ORAC: A Molecular Dynamics Program to Simulate Complex Molecular Systems with Realistic Electrostatic Interactions

PIERO PROCACCI,[1*] TOM A. DARDEN,[2] EMANUELE PACI,[3†] MASSIMO MARCHI[3]

[1]*Centre Européen de Calcul Atomique et Moleculaire (CECAM) Ecole Normale Superieure de Lyon, Lyon, France*
[2]*National Institute of Environmental Health, Sciences Research, Triangle Park, North Carolina*
[3]*Section de Biophysique des Protéines et des Membranes, DBCM, DSV, CEA, Centre d'Études, Saclay, 91191 Gil-sur-Yvette Cedex, France*

Received 25 February 1997; accepted 13 June 1997

ABSTRACT: In this study, we present a new molecular dynamics program for simulation of complex molecular systems. The program, named ORAC, combines state-of-the-art molecular dynamics (MD) algorithms with flexibility in handling different types and sizes of molecules. ORAC is intended for simulations of molecular systems and is specifically designed to treat biomolecules efficiently and effectively in solution or in a crystalline environment. Among its unique features are: (i) implementation of reversible and symplectic multiple time step algorithms (or r-RESPA, reversible reference system propagation algorithm) specifically designed and tuned for biological systems with periodic boundary conditions; (ii) availability for simulations with multiple or single time steps of standard Ewald or smooth particle mesh Ewald (SPME) for computation of electrostatic interactions; and (iii) possibility of simulating molecular systems in a variety of thermodynamic ensembles. We believe that the combination of these algorithms makes ORAC more advanced than other MD programs using standard simulation algorithms. © 1997 John Wiley & Sons, Inc. *J Comput Chem* **18**: 1848–1862, 1997

Keywords: molecular dynamics; biomolecules; electrostatics; software; reversible multiple time-step algorithms

Correspondence to: M. Marchi

*Permanent address: Dipartimento di Chimica, Università degli Studi di Firenze, 50120 Firenze, Italy

[†]Present address: Laboratoire de Chimie Biophysique, Institut Le Bel, Université Louis Pasteur, 67000 Strasbourg, France

This article includes Supplementary Material available from the authors upon request or via the Internet at ftp.wiley.com/public/journals/jcc/suppmat/18/1848 or http://journals.wiley.com/jcc/

CCC 0192-8651 / 97 / 151848-15

Introduction

The complexity of the system studied by molecular dynamics (MD) simulations has increased steadily since the early 1960s. From the first pioneering simulations of a few hundred Lennard–Jones particles carried out by Rahman[1] and Verlet[2] on large mainframe computers, computer technology has evolved considerably to allow today routine large-scale simulations of 10,000-particle systems on personal workstations. Following the early work on atomic liquids, the MD technique has been extended to the study of molecular liquids, molecular solids, and finally biomolecular systems. In these latter systems, many important events occur normally on a much longer time scale than that typical of homogeneous liquids or solids. Moreover, due to the polar and charged nature of biomolecules and of their natural environment (ionic solutions) long-range electrostatic interactions play a fundamental role in determining their structure and dynamics. Unfortunately, the need to compute accurately the long-range tails of the electrostatic interactions and to sample longer time scales greatly increase the computational task to be carried out by MD simulation.

Nevertheless, along with the increase in computer power, novel computational techniques have been proposed in the past to significantly improve simulation efficiency. Reversible and symplectic multiple time step algorithms (or r-RESPA, reversible reference system propagation algorithm), developed by Berne and coworkers,[3–7] considerably increase the simulation time step by subdividing the system into a series of subsystems of increasingly longer time scale. On the other hand, fast multiple methods[8, 9] and particle mesh Ewald (PME) techniques[10, 11] substantially speed up the computation of long-range electrostatic interactions in large systems. Recently, an algorithm that combines r-RESPA and PME methods has appeared.[12]

This article describes in detail our MD program which implements, along with more standard simulation algorithms, the techniques just mentioned. The program, named ORAC,* is intended for simulations of molecular systems and is specifically designed for simulation of biomolecules in solution or in a crystalline environment. ORAC has been written to be a "number cruncher" and does not provide a graphical interface or sophisticated facilities to analyze trajectories produced by MD simulations.

Among the unique features of ORAC is the availability of multiple time step r-RESPA algorithms specifically designed and tuned for biological systems with periodic boundary conditions. These integration algorithms can be used either with standard Ewald summation[7] or with the smooth particle mesh Ewald (SPME) technique.[11] In our implementation of combined r-RESPA and SPME,[12] no degrees of freedom of the system are constrained, and the force associated with each particle is partitioned into five components, which evolve in time with distinct and increasingly longer time scales. A suitable time scale separation is achieved by subdividing the direct space nonbonded interactions, including Coulombic and Van der Waals contributions, into short-, medium-, and long-range shells. Bonded forces are associated with the two shorter time scales, whereas the reciprocal space nonbonded interaction is included with the medium range direct space forces.

When combined with the standard Ewald summation technique the algorithm achieves up to a Four-fold reduction in computational cost for system as large as 20,000 atoms with respect to a standard algorithm involving constraints. If standard Ewald is replaced by SPME speed-ups up to one order of magnitude can be achieved with respect to simulations employing standard Ewald and rigid bond constraints. When compared to simulations making use of constraints and a spherical cutoff at 10 Å (the radius of truncation of the direct space sum) our combination of r-RESPA and SPME is about 2.5 times faster. Although other multiple time step approaches such as the "twin-range" method[13] have previously been implemented in MD programs for biomolecules, this class of methods, in contrast with r-RESPA techniques, alter the sampling distribution function of the simulation in an uncontrolled way. Thus, the availability in ORAC of efficient techniques such as r-RESPA and SPME for simulating large systems eliminates the need for uncontrolled approximations such as the spherical potential truncation, constraint dynamics, and twin-range techniques. Unphysical consequences of these approximations have been reported many times.[5, 7, 14, 15]

The SPME technique can also be used in ORAC to handle electrostatics in any ensemble, while at present r-RESPA is implemented only for NVE

*ORAC is distributed free of charge to academia. Information on how to obtain the code can be found on the web site http://www.cecam.fr/orac.

ensemble simulations. Because algorithms for NPH ensemble simulation already exist in the literature,[16] extension of r-RESPA to ensembles other than microcanonical is straightforward and its implementation in ORAC is planned for the near future.

As to more "standard" MD features ORAC can perform simulations in the microcanonical (NVE), canonical (NVT), isobaric (NPH), and isothermal–isobaric (NPT) ensembles. Constant pressure and constant temperature are implemented using the so-called extended Lagrangian methods.[17–19] For NPH, NVT, and NPT simulations, extended Lagrangians are designed to yield at equilibrium the correct distribution function of the corresponding ensemble. We stress that the extended systems method, routinely employed in the study of liquids and solids, has been rarely used in the simulation of biomolecules, for which simpler and less rigorous techniques have been normally favored.[13] ORAC's force routines, where most of the CPU time is spent, have been carefully optimized for both vector and scalar machines, and linked cell neighbor lists can be used for systems of large size. The standard part of the program is thus both modern and efficient and, in our experience, the program compares favorably with other popular MD packages such as CHARMM[20] or AMBER,[21,22] when using standard algorithms.

The general features of ORAC are described in the Supplementary Material, which also contains an analysis of a practical example: the preparation and simulation of a bovine pancreatic trypsin inhibitor molecule in water solution.

The Simulated System

INTERACTION POTENTIAL

ORAC considers two molecular types: (i) solvent molecules, ideally with only a few internal degrees of freedom, which are small in size and present in great numbers within the system; and (ii) solute molecules, large in size and having a complicated topology, their prototype being a large macromolecule (i.e., a protein or a DNA molecule). Although this subdivision is rather artificial, it was created to obtain the best computational performance on vector supercomputers. On machines such as the Cray C90, the subdivision between solute and solvent is very useful, allowing excellent performance, especially when simulating solvated macromolecules.

Solvent

In this version of ORAC, the internal degrees of freedom of the solvent molecules are limited to bond stretching and angle bendings. In addition to bond constraints, generalized bond constraints[23] can also be enforced in solvent molecules. This feature can be useful to handle model potentials of rigid molecules involving interaction sites displaced with respect to atomic sites (e.g., the TIP4P model for water[24]). The intermolecular interaction potential of the solvent includes electrostatic terms, modeled by point charges (PC) on specific atomic sites, and a Lennard–Jones (LJ) dispersion–repulsion term. The PC/LJ representation is widely used by the most popular force fields for simulation of simple liquids, protein, DNA, and phospholipids.

Thus, the total interaction potential acting among solvent molecules can be written as:

$$\begin{aligned} V_{slv} = \sum_{\alpha=1,N} \Bigg\{ & \sum_{\text{Bonds}} K_r (r_\alpha - r_0)^2 \\ & + \sum_{\text{Angles}} K_\theta (\theta_\alpha - \theta_0)^2 \qquad (1) \\ & + \sum_{\beta=\alpha+1,N} \sum_{i=1,n_\alpha} \sum_{j=1,n_\beta} 4\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{i\alpha,j\beta}} \right)^{12} \right. \\ & \left. - \left(\frac{\sigma_{ij}}{r_{i\alpha,j\beta}} \right)^{6} \right] + V_{el}(r_{i\alpha,j\beta}) \Bigg\} \qquad (2) \end{aligned}$$

Here, α and β are the molecular indices, N is the total number of solvent molecules, and n_a is the number of atoms per molecule. K_r and K_θ are the stretching and bending force constants, and r_0 and θ_0 are the equilibrium bond distance and angle, respectively. In addition, $\sigma_{i,j}$ and $\epsilon_{i,j}$ are the Lennard–Jones parameters for the coupling of $i\alpha$th and $j\beta$th atoms. Finally, V_{el} is the electrostatic potential. Its detailed description is provided later in this work.

The Lennard–Jones interaction parameters between atoms of different types (cross-parameters) are calculated by default from standard mixing rules, namely:

$$\begin{aligned} \sigma_{ij} &= \frac{1}{2}(\sigma_i + \sigma_j) \\ \epsilon_{ij} &= \sqrt{\epsilon_i \epsilon_j} \qquad (3) \end{aligned}$$

where σ_i and ϵ_i are the Lennard–Jones parameters of the ith atom. Arbitrary cross-parameters may also be provided.

Solute

Solute molecules are flexible and can have rigid bonds. The interaction among solute atoms is described by a general potential function similar to that used by the CHARMM and AMBER force fields and given by:

$$V_{tot} = V_{nb} + V_b \tag{4}$$

$$V_{nb} = V_{vdw} + V_{el} \tag{5}$$

$$V_b = V_{stretch} + V_{bend} + V_{14} + V_{tors} \tag{6}$$

The first term in eq. (4)a is the nonbonded potential energy, which includes [eq. (5)], Van der Waals and electrostatic energy terms similar to the solvent. The bonded potential energy, V_b, in eq. (6) is composed of $V_{stretch}$, V_{bend}, and V_{tors} due to stretching, bending, and torsion interactions, respectively. The bond stretchings and angle bending terms are given by:

$$V_{stretch} = \sum_{\text{Bonds}} K_r(r - r_0)^2 \tag{7}$$

$$V_{bend} = \sum_{\text{Angles}} K_\theta(\theta - \theta_0)^2 \tag{8}$$

where K_r and K_θ are the bonded force constants associated with bond stretching and angle bending, respectively, and r_0 and θ_0 are their respective geometrical reference values.

The term V_{14} is typical of biomolecular force fields. In general, it is a nonbonded interaction between two atoms separated by three consecutive bonds. This interaction, composed of an electrostatic and a Lennard–Jones term, is appropriately reduced by force-field-specific multiplicative factors. In ORAC, V_{14} is computed as:

$$V_{14} = V_{14}^{el}C_{14}^{el} + V_{14}^{LJ}C_{14}^{LJ} \tag{9}$$

where V_{14}^{el} and V_{14}^{LJ} are standard electrostatic and Lennard–Jones energy terms, respectively. The two multiplicative factors can assume values between 0 and 1.

Torsion potentials are of two types: the so-called *improper* torsion potential and the normal or *proper* torsion potential. They have the following form:

$$V_{tors} = V_{p\text{-}tors} + V_{i\text{-}tors} \tag{10}$$

$$V_{p\text{-}tors} = \sum_{\text{Proper}} K_\phi[1 + \cos(n\phi - \gamma)] \tag{11}$$

$$V_{i\text{-}tors} = C_1 \sum_{\text{Improper}} K_\chi(\chi - \chi_0)^2 \tag{12}$$

$$+(1 - C_1) \sum_{\text{Improper}} K_\chi[1 + \cos(n\chi - \gamma)] \tag{13}$$

where ϕ and χ are the dihedral angles and K_ϕ, K_χ, n, γ, and χ_0 are constants. C_1 can be either $C_1 = 1$ for CHARMM-like or $C_1 = 0$ for AMBER-like force fields. ORAC can simulate systems containing any type of solute molecules and as many identical solute molecules as needed. Also, bond stretchings and angle bendings can be kept rigid by standard bond constraints.

COORDINATES AND BOUNDARY CONDITIONS

ORAC simulates infinite systems by applying standard periodic boundary conditions (PBC). Following a technique developed for simulation of molecular liquids, ORAC works in scaled coordinates, which allow for a natural and simple way to handle PBC.

If a, b, c, α, β, and γ are the cell parameters of the real simulation box, the transformation from scaled coordinates to real coordinates can be carried out by the matrix **h**:

$$\mathbf{h} = \frac{1}{2}\begin{pmatrix} a & b\cos(\gamma) & c\cos(\beta) \\ 0 & b\sin(\gamma) & c[\cos(\alpha) - \cos(\beta)\cos(\gamma)]/\sin(\gamma) \\ 0 & 0 & (c/\sin(\gamma))\sqrt{\sin^2(\beta)\sin^2(\gamma) - (\cos\alpha - \cos\beta\cos\gamma)^2} \end{pmatrix} \tag{14}$$

Conversely, $\mathbf{h}^{-1}$ transforms real coordinates to scaled coordinates. With the chosen **h** the scaled box has a length of 2 and PBC can be computed using the generic FORTRAN function ANINT or faster alternatives.[25] Because ORAC always employs scaled coordinates, simulation of finite systems, such as clusters or isolated proteins, can be accomplished using a very large simulation cell to set to zero the interactions among images.

Long-Range Interactions

ORAC can handle long-range electrostatic interactions by three techniques: spherical truncation; the standard Ewald summation; and the smooth particle mesh Ewald (SPME) method. In the following subsections, we provide a general description of these techniques.

SPHERICAL CUTOFF

So far, the great majority of protein simulation studies have employed a spherical cutoff of the nonbonded interactions. Generally, the nonbonded interactions are spherically truncated at a distance of the order of 9–10 Å. To improve the convergence of the Coulombic term, which decays with the distance r as $1/r$, ORAC imposes a cutoff between atomic groups rather than atom pairs. Such groups are defined as having a total charge of approximately zero. Thus, the Coulombic term acquires an r dependence of $1/^{r3}$ due to the interactions between the dipoles of the two separate groups. The total nonbonded potential can then be written:

$$V = \sum_{\alpha<\beta}\sum_{ij} V(r_{i\alpha,j\beta}) + \sum_{\alpha}\sum_{ij} V(r_{i\alpha,j\alpha}) \tag{15}$$

where α and β run on the groups of the protein and i and j labels the atoms belonging to each group.

Because an abrupt cutoff in the potential corresponds to a discontinuity in the potential function itself and its derivatives, an MD simulation carried out in these conditions will not conserve energy. Thus, ORAC uses a so-called switching function (a third-order spline) to smoothly switch the potential to zero. The function used is defined as:

$$S(r, r_{on}, r_{off}) = \begin{cases} 1 \qquad r \le r_{on} \\ \dfrac{(r_{off}-r)^2(r_{off}+2r-3r_{on})}{(r_{off}-r_{on})} \\ \qquad r_{on} \le r \le r_{off} \\ 0 \qquad r \ge r_{off} \end{cases} \tag{16}$$

The function is one before and at r_{on} and becomes zero at r_{off}. Thus, in case of cutoff on groups of atoms, eq. (15) becomes:

$$V = \sum_{\alpha<\beta}\left[\sum_{ij} V(r_{\alpha i,\beta j})\right] S(r_{\alpha,\beta}, r_{on}, r_{off}) + \sum_{\alpha}\sum_{ij} V(r_{\alpha i,\alpha j}) \tag{17}$$

Because of the availability of the SPME and r-RESPA methods, the simple spherical truncation for electrostatic systems under PBC is becoming an unnecessary option of ORAC. Still, cutoff truncation remains the only possibility in ORAC for LJ systems and can be useful for cluster dynamics, for testing purposes or for periodic polar systems, where Ewald summation is less critical.

STANDARD EWALD SUMMATION

For periodic systems, the total electrostatic energy is given by:

$$V_{el} = \sum_{\mathbf{n}}{}' \sum_{ij} \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{r}_n|^2} \tag{18}$$

where $\mathbf{r}_n$ is a generic point of the direct lattice and the prime indicates that the term $i = j$ and $\mathbf{n} = 0$ is omitted. The sum in eq. (18) can be calculated exactly using the Ewald method.[26]

The problem of computing the sum in eq. (18) is solved in this technique by multiplying the general term by a Gaussian convergence factor $e^{-s^2 r_n^2}$ and by splitting the resulting sum into two absolutely convergent sums, one in the direct lattice and the other in the reciprocal lattice. Taking the limit $s \to 0$[27] and exploiting electroneutrality to eliminate the singularity in the reciprocal lattice at $\mathbf{k} = 0$, one obtains the following expression for the electrostatic energy of eq. (18):

$$V_{el} = V_{qd} + V_{qr} \tag{19}$$

where V_{qd} and V_{qr} are the direct and reciprocal space components of the electrostatic energy, defined as:

$$V_{qd} = \frac{1}{2}\sum_{ij}^{N} q_i q_j \sum_{\mathbf{n}}{}' \frac{1}{|\mathbf{r}_{ij} + \mathbf{r}_n|} erfc\left(\alpha|\mathbf{r}_{ij} + \mathbf{r}_n|\right) - \frac{\alpha}{\pi^{1/2}}\sum_i q_i^2 \tag{20}$$

$$V_{qr} = \frac{1}{2\pi V}\sum_{\mathbf{m}\neq 0}^{\infty}\frac{\exp\left(-\pi^2|\mathbf{m}|^2/\alpha^2\right)}{|\mathbf{m}|^2} \times S(\mathbf{m})S(-\mathbf{m}) - V_{intra} \tag{21}$$

Here, *erfc* is the complementary error function, V is the unit cell volume, $\mathbf{m}$ is a reciprocal lattice vector, and α is an arbitrary parameter. In eq. (21), $S(\mathbf{m})$ is the charge-weighted structure factor defined as:

$$S(\mathbf{m}) = \sum_i^N q_i \exp(2\pi i \mathbf{m}\cdot\mathbf{r}_i) \tag{22}$$

Finally, the term V_{intra} is given by:

$$V_{intra} = \sum_{ij} q_i q_j \frac{erf(\alpha r_{ij})}{r_{ij}} \tag{23}$$

where the sum over i and j includes all excluded bonded contacts. This last term subtracts, in direct space, the intramolecular energy between bonded pairs, which is automatically included in the right-hand side of eq. (21).

In the Ewald method the electrostatic potential is calculated with a constant accuracy (usually several orders of magnitude higher than that of spherical-cutoff based methods) everywhere in the sample, irrespective of the charge distribution. Thus, this method is rigorous, but, unfortunately, more expensive than spherical truncation. When implemented by means of eqs. (20) and (21), the computational cost of the Ewald sum grows with $N^{3/2}$, with N being the number of charges in the system.

In simulation of solids, periodic boundary conditions reflect the physical nature of the systems. On the other hand, for liquids or proteins in solutions, PBC must be considered as a simple device to eliminate boundary effects and, certainly, enhances beyond physical reality the periodic nature of the systems. Nonetheless, for samples of large size, the effects of this artificial periodicity have been found to be negligible.[10,14,15,28]

SMOOTH PARTICLE MESH EWALD

For increasingly large systems the computational cost of standard Ewald summation quickly becomes too great for practical applications. Alternative algorithms which scale with a smaller power of N than standard Ewald have been proposed in the past. Among the faster algorithms designed for periodic systems is the particle mesh Ewald (PME) technique,[10,11] inspired by the particle mesh method of Hockney and Eastwood.[29] Here, a multidimensional piecewise interpolation approach is used to compute the reciprocal lattice energy, V_{qr}, of eq. (21), whereas the direct part, V_{qd}, is computed in a standard way according to eq. (20). The low computational cost of the PME method allows the choice of large values of the Ewald convergence constant α with respect to those used in conventional Ewald. This leads to shorter cutoffs in the direct space Ewald sum, V_{qd}. If $\mathbf{u}_\mathrm{j}$ is the scaled fractional coordinate of the ith particle, the charge-weighted structure factor, $S(\mathbf{m})$, in eq. (22), can be rewritten as:

$$S(\mathbf{m}) = \sum_1^N q_j \exp\left[2\pi i\left(\frac{m_1 u_{j1}}{K_1} + \frac{m_2 u_{2j}}{K_2} + \frac{m_3 u_{3j}}{K_3}\right)\right] \tag{24}$$

where K_1, K_3, K_3 and m_1, m_2, m_3 are integers. The α component of the scaled fractional coordinate for the ith atom can be written as[†]:

$$u_{i\alpha} = K_\alpha \mathbf{k}_\alpha \cdot \mathbf{r}_\mathbf{i} \tag{25}$$

where $\mathbf{k}_\alpha$ $(\alpha = 1, 2, 3)$ are the reciprocal lattice basic vectors.

$S(\mathbf{m})$ in eq. (24) can be looked at as a discrete Fourier transform of a set of charges placed irregularly within the unit cell. Techniques have been devised in the past to approximate $S(\mathbf{m})$ with expression involving Fourier transforms on a regular grid of points. Such approximations of the weighted structure factor are computationally advantageous because they can be evaluated by fast Fourier transforms (FFT). All these FFT-based approaches involve, in some sense, a smearing of the charges over nearby grid points to produce a regularly gridded charge distribution. The PME method accomplishes this by interpolation. Thus, the complex exponentials, $\exp(2\pi i m_\alpha u_{i\alpha}/K_\alpha)$, computed for the ith charge in eq. (24), are rewritten as a sum of interpolation coefficients multiplied by their values at the nearby grid points. In the smooth version of PME (SPME),[11] which uses cardinal B-splines in place of the Lagrangian coefficients adopted by PME, the sum is additionally multiplied by an appropriate factor, namely:

$$\exp(2\pi i m_\alpha u_{i\alpha}/K_\alpha) = b(m_\alpha)\sum_k M_n(u_{i\alpha} - k)\exp(2\pi i m_\alpha k/K_\alpha) \tag{26}$$

[†] The scaled fractional coordinate is related to the scaled coordinates given in the subsection, "Coordinates and Boundary Conditions," by the relation $s_{i\alpha} = 2\frac{u_{i\alpha}}{K_\alpha}$.

where n is the order of the spline interpolation, and $M_n(u_{i\alpha} - k)$ defines the coefficients of the cardinal B-spline interpolation at the scaled coordinate $u_{i\alpha}$. In eq. (26), the sum over k, representing the grid points, is only over a finite range of integers, because the functions $M_n(u)$ are zero outside the interval $0 \le u \le n$. It must be stressed that the complex coefficients $b(m_\alpha)$ are independent of the charge coordinates $\mathbf{u_i}$ and need be computed only at the very beginning of a simulation. A detailed discussion of the $M_n(u)$ functions and of the b_α coefficients is given in Essmann et al.[11]

By inserting eq. (26) into eq. (24), $S(\mathbf{m})$ can be approximated as:

$$S(\mathbf{m}) = b_1(m_1)b_2(m_2)b_3(m_3)\mathscr{F}[Q](m_1, m_2, m_3) \tag{27}$$

where $\mathscr{F}[Q](m_1, m_2, m_3)$ stands for the discrete FT at the grid point m_1, m_2, m_3 of the array $Q(k_1, k_2, k_3)$, with $1 \le k_i \le K_i$ $(i = 1, 2, 3)$. The gridded charge array, $Q(k_1, k_2, k_3)$, is defined as:

$$Q(k_1, k_2, k_3) = \sum_{i=1,N} q_i M_n(u_{i1} - k_1) M_n(u_{i2} - k_2) \times M_n(u_{i3} - k_3) \tag{28}$$

Inserting the approximated structure factor of eq. (27) into eq. (21), and using the fact that $\mathscr{F}[Q](-m_1, -m_2, -m_3) = K_1 K_2 K_3 \mathscr{F}^{-1}[Q](m_1, m_2, m_3)$, the SPME reciprocal lattice energy can then be written as:

$$\begin{aligned} V_{qr} &= \frac{1}{2} \sum_{m_1=1}^{K_1} \sum_{m_2=1}^{K_2} \sum_{m_3=1}^{K_3} B(m_1, m_2, m_3) \\ &\quad \times C(m_1, m_2, m_3) \\ &\quad \times \mathscr{F}[Q](m_1, m_2, m_3) \\ &\quad \times \mathscr{F}[Q](-m_1, -m_2, -m_3) \qquad (29) \\ &= \frac{1}{2} \sum_{m_1=1}^{K_1} \sum_{m_2=1}^{K_2} \sum_{m_3=1}^{K_3} \mathscr{F}^{-1}[\Theta_{rec}](m_1, m_2, m_3) \\ &\quad \times \mathscr{F}[Q](m_1, m_2, m_3) \\ &\quad \times K_1 K_2 K_3 \mathscr{F}^{-1}[Q](m_1, m_2, m_3) \qquad (30) \end{aligned}$$

with:

$$B(m_1, m_2, m_3) = |b_1(m_1)|^2 |b_2(m_2)|^2 |b_3(m_3)|^2 \tag{31}$$

$$C(m_1, m_2, m_3) = (1/\pi V)\exp(-\pi^2 \mathbf{m}^2/\alpha^2)/\mathbf{m}^2 \tag{32}$$

$$\Theta_{rec} = \mathscr{F}[BC] \tag{33}$$

We first notice that Θ_{rec} does not depend on the charge positions and that $M_n(u_{i\alpha} - k)$ is differentiable for $n > 2$, which is always the case in practical applications. Thus, the force on each charge can be obtained by rearranging eq. (29) as a product of Q times the convolution $Q * \Theta_{rec}$ and by taking its derivative, namely:

$$\begin{aligned} F_{i\alpha}^{(qr)} &= -\frac{\partial V_{qr}}{\partial r_{i\alpha}} \\ &= \sum_{m_1=1}^{K_1} \sum_{m_2=1}^{K_2} \sum_{m_c3=1}^{K_3} \frac{\partial Q(m_1, m_2, m_3)}{\partial r_{i\alpha}} \\ &\quad \times (\theta_{rec} * Q)(m_1, m_2, m_3) \qquad (34) \end{aligned}$$

ORAC performs the calculation of the reciprocal lattice energy and its corresponding forces using the original SPME code and implementation described by Essmann et al.[11] In practice, the calculation is carried out according to the following scheme: (i) At each simulation step ORAC computes the grid scaled fractional coordinates $u_{i\alpha}$ and, subsequently, Q according to eq. (28). At this stage, the derivative of the M_n functions are also computed and stored in memory. (ii) The array containing Q is then overwritten by $\mathscr{F}[Q]$, Q's three-dimensional Fourier transform. (iii) The electrostatic energy is then computed via eq. (30). At the same time, the array containing $\mathscr{F}[Q]$ is overwritten by the product of itself with the array containing BC (computed at the very beginning of the run). (iv) The resulting array is then Fourier transformed to obtain the convolution $\theta_{rec} * Q$. (v) Finally, the forces are computed via eq. (34) using the previously stored derivatives of the M_n functions to recast $\partial Q/\partial r_{i\alpha}$.

The memory requirement of the SPME method are limited. $2K_1K_2K_3$ double precision reals are needed for the grid charge array Q, whereas the calculation of the functions $M_n(u_{i\alpha} - j)$ and their derivatives requires only $6 \times n \times N$ double precision real numbers. The K_α integers determine the fineness of the grid along the αth lattice vector of the unit cell. The output accuracy of the energy and forces depends on the SPME parameters: the α convergence parameter, the grid spacing, and the order n of the B-spline interpolation. For a typical $\alpha \simeq 0.4$ Å^{-1} relative accuracies between 10^{-4} and 10^{-5} for the electrostatic energy are obtained when the grid spacing is around 1 Å along each axis, and the order n of the B-spline interpolation is 4 or 5. A rigorous error analysis and a comparison with regular Ewald summation can be found in Refs. 11 and 30. For further information on the PME and SPME techniques we refer to the original works.[10,11,14,30]

ORAC r-RESPA Implementation

The multiple time-step integration technique adopted in ORAC is based on the reversible version[3] of the reference system propagator algorithm (r-RESPA) developed by Berne and coworkers.[3,6,31–33] The interaction potential V is appropriately decomposed in m part $V = V_1 \ldots V_m$, corresponding to m reference systems, each of which has a characteristic time scale. This subdivision is exploited in the numerical integration of the equations of motion by associating an increasingly large time step to forces of increasingly slow time scales. Although the r-RESPA integration algorithm in ORAC has been specifically optimized for the simulation of solvated proteins,[12] its implementation is general. r-RESPA can thus be efficiently appli ed to any system from liquid argon to large biomolecular systems.

POTENTIAL SUBDIVISION

ORAC allows up to the fifth-order potential subdivision. For all systems, the potential breakup is as follows:

$$V_{n0} = V_{stretch} + V_{bend} + V_{i\text{-}tor} \tag{35}$$

$$V_{n1} = V_{p-tors} + V_{14} \tag{36}$$

$$V_m = V_{vdw}^{(m)} + V_{qd}^{(m)} \tag{37}$$

$$V_l = V_{vdw}^{(l)} + V_{qd}^{(l)} + V_{qr} \tag{38}$$

$$V_h = V_{vdw}^{(h)} + V_{qd}^{(h)} \tag{39}$$

where the superscripts m, l, h of the direct space term V_{vdw} and V_{qd} refer to the short-medium-, and long-range nonbonded interactions, respectively. The fastest term $V_{\mathrm{n}0}$ includes bending, stretching, and *improper* torsional potentials.[‡] The proper torsion term along with all 1–4 nonbonded interactions are instead assigned to the slower intramolecular potential V_{n1}. The subdivision for bonded interactions in eq. (35)–(37) is maintained in the r-RESPA algorithm even if one or more potential terms are zero. This is the case, for instance, in the simulation of a protein, where V_{n1} is nonzero only for intraprotein interactions.

The mth reference system includes nonbonded direct space interactions at short range, typically between 0 and 4.3–5.3 Å. V_l contains both the medium-range direct space potential, with a typical range of 4.3–5.3 to 6.9–7.3 Å, and the reciprocal space term V_{qr}. Finally, the hth reference system, which is the most slowly varying, contains the remaining direct space interactions from 6.9 to 7.3 Å to cutoff distance. This direct space separation for the nonbonded interactions is carried out by multiplying the full potential by switching functions between pairs of atomic groups.[6,7,32] Thus, at any distance R, the potential V can be written schematically as:

$$V(R) = V_m(R) + V_l(R) + V_h(R) \tag{40}$$

with:

$$V_m(R) = V(R) S_m(R) \tag{41}$$

$$V_l(R) = V(R)[S_l(R) - S_m(R)] \tag{42}$$

$$V_h(R) = V(R)[S_h(R) - S_l(R)] \tag{43}$$

$$\tag{44}$$

where S_j is the switching function for the three shells, $j = m, l, h$, defined as:

$$S_j(R) = \begin{Bmatrix} 1 & R_{j-1} \le R < R_j \\ S_{3p}^{(j)} & R_j \le R < R_j + \lambda_j \\ 0 & R_j + \lambda_j < R \end{Bmatrix} \tag{45}$$

Here, R is the intergroup distance and λ_j is the healing interval for the jth shell. While R_0 is zero, $R_1 = R_m$, $R_2 = R_l$, and $R_3 = R_h$ are the short-, medium-, and long-range shell radii, respectively. The switching function $S_{3p}^{(j)}(R)$ is 1 at R_j and goes monotonically to 0 at $R_j + \lambda_j$. Provided that $S_{3p}^{(j)}$ and its derivatives are continuous at R_j and $R_j + \lambda_j$, the analytical form of S_{3p} in the healing interval is arbitrary. ORAC uses a third-order spline function.[7] In Figure 1 we illustrate the use of the switching functions defined in eq. (45) for the breakup of an LJ typical potential $V(R)$. The cutoffs of the three shells are $R_m = 3.5$ Å, $R_l = 6.5$ Å, and $R_h = 9.5$ Å, and in all three cases the healing length is $\lambda = 0.5$ Å.

[‡] Improper torsions in proteins involving more than one hydrogen atom exceed 1000 cm^{-1} in both AMBER[22] and CHARMM[20,34] force fields.

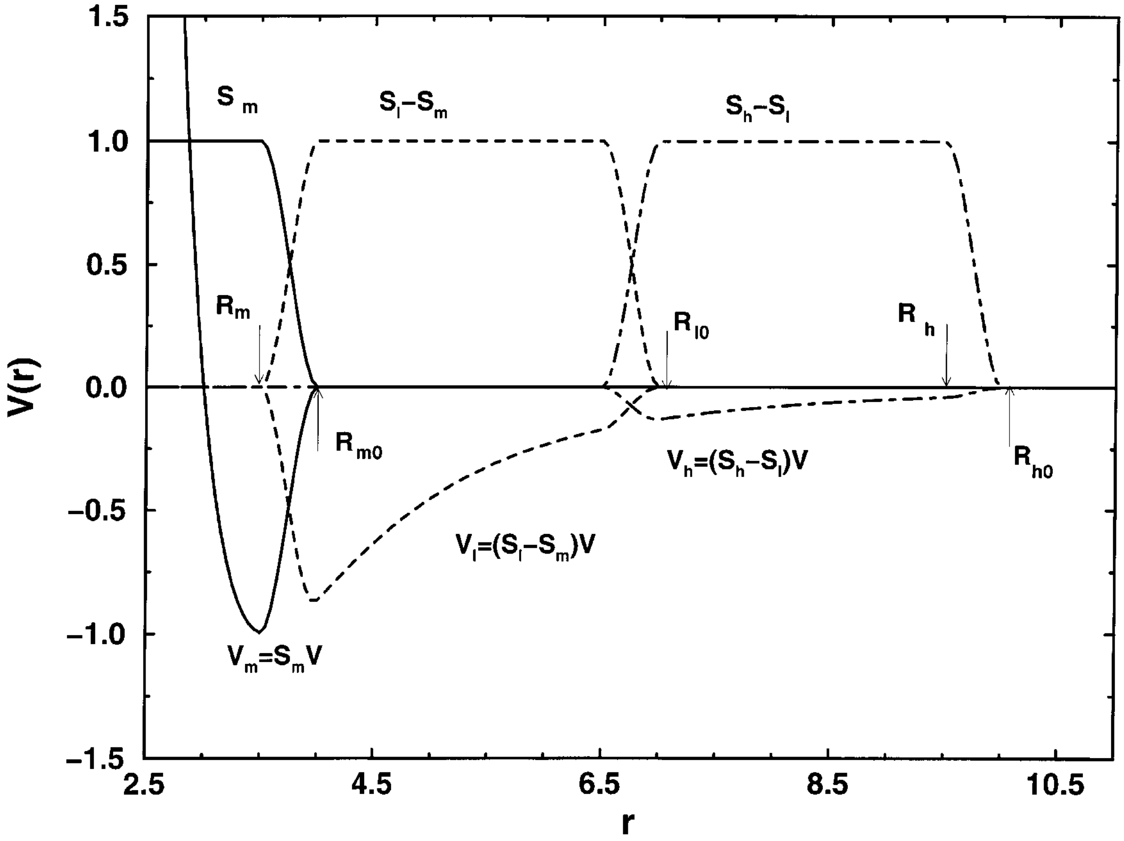

FIGURE 1. Subdivision for a Lennard–Jones function V by means of the switching functions S_m, S_l, and S_h in short-, medium-, and long-range parts, V_m, V_l, and V_h, respectively. The healing lengths are defined as $\lambda_i = R_i - R_{i0}$, $i = l, m, h$ (see text).

The time scale of V_{qr} depends on the α convergence parameter of the Ewald sum: The larger the α (large α values are very suitable to SPME) the more "short ranged" V_{qr} becomes and the faster it varies with time. The assignment of V_{qr} to a specific shell also depends on the size of the time steps chosen for the short, medium, and long range. In the subdivision defined in eqs. (35)–(36), the reciprocal lattice term, V_{qr}, was assigned to the medium shell based on our previous experience. Indeed, algorithms based on this subdivision have been thoroughly tested and studied in Procacci et al.[12], where the "best" numerical parameters for solvated proteins have also been determined.

INTEGRATION SCHEME

It has been shown in the past[3,6,16,35–37] that multiple time-step algorithms that are reversible and symplectic[38,39] can be easily derived by application to the phase space vector $\{\mathbf{p}, \mathbf{q}\}$ of an appropriately partitioned and discretized classical propagator, $e^{iL\Delta t}$, which propagates the microscopic systems for the time slice Δt. Here, L is the Liouvillean of the system, defined as:

$$iL = \sum_i^N \left[\dot{\mathbf{r}}_i \frac{\partial}{\partial \mathbf{r}_i} + \mathbf{f}_i \frac{\partial}{\partial \mathbf{p}_i} \right] \tag{46}$$

where $\mathbf{r}_i$ and $\mathbf{p}_i$ are, respectively, the coordinates and the momenta of the particles in the system, whereas $\dot{\mathbf{r}}_i$ and $\mathbf{f}_i$ are the corresponding velocities and forces. For the subdivision found in eqs. (35)–(39) there is a corresponding partition of the Liouvillean, iL. Thus, the classical propagator can be rewritten as:

$$e^{iL\Delta t} = e^{(iL_{n0} + iL_{n1} + iL_m + iL_l + iL_h)\Delta t} \tag{47}$$

with:

$$iL_{n0} = \sum_i^{3N} \dot{q} \frac{\partial}{\partial q_i} - \frac{\partial V_{n0}}{\partial q_i} \frac{\partial}{\partial p_i}$$

$$
\begin{aligned}
iL_{n1} &= -\sum_{i}^{3N} \frac{\partial V_{n1}}{\partial q_i} \frac{\partial}{\partial p_i} \\
&\ \vdots \\
iL_{h} &= -\sum_{i}^{3N} \frac{\partial V_{h}}{\partial q_i} \frac{\partial}{\partial p_i}
\end{aligned}
\tag{48}
$$

Here the sums are extended to all $3N$ degrees of freedom, with N being the number of atoms in the systems.

To obtain a more manageable expression for the propagator containing only exponential products we use the following symmetric hermitian approximant:

$$
e^{i(L_1+L_2)\Delta t} \simeq e^{(iL_1\Delta t/2)} e^{(iL_2\Delta t)} e^{(iL_1\Delta t/2)} + O((\Delta t)^3)
\tag{49}
$$

By repeatedly using the identity $e^{iL\Delta t} = (e^{iL\Delta t/k})^k$ and applying eq. (49) to eq. (47) a five-time-step propagator can be easily obtained:

$$
\begin{aligned}
e^{iL\Delta t} = {} & \Bigg\{ \exp\left[-\frac{\partial V_h}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_h}{2} \right] \Bigg\{ \exp\left[-\frac{\partial V_l}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_l}{2} \right] \Bigg\{ \exp\left[-\frac{\partial V_m}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_m}{2} \right] \\
& \times \Bigg\{ \exp\left[-\frac{\partial V_{n1}}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_{n1}}{2} \right] \Bigg\{ \exp\left[-\frac{\partial V_{n0}}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_{n0}}{2} \right] \exp\left[\dot{\mathbf{r}}_i \frac{\partial}{\partial \mathbf{r}_i} \Delta t_0 \right] \\
& \times \exp\left[-\frac{\partial V_{n0}}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_{n0}}{2} \right] \Bigg\}^{n0} \exp\left[-\frac{\partial V_{n1}}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_{n1}}{2} \right] \Bigg\}^{n1} \exp\left[-\frac{\partial V_m}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_m}{2} \right] \Bigg\}^{m} \\
& \times \exp\left[-\frac{\partial V_l}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_l}{2} \right] \Bigg\}^{l} \exp\left[-\frac{\partial V_h}{\partial \mathbf{r}_i} \frac{\partial}{\partial \mathbf{p}_i} \frac{\Delta t_h}{2} \right] \Bigg\}^{h}
\end{aligned}
\tag{50}
$$

where the sums over the atoms have been left implicit and the time steps are defined as:

$$
\begin{aligned}
\Delta t_{n0}, \Delta t_{n1} &= n0\Delta t_{n0}, \Delta t_m = n1\Delta t_{n1}, \Delta t_l \\
&= m\Delta t_m, \Delta t_h = l\Delta t_l, \text{and } \Delta t = h\Delta t_h
\end{aligned}
\tag{51}
$$

The corresponding multiple time step integration algorithm for advancing positions and velocities can be straightforwardly derived[3] by applying all exponential operators appearing in eq. (50) to the phase space vector $\{\mathbf{p}, \mathbf{q}]^3$ from right to left, using the only rule:

$$
\exp\left(a \frac{\partial}{\partial x} \right) x = x + a
\tag{52}
$$

where a is constant independent of x.

From the structure of $e^{iL\Delta t}$ it is clear that the slower forces are recalculated only after the faster (and less expensive) forces have been computed a given number of times. As an example, the medium range shell force, $F_l = \partial V_l/\partial q$, is recalculated only after F_m has been computed m times.

In a recent publication[12] we have investigated the efficiency of this type of r-RESPA algorithm in combination with the SPME technique to handle electrostatics. Indeed, with respect to single time-step simulations employing standard Ewald sum and rigid bond constraints, we obtained a speed-up of about one order of magnitude. When compared to simulations making use of constraints and a spherical cutoff at 10 Å (the radius of truncation of the direct space sum) our technique is about 2.5 times faster.

Simulation in Other Ensembles: Extended Lagrangian Method

ORAC can carry out MD simulation in ensembles other than the NVE using the extended Lagrangian (EL) method.[40] The fundamental idea of EL is to represent the effect of a suitable external reservoir by adding new degrees of freedom to the system.[17–19] The equations of motion to simulate the isobaric–isothermal (NPT) ensemble can be derived from the Lagrangian or Hamiltonian formalism adopting a set of *virtual coordinates* related to the real set of coordinates by a noncanonical transformation. The particular form of the Lagrangian or Hamiltonian is chosen in a way that the trajectory of the system in the *extended* phase space generates the desired equilibrium distribution function upon integration over the extra (extended) variables. From the simulation point of view, an important feature of this approach is that the Hamiltonian of the extended system is a constant of motion and is, therefore, a conserved quantity during a simulation. This characteristic can be used to control the quality of the simulation.

Because a detailed derivation of the EL method to simulate the NPT ensemble can easily be found in the literature,[41–43] we give here only the resulting equations of motion for a system composed of interacting molecules. These equations are used by ORAC.

When dealing with molecules, one has the choice of coupling the barostat to each atomic degree of freedom or to each molecular center of mass.[41] In the former case, the effect of the barostat corresponds to a space scaling that affects the position of each atom. In the latter, the scaling occurs only for the position of the molecular centers of mass, thus affecting only intermolecular distances.

It is clear that the definition of the pressure tensor will depend on the chosen coupling scheme. Thus, according to the virial theorem one can write the atomic pressure for a system of N molecules:

$$P_{atm} = \langle \Pi_{atm} \rangle = \left\langle \frac{1}{3V} \sum_{\alpha}^{N} \sum_{i}^{N_\alpha} \left(\frac{\mathbf{p}_{\alpha i}^2}{m_{\alpha i}} + \mathbf{r}_{\alpha i} \cdot \mathbf{f}_{\alpha i} \right) \right\rangle \tag{53}$$

and the molecular pressure:

$$P_{mol} = \langle \Pi_{mol} \rangle = \left\langle \frac{1}{3V} \sum_{\alpha}^{n} \left(\frac{\mathbf{P}_{\alpha}^2}{M_{\alpha}} + \mathbf{R}_{\alpha} \cdot \mathbf{F}_{\alpha} \right) \right\rangle \tag{54}$$

Here, upper and lower case letters are used for molecular and atomic properties, respectively. r is the coordinate, f is the force, and V is the volume. Indices α and i apply to molecules and atoms, respectively, and N_α is the number of atoms of the αth molecule. The symbol $\langle \ldots \rangle$ indicates an ensemble average.

If the system is ergodic and the molecules do not dissociate (which is true for nondissociative potentials), the above pressure equations have been demonstrated equivalent for flexible molecules.[44] In ORAC we have chosen instead to couple the barostat to the molecular center of mass even when simulating a protein in solution.[40]

For ease of numerical implementation, we write all the equations of motion at constant T and P in terms of scaled coordinates. If $\mathbf{h}$ is the 3×3 matrix defined earlier, the scaled coordinates $\mathbf{s}_{\alpha i}$ of the ith atoms belonging to the αth molecule are defined according to:

$$\mathbf{s}_{\alpha i} = \mathbf{h}^{-1} \mathbf{r}_{\alpha i} \tag{55}$$

where $\mathbf{r}_{\alpha i}$ are the Cartesian coordinates of the same atom. In the extended Lagrangian formulation each element of the matrix $\mathbf{h}$ is treated as an extra degree of freedom of the system. Thus, the equations of motion for the extended molecular system at external temperature T_{ext} and pressure P_{ext} are:

$$\begin{aligned} m_{\alpha i}\ddot{\mathbf{s}}_{\alpha i} = {} & \mathbf{h}^{-1}\mathbf{F}_{\alpha i} - m_{\alpha i}\left(\mathbf{h}^{-1}(\mathbf{h}^{t})^{-1}\dot{\mathbf{h}}^{t}\mathbf{h} + \mathbf{h}^{-1}\dot{\mathbf{h}}\right)\dot{\mathbf{S}}_{\alpha} \\ & + m_{\alpha i}\Big[(\ddot{\mathbf{h}}^{-1} + 2\dot{\mathbf{h}}^{-1}\dot{\mathbf{h}}\mathbf{h}^{-1})\mathbf{h}(\mathbf{s}_{\alpha i} - \mathbf{S}_{\alpha}) \\ & + 2\dot{\mathbf{h}}^{-1}\mathbf{h}\left(\dot{\mathbf{s}}_{\alpha i} - \dot{\mathbf{S}}_{\alpha}\right)\Big] \\ & + m_{\alpha i}\zeta\left[\dot{\mathbf{h}}^{-1}\mathbf{h}(\mathbf{s}_{\alpha i} - \mathbf{S}_{\alpha}) - \dot{\mathbf{s}}_{\alpha i}\right] \end{aligned} \tag{56}$$

$$Q\ddot{\mathbf{h}} = (\mathbf{\Pi}_{mol} - P_{ext}\mathbf{1})V(\mathbf{h}^{-1})^{t} - Q\zeta\dot{\mathbf{h}} \tag{57}$$

$$W\dot{\zeta} = \sum_{\alpha=1}^{N}\sum_{i=1}^{N_\alpha} m_{\alpha i}(\mathbf{h}\dot{\mathbf{s}}_{\alpha i})^2 + QTr[\dot{\mathbf{h}}^{t}\dot{\mathbf{h}}] - gk_BT_{ext} \tag{58}$$

Here, $\mathbf{S}_\alpha$ is the center of mass position of molecule α in scaled coordinates, $m_{\alpha i}$ is the atomic mass, and g is a constant equal to the total number of degrees of freedom of the extended system. Also, Q and W are the masses of the extended variables $\mathbf{h}$ and ζ associated with the control of pressure and temperature respectively. While P_{ext} is the applied external pressure, the molecular pressure tensor is defined by:

$$\mathbf{\Pi}_{mol} = \frac{1}{V}\left[\sum_{\alpha=1}^{N} M_{\alpha}\left(\mathbf{h}\dot{\mathbf{S}}_{\alpha}\right)\left(\mathbf{h}\dot{\mathbf{S}}_{\alpha}\right)^{t} + \sum_{\alpha=1}^{N} \mathbf{F}_{\alpha}(\mathbf{h}\mathbf{S}_{\alpha})^{t}\right] \tag{59}$$

where V is the volume of the simulation cell, M_α is the mass of the αth molecule, and $\mathbf{F}_\alpha = \sum_i^{N_\alpha} \mathbf{F}_{\alpha i}$ is the force acting on the center of mass of molecule α. The total internal pressure, P, can be obtained by taking the trace of eq. (59), that is:

$$P_{mol} = \langle \Pi_{mol} \rangle = \frac{1}{3} Tr \mathbf{\Pi}_{mol} \tag{60}$$

The first term on the right-hand side of eq. (56) is the usual Newton equation of motion. The additional contributions in the second and third terms are due to the coupling with the barostat. The equations of motion for the MD cell vectors, given in eq. (57), contain a force term proportional to the difference between the instantaneous molecular pressure tensor and the externally imposed pressure. In both eqs. (56) and (57) the presence of a friction term proportional to ζ controls the total kinetic energy of the system. Finally, eq. (58) is a first-order equation for the variable ζ.

In addition to the standard Nosè–Hoover thermostat coupled to the barostat in eq. (58), ORAC can also employ a Nosè–Hoover chain[45] for simulation in the NVT and NPT ensembles.

ORAC integrates the equations of motion using a Verlet-based algorithm modified to treat velocity-dependent forces.[46] With respect to standard Verlet an iterative procedure is added to calculate the velocities of the extended system at the same time step and with similar precision as the coordinates. In ORAC's implementation, convergence is reached when the maximum relative change between the forces on the particles and on the extended variables of two successive iterations is less than 10^{-6}. The iterative procedure used in the integration breaks the time reversibility of the algorithm. As a consequence, in runs longer than 1 nanosecond, a very small energy drift is sometimes detectable.

In the EL method the detailed dynamics of the extended variables depends upon the value of the corresponding masses chosen. Within certain approximations the associated fundamental frequencies can be readily estimated by linearizing the equations of motion in $\mathbf{h}$ and ζ—eqs. (57) and (58), respectively. Thus, in the limit of large Q, the barostat mass, and small W, the thermostat mass, it can be shown[47] that the characteristic oscillatory frequencies for variables $\mathbf{h}$ and ζ are given by:

$$\omega_Q = \left(\frac{3LB}{Q} \right)^{1/2} \tag{61}$$

while ζ behaves as an harmonic oscillator with frequency:

$$\omega_W = \left(\frac{2Nk_BT}{W} \right)^{1/2} \tag{62}$$

where $L = V^{1/3}$ with V the volume of the system and B is the bulk modulus (or inverse compressibility):

$$B^{-1} \equiv -\frac{1}{V} \left(\frac{\partial V}{\partial P} \right) \tag{63}$$

When the time scales of $\mathbf{h}$ and ζ are similar, eqs. (57) and (58) couple together and their characteristic frequencies can differ significantly from their harmonic values. The best choice of pressure and temperature mass parameters would provide a direct coupling to those vibrational modes of the system directly involved in the processes of sound and thermal diffusion, respectively. Because low-frequency modes ($\nu < 100$ cm^{-1}) bring the highest weight in both phenomena, one is obliged to choose a barostat and thermostat masses with the same order of magnitude. Thus, the harmonic frequencies given by eqs. (61) and (62) can be used as only rough estimates of the real time scales. As an example, if we choose Q and W corresponding to ω_Q and ω_W values of 50 and 150 cm^{-1}, respectively, then the *experimental* ω_Q is found at 130 cm^{-1}. To determine the mass Q from eq. (61) ORAC uses, by default, a bulk modulus of 1890 MPa, which is suitable for simulations of liquid water and solvated proteins.

Organization of Code and Data Flow

The ORAC program has been written in FORTRAN 77. Very few of the program subroutines have a global character (i.e., share data with others), whereas the rest are self-contained modular subroutines. In general, global routines accomplish general tasks; examples include setting up the simulation box, running a minimization or MD simulation, etc. On the contrary, modular routines perform more specific tasks, such as initialization of velocities, computing the potential and forces due to a specific type of interaction, writing trajectory files, etc. This subdivision facilitates program maintenance and development. It also makes it easier to read and understand the code.

Figure 2 shows the essential flow chart of the program. Here, the main program first calls the global routines start, bldbox and then, alternatively, drvmin, drvmd, and mtsmd.

The fundamental task of start is to read and elaborate all information regarding the simulated system except its coordinates. First, start reads and interprets the input and the auxiliary files (see the

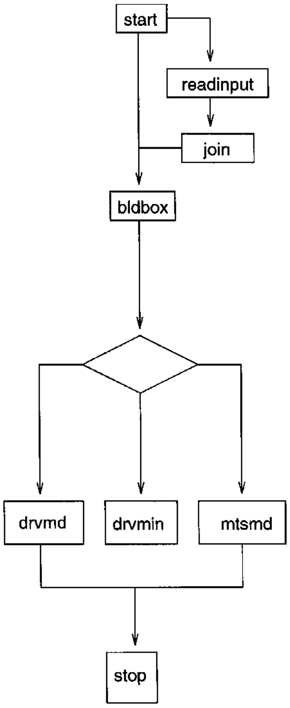

FIGURE 2. Schematic flow chart of the ORAC program. Only the most significant routines are shown.

next two subsections) containing all of the simulation options and all necessary information on the solute and solvent potential field and topology. This initial task is accomplished by invoking the global module read_input to parse the ORAC main input file. Before jumping back to the calling routine, read_input verifies the input by checking the syntax and options compatibility. Second, based on the information gathered previously by read_input, start constructs the total connectivity and topology of the solute molecules through join. This corresponds to setting up arrays for each topology element (i.e., stretching, bending, torsions, etc.) identifying the atoms involved and the potential parameters to be used in the interaction.

The routine bldbox reads the coordinates from a protein data bank (PDB) format file, assigns the *nonbonded* potential parameters according to the types defined in the topological file, and builds up the basic simulation cell. Unlike highly interactive packages, ORAC cannot construct a solute molecule from scratch starting from the sequence of the component residues. In ORAC the system coordinates must always be read from a PDB or a restart file. On the contrary, solvent molecules can be generated on a regular lattice of the user's choice and added to solute molecules. In addition, symmetry operations can be applied by the routine bldbox to generate a crystal lattice according to any given space group.

The three global routines read_input, join, and bldbox initialize a large common block area containing the data needed to run a simulation. This storage area, consisting of several common blocks, is defined in the file *cpropar.inc*, which is included through a FORTRAN INCLUDE directive in many of the global modules.

When ORAC returns from bldbox, the system coordinates, the system, and simulation parameters are known. With this information, minimizations and single and multiple time step simulations can be carried out by the global modules drvmin, drvmd, and mtsmd, respectively.

Conclusion and Perspective

In this article, we have described a molecular dynamics program, ORAC, specifically designed for efficiently and effectively simulating complex molecular systems such as proteins and membranes. ORAC can carry out simulation in the microcanonical (NVE), canonical (NVT), isobaric (NPH), and isothermal–isobaric (NPT) ensembles. Runs in the NVE ensemble can be carried out with single and multiple time step r-RESPA algorithms. In the latter case, fast algorithms to evaluate nonbonded forces for complex molecular systems area available. Standard Ewald summation and the faster particle mesh Ewald techniques are also available to compute accurately electrostatic energies and forces. The above techniques can be used with single time step integration schemes in the NVT, NPH, and NPT ensembles and in conjunction with r-RESPA method and the particle mesh technique, constitute a major asset for the program and can reduce the average cost of a simulation by a factor of three to four. In Table I, we show the CPU time per picosecond of simulation of a typical multiple time-step algorithm applied to a solvated BPTI protein on various machines. A nanosecond simulation, running with Ewald and integrating *all* 3×3912 degrees of freedom of the system, takes about 311 hours on a medium-sized workstation such as an HP-735.

ORAC is sufficiently flexible that a variety of molecules can be easily handled by defining the

TABLE I.
Performance of ORAC on a Series of Desktop Workstations.[a]

	HP 735	DEC Alpha 3000 / 800	IBM 580H	SGI R10000
System 1	1.10	1.33	1.22	0.41
System 2	9.62	13.61	9.82	4.13

[a] Simulation runs were carried out on two different systems: (i) System 1 corresponded to the hydrated BPTI molecules discussed in the example and contained 3906 atoms. (ii) System 2 was a hydrated reaction center of *Rhodobacter sphaeroides* containing 33,495 atoms in a nonorthogonal box of dimensions $a = 73.14$ Å, $b = 82.07$ Å, $c = 57.85$ Å, and $\alpha = 90.0$, $\beta = 88.6$, $\gamma = 90.5$ degrees. The algorithm used in both cases was the five-step r-RESPA algorithm discussed in step IV of the Supplementary Material. To obtain the same level of convergence in the electrostatic sum, the same SPME parameters used in the examples were adopted for both systems except that for system 2 the number of grid points was increased to 64, 72, and 48 in the three directions, respectively. This compensated for the larger dimensions of the simulation box. The timings reported are given in units of CPU seconds needed to run 1 femtosecond of simulation.

molecule topology and the force field in a clear and concise way. This is not always the case in other simulation programs. Although ORAC can simulate isolated molecules, it is specifically designed to investigate periodic systems which minimize finite size effects. Thus, the best performance with respect to other MD codes is reached when simulating crystals, liquids, and heterogeneous solutions (i.e., proteins or membranes in solutions).

In the future, efforts will be made toward improving the efficiency of r-RESPA and SPME algorithms. The extension of ORAC's capability to handle constant pressure and/or temperature ensembles by implementing new r-RESPA algorithms for extended systems[16,52] has been recently carried out and a public version of the new code will be made available soon.

Acknowledgments

The authors acknowledge Marc Souaille for writing the routines to compute the linked cell neighbor lists for nonorthogonal boxes. One of us (M.M.) thanks E. Clementi, G. R. Kneller, and U. Niesar for useful discussions in the early stages of program development. Stefano Baroni is gratefully acknowledged for his continuous support.

References

1. A. Rahman, *Phys. Rev.*, **136,** 405 (1964).
2. L. Verlet, *Phys. Rev.*, **159,** 98 (1967).
3. M. E. Tuckerman, B. J. Berne, and G. J. Martyna, *J. Chem. Phys.*, **97,** 1990 (1992).
4. D. D. Humphreys, R. A. Friesner, and B. J. Berne, *J. Phys. Chem.*, **98,** 6885 (1994).
5. M. Watanabe and M. Karplus, *J. Phys. Chem.*, **99,** 5680 (1995).
6. P. Procacci and B. J. Berne, *J. Chem. Phys.*, **1015,** 2421 (1994).
7. P. Procacci and M. Marchi, *J. Chem. Phys.*, **104,** 3003 (1996).
8. L. Greengard and V. Rokhlin, *J. Comput. Phys.*, **73,** 325 (1987).
9. K. E. Schmidt and M. A. Lee, *J. Stat. Phys.*, **63,** 1223 (1991).
10. T. Darden, D. York, and L. Pedersen, *J. Chem. Phys.*, **98,** 10089 (1993).
11. U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. G. Pedersen, *J. Chem. Phys.*, **101,** 8577 (1995).
12. P. Procacci, T. Darden, and M. Marchi, *J. Phys. Chem.*, **100,** 10464 (1996).
13. H. Berendsen, *Molecular Dynamics and Protein Structure*, Polycrystal Book Service, Western Spring, IL, 1985.
14. H. Lee, T. A. Darden, and L. G. Pedersen, *J. Chem. Phys.*, **102,** 3830 (1995).
15. M. Saito, *J. Chem. Phys.*, **101,** 4055 (1994).
16. P. Procacci and B. J. Berne, *Mol. Phys.*, **83,** 255 (1994).
17. H. C. Andersen, *J. Chem. Phys.*, **72,** 2384 (1980).
18. M. Parrinello and A. Rahman, *Phys. Rev. Lett.*, **45,** 1196 (1980).
19. S. Nosè, *J. Chem. Phys.*, **81,** 511 (1984).
20. B. R. Brooks, R. E. Bruccoeri, B. D. Olafson, D. J. States, S. Swaminanthan, and M. Karplus, *J. Comput. Chem.*, **4,** 187 (1983).
21. S. J. Wiener, P. A. Kollmann, D. T. Nguyen, and D. A. Case, *J. Comput. Chem.* **7,** 230 (1986).
22. W. D. Cornell, P. Cieplak, C. I. Bavly, I. R. Gould, K. M. Merz Jr., D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. Kollmann, *J. Am. Chem. Soc.*, **117,** 5179 (1995).
23. G. Ciccotti, M. Ferrario, and J.-P. Ryckaert, *Mol. Phys.*, **47,** 1253 (1982).
24. W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein, *J. Chem. Phys.*, **79,** 926 (1983).
25. M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, Oxford University Press, Oxford, 1989.
26. P. Ewald, *Ann. Phys.*, **64,** 253 (1921).
27. S. W. deLeeuw, J. W. Perram, and E. R. Smith, *Proc. R. Soc. Lond. A*, **373,** 27 (1980).
28. P. E. Smith and B. M. Pettitt, *J. Chem. Phys.*, **105,** 4289 (1996).
29. R. W. Hockney, *Computer Simulation Using Particles*, McGraw-Hill, New York, 1989.
30. H. G. Petersen, *J. Chem. Phys.*, **103,** 3668 (1995).

31. M. E. Tuckerman and B. J. Berne, *J. Chem. Phys.*, **95,** 8362 (1991).
32. M. E. Tuckerman, G. J. Martyna, and B. J. Berne, *J. Chem. Phys.*, **94,** 6811 (1991).
33. M. E. Tuckerman, B. J. Berne, and A. Rossi, *J. Chem. Phys.*, **94,** 1465 (1990).
34. Polygen Corp. Parameter and Topology files for CHARMm, Version 22 (copyright 1986, release 1992).
35. M. E. Tuckerman and M. Parrinello, *J. Chem. Phys.*, **101,** 1302 (1994).
36. M. E. Tuckerman and M. Parrinello, *J. Chem. Phys.*, **101,** 1316 (1994).
37. H. de Raedt and B. de Raedt, *Phys. Rev. A*, **28,** 3575 (1983).
38. H. Goldstein, *Classical Mechanics*, Addison-Wesley, Reading, MA, 1980.
39. S. K. Grey, *J. Chem. Phys.*, **101,** 4062, 1994.
40. E. Paci and M. Marchi, *J. Phys. Chem.*, **104,** 3003 (1996).
41. M. Ferrario and J.-P. Ryckaert, *Mol. Phys.*, **78,** 7368 (1985).
42. S. Nosé, *Prog. Theor. Phys.*, **103**(Suppl.), 1 (1991).
43. M. Ferrario, *Computer Simulation in Chemical Physics*, M. P. Allen and D. J. Tildesley, Eds., Kluwer, Dordrecht, 1993.
44. G. Ciccotti and J.-P. Ryckaert, *Computer Phys. Rep.*, **4,** 345 (1986).
45. G. L. Martyna, M. L. Klein, and M. Tuckerman, *J. Chem. Phys.*, **97,** 2635 (1992).
46. J.-P. Ryckaert and G. Ciccotti, *J. Chem. Phys.*, **78,** 7368 (1983).
47. S. Nosé and M. L. Klein, *Mol. Phys.*, **50,** 1055 (1983).
48. M. Marchi and P. Procacci, ORAC *Manual and Guide*, CECAM-ENS, Lyon 1997 (available at ftp.cecam.fr:/pub/orac/doc/manual.ps).
49. J. J. P. Stewart, *J. Comput. Chem.*, **10,** 221 (1989).
50. S. J. Weiner, P. A. Kollmann, D. A. Case, U. C. Singh, C. Ghio, G. Alagona, S. Profeta Jr., and P. Weiner, *J. Am. Chem. Soc.*, **106,** 765 (1984).
51. S. Parkin, B. Rupp, and H. Hope, The structure of bovine pancreatic trypsin inhibitor at 125k: Definition of carboxyl-terminal residues glycine-57 and alanine-58 (manuscript submitted).
52. G. J. Martyna, M. E. Tuckerman, D. J. Tobias, and M. L. Klein, *Mol. Phys.*, **87,** 1117 (1996).