A Fast SHAKE Algorithm to Solve Distance Constraint Equations for Small Molecules in Molecular Dynamics Simulations

VINCENT KRÄUTLER, WILFRED F. VAN GUNSTEREN, PHILIPPE H. HÜNENBERGER

Laboratorium für Physikalische Chemie, ETH Zentrum, Universitätstrasse 6, Zürich, Switzerland

Received 29 August 2000; accepted 11 November 2000

ABSTRACT: A common method for the application of distance constraints in molecular simulations employing Cartesian coordinates is the SHAKE procedure for determining the Lagrange multipliers regarding the constraints. This method relies on the linearization and decoupling of the equations governing the atomic coordinate resetting corresponding to each constraint in a molecule, and is thus iterative. In the present study, we consider an alternative method, M-SHAKE, which solves the coupled equations simultaneously by matrix inversion. The performances of the two methods are compared in simulations of the pure solvents water, dimethyl sulfoxide, and chloroform. It is concluded that M-SHAKE is significantly faster than SHAKE when either (1) the molecules contain few distance constraints (solvent), or (2) when a high level of accuracy is required in the application of the constraints. J Comput Chem 22: 501–508, 2001

Keywords: molecular dynamics; constraints; SHAKE procedure

Introduction

The application of constraints in molecular dynamics (MD) simulations of (bio-)molecular systems is a very common practice. It offers three main advantages. First, the maximal time step permitting an accurate integration of the equations of motion is directly related to the frequency of the fastest motions in the system. Because the highest frequency is generally associated with bond vibrations, constraining bond lengths permits an increase in the simulation time step (roughly by a factor 2–3[1]), and thus a reduction of the computational effort for obtaining a trajectory of a given length. The application of bond-angle constraints may permit a further small increase in the time step, but is generally not recommended, because it may bias both the equilibrium and dynamical properties of

Corresponding author: P. H. Hünenberger; e-mail: phil@igc.phys.chem.ethz.ch

the system.[2,3] However, for completely rigid molecules (no internal degrees of freedom), such artifacts disappear, and the simultaneous application of bond-length and bond-angle constraints is common practice (e.g., for solvent molecules). Second, when the range of motional frequencies in a molecular system is very broad, the exchange of energy between the high- and low-frequency modes may be slow. Differences in the accuracy of the forces computed for the different modes may result in different heating rates, and, when the heat exchange is slow, different effective temperatures. The application of bond-length constraints largely remedies this problem by narrowing the range of frequencies in the system. Third, if the vibrational frequency of a mode is larger than $k_B T/h$, where k_B is Boltzmann's constant, T the absolute temperature, and h Planck's constant, the vibration must be treated quantum mechanically. At room temperature, this is the case for most bond-stretching frequencies. In this case, treating bonds as constraints is probably a better approximation of their quantum-mechanical behavior than treating them as classical harmonic oscillators.[4]

Although a number of algorithms have been developed for the application of distance constraints in MD simulations,[5–8] the most popular of them is certainly the SHAKE procedure.[5] In this method, the N_c constraint equations determining the bond (and possibly angle) constraints in a molecule are multiplied by as many Lagrange multipliers and added to the physical potential energy function governing the motion of the system. When inserted into the constraint equations, the equations of motion lead to a system of N_c quadratic equations that must be solved simultaneously for the Lagrange multipliers. Two approximations are then made: (1) the system of equations is linearized by neglecting any term quadratic in the multipliers, and (2) the multipliers are determined independently in sequence by omitting the coupling between distance constraints involving a common atom. As a consequence of these approximations, the procedure must be performed iteratively until satisfaction of all constraints within a specified tolerance. Note, however, that the second approximation can easily be removed. In this case, the linearized system of coupled equations is solved exactly through the inversion of an $N_c \times N_c$ matrix.[5] This method will be further referred to as M-SHAKE (Matrix-inversion SHAKE), and is generally not applied in molecular simulations because the required computing time rapidly rises with increasing number of constraints. Iteration is still required due to the linearization approximation, but the number of iterations is expected to be significantly reduced.

In the case of a complex solute (e.g., macromolecule), the matrix to be inverted becomes very large, and M-SHAKE is not advantageous compared to SHAKE, both in terms of computational speed and memory requirement. However, for small molecules, M-SHAKE may become competitive, because it requires fewer iterations of a simple matrix inversion. In typical simulations of (bio-)molecular systems, most of the distance constraints are located within solvent molecules. Consequently, application of M-SHAKE for the solvent may result in a very significant overall performance improvement.

This possibility was already exploited previously in the algorithm SETTLE.[6] This algorithm is based on an analytical expression for the constraint forces in rigid triatomic molecules. The method avoids the linearization of the system of equations, and is thus non-iterative. However, it requires the evaluation of ten (expensive) square-root functions per tri-atomic molecule, and consequently, is not necessarily faster than M-SHAKE. Furthermore, the generalization to molecules consisting of more than three atoms is not straightforward.

In the present study, we compare the performances of SHAKE, M-SHAKE (with different matrix inversion procedures) and SETTLE (tri-atomic molecule only) in simulations of the pure solvents water, dimethyl sulfoxide (DMSO), and chloroform.

Theory

The N_c distance constraints within a specific molecule can be written in the form of N_c constraint equations

$$\sigma_k(\{\mathbf{r}_i\}) = r_{k_1 k_2}^2 - d_{k_1 k_2}^2 = 0, \qquad k = 1, \ldots, N_c, \quad (1)$$

where $\mathbf{r}_i$ is the position vector of atom i, $r_{k_1 k_2}$ the distance between the atoms k_1 and k_2 involved in constraint k, and $d_{k_1 k_2}$ the corresponding constraint distance. These conditions must be enforced while integrating the classical (Newtonian) equations of motion. This can be accomplished by applying Lagrange's method of undetermined multipliers. A zero term is added to the physical potential energy function $V(\{\mathbf{r}_i\})$, which yields the equations of motion

$$m_i \frac{d^2 \mathbf{r}_i(t)}{dt^2} = -\frac{\partial}{\partial \mathbf{r}_i}\left[V(\{\mathbf{r}_i\}) + \sum_{k=1}^{N_c} l_k(t)\sigma_k(\{\mathbf{r}_i\}) \right], \quad (2)$$

where m_i is the mass of atom i, and the coefficients $l_k(t)$ represent the (unknown) Lagrange multipliers.

The first term in the right-hand side of this equation represents the unconstrained force $\mathbf{f}_i^{uc}$ acting on atom i, while the second term represents the constraint force $\mathbf{f}_i^c$

$$\mathbf{f}_i^c(t) = -\sum_{k=1}^{N_c} l_k(t)\frac{\partial \sigma_k(\{\mathbf{r}_i(t)\})}{\partial \mathbf{r}_i(t)} = -2\sum_{k=1}^{N_c} l_k(t)(\delta_{i,k_1} - \delta_{i,k_2})\mathbf{r}_{k_1k_2}(t), \quad (3)$$

where $\mathbf{r}_{k_1k_2} \equiv \mathbf{r}_{k_1} - \mathbf{r}_{k_2}$. If the leap-frog algorithm[9] is used to integrate the equations of motion, the unconstrained atomic coordinates resulting from the action of the unconstrained forces are given by

$$\mathbf{r}_i^{uc}(t+\Delta t) = \mathbf{r}_i(t) + \Delta t\,\mathbf{v}_i(t - \Delta t/2) + (\Delta t)^2 m_i^{-1}\mathbf{f}_i^{uc}(t), \quad (4)$$

where $\mathbf{v}_i$ is the velocity of atom i and Δt the time step size. Adding the effect of the constraint forces leads to

$$\mathbf{r}_i(t+\Delta t) = \mathbf{r}_i^{uc}(t+\Delta t) + (\Delta t)^2 m_i^{-1}\mathbf{f}_i^c(t). \quad (5)$$

These new coordinates should satisfy the constraint equations [eq. (1)], and thus

$$\left[\mathbf{r}_{k_1k_2}^{uc}(t+\Delta t) + (\Delta t)^2 m_{k_1}^{-1}\mathbf{f}_{k_1}^c(t) - (\Delta t)^2 m_{k_2}^{-1}\mathbf{f}_{k_2}^c(t)\right]^2 - d_{k_1k_2}^2 = 0, \qquad k = 1,\ldots,N_c. \quad (6)$$

Inserting eq. (3) leads to a set of N_c quadratic equations, which is to be solved for the Lagrange multipliers $l_k(t)$

$$\left\{\mathbf{r}_{k_1k_2}^{uc}(t+\Delta t) - 2(\Delta t)^2\sum_{k'=1}^{N_c} l_{k'}(t)\mathbf{r}_{k_1'k_2'}(t) \times \left[m_{k_1}^{-1}(\delta_{k_1,k_1'} - \delta_{k_1,k_2'}) + m_{k_2}^{-1}(\delta_{k_2,k_2'} - \delta_{k_2,k_1'})\right]\right\}^2 - d_{k_1k_2}^2 = 0, \qquad k = 1,\ldots,N_c. \quad (7)$$

In the SHAKE method,[5] the following two approximations are made: (1) the system of equations is linearized by neglecting the terms quadratic in the Lagrange multipliers, and (2) the equations are decoupled by assuming that the atoms k_1 and k_2 involved in constraint k are not involved in any other constraint. With these simplifications, approximate solutions for the multipliers are given by

$$l_k(t) = \frac{[\mathbf{r}_{k_1k_2}^{uc}(t+\Delta t)]^2 - d_{k_1k_2}^2}{4(\Delta t)^2(m_{k_1}^{-1} + m_{k_2}^{-1})\mathbf{r}_{k_1k_2}(t)\cdot\mathbf{r}_{k_1k_2}^{uc}(t+\Delta t)}, \qquad k = 1,\ldots,N_c, \quad (8)$$

and the corresponding coordinate adjustments are performed according to eqs. (3) and (5) as

$$\begin{aligned}\mathbf{r}_{k_1}(t+\Delta t) &= \mathbf{r}_{k_1}^{uc}(t+\Delta t) - 2(\Delta t)^2 m_{k_1}^{-1} l_k(t)\mathbf{r}_{k_1k_2}(t) \qquad \text{and}\\ \mathbf{r}_{k_2}(t+\Delta t) &= \mathbf{r}_{k_2}^{uc}(t+\Delta t) + 2(\Delta t)^2 m_{k_2}^{-1} l_k(t)\mathbf{r}_{k_1k_2}(t).\end{aligned} \quad (9)$$

Due to the approximations made above, the procedure must be iterated, taking the coordinates $\mathbf{r}_i$ obtained through eq. (9) at each iteration as the coordinates $\mathbf{r}_i^{uc}$ for the next iteration. The iteration cycle for a given molecule is terminated when all constraint conditions [eq. (1)] are satisfied within a given tolerance

$$\frac{|r_{k_1k_2}(t+\Delta t) - d_{k_1k_2}|}{d_{k_1k_2}} \le \tau, \qquad k = 1,\ldots,N_c, \quad (10)$$

where τ is the relative geometric tolerance on the constraint length. Using the Taylor expansion of $r^2 - d^2$ around $r = d$

$$r^2 - d^2 = 2d(r-d) + O\big[(r-d)^2\big], \quad (11)$$

eq. (10) can be conveniently approximated by the expression

$$\frac{|r_{k_1k_2}^2(t+\Delta t) - d_{k_1k_2}^2|}{2d_{k_1k_2}^2} \le \tau, \qquad k = 1,\ldots,N_c, \quad (12)$$

which avoids the evaluation of a square-root function.

In the M-SHAKE method, eq. (7) is also linearized, but the approximation of independent constraints is not made. Instead, eq. (7) is rewritten (after linearization) in the matrix form

$$\underline{\mathbf{A}}\mathbf{l} = \mathbf{c}, \quad (13)$$

where $\mathbf{l}$ is an N_c-dimensional vector containing the Lagrange multipliers, the elements of the vector $\mathbf{c}$ are given by

$$c_k = \frac{[r_{k_1k_2}^{uc}(t+\Delta t)]^2 - d_{k_1k_2}^2}{4(\Delta t)^2}, \quad (14)$$

and the elements of the matrix $\underline{\mathbf{A}}$ by

$$A_{kk'} = \left[m_{k_1}^{-1}(\delta_{k_1,k_1'} - \delta_{k_1,k_2'}) + m_{k_2}^{-1}(\delta_{k_2,k_2'} - \delta_{k_2,k_1'})\right] \times \mathbf{r}_{k_1'k_2'}(t)\cdot\mathbf{r}_{k_1k_2}^{uc}(t+\Delta t). \quad (15)$$

It is easily seen that the diagonal elements A_{kk} characterize the force directly due to constraint k, whereas the off-diagonal elements $A_{kk'}$ ($k' \neq k$) account for the effect along a constraint k of the forces due to a constraint k'. These off-diagonal elements are neglected in SHAKE. In M-SHAKE, eq. (13) is solved directly by matrix inversion. The following

methods were considered for performing this inversion.

1. Explicit (hard-coded) calculation of $\mathbf{A}^{-1}$. This was only done for water ($N_c = 3$), because the expressions quickly become very complex for larger matrices.
2. Use of Cramer's rule. In this method, each component of the vector $\mathbf{l}$ is calculated as $l_k = |\underline{\mathbf{A}}_k|/|\underline{\mathbf{A}}|$, where $|\underline{\mathbf{A}}_k|$ is the determinant obtained by replacing the kth column of $\underline{\mathbf{A}}$ by the vector $\mathbf{c}$. Here again, the expressions quickly become untractable for larger matrices and the method was only tested for water.
3. Use of the LU-factorization method.[10] In this method, a square, non-singular matrix $\underline{\mathbf{A}}$ is factorized into a lower triangular matrix $\underline{\mathbf{L}}$ and an upper triangular matrix $\underline{\mathbf{U}}$, such that $\underline{\mathbf{L}}\,\underline{\mathbf{U}} = \underline{\mathbf{A}}$. The computational cost of the factorization scales as N_c^3, whereas the inversion of the triangular matrices only scales as N_c^2.
4. Use of the LDL^t-factorization method.[10] In this case, the matrix $\underline{\mathbf{A}}$ is approximated by a symmetric matrix $\underline{\mathbf{A}}'$ with identical diagonal elements, but in which any occurrence of $\mathbf{r}^{uc}_{k_1k_2}(t + \Delta t)$ in the off-diagonal elements is replaced by $\mathbf{r}_{k_1k_2}(t)$. The symmetric matrix $\underline{\mathbf{A}}'$ is then factorized as $\underline{\mathbf{L}}\,\underline{\mathbf{D}}\,\underline{\mathbf{L}}^t = \underline{\mathbf{A}}$, where $\underline{\mathbf{D}}$ is a diagonal matrix, $\underline{\mathbf{L}}$ a lower triangular matrix, and the t superscript denotes the transpose. This factorization method also scales as N_c^3, but is about twice less expensive in terms of floating-point operations compared to LU-factorization.

Because M-SHAKE (just as SHAKE) involves the linearization of eq. (7), it is also an iterative procedure. As in SHAKE, the iteration cycle is terminated when eq. (12) is satisfied. However, it is expected that M-SHAKE will require fewer iterations compared to SHAKE because it takes into account the coupling between the constraints within a molecule.

After application of the constraints to the atomic coordinates using either SHAKE or M-SHAKE, components of the atomic velocities along the constraints must be removed. This is simply done by replacing the leap-frog equation for the velocities by the equation

$$\mathbf{v}_i(t + \Delta t/2) = \big[\mathbf{r}_i(t + \Delta t) - \mathbf{r}_i(t)\big]/\Delta t, \qquad (16)$$

where the atomic velocities are recalculated based on the constrained coordinates.

Computational Details

The different methods for applying constraints were compared for three solvent models: simple point charge water (SPC, three sites[11]), dimethyl sulfoxide (DMSO, four sites[12]), and chloroform (five sites[13–15]). For the three solvents, the simulations involved cubic unit cells containing 5000, 1000, and 1000 molecules, respectively, at the experimental densities of 0.997, 1.095, and 1.489 $g \cdot cm^{-3}$ (room temperature). Electrostatic interactions were truncated at 0.9 nm, and a reaction-field correction[16, 17] was included using the solvent experimental relative dielectric permittivities of 78, 46, and 4.8, respectively. The temperature was maintained by weak coupling to a heat bath[18] at 298 K (unless otherwise specified) using a coupling constant of 0.1 ps. A time step of 2 fs (unless otherwise specified) was used to integrate the equations of motion. Simulations were carried out using a modified version of the GROMOS simulation program[19] in which SHAKE, different versions of M-SHAKE, and the subroutine SETTLE (taken from AMBER[20]) were implemented. For fair comparison, all routines involved in the application of constraints were cleaned to remove any superfluous code, and optimized with respect to the number of floating-point operations. Values of the relative tolerance τ [eq. (12)] ranging from 10^{-2} to 10^{-12} were considered. The average number of iterations per molecule, N_{iter}, and the average CPU time per molecule to satisfy the constraints, t_{CPU} (measured on a Sun Ultra 10 workstation with a 440 MHz UltraSPARC-IIi processor), were monitored. These averages were calculated over 50 steps of constrained MD simulation, starting from well equilibrated solvent configurations. For simplicity, the results are reported as a function of the quantity $p\tau \equiv -\log \tau$, which increases with the required accuracy.

Results

The average number of iterations and CPU time per molecule to satisfy the constraint equations for the different methods as applied to water ($N_c = 3$) are displayed in Figure 1. For SHAKE, the number of iterations increases almost linearly as a function of $p\tau$, with a slope of about 5.1. In contrast, for M-SHAKE, this increase is much smaller (from one to about 3.5 over 10 orders of magnitude in accuracy), and proceeds in a stepwise fashion. On average, about one iteration is needed for $p\tau = 2$, about two

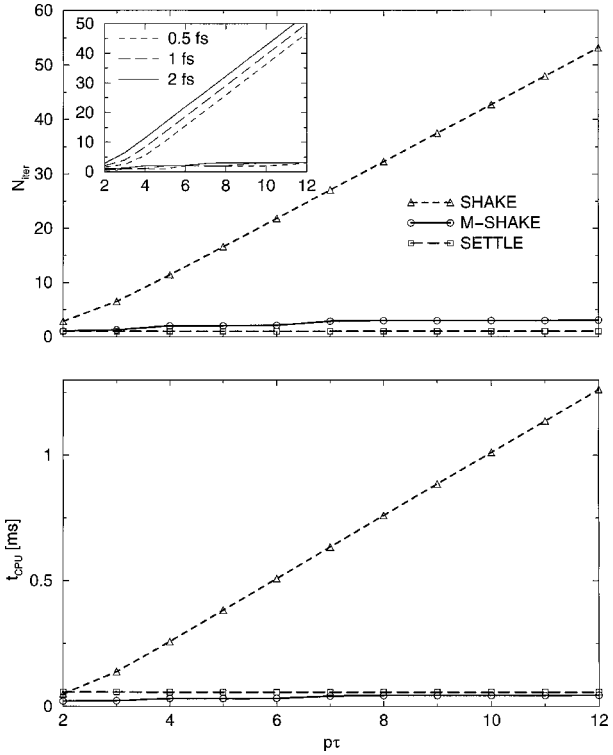

FIGURE 1. Average number of iterations (top) and CPU times (bottom) per molecule to satisfy the constraint equations for water (number of constraints $N_c = 3$) as a function of $p\tau \equiv -\log \tau$, where τ is the relative tolerance on the constraint distances. Curves corresponding to SHAKE, M-SHAKE (explicit matrix inversion), and SETTLE are represented. Curves corresponding to M-SHAKE using Cramer's rule are indistinguishable from those with explicit matrix inversion. Inset (top graph): average number of iterations using different MD time step sizes $\Delta t = 0.5$, 1.0, or 2.0 fs, displayed as a function of $p\tau$, using SHAKE (upper curves) and M-SHAKE (explicit matrix inversion, lower curves).

for $p\tau = 4$–6, and about three for $p\tau = 7$–11. Thus, M-SHAKE requires fewer iterations compared to SHAKE over the whole range of tolerances considered, the difference being especially large when a high level of accuracy is required.

Similar considerations apply to the CPU times required to solve the constraint equations. Because each M-SHAKE iteration (0.014–0.021 ms per molecule, depending on $p\tau$) requires about the same time as a SHAKE iteration (0.018–0.024 ms per molecule), the much higher number of iterations required by SHAKE to reach convergence makes it a much slower method. On the other hand, although SETTLE only performs one iteration per molecule, this single iteration requires 0.058 ms. As a consequence, M-SHAKE is found to be the fastest of the three algorithms over the whole range of tolerances considered. For example, at $p\tau = 4$ (a value typical of standard molecular simulations), M-SHAKE takes 0.033 ms per molecule, SETTLE is 1.7 times slower and SHAKE about eight times slower. At $p\tau = 8$, M-SHAKE takes 0.044 ms per molecule, SETTLE is 1.3 times slower, while SHAKE is about 17 times slower. The application of a version of M-SHAKE based on Cramer's rule led to very similar results, the corresponding CPU times being only about 1 to 2% larger compared to the version of M-SHAKE based on explicit matrix inversion.

In the case of water, the dependence of the number of iterations and CPU times on the time step size Δt and on the temperature were also tested. The time step size was found to have some influence on the number of iterations required to reach convergence, as is illustrated in the inset of Figure 1. For SHAKE and for $p\tau \geq 4$, multiplying the time step size by a factor two leads to about three additional iterations per molecule, leading to an offset in the curves while the slope remains unchanged. For M-SHAKE, the stepwise increases in the number of iterations as a function of $p\tau$ are shifted towards lower $p\tau$ values when Δt is increased, whereas the size of the plateaus remains practically unchanged. These observations are easy to rationalize. When Δt is increased, the unconstrained coordinates $\mathbf{r}_i^{uc}(t + \Delta t)$ deviate more significantly from the initial coordinates $\mathbf{r}_i(t)$. Constraints are violated to a larger extent, thereby requiring more SHAKE or M-SHAKE iterations. On the other hand, the effect of a temperature change (from 298 to 400 or 500 K) remained practically unnoticeable (data not shown). This absence of effect can be explained in two ways: (1) an increase in temperature from 298 to 500 K only implies an increase in the average atomic velocities by a factor 1.3, and (2) the velocities $\mathbf{v}_i(t - \Delta t/2)$ [see eq. (4)] possess no components along the constraints due to application of eq. (16). For these reasons, the effect of increasing the temperature is expected to be much more limited than the effect of increasing the time step size.

The average number of iterations and CPU time per molecule to satisfy the constraints for the different methods as applied to DMSO ($N_c = 6$) are displayed in Figure 2. The results are qualitatively similar to those for water (Fig. 1). Note, however, that the curve describing the number of iterations for SHAKE only becomes linear for $p\tau \geq 5$, with a slope of about 4.8. Interestingly, for both SHAKE and M-SHAKE, the number of iterations is lower by 30–50% for DMSO compared to water. This may be a consequence of two effects: (1) DMSO is much less

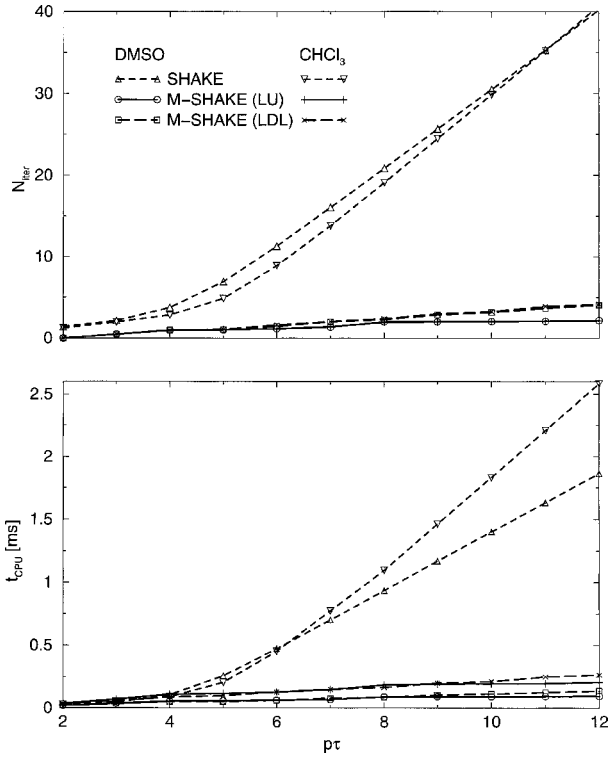

FIGURE 2. Average number of iterations (top) and CPU times (bottom) per molecule to satisfy the constraint equations for DMSO (number of constraints $N_c = 6$) and chloroform (number of constraints $N_c = 9$) as a function of $p\tau \equiv -\log \tau$, where τ is the relative tolerance on the constraint distances, for SHAKE and M-SHAKE (with *LU* or LDL^t factorization).

polar than water, and its atoms are thus subject to smaller electrostatic forces; and (2) DMSO does not contain hydrogen atoms, which, due to their small masses, are subject to larger unconstrained displacements compared to non-hydrogen atoms.

For DMSO and for $p\tau \geq 5$ each M-SHAKE iteration (0.045–0.054 ms per molecule, depending on $p\tau$, for the *LU*-factorization) requires about the same amount of CPU time compared to a SHAKE iteration (0.041–0.047 ms per molecule). Note that these times are about twice as large as the corresponding times for water, reflecting the ratio of the number of constraints in the two molecules. For $p\tau < 5$, the CPU time per iteration becomes somewhat more advantageous for SHAKE compared to M-SHAKE. Nevertheless, for DMSO, M-SHAKE is also faster than SHAKE over the whole range of tolerances considered, but the speed increase is only marginal for the lowest values of $p\tau$. For example, at $p\tau = 4$, M-SHAKE (*LU*-factorization) takes about 0.057 ms per molecule and SHAKE is 1.6 times slower, whereas at $p\tau = 8$, M-SHAKE takes about 0.090 ms per molecule and SHAKE is about 12 times slower. The comparison between the *LU*-factorization and LDL^t-factorization methods for applying M-SHAKE shows that the latter one is somewhat faster for $p\tau \leq 6$. For higher accuracies, the LDL^t-method requires more iterations (because it approximates the $\underline{\mathbf{A}}$ matrix by a symmetric matrix), and becomes slower.

The average number of iterations and CPU time per molecule to satisfy the constraints for the different methods as applied to chloroform ($N_c = 9$) are also displayed in Figure 2. The results are qualitatively similar to those for water (Fig. 1) and DMSO. As for DMSO, the curve describing the number of iterations for SHAKE only becomes linear for $p\tau \geq 5$, with a slope of about 5.3. For chloroform and for $p\tau \geq 5$, each M-SHAKE iteration (0.096–0.114 ms per molecule, depending on $p\tau$, for the *LU*-factorization) now requires about twice as much CPU time compared to a SHAKE iteration (0.050–0.063 ms per molecule). For SHAKE, the CPU times are about 3 and 1.5 times as large, respectively, as the corresponding times for water and DMSO, respectively, reflecting again the ratio of the number of constraints in the different molecules. For $p\tau < 5$, the CPU time per iteration becomes even more advantageous for SHAKE (0.024–0.042 ms) compared to M-SHAKE (0.115–0.154 ms). As a consequence, SHAKE becomes noticeably faster than M-SHAKE for low values of $p\tau$. For example, at $p\tau = 4$, M-SHAKE (*LU*-factorization) takes about 0.111 ms per molecule and SHAKE is 1.2 times faster, whereas at $p = 8$, M-SHAKE takes about 0.186 ms per molecule and SHAKE is six times slower.

When considering in turn water, DMSO, and chloroform, SHAKE iteration numbers and timings describe an increasingly marked plateau for low values of $p\tau$. This is due to the fact that SHAKE can skip individual constraints in molecules as soon as these are satisfied within the required tolerance, thereby increasing the efficiency of the algorithm. On the other hand, because M-SHAKE simultaneously solves all constraint equations for one whole molecule, this algorithm does not benefit from this advantage.

For the three solvents considered, M-SHAKE is considerably faster than SHAKE for small relative tolerances (large values of $p\tau$). For $p\tau = 4$ (as typically used in current simulations), M-SHAKE is eight times faster for water, marginally faster for DMSO, and marginally slower for chloroform. However, the value $p\tau = 4$ may already be rather high for ensuring accurate energy conservation in

MD simulations. To assess the effect of $p\tau$ on energy conservation, we performed a few additional 50 ps MD simulations of 512 SPC water molecules in a cubic box of edge $L = 2.486$ nm, employing the P^3M method to handle electrostatic interactions (hat charge-shaping function of width 1.0 nm, $32\times32\times32$ grid points). These simulations were started from a configuration equilibrated by 50 ps MD simulation at $T = 298$ K using $p\tau = 12$, and carried out for 50 ps in the absence of temperature coupling and with different values of $p\tau$. The time evolution of the total (kinetic + potential) energy during these simulations, scaled by the number of molecules, could be adequately fitted by straigth lines with intercept E_o and slope α. The choice $p\tau = 12$ led to $E_o = -34.353\ \mathrm{kJ\cdot mol^{-1}}$ and $\alpha = 6.1\cdot10^{-5}\mathrm{kJ\cdot mol^{-1}\cdot ps^{-1}}$, indicating a slight heating of the system over time, probably caused by noise in the calculation of the electrostatic forces and in the integration of the equations of motion. In contrast, the choices $p\tau = 4$ and $p\tau = 3$ led to $E_o = -34.356$ and $-34.370\ \mathrm{kJ\cdot mol^{-1}}$, respectively, and $\alpha = -29.0\cdot10^{-5}$ and $-125.7\cdot10^{-5}\ \mathrm{kJ\cdot mol^{-1}\cdot ps^{-1}}$, respectively, evidencing (1) an initial decrease in the total energy caused by a slight increase of the average O—H bond lengths, and (2) a progressive cooling of the system caused by the reduced SHAKE accuracy. These cooling rates are moderate but nonnegligible, leading to a decrease of 1% in the total potential energy after about 1.1 ns ($p\tau = 4$) or 0.3 ns ($p\tau = 3$). On the other hand, $p\tau = 6$ gives a negligible cooling ($\alpha = -1.2\cdot10^{-5}\ \mathrm{kJ\cdot mol^{-1}\cdot ps^{-1}}$), and $p\tau = 8$ a heating rate comparable with the $p\tau = 12$ case. When cutoff truncation is used to handle electrostatic interactions (rather than a lattice-sum method), the cooling arising from SHAKE inaccuracies becomes negligible compared to the heating caused by the cutoff noise. For the same system of 512 SPC water molecules, a cutoff of 1.2 nm, and a reaction-field correction with $\epsilon_{RF} = 78$, we find $\alpha = 0.45\ \mathrm{kJ\cdot mol^{-1}\cdot ps^{-1}}$, which is over 300 times larger than the effect of SHAKE at $p\tau = 4$.

Conclusion

The results of the present study show that for small molecules (solvent), the SHAKE algorithm, which solves the constraint equations independently, can be advantageously replaced by the M-SHAKE procedure, proceeding by matrix inversion. In addition, for a tri-atomic molecule, M-SHAKE turns out to be also slightly faster than the analytical solution provided by the SETTLE algorithm.

The speed gain in the M-SHAKE procedure is essentially due to a very limited number of iterations compared to SHAKE. On the other hand, the computational cost of each iteration scales in principle as N_c^3 (LU- or LDL^t-factorization), whereas the cost of one SHAKE iteration scales as N_c or less (because individual constraints can be skipped as soon as these are satisfied within the required tolerance). Consequently, M-SHAKE will only be competitive when (1) there are few constraints in each molecule, or (2) a high level of accuracy (low tolerance) is required. For a relative tolerance $\tau = 10^{-4}$ (as typically used in current simulations), use of M-SHAKE is very advantageous for water (eight times faster), somewhat faster than SHAKE for DMSO, but slightly slower for chloroform (and probably also for any larger molecule). However, for applications with a high accuracy of the force evaluation (e.g., employing lattice-sum methods), a value of $\tau = 10^{-4}$ may already lead to significant inaccuracies and, in particular, prevent good energy conservation in simulations. Chosing a value of $\tau = 10^{-8}$ instead of 10^{-4}, the computational cost of SHAKE is multiplied by 2.9 (water), 8.8 (DMSO), and 11.9 (chloroform), whereas for M-SHAKE, these scaling factors are only 1.3, 1.6, and 1.7, respectively. Because most of the effort in the application of constraints generally involves solvent molecules, the use of M-SHAKE for the solvent molecules would permit this increase in precision for little additional computational costs.

References

1. van Gunsteren, W. F.; Berendsen, H. J. C. Mol Phys 1977, 34, 1311.
2. van Gunsteren, W. F. Mol Phys 1980, 40, 1015.
3. van Gunsteren, W. F.; Karplus, M. Macromolecules 1982, 15, 1528.
4. Tironi, I. G.; Brunne, R. M.; van Gunsteren, W. F. Chem Phys Lett 1996, 250, 19.
5. Ryckaert, J.-P.; Ciccotti, G.; Berendsen, H. J. C. J Comput Phys 1977, 23, 327.
6. Miyamoto, S.; Kollman, P. A. J Comput Chem 1992, 13, 952.
7. Hess, B.; Bekker, H.; Berendsen, H. J. C.; Fraaije, J. G. E. M. J Comput Chem 1997, 18, 1463.
8. Andersen, H. C. J Comput Phys 1983, 52, 24.
9. Hockney, R. W.; Eastwood, J. W. Computer Simulation Using Particles; Institute of Physics Publishing: Bristol, 1981.
10. Golub, G. H.; van Loan, C. F. Matrix Computations; Johns Hopkins University Press: London, 1996.
11. Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; Hermans, J. In Intermolecular Forces; Pullman, B., Ed.; Reidel, Dordrecht: The Netherlands, 1981, p. 331.

12. Liu, H.; Müller-Plathe, F.; van Gunsteren, W. F. J Am Chem Soc 1995, 117, 4363.
13. Tironi, I. G.; van Gunsteren, W. F. Mol Phys 1994, 83, 381.
14. Dietz, W.; Heinzinger, K. Ber Bunsenges Chem 1984, 88, 543.
15. Dietz, W.; Heinzinger, K. Ber Bunsenges Chem 1985, 89, 968.
16. Barker, J. A.; Watts, R. O. Mol Phys 1973, 26, 789.
17. Tironi, I. G.; Sperb, R.; Smith, P. E.; van Gunsteren, W. F. J Chem Phys 1995, 102, 5451.
18. Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; DiNola, A.; Haak, J. R. J Chem Phys 1984, 81, 3684.
19. van Gunsteren, W. F.; Billeter, S. R.; Eising, A. A.; Hünenberger, P. H.; Krüger, P.; Mark, A. E.; Scott, W. R. P.; Tironi, I. G. Biomolecular Simulation: The GROMOS96 Manual and User Guide; Verlag der Fachvereine: Zürich, 1996.
20. Pearlman, D. A.; Case, D. A.; Caldwell, J. D.; Ross, W. S.; Cheatham, T. E., III; DeBolt, S.; Fergusson, D.; Seibel, G.; Kollman, P. Comput Phys Commun 1995, 91, 1.