigma \underline{\lambda} = -\underline{\sigma} \tag{6}$$

for the λ_k.

In Ref. [1], Eq. (6) is approximated using the Gauss–Seidel method, which considers only the diagonal elements of $\mathbf{J}_\sigma$. This approach, which is the well-known SHAKE algorithm, is equivalent to treating the constraints as if they were independent and solving the kth constraint equation for λ_k using Newton's Method:

$$\lambda_l = -\frac{\sigma_l}{\partial \sigma_l / \partial \lambda_l}\Bigg|_{\lambda_k = 0, k = 1 \ldots n_c}. \tag{7}$$

This straightforward approach, described in Algorithm 1, requires only $\mathcal{O}(n_c)$ operations per iteration, yet converges only linearly.

Using the constraints in Eq. (3), the assignment in line 7 reduces to

$$\lambda_i \leftarrow \frac{\|\mathbf{x}_{i\alpha} - \mathbf{x}_{i\beta}\|^2 - d_i^2}{2(\mathbf{x}_{i\alpha} - \mathbf{x}_{i\beta}) \cdot (\mathbf{v}_{i\alpha,i} - \mathbf{v}_{i\beta,i})} \tag{8}$$

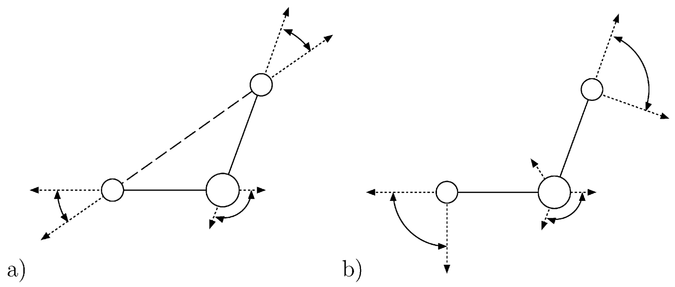

Fig. 2. The magnitude with which one constraint interferes with another is given by the cosine of the angle between the gradients of the two constraints over the common particle. For the case of the water molecule shown, this is quite large over the hydrogen atoms when a fictitious bond is used for the H–O–H angle (a). When angular constraints are used, the gradients are orthogonal over the hydrogen atoms (b).

and the assignments in lines 2 and 3 become

$$\mathbf{v}_{i\alpha,i} \leftarrow 2(\mathbf{x}_{i\alpha} - \mathbf{x}_{i\beta})(\Delta t)^2 m_{i\alpha}^{-1}, \tag{9}$$

$$\mathbf{v}_{i\beta,i} \leftarrow 2(\mathbf{x}_{i\beta} - \mathbf{x}_{i\alpha})(\Delta t)^2 m_{i\beta}^{-1}. \tag{10}$$

For independent constraints (*i.e.* all the off-diagonal elements of $\mathbf{J}_\sigma$ are zero) SHAKE converges quadratically. If, however, the constraints are coupled (*i.e.* two constraints share a common particle) the Jacobian will contain off-diagonal elements and convergence will no longer be quadratic but linear. If the off-diagonal elements are too large, the Gauss–Seidel iteration may even fail to converge. The magnitude of this interference is given by the cosine of the angle between the gradients of the two constraints over the common particle (Fig. 2a).

The off-diagonal elements can be eliminated by linearly recombining the constraint gradients, thus restoring quadratic convergence, as is done in P-SHAKE [12]. This diagonalization, however, is a computationally expensive operation and is advantageous only for rigid molecules, where the linear recombination is invariant so that it can be pre-computed off-line and simply applied in each iteration. The resulting algorithm converges quadratically at a cost of only $\mathcal{O}(n_c^2)$ operations per iteration.

In M-SHAKE [7], Eq. (6) is solved using an LU-decomposition of the matrix $\mathbf{J}_\sigma$ at a cost of $\mathcal{O}(n_c^3)$ operations per iteration. In LINCS [6] the solution of Eq. (6) is approximated by iteratively estimating the inverse of $\mathbf{J}_\sigma$ in $\mathcal{O}(n_c^2 m)$ operations, where m is the order of the approximation. M-SHAKE converges quadratically and LINCS converges quadratically for sufficiently large m.

An alternative way of solving Eq. (6) is given in Ref. [2], where the inverse of the Jacobian is computed once per time-step and re-used in each iteration. This is otherwise known as the *simplified* Newton iteration. This reduces the cost of each iteration to $\mathcal{O}(n_c^2)$ operations with an initial cost of $\mathcal{O}(n_c^3)$. Convergence, however, is no longer quadratic but linear. In [5], several other variants are proposed such as using a symmetrized Jacobian or re-using the same inverted Jacobian over several time-steps until convergence falls below a given threshold.

3. The ϑ-SHAKE algorithm

In molecules with angular constraints—*e.g.*, rigid water in which the H–O–H angle is constrained—fictitious bonds over the particles at the angle ends are used—*i.e.* an H–H fictitious bond for rigid water. These fictitious bonds, however practical in their formulation, result in large overlaps between the constraint gradients, thus slowing the convergence of algorithms such as SHAKE. In order to avoid such overlaps, we can re-define the constraint equation for angular constraints as

$$\sigma_l^\angle = \cos^2 \vartheta_{l\alpha,l\beta,l\gamma} - \cos^2 \vartheta_l = 0, \tag{11}$$

where $\vartheta_{l\alpha,l\beta,l\gamma}$ is the angle spanning the particles $\mathbf{x}_{l\alpha}$, $\mathbf{x}_{l\beta}$ and $\mathbf{x}_{l\gamma}$ and ϑ_l is the prescribed constraint angle.

Using the cosine rule, we can re-write Eq. (11) as

$$\sigma_l^\angle = \frac{[(\mathbf{x}_{l\alpha} - \mathbf{x}_{l\beta}) \cdot (\mathbf{x}_{l\gamma} - \mathbf{x}_{l\beta})]^2}{\|\mathbf{x}_{l\alpha} - \mathbf{x}_{l\beta}\|^2 \|\mathbf{x}_{l\gamma} - \mathbf{x}_{l\beta}\|^2} - \cos^2 \vartheta_l = 0. \tag{12}$$

Given its gradients with respect to the particles $\mathbf{x}_{l\alpha}$, $\mathbf{x}_{l\beta}$ and $\mathbf{x}_{l\gamma}$, this equation can be used analogously to the distance constraints within SHAKE. These gradients are more complex than those of the linear constraint equations, yet they can be pre-computed outside of the iteration loop.

The gradients of $\sigma_l^\angle$ in the particles $\mathbf{x}_{l\alpha}$, $\mathbf{x}_{l\beta}$ and $\mathbf{x}_{l\gamma}$ are

$$\frac{\partial \sigma^\angle}{\partial \mathbf{x}_\alpha} = 2(\mathbf{x}_{\alpha-\beta} \cdot \mathbf{x}_{\gamma-\beta}) \frac{\mathbf{x}_{\gamma-\beta}\|\mathbf{x}_{\alpha-\beta}\|^2 - \mathbf{x}_{\alpha-\beta}(\mathbf{x}_{\alpha-\beta} \cdot \mathbf{x}_{\gamma-\beta})}{\|\mathbf{x}_{\alpha-\beta}\|^4 \|\mathbf{x}_{\gamma-\beta}\|^2}, \tag{13}$$

$$\frac{\partial \sigma^\angle}{\partial \mathbf{x}_\gamma} = 2(\mathbf{x}_{\gamma-\beta} \cdot \mathbf{x}_{\alpha-\beta}) \frac{\mathbf{x}_{\alpha-\beta}\|\mathbf{x}_{\gamma-\beta}\|^2 - \mathbf{x}_{\gamma-\beta}(\mathbf{x}_{\gamma-\beta} \cdot \mathbf{x}_{\alpha-\beta})}{\|\mathbf{x}_{\gamma-\beta}\|^4 \|\mathbf{x}_{\alpha-\beta}\|^2}, \tag{14}$$

$$\frac{\partial \sigma^\angle}{\partial \mathbf{x}_\beta} = -\left(\frac{\partial \sigma^\angle}{\partial \mathbf{x}_\alpha} + \frac{\partial \sigma^\angle}{\partial \mathbf{x}_\gamma}\right), \tag{15}$$

where $\mathbf{x}_{\alpha-\beta} = \mathbf{x}_\alpha - \mathbf{x}_\beta$ and $\mathbf{x}_{\gamma-\beta} = \mathbf{x}_\gamma - \mathbf{x}_\beta$. Note that the sum of the resulting forces and moments on the particles is zero, meaning that neither the center of mass of the particles nor their rotational energy is affected, regardless of the Lagrange multiplier applied.

Note also that for all practical purposes the common factor

$$\frac{2(\mathbf{x}_{\alpha-\beta} \cdot \mathbf{x}_{\gamma-\beta})}{\|\mathbf{x}_{\alpha-\beta}\|^2 \|\mathbf{x}_{\gamma-\beta}\|^2} \tag{16}$$

in the constraint gradients can be eliminated and accounted for in the resulting λ_k, thus simplifying the expressions.

The gradient of $\sigma_l^\angle$ in λ_l, assuming all $\lambda_k = 0$, is

$$\frac{\partial \sigma_l^\angle}{\partial \lambda_l} = 2(\mathbf{x}_{\alpha-\beta} \cdot \mathbf{x}_{\gamma-\beta}) \left[\frac{(\mathbf{v}_{\alpha-\beta,l} \cdot \mathbf{x}_{\gamma-\beta} + \mathbf{v}_{\gamma-\beta,l} \cdot \mathbf{x}_{\alpha-\beta})}{\|\mathbf{x}_{\alpha-\beta}\|^2 \|\mathbf{x}_{\gamma-\beta}\|^2} - \frac{(\mathbf{x}_{\alpha-\beta} \cdot \mathbf{x}_{\gamma-\beta})[\|\mathbf{x}_{\alpha-\beta}\|^2 (\mathbf{x}_{\gamma-\beta} \cdot \mathbf{v}_{\gamma-\beta,l}) + \|\mathbf{x}_{\gamma-\beta}\|^2 (\mathbf{x}_{\alpha-\beta} \cdot \mathbf{v}_{\alpha-\beta,l})]}{\|\mathbf{x}_{\alpha-\beta}\|^4 \|\mathbf{x}_{\gamma-\beta}\|^4} \right], \tag{17}$$

where $\mathbf{v}_{\alpha-\beta,l} = \mathbf{v}_{\alpha,l} - \mathbf{v}_{\beta,l}$ and $\mathbf{v}_{\gamma-\beta,l} = \mathbf{v}_{\gamma,l} - \mathbf{v}_{\beta,l}$.

In the algorithm, λ_l is then updated using

$$\lambda_l \leftarrow \left. \frac{\partial \sigma_l^\angle}{\partial \lambda_l} \right|_{\lambda_k = 0, k = 1 \ldots n_c}. \tag{18}$$

Eq. (17) is rather cumbersome and costly to evaluate yet by applying the cosine rule it can be reduced to the equivalent, though much simpler, equation for the fictitious bond over $\mathbf{x}_\alpha$ and $\mathbf{x}_\gamma$:

$$\lambda_l \leftarrow \frac{\|\mathbf{x}_{l\alpha} - \mathbf{x}_{l\gamma}\|^2 - d_l^2}{2(\mathbf{x}_{l\alpha} - \mathbf{x}_{l\gamma}) \cdot (\mathbf{v}_{l\alpha,l} - \mathbf{v}_{l\gamma,l})}. \tag{19}$$

The angular constraints are solved for in the same iteration as distance constraints, as shown in Algorithm 2, which we will call ϑ-SHAKE.

The main advantage of this approach is that it renders constraint gradients on the end particles of the angle orthogonal to the gradients of the distance constraints over the angle rays (Fig. 2b). In more complex geometries where the particles in the angle would also be subject to other constraints, the new constraint gradients may overlap with other constraints, quickly annihilating whatever advantage was gained over the use of a fictitious bond or even failing to converge altogether. This approach therefore only has an advantage when the particles at the ends of the angular constraint are not subject to a significant number of overlapping constraints.

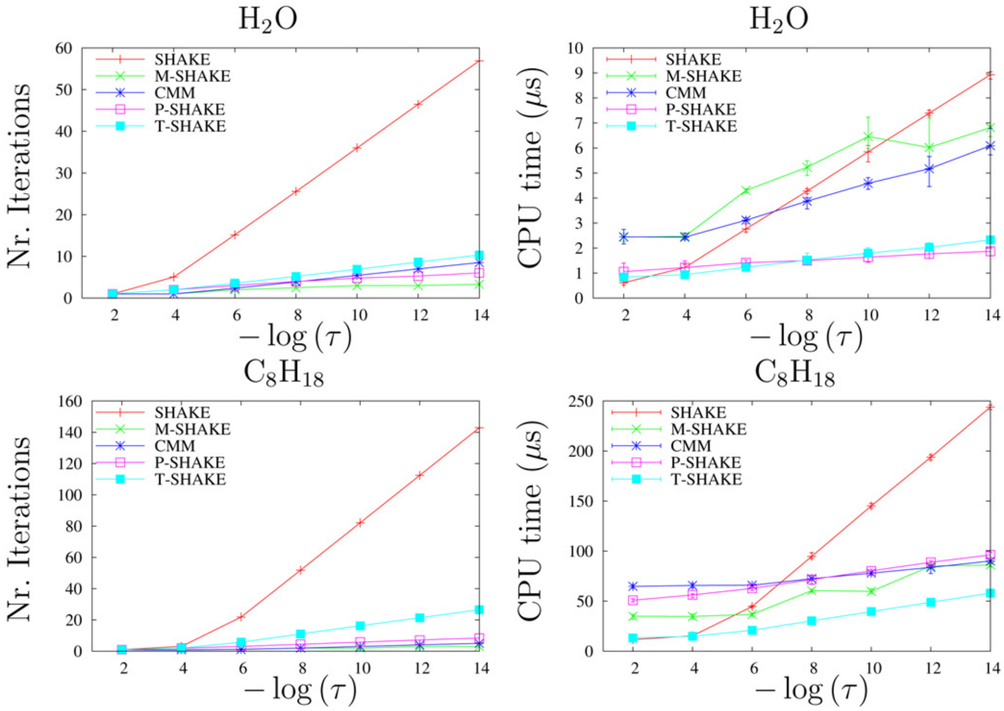

Fig. 3. Average number of iterations and average computation time for SHAKE, M-SHAKE, CMM, P-SHAKE, and ϑ-SHAKE algorithms. The performance is measured on an 2.4 GHz AMD Opteron 250 processor.

```
for $i = 1 \ldots n_c$ do
  if $\sigma_i$ is a distance constraint then
    $\mathbf{v}_{i\alpha,i} \leftarrow \frac{\partial \sigma_i^{(t)}}{\partial \mathbf{x}_{i\alpha}} (\Delta t)^2 m_{i\alpha}^{-1}$
    $\mathbf{v}_{i\beta,i} \leftarrow \frac{\partial \sigma_i^{(t)}}{\partial \mathbf{x}_{i\beta}} (\Delta t)^2 m_{i\beta}^{-1}$
  els if $\sigma_i$ is an angular constraint then
    $\mathbf{v}_{i\alpha,i} \leftarrow \frac{\partial \sigma_i^{\angle(t)}}{\partial \mathbf{x}_{i\alpha}} (\Delta t)^2 m_{i\alpha}^{-1}$
    $\mathbf{v}_{i\beta,i} \leftarrow \frac{\partial \sigma_i^{\angle(t)}}{\partial \mathbf{x}_{i\beta}} (\Delta t)^2 m_{i\beta}^{-1}$
    $\mathbf{v}_{i\gamma,i} \leftarrow \frac{\partial \sigma_i^{\angle(t)}}{\partial \mathbf{x}_{i\gamma}} (\Delta t)^2 m_{i\gamma}^{-1}$
  end if
end for
repeat
  for $i = 1 \ldots n_c$ do
    if $\sigma_i$ is a distance constraint then
      $\lambda_i \leftarrow \frac{\|\mathbf{x}_{i\alpha} - \mathbf{x}_{i\beta}\|^2 - d_i^2}{2(\mathbf{x}_{i\alpha} - \mathbf{x}_{i\beta}) \cdot (\mathbf{v}_{i\alpha,i} - \mathbf{v}_{i\beta,i})}$
      $\mathbf{x}_{i\alpha} \leftarrow \mathbf{x}_{i\alpha} + \lambda_i \mathbf{v}_{i\alpha,i}$
      $\mathbf{x}_{i\beta} \leftarrow \mathbf{x}_{i\beta} + \lambda_i \mathbf{v}_{i\beta,i}$
    else if $\sigma_i$ is an angular constraint then
      $\lambda_i \leftarrow \frac{\|\mathbf{x}_{i\alpha} - \mathbf{x}_{i\gamma}\|^2 - d_i^2}{2(\mathbf{x}_{i\alpha} - \mathbf{x}_{i\gamma}) \cdot (\mathbf{v}_{i\alpha,i} - \mathbf{v}_{i\gamma,i})}$
      $\mathbf{x}_{i\alpha} \leftarrow \mathbf{x}_{i\alpha} + \lambda_i \mathbf{v}_{i\alpha,i}$
      $\mathbf{x}_{i\beta} \leftarrow \mathbf{x}_{i\beta} + \lambda_i \mathbf{v}_{i\beta,i}$
      $\mathbf{x}_{i\gamma} \leftarrow \mathbf{x}_{i\gamma} + \lambda_i \mathbf{v}_{i\gamma,i}$
    end if
  end for
until $\max_i |\sigma_i| \leqslant \tau$
```

Algorithm 2. ϑ-SHAKE.

4. Results and discussion

We assess the efficiency of ϑ-SHAKE by comparing its performance to SHAKE, M-SHAKE, CMM, and P-SHAKE for rigid water and octane. Water was chosen as a prototypical example of a constrained angle and because it is a common molecule in MD simulations. Octane was chosen since it contains many angular groups, namely the H–C–H over each carbon and the CH_3 groups at the ends, in which the particles at the ends of the rays (*i.e.* the hydrogens) are not subject to many other constraints.

The molecular dynamics simulations were performed using the FASTTUBE package [13], with all non-bonded interactions truncated at 1.0 nm and smoothed and truncated Coulomb potential as described in [14]. The systems are equilibrated for 10 000 steps using a Berendsen heat bath [15] and the performance of the different algorithms (of iterations and timing) is averaged out from the 1000 time steps following the equilibration.

The water is described by the rigid SPC/E water model [16] with constrained bonds length $d_{\text{O–H}} = 0.1000$ nm and angle $\vartheta_{\text{H–O–H}} = 109.47°$, while the length of the fictitious bond extending between the hydrogens is $d_{\text{H–H}} = 0.1633$ nm ($n_c = 3$). The computational domain is cubic with an edge length of 3.000 nm. The system contains 900 molecules at 298 K with a density of 0.997 $\text{g}\,\text{cm}^{-3}$. The time step is 2.0 fs.

For the simulations of liquid octane (C_8H_{18}) we consider the all atom model by Forester and Smith [17]. The model is based on the AMBER force field [18], including Lennard-Jones and Coulomb interactions, and harmonic C–C–C angle and C–C–C–C torsion potentials. The C–H, and C–C bonds and the H–C–H angles are constrained, with $d_{\text{C–H}} = 0.1090$ nm, $d_{\text{C–C}} = 0.1526$ nm and $\vartheta_{\text{H–C–H}} = 109.5°$. The fictitious bond extending between the hydrogens is $d_{\text{H–H}} = 0.1780$ nm ($n_c = 37$). The system considered involves 64 octane molecules maintained at 320 K in a cubic box of side length 2.720 nm. A time step of 2 fs was found to ensure conservation of energy during the simulation.

The simulations were performed on an 2.4 GHz AMD Opteron 250 processor using the Intel Fortran Compiler version 10.1 and the Intel Math Kernel (MKL) Library 10.0 BLAS and LAPACK routines for the LU decomposition in CMM and M-SHAKE. The number of iterations and the average CPU time required to converge to a specified absolute tolerance τ is recorded for $\tau = 10^{-2}, 10^{-4}, \ldots, 10^{-14}$ in Fig. 3.

For the simulation of rigid SPC/E water we find that the proposed ϑ-SHAKE algorithm reduces the number of iterations from the 57 iteration required by the standard SHAKE algorithm ($\tau = 10^{-14}$) to 10 iterations which is still higher than the iterations required by M-SHAKE (3 iterations), P-SHAKE (6 iterations), and CMM (9 iterations). Note however, that the low number of operations required per iteration by the ϑ-SHAKE algorithm results in a CPU time comparable to the one required by P-SHAKE, being approximately 20% faster for large tolerances (10^{-2}) and 20% slower for small tolerances (10^{-14}).

In the case of octane, ϑ-SHAKE outperforms the other constraint solvers for all tolerances. The large number of constraints has a strong influence on the performance of M-SHAKE and CMM, which require $\mathcal{O}(n_c^3)$ and $\mathcal{O}(n_c^2)$ operations per iteration respectively. P-SHAKE, which is more efficient for rigid and semi-rigid molecules, is also at a disadvantage, since the octane molecules are flexible.

Simulations of rigid DMSO and $CHCl_3$ were also considered, yet due to the high connectivity of their constraints it was difficult to find a configuration for which the Gauss–Seidel iteration would converge and where it did, it showed only a small advantage of ϑ-SHAKE over SHAKE. These results are therefore not included in the analysis.

The results show that for molecules with low internal connectivity and angular constraints (*e.g.*, alkanes, lipid chains and other bio-molecules), ϑ-SHAKE, due to its cost of $\mathcal{O}(n_c)$ per iteration and faster convergence than SHAKE, not only performs almost as well as the more complex P-SHAKE for small molecules, but scales better as the number of constraints increases.

Acknowledgements

We wish to thank Professor Bertrand Meyer (ETHZ), for making this paper possible. P.G. would like to thank Dr. Urs Zimmerli for the useful discussions regarding constraints, molecular dynamics and chemistry in general. J.H.W. and P.K. acknowledge partial support by the Myhrvold and Otto Mønsted Foundations.

References

[1] J.-P. Ryckaert, G. Ciccotti, H.J.C. Berendsen, J. Comput. Phys. 23 (1977) 327.
[2] G. Ciccotti, J.-P. Ryckaert, Comput. Phys. Rep. 4 (1986) 345.
[3] G. Ciccotti, M. Ferrario, J.-P. Ryckaert, Mol. Phys. 47 (1982) 1253.
[4] J.P. Ryckaert, Mol. Phys. 55 (1985) 549.
[5] E. Barth, K. Kuczera, B. Leimkuhler, R.D. Skeel, J. Comput. Chem. 16 (1995) 1192.
[6] B. Hess, H. Bekker, H.J.C. Berendsen, J.G.E.M. Fraaije, J. Comput. Chem. 18 (1997) 1463.
[7] V. Kräutler, W.F. van Gunsteren, P.H. Hünenberger, J. Comput. Chem. 22 (2001) 501.
[8] S. Miyamoto, P.A. Kollman, J. Comput. Chem. 13 (1992) 952.
[9] H.C. Andersen, J. Comput. Phys. 52 (1983) 24.
[10] S.G. Lambrakos, J.P. Boris, E.S. Oran, I. Chandrasekhar, M. Nagumo, J. Comput. Phys. 85 (1989) 473.
[11] S.-H. Lee, K. Palmo, S. Krimm, J. Comput. Phys. 210 (2005) 171.
[12] P. Gonnet, J. Comput. Phys. 220 (2007) 740.
[13] J.H. Walther, R. Jaffe, T. Halicioglu, P. Koumoutsakos, J. Phys. Chem. B 105 (2001) 9980.
[14] M. Levitt, M. Hirshberg, K.E. Laidig, V. Daggett, J. Phys. Chem. B 101 (1997) 5051.
[15] H.J.C. Berendsen, J.P.M. Postma, W.F. van Gunsteren, A. DiNola, J.R. Haak, J. Chem. Phys. 81 (1984) 3684.
[16] H.J.C. Berendsen, J.R. Grigera, T.P. Straatsma, J. Phys. Chem. 91 (1987) 6269.
[17] T.R. Forester, W. Smith, J. Comput. Chem. 19 (1998) 102.
[18] S.J. Weiner, P.A. Kollman, D.T. Nguyen, D.A. Case, J. Comput. Chem. 7 (1986) 230.