



# HHS Public Access

Author manuscript

*J Chem Theory Comput.* Author manuscript; available in PMC 2015 October 26.

Published in final edited form as:

*J Chem Theory Comput.* 2014 February 28; 10(4): 1638–1651. doi:10.1021/ct401096t.

## Scalable Evaluation of Polarization Energy and Associated Forces in Polarizable Molecular Dynamics: II. Towards Massively Parallel Computations using Smooth Particle Mesh Ewald

Louis Lagardère<sup>†,‡</sup>, Filippo Lipparini<sup>¶,‡,†,§</sup>, Étienne Polack<sup>¶,‡</sup>, Benjamin Stamm<sup>¶,||</sup>, Éric Cancès<sup>⊥</sup>, Michael Schnieders<sup>#</sup>, Pengyu Ren<sup>@</sup>, Yvon Maday<sup>¶,△,▽</sup>, and Jean-Philip Piquemal<sup>\*,‡</sup>

<sup>†</sup>UPMC Univ. Paris 06, Institut du Calcul et de la Simulation, F-75005, Paris, France

<sup>‡</sup>UPMC Univ. Paris 06, UMR 7617, Laboratoire de Chimie Théorique, F-75005, Paris, France

<sup>¶</sup>UPMC Univ. Paris 06, UMR 7598, Laboratoire Jacques-Louis Lions, F-75005, Paris, France

<sup>||</sup>CNRS, UMR 7598 and 7616, F-75005, Paris, France

<sup>⊥</sup>Université Paris-Est, CERMICS, Ecole des Ponts and INRIA, 6 & 8 avenue Blaise Pascal, 77455 Marne-la-Vallée, France

<sup>#</sup>Departments of Biomedical Engineering and Biochemistry, The University of Iowa, Iowa City, Iowa 52358, United States

<sup>@</sup>Department of Biomedical Engineering, The University of Texas at Austin, Austin, Texas 78712, United States

<sup>△</sup>Institut Universitaire de France

<sup>▽</sup>Brown Univ, Division of Applied Maths, Providence, RI, USA

### Abstract

In this paper, we present a scalable and efficient implementation of point dipole-based polarizable force fields for molecular dynamics (MD) simulations with periodic boundary conditions (PBC). The Smooth Particle-Mesh Ewald technique is combined with two optimal iterative strategies, namely, a preconditioned conjugate gradient solver and a Jacobi solver in conjunction with the Direct Inversion in the Iterative Subspace for convergence acceleration, to solve the polarization equations. We show that both solvers exhibit very good parallel performances and overall very competitive timings in an energy-force computation needed to perform a MD step. Various tests on large systems are provided in the context of the polarizable AMOEBA force field as implemented in the newly developed Tinker-HP package which is the first implementation for a polarizable model making large scale experiments for massively parallel PBC point dipole models possible. We show that using a large number of cores offers a significant acceleration of the overall process involving the iterative methods within the context of *SPME* and a noticeable improvement of the memory management giving access to very large systems (hundreds of

\*To whom correspondence should be addressed jpp@lct.jussieu.fr.

§current address: Institut für Physikalische Chemie, Universität Mainz, Duesbergweg 10-14, D-55128 Mainz, Germany

thousands of atoms) as the algorithm naturally distributes the data on different cores. Coupled with advanced MD techniques, gains ranging from 2 to 3 orders of magnitude in time are now possible compared to non-optimized, sequential implementations giving new directions for polarizable molecular dynamics in periodic boundary conditions using massively parallel implementations.

## 1 Introduction

Polarizable force fields have been, in the last decade, the subject of an intense development.<sup>1–8</sup> The ability of anisotropic polarizable molecular mechanics (APMM) to model complex systems, including charged or very polar ones, biological substrates containing metal ions, weakly interacting molecules or ionic liquids is of prime interest in the molecular dynamics community,<sup>8–12</sup> as is its robustness and transferability.<sup>8</sup> Furthermore, such force fields have been recently successfully employed in conjunction with quantum mechanical models<sup>13–23</sup> in order to accurately reproduce environmental effects on structural and photophysical properties of chromophores in complex biological substrates, expanding their use to the QM/MM community. Several strategies can be used to introduce polarization in a classical force field, including fluctuating charges,<sup>20,24–26</sup> Drude's oscillators,<sup>23,27</sup> the Kriging method<sup>28</sup> and induced dipoles.<sup>29–33</sup> Independent of the model, all polarizable force fields share the need to determine the polarization degrees of freedom for a given geometry, which usually requires one to solve a set of linear equations;<sup>34,35</sup> furthermore, a polarization term is added to the energy and thus it is required to compute the associated forces. With respect to standard, additive force fields, such a characteristic introduces a heavy computational overhead, as for each step of a molecular dynamics simulation (or for each value of the QM density in a QM/MM computation), one needs to solve a linear system whose size depends on the number of the polarizable sites. A standard, direct approach, such as the use of LU or Cholesky decomposition, becomes rapidly unfeasible for large systems, as it requires a computational effort that scales as the cube of the number of atoms, together with quadratic storage: the development of efficient iterative techniques is therefore a mandatory step in extending the range of applicability of polarizable force fields to large and very large molecular systems.

Iterative techniques require to perform several matrix-vector multiplications, representing a computational effort quadratic in the size of the system: an efficient implementation has therefore three different points to address: i) a fast convergent iterative procedure, in order to limit as much as possible the number of matrix-vector multiplications, ii) a fast matrix-vector multiplication technique, and in particular a linear scaling technique that allows one to overcome the quadratic bottleneck and iii) an efficient parallel setup, in order to distribute the computational work among as many processors as possible, exploiting thus modern, parallel computers.

In a recent paper,<sup>34</sup> we focused on the various iterative techniques suitable to solve the polarization equations for dipole-based polarizable force fields and we found that two algorithms, namely, the preconditioned conjugate gradient (PCG) method and Jacobi iterations with convergence acceleration based on Pulay's Direct Inversion in the Iterative Subspace (DIIS), were particularly suitable for the purpose, both in terms of convergence

properties and in terms of parallel implementation, addressing therefore the first and the third points of an optimal implementation.

The aim of our paper is to introduce new algorithmic strategies to solve the polarization equations for polarizable molecular dynamics in periodic boundary conditions for which we present a new and tailored parallel implementation together with the specific procedure to compute the associated forces. We first analyze the coupling of previously proposed iterative solvers for polarization, that we only studied in the case of direct space computations, to the Smooth Particle Mesh Ewald (SPME) technique. This technique allows one to compute the involved matrix-vector products required to determine the induced dipoles in iterative procedures with a  $\mathcal{O}(N \log N)$  computational effort, addressing therefore the need of an efficient and scalable implementation. Indeed, SPME for distributed multipoles was introduced by Sagui et al.<sup>36</sup> and Wang and Skeel<sup>35</sup> demonstrated that PCG was compatible with SPME but the study was limited to the case of permanent point charges. We then propose to assemble such ingredients into a new parallel implementation using SPME for distributed multipoles and improved polarization solvers including the newly developed JI/DIIS. This strategy is oriented towards the needs of massive parallelism allowing one to tackle large systems by distributing both computation and memory on a large number of cores, resorting into a newly developed module of Tinker called Tinker-HP. In fact, the SPME method computes the electrostatic interactions as the sum of a direct space contribution, which is limited to close neighbors, plus a reciprocal (Fourier) space sum, which can be efficiently computed thanks to the Fast Fourier transform (FFT). However, the heavy cost in communication of any parallel FFT implementation makes achieving a scalable parallel global implementation a more complex task than for direct space simulations. Such an issue is expected to be the main bottleneck for the parallel efficiency of the presented iterative solvers and will be addressed in our study.

The paper is organized as follows. In Section 2, we introduce the SPME formalism in the general context of APMM using distributed point multipoles. In Section 3, we explicitly formulate the polarization energy and introduce the different iterative methods to solve the polarization equations. In Section 4, we explore the parallel behavior of the different iterative algorithms as well as the parallel behavior of the computation of the forces. In Section 5, some numerical results are provided. We conclude the paper in Section 6 with some conclusions and perspectives.

## 2 PME with multipolar interactions

The use of periodic boundary conditions is a natural choice when simulating intrinsically periodic systems, such as crystals or pure liquids, and is a commonly used procedure to simulate solvated systems with explicit solvent molecules including proteins, ions, etc. Notice that an alternative approach, based on non-periodic boundary conditions and polarizable continuum solvation models, has also been recently proposed for APMM.<sup>37,38</sup>

When PBC are employed, the use of the particle-mesh Ewald method becomes particularly interesting because of both its computational efficiency and the physical accuracy of the results it provides. The PME method is based on the Ewald summation<sup>39</sup> introduced to

compute the electrostatic energy of a system in PBC and has been originally formulated by Darden et al.<sup>40</sup>. This first formulation, that involved Lagrange polynomials, introduced forces that were not analytical derivatives of the energy and another formulation using B-splines, the Smooth Particle Mesh Ewald (SPME), was given by Essmann et al.<sup>41</sup>, allowing analytical differentiation to get the forces and so ensuring energy conservation. Such an approach, first developed for distributed point charges only, was extended to distributed multipoles by Sagui et al.<sup>36</sup> and to distributed Hermite gaussian densities by Cisneros et al.<sup>42</sup>.

## 2.1 Notation

In this paper, we will use the same notation as in our previous paper.<sup>34</sup> Let  $\mathbf{r}^{(N)}$  be the system composed of a neutral unit cell  $U$  containing  $N$  atoms at positions  $\{\vec{r}_i\}_{i=1}^N$  replicated in all directions. Let  $(\vec{a}_1, \vec{a}_2, \vec{a}_3)$  be the vectors defining the unit cell so that its volume is  $V = \vec{a}_1 \cdot (\vec{a}_2 \times \vec{a}_3)$ . We will indicate vector quantities by using an arrow if the vectors are in  $\mathbb{R}^3$  and with the bold font if they represent a collection of three-dimensional vectors. For instance,  $\boldsymbol{\mu}$  will be a  $3N$ -dimensional column vector  $(\vec{\mu}_1, \dots, \vec{\mu}_N)^T$ , where each  $\vec{\mu}_i = (\mu_i^1, \mu_i^2, \mu_i^3)^T$  is a three-dimensional column vector (in particular, a dipole). We will use Latin indexes as a subscript to refer to different atomic sites and Greek indexes as superscripts to indicate the Cartesian component of a vector, so that  $\mu_i^\alpha$  denotes the  $\alpha$ -th Cartesian component ( $\alpha = 1, 2, 3$ ) of the vector  $\vec{\mu}_i$  ( $i=1, \dots, N$ ).

For any integers  $n_1, n_2, n_3$  and  $m_1, m_2, m_3$  we will denote by

$$\vec{n} = n_1 \vec{a}_1 + n_2 \vec{a}_2 + n_3 \vec{a}_3 \quad \vec{m} = m_1 \vec{a}_1^* + m_2 \vec{a}_2^* + m_3 \vec{a}_3^*$$

where  $(\vec{a}_1^*, \vec{a}_2^*, \vec{a}_3^*)$  is the dual basis of  $(\vec{a}_1, \vec{a}_2, \vec{a}_3)$  (i.e.  $\vec{a}_\alpha^* \cdot \vec{a}_\beta = \delta_{\alpha\beta}$ ). Hence in what follows  $\vec{n}$  will always be in the direct space and  $\vec{m}$  in the reciprocal space and we shall identify  $\vec{n}$  with  $(n_1, n_2, n_3)$  and  $\vec{m}$  with  $(m_1, m_2, m_3)$ .

Furthermore, we will call fractional coordinates of the atoms the quantities:

$$\vec{a}_\alpha^* \cdot \vec{r}_i \in \left[ -\frac{1}{2}, \frac{1}{2} \right] \quad \alpha=1, 2, 3; i=1, \dots, N. \quad (1)$$

In the AMOEBA polarizable force field with which we have been working, the charge density of a system is approximated by a set of permanent atomic multipoles (up to quadrupoles) located at the atomic positions  $\vec{r}_i$ , which we will denote as follows:

$$q_{pi}, \vec{\mu}_{pi}, \Theta_{pi} \text{ for } i=1, \dots, N.$$

## 2.2 Ewald Summation and Smooth Particle Mesh Ewald

In this subsection, we will review how to compute electrostatic interactions between permanent multipoles in periodic boundary conditions by using the Ewald summation and the Smooth Particle Mesh Ewald method. The idea is to split the electrostatic energy into

three terms, two being sums with good convergence properties and the third one being independent of the atomic positions.

The electrostatic potential at a point  $\vec{x}$  created by the permanent multipoles is obtained by applying the associated multipole operator  $L_i$  to  $L_i$  to  $|\vec{r}_i - \vec{x}|^{-1}$  where, for  $i = 1, \dots, N$ :

$$L_i = q_{pi} + \vec{\mu}_{pi} \cdot \nabla_i + \Theta_{pi} : \nabla_i \nabla_i, \quad (2)$$

where  $\nabla_i \nabla_i$  is the  $3 \times 3$  matrix defined by:  $(\nabla_i \nabla_i)_{\alpha\beta} = \partial_{r_i^\alpha} \partial_{r_i^\beta}$ . Let us define the multipole structure factor  $S(\vec{m})$

$$S(\vec{m}) = \sum_{j=1}^N \tilde{L}_j(\vec{m}) \exp(2i\pi \vec{m} \cdot \vec{r}_j) \quad (3)$$

with  $\tilde{L}_j$  being the Fourier transform of the multipole operator associated to site  $j$ :

$$\tilde{L}_j(\vec{m}) = q_{pj} + 2i\pi \vec{\mu}_{pj} \cdot \vec{m} - (2\pi)^2 \Theta_{pj} : M, \quad (4)$$

where  $(M)_{\alpha\beta} = m_\alpha m_\beta$ .

The electrostatic potentials  $\phi_{\text{elec}}(\vec{r}_i)$  generated by the permanent multipoles at site  $i$  and the corresponding electrostatic energy  $E_{\text{elec}}(\mathbf{r}^{\{N\}})$  are formally given by:

$$\phi_{\text{elec}}(\vec{r}_i) = \sum_{\vec{n} \in \mathbb{Z}^3} \sum_{1 \leq j \leq N}^{(n)} L_j \left( \frac{1}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right), \quad (5)$$

$$E_{\text{elec}}(\mathbf{r}^{\{N\}}) = \frac{1}{2} \sum_{\vec{n} \in \mathbb{Z}^3} \sum_{1 \leq i, j \leq N}^{(n)} L_i L_j \left( \frac{1}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right), \quad (6)$$

where the exponent  $(n)$  means that the terms  $i = j$  are not summed up when  $\vec{n} = \vec{0}$ . The Ewald summation technique<sup>39,43</sup> allows one to decompose the electrostatic energy of the geometric configuration  $\mathbf{r}^{\{N\}}$  (equation (6)) into three components: a contribution  $E_{\text{dir}}$  consisting of short range interactions computed in the real space, a global contribution  $E_{\text{rec}}$  computed in the reciprocal space and a self energy term  $E_{\text{self}}$ , which is a bias arising from the Ewald summation. These contributions are given by:

$$E_{\text{dir}} = \frac{1}{2} \sum_{\vec{n} \in \mathbb{Z}^3} \sum_{1 \leq i, j \leq N}^{(n)} L_i L_j \left( \frac{\text{erfc}(\beta |\vec{r}_j - \vec{r}_i + \vec{n}|)}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right), \quad (7)$$

$$E_{\text{rec}} = \frac{1}{2\pi V} \sum_{\substack{\vec{m} \in \mathbb{Z}^3 \\ \vec{m} \neq \vec{0}}} \frac{\exp(-\pi^2 \vec{m}^2 / \beta^2)}{\vec{m}^2} S(\vec{m}) S(-\vec{m}), \quad (8)$$

$$E_{\text{self}} = - \sum_{1 \leq j \leq N} \left( \frac{\beta}{\sqrt{\pi}} q_{pj}^2 + \frac{2\beta^3}{3\sqrt{\pi}} \vec{\mu}_{pj}^2 + \frac{2\beta^3}{9\sqrt{\pi}} q_{pj} \text{Tr}(\Theta_{pj}) + \frac{8\beta^5}{45\sqrt{\pi}} \Theta_{pj} : \Theta_{pj} \right). \quad (9)$$

We therefore have:

$$E_{\text{elec}} = E_{\text{dir}} + E_{\text{rec}} + E_{\text{self}}. \quad (10)$$

This formulation introduces a positive parameter  $\beta$  that controls the range of the direct term and conversely the importance of the reciprocal term. The direct term is computed using a cutoff (that depends on  $\beta$ ) as the erfc function in this term decreases to zero on a characteristic distance that depends on this parameter. Note that, as done by Wang and Skeel<sup>35</sup>, we neglect the surface term that arises during the derivation of the Ewald summation. We have thus chosen to use the self energy that is commonly used in the literature<sup>31,44–46</sup> and this does not change the associated forces in molecular dynamics. However, as will be shown by some of us, other forms can be considered for this term.

Scaling factors and damping functions<sup>30,31,47</sup> are often used in force fields to parameterize electrostatic interactions at short range, for instance in order to exclude close (1–2, 1–3, 1–4) neighbors; here, we assume that the only interactions concerned are between atoms within the unit cell. As the Ewald summation corresponds to unscaled interactions, one has to take this into account by adding a correction to the energy computed with this method. Suppose that the interaction between multipoles of sites  $i$  and  $j$  is scaled and/or damped by a factor  $s_{ij}$  ( $s_{ij} = s_{ji}$ ), then the correction

$$(s_{ij} - 1) L_i L_j \left( \frac{1}{|\vec{r}_j - \vec{r}_i|} \right) \quad (11)$$

has to be added to the total energy. Let us define  $E_{\text{corr}}$  the sum of all these terms:

$$E_{\text{corr}} = \frac{1}{2} \sum_{i=1}^N \sum_{j \in M(i)} (s_{ij} - 1) L_i L_j \left( \frac{1}{|\vec{r}_j - \vec{r}_i|} \right) \quad (12)$$

where  $M(i)$  is the list of atom sites  $j$  for which the multipolar interactions between sites  $i$  and  $j$  are scaled. For the case of the computation of the polarization energy, where all dipole-dipole interactions are included, such scaling parameters originate from Thole's damping, which is introduced in order to avoid the so-called polarization catastrophe.<sup>30,47</sup>

As explained in Ref. 45, the total electrostatic potential can be split into four contributions  $\Phi_{\text{dir}}$ ,  $\Phi_{\text{rec}}$ ,  $\Phi_{\text{self}}$  and  $\Phi_{\text{corr}}$ , with

$$\Phi_{\text{rec}}(\vec{r}_j) = \frac{1}{\pi V} \sum_{\substack{\vec{m} \in \mathbb{Z}^3 \\ \vec{m} \neq \vec{0}}} \frac{\exp(-\pi^2 \vec{m}^2 / \beta^2)}{\vec{m}^2} S(\vec{m}) \exp(-2i\pi \vec{m} \cdot \vec{r}_j). \quad (13)$$

In the context of the  $\text{SPME}$  method, the fractional coordinates of the atoms that appear in the reciprocal term are scaled by three factors (one for each coordinate). Indeed, let  $K_1, K_2, K_3$  be positive integers, then the corresponding scaled fractional coordinates of atom  $i$  are given by  $u_{1i}, u_{2i}, u_{3i}$ :

$$u_{\alpha j} = K_\alpha \vec{a}_\alpha^* \cdot \vec{r}_j \in \left[ -\frac{K_\alpha}{2}, \frac{K_\alpha}{2} \right]. \quad (14)$$

In consequence, the complex exponential functions present in the reciprocal potential and the structure factor can be written:

$$\exp(2i\pi \vec{m} \cdot \vec{r}_j) = \exp\left(2i\pi m_1 \frac{u_{1j}}{K_1}\right) \exp\left(2i\pi m_2 \frac{u_{2j}}{K_2}\right) \exp\left(2i\pi m_3 \frac{u_{3j}}{K_3}\right). \quad (15)$$

The  $\text{SPME}$  consists in replacing each of the complex exponentials  $\exp\left(2i\pi m_\alpha \frac{u_{\alpha j}}{K_\alpha}\right)$ , as a function of  $u_{\alpha j}$ , by a B-spline interpolation of degree  $p$  on a grid of size  $K_1 \times K_2 \times K_3$ . Using B-splines makes the  $\text{SPME}$  approximation of the reciprocal potential differentiable and allows one to derive forces that are analytical derivatives of the  $\text{SPME}$  approximation of  $E_{\text{rec}}$ . Furthermore, the sums appearing in the reciprocal potential when using the approximation through B-spline interpolation can be interpreted as discrete Fourier transformations, allowing one to use the  $\text{FFT}$  summation technique. After some cumbersome algebra,<sup>48</sup> the reciprocal potential can be approximated by:

$$\Phi_{\text{rec}}(\vec{r}_j) \approx \sum_{\vec{n} \in \mathbb{Z}^3} \theta_p(u_{1j} - n_1) \theta_p(u_{2j} - n_2) \theta_p(u_{3j} - n_3) \left(G^R \star Q^R\right)(\vec{n}) \quad (16)$$

where  $\theta_p$  are the B-splines of degree  $p$  of Sagui et al.<sup>48</sup>,  $G^R$  is the influence function defined by its Fourier transform in Sagui et al.<sup>48</sup> and  $Q^R$  is the real space multipolar array associated to the atomic multipoles of the system given by:

$$Q^R(k_1, k_2, k_3) = \sum_{\vec{n} \in \mathbb{Z}^3} \sum_{1 \leq j \leq N} L_j \left( \theta_p(u_{1j} - k_1 - K_1 n_1) \theta_p(u_{2j} - k_2 - K_2 n_2) \theta_p(u_{3j} - k_3 - K_3 n_3) \right). \quad (17)$$

Thus, two sets of parameters control the approximation level of the  $\text{SPME}$  algorithm:  $K_1, K_2, K_3$ , that define the size of the interpolation grid along each axis, and  $p$ , the degree of the B-splines that these complex exponentials are interpolated with.

From a computational point of view, the standard  $\text{SPME}$  algorithm follows these steps:

1. Compute the real space potential. The derivatives of the erfc function can be computed recursively as it is shown in Ref. 49.
2. Compute the correction potential due to scaling factors of the electrostatic interactions (usually done at the same time as the computation of the real space potential).
3. Compute the reciprocal potential:

- (a) Build the multipolar array  $Q^R$  associated with the atomic multipoles of the system;
- (b) Compute the FFT of this array  $Q^F$ ;
- (c) Multiply  $Q^F$  with  $G^F$ , i.e., the Fourier transform of  $G^R$ , which corresponds to a convolution in direct space;
- (d) Compute the backward FFT of the result to get the convolution of  $Q^R$  and  $G^R$ ;
- (e) Multiply the result with the values of the B-splines at the position where the potential is to be evaluated following equation (16).

### 3 Evaluation of the Polarization Energy and its Derivatives in Periodic Boundary Conditions

In this section we will see how the previously described methods, i.e., the Ewald summation and Smooth Particle Mesh Ewald, can be used in order to compute the polarization energy and the associated forces in a dipole-based polarizable force-field.

The static multipoles create, at each atom  $i$ , an electric field  $\vec{E}_i$  that is responsible for an induced dipole moment  $\vec{\mu}_i$ , the unknown of the polarization problem, in such a way that the (favorable) interaction between the induced dipoles and the inducing field is maximized, while the sum of the work to polarize each site and the repulsion between the induced dipoles is minimized. In other words, the electrostatic equilibrium is reached when the total polarization energy  $\mathcal{E}_{\text{pol}}$  (defined below) is minimized.

#### 3.1 The Polarization Energy

Let us introduce the vectors

$$\mathbf{E} = \begin{pmatrix} \vec{E}_1 \\ \vec{E}_2 \\ \vdots \\ \vec{E}_N \end{pmatrix}, \boldsymbol{\mu} = \begin{pmatrix} \vec{\mu}_1 \\ \vec{\mu}_2 \\ \vdots \\ \vec{\mu}_N \end{pmatrix}, \quad (18)$$

where  $\mathbf{E}$  collects the electric fields created by the permanent multipoles on the atom sites and  $\boldsymbol{\mu}$  the induced dipoles. The polarization energy functional can be written in a compact way:

$$\mathcal{E}_{\text{pol}} = -\mathbf{E}^\dagger \boldsymbol{\mu} + \frac{1}{2} \boldsymbol{\mu}^\dagger \mathbf{T} \boldsymbol{\mu}, \quad (19)$$

where the first term represents the interactions between the dipoles and the inducing field and the second one the interactions between dipoles expressed through the polarization



matrix  $\mathbf{T}$  that will be introduced below. The minimizer  $\boldsymbol{\mu}$  of equation (19) is the total vector of the induced dipole moments we introduced above; it satisfies the optimality condition:

$$\mathbf{T}\boldsymbol{\mu}=\mathbf{E}. \quad (20)$$

**3.1.1 Permanent Fields**—One can apply the Ewald summation to the electric fields  $\vec{E}_i$  for  $i = 1, \dots, N$ , in a similar fashion to what was described in the last section:

$$\vec{E}_i = \vec{E}_{i,\text{dir}} + \vec{E}_{i,\text{rec}} + \vec{E}_{i,\text{corr}} + \vec{E}_{i,\text{self}}, \quad (21)$$

where

$$\vec{E}_{i,\text{dir}} = \sum_{\vec{n} \in \mathbb{Z}^3} \sum_{1 \leq j \leq N}^{(n)} -\frac{\partial}{\partial \vec{r}_i} L_j \left( \frac{\text{erfc}(\beta |\vec{r}_j - \vec{r}_i + \vec{n}|)}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right), \quad (22)$$

$$\vec{E}_{i,\text{rec}} = \frac{1}{\pi V} \sum_{\substack{\vec{m} \in \mathbb{Z}^3 \\ \vec{m} \neq \vec{0}}} -\frac{\partial}{\partial \vec{r}_i} \left( \frac{\exp(-\pi^2 \vec{m}^2 / \beta^2)}{\vec{m}^2} S(\vec{m}) \exp(-2i\pi \vec{m} \cdot \vec{r}_i) \right), \quad (23)$$

$$\vec{E}_{i,\text{corr}} = \sum_{j \in M(i)} -(s_{ij} - 1) \frac{\partial}{\partial \vec{r}_i} L_j \left( \frac{1}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right), \quad (24)$$

$$\vec{E}_{i,\text{self}} = \frac{4\beta^3}{3\sqrt{\pi}} \vec{\mu}_{pi}. \quad (25)$$

The field  $\vec{E}_{i,\text{self}}$  is a bias arising from the Ewald summation. Applying the SPME method, one can approximate  $\vec{E}_{i,\text{rec}}$  by

$$\vec{E}_{i,\text{rec}} \approx \sum_{\vec{n} \in \mathbb{Z}^3} -\frac{\partial}{\partial \vec{r}_i} (\theta_p(u_{1i} - n_1) \theta_p(u_{2i} - n_2) \theta_p(u_{3i} - n_3)) (G^R * Q^R)(\vec{n}) \quad (26)$$

as explained in section 2.

**3.1.2 The Polarization Matrix**—One can also apply the Ewald summation to the electric fields at site  $i$  generated by the other induced dipoles and their images, and by the images of  $\vec{\mu}_i$  which leads to the expression of the off-diagonal term of the polarization matrix:

$$T_{ij} = T_{ij,\text{dir}} + T_{ij,\text{rec}} + T_{ij,\text{corr}}. \quad (27)$$

The expressions of the three terms of the right-hand side of Equation (27) are:

$$T_{ij,\text{dir}}^{\alpha\beta} = \sum_{\vec{n} \in \mathbb{Z}^3} \binom{n}{\vec{n}} \frac{\partial}{\partial r_i^\alpha} \frac{\partial}{\partial r_j^\beta} \frac{\text{erfc}(\beta|\vec{r}_j - \vec{r}_i + \vec{n}|)}{|\vec{r}_j - \vec{r}_i + \vec{n}|}, \quad (28)$$

$$T_{ij,\text{rec}}^{\alpha\beta} = \frac{2i}{V} \sum_{\substack{\vec{m} \in \mathbb{Z}^3 \\ \vec{m} \neq \vec{0}}} \frac{2i\pi \exp(-\pi^2 \vec{m}^2 / \beta^2)}{\vec{m}^2} m^\alpha m^\beta \exp(2i\pi \vec{m} \cdot (\vec{r}_j - \vec{r}_i)), \quad (29)$$

$$T_{ij,\text{corr}}^{\alpha\beta} = \sum_{j \in M(i)} (s_{ij} - 1) \frac{\partial}{\partial r_i^\alpha} \frac{\partial}{\partial r_j^\beta} \left( \frac{1}{|\vec{r}_j - \vec{r}_i + \vec{n}|} \right). \quad (30)$$

Applying the SPME method we have then:

$$T_{ij,\text{rec}}^{\alpha\beta} \approx \sum_{\vec{n} \in \mathbb{Z}^3} \frac{\partial}{\partial r_i^\alpha} (\theta_p(u_{1i} - n_1) \theta_p(u_{2i} - n_2) \theta_p(u_{3i} - n_3)) \left( G^R * Q_j^{R\beta} \right) (\vec{n}) \quad (31)$$

where  $Q_j^{R\beta}$  is the multipolar array associated with the  $\beta$  component of the  $j$ -th induced dipole:

$$Q_j^{R\beta}(k_1, k_2, k_3) = \sum_{\vec{n} \in \mathbb{Z}^3} \frac{\partial}{\partial r_j^\beta} (\theta_p(u_{1j} - k_1 - K_1 n_1) \theta_p(u_{2j} - k_2 - K_2 n_2) \theta_p(u_{3j} - k_3 - K_3 n_3)). \quad (32)$$

From the Ewald summation, the self-fields  $T_{i,\text{self}} \vec{\mu}_i$ , where

$$T_{i,\text{self}}^{\alpha\beta} = -\frac{4\beta^3}{3\sqrt{\pi}} \delta_{\alpha\beta}, \quad (33)$$

also have to be included to the expression of the polarization energy that reads:

$$\mathcal{E}_{\text{pol}} = -\sum_{i=1}^N E_i^\alpha \mu_i^\alpha + \frac{1}{2} \sum_{i=1}^N (\alpha_i^{-1})^{\alpha\beta} \mu_i^\alpha \mu_i^\beta + \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i} T_{ij}^{\alpha\beta} \mu_i^\alpha \mu_j^\beta + \frac{1}{2} \sum_{i=1}^N T_{i,\text{self}}^{\alpha\beta} \mu_i^\alpha \mu_i^\beta, \quad (34)$$

where the polarizabilities  $\alpha_i \in \mathbb{R}^{3 \times 3}$  describe the linear response to an electric field of the contribution of atom  $i$  to the density of charge. Recall that the first term represents the interactions between the inducing field and the dipoles, and the others the interactions between dipoles.

Finally, the  $3N \times 3N$  polarization matrix  $\mathbf{T}$ , takes the form:

$$\mathbf{T} = \begin{pmatrix} \alpha_1^{-1} + T_{1,\text{self}} & T_{12} & T_{13} & \cdots & T_{1N} \\ T_{21} & \alpha_2^{-1} + T_{2,\text{self}} & T_{23} & \cdots & T_{2N} \\ T_{31} & T_{32} & \ddots & & \\ \vdots & \vdots & & & \vdots \\ T_{N1} & T_{N2} & \cdots & \alpha_N^{-1} + T_{N,\text{self}} \end{pmatrix}. \quad (35)$$

### 3.2 Forces

The forces associated with the polarization energy are obtained by differentiating the latter with respect to the positions of the atoms. At this point, we should remind the reader that not only does  $\mathcal{E}_{\text{pol}}$  depends on the atomic positions but so does the minimizer  $\boldsymbol{\mu}$ . We thus obtain:

$$-F_k^\alpha = \frac{d\mathcal{E}_{\text{pol}}}{dr_k^\alpha} = \frac{\partial \mathcal{E}_{\text{pol}}}{\partial r_k^\alpha}, \quad k = 1, \dots, N,$$

where we have exploited the variational formulation of the polarization energy. More precisely, we can develop:

$$\begin{aligned} -F_k^\alpha &= -\boldsymbol{\mu}^\dagger \frac{\partial \mathbf{E}}{\partial r_k^\alpha} + \frac{1}{2} \boldsymbol{\mu}^\dagger \frac{\partial \mathbf{T}}{\partial r_k^\alpha} \boldsymbol{\mu} \\ &= -\sum_{i=1}^N \frac{\partial E_i^\beta}{\partial r_k^\alpha} \mu_i^\beta + \frac{1}{2} \sum_{i=1}^N \frac{\partial (\alpha_i^{-1})^{\beta\gamma}}{\partial r_k^\alpha} \mu_i^\beta \mu_i^\gamma + \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i} \frac{\partial T_{ij}^{\beta\gamma}}{\partial r_k^\alpha} \mu_i^\beta \mu_j^\gamma. \end{aligned} \quad (36)$$

The second term of the right hand side of equation (36) originates from the fact that the polarizability tensor needs to be first defined in some molecular frame and then rotated in the lab frame; similar contributions are also present in the first term due to the static dipoles and quadrupoles. Such contributions to the forces require to take the derivatives of the rotation matrices used to switch from the local frame, where the polarizabilities and multipoles are expressed, to the global frame into account. A complete derivation of these terms, which is straightforward, but very cumbersome, can be found in Ref. 34.

### 3.3 Iterative Schemes to Solve the Polarization Equations

In section 3.1 we introduced the polarization equations (20) which require to invert the  $3N \times 3N$  polarization matrix  $\mathbf{T}$  to compute the induced dipoles. As  $N$  can be very large it is usually not possible to use exact methods such as LU or Cholesky factorizations whose computational cost scales as  $N^3$  (for a  $N \times N$  matrix) and which require quadratic storage. A more convenient strategy is to use an iterative method. In a recent paper<sup>34</sup> we explored various iterative strategies to solve the polarization equations. We suggested that two methods seem to be especially suitable for the computation of the induced dipoles within a parallel implementation: the Preconditioned Conjugate Gradient (PCG) method and the Jacobi method coupled with Direct Inversions in the Iterative Subspace (DIIS).

We observed that the spectrum of the Polarization Matrix with Periodic Boundary Conditions computed with  $\text{SPME}$  has in general almost the same structure as for nonperiodic

boundary conditions, both methods have the same rate of convergence for our particular problem as what was described in our previous paper. The convergence of such methods is shown in figure 2 in comparison with the Jacobi Over-Relaxation method (JOR) which has traditionally been used in conjunction with the AMOEBA force field. Similarly to what was observed in direct space computation, both the JI/DIIS and PCG exhibit good convergence properties. Notice that the simple Jacobi iterations are not convergent. Although the rate of convergence is the same as for the previously studied direct space case, the decomposition of the polarization matrix  $\mathbf{T}$  in three matrices (four with the diagonal) with particular properties makes the parallel implementation of the solvers different.

## 4 Parallel Implementation

Let us recall the steps of both the PCG and JI/DIIS iterative methods in order to comment their parallel implementations with MPI.

Starting from an initial guess  $\boldsymbol{\mu}_0$ , they define a new set of induced dipoles at each iteration. In the case of the JI/DIIS solver, the new induced dipoles are obtained after one block-type Jacobi iteration followed by a DIIS extrapolation. This extrapolation requires one to assemble the so called DIIS matrix which is done by making scalar products of the past increments with the newest one.<sup>50</sup>

Let  $\varepsilon$  be the convergence threshold and *maxit* the maximum number of iterations. Also, let  $\mathbf{O}$  be the off-diagonal part of the polarization matrix  $\mathbf{T}$  and  $\boldsymbol{\alpha}$  be the  $(3N \times 3N)$  block-diagonal matrix with the polarizability tensors of the atoms of the system on its diagonal.

Furthermore, let  $\mathbf{T}_{self}$  be the  $(3N \times 3N)$  block-diagonal matrix collecting the self terms, and  $\mathbf{D}$  the  $(3N \times 3N)$  block-diagonal part of  $\mathbf{T}$  given by:  $\mathbf{D} = \boldsymbol{\alpha}^{-1} + \mathbf{T}_{self}$  so that  $\mathbf{T} = \mathbf{O} + \mathbf{D}$ . Then JI/DIIS can be summarized as follows:

```

while it < maxit and inc >  $\varepsilon$  do

   $\tilde{\boldsymbol{\mu}}_{it+1} = \mathbf{D}^{-1} (\mathbf{E} - \mathbf{O}\boldsymbol{\mu}_{it})$ : Jacobi step

  inc =  $\|\tilde{\boldsymbol{\mu}}_{it+1} - \tilde{\boldsymbol{\mu}}_{it}\|_{l^2(\mathbb{R}^{3N})}$ 

   $\boldsymbol{\mu}_{it+1} \leftarrow \tilde{\boldsymbol{\mu}}_{it+1}$ : DIIS extrapolation

end while

```

The procedure is slightly more complicated for the conjugate gradient method because the new set of induced dipoles are obtained by updating them using a descent direction.

Let  $\mathbf{p}_{it}$  be such a descent direction at iteration *it*,  $\mathbf{r}_{it}$  the residual at the same iteration and  $\mathbf{p}_0 = -\mathbf{r}_0$ .

Then, conjugate gradient is described by the following pseudo code (preconditioned conjugate gradient only differs by the fact that  $\mathbf{T}$  is replaced by  $\mathbf{D}^{-1}\mathbf{T}$  and  $\mathbf{E}$  by  $\mathbf{D}^{-1}\mathbf{E}$ ):

```

while it < maxit and  $\|\mathbf{r}_{it}\|_{l^2(\mathbb{R}^{3N})} \geq \varepsilon$  do

```

$$\gamma_{it} = \frac{\mathbf{r}_{it}^T \mathbf{r}_{it}}{\mathbf{p}_{it}^T \mathbf{T} \mathbf{p}_{it}}$$

$\boldsymbol{\mu}_{it+1} = \boldsymbol{\mu}_{it} + \gamma_{it} \mathbf{p}_{it}$  : the new dipoles along the descent direction  $\mathbf{p}_{it}$

$\mathbf{r}_{it+1} = \mathbf{r}_{it} + \gamma_{it} \mathbf{T} \mathbf{p}_{it}$  : the updated residual

$$\beta_{it+1} = \frac{\mathbf{r}_{it+1}^T \mathbf{r}_{it+1}}{\mathbf{r}_{it}^T \mathbf{r}_{it}}$$

$\mathbf{p}_{it+1} = -\mathbf{r}_{it+1} + \beta_{it+1} \mathbf{p}_{it}$  : new descent direction

end while

In both cases, matrix-vector products are required at each iteration, these involve the current values of the dipoles for the JI/DIIS and the descent direction for the Conjugate Gradient.

In terms of a parallel implementation, the parts that require communication between processes are these matrix-vector products (communication of the dipoles or of the descent direction) and the scalar products (reductions).

A comparison between the two solvers reveals that, because of the global reductions needed to compute the DIIS matrix at each iteration, the JI/DIIS might in principle be less suited for parallelization than the PCG. There are two intertwined considerations that need to be stated. First, our implementation is based on non-blocking communication, both for the dipoles/descent directions and for the global reductions. This means that the MPI communication calls return immediately even if the processes involved are not complete: communication is performed while the computation is still going. Notice that a “MPI\_Wait” or “MPI\_Probe” has to be executed after the computation in order to make sure that all the processes received the needed information. The use of non-blocking communication allows one to efficiently cover the communication time, limiting thus the impact of assembling the DIIS matrix on the performances of the JI/DIIS. Second, as the communication is done before the convergence check, the converged dipoles/descent directions are always broadcast. As one needs to compute the forces in MD simulation, which require the converged dipoles to be assembled, one needs a further communication step in the PCG solver, i.e., the dipoles at convergence, whereas this is not necessary for the JI/DIIS, as the converged dipoles are already communicated. These two considerations combined explain the slightly superior performance of the JI/DIIS solver.

Note that we have improved our last parallel implementation of the solvers (that was coupled with direct space computations only) so that the only important difference between them is an additional round of communications (of the converged induced dipoles) in the case of the PCG. For this reason, we expect them to have a similar parallel behaviour.

#### 4.1 Direct Part (real space)

In this section, we first recall the main steps of the previously studied parallel implementation of the polarization solvers without periodic boundary conditions and then explain the differences with respect to the case of the direct part of the interactions when SPME is used.

When no periodic boundary conditions are imposed, as studied in Ref. 34, no cutoff is used so that global communications of the current dipoles (JI/DIIS) or descent direction (PCG) are mandatory and a particle decomposition, where the atoms are distributed among processes without taking into account their positions, suitable. These global communications, whose number grows quadratically with the number of processes, are then the obvious bottleneck to the effectiveness of the parallel implementation.

Things are more complicated for  $_{\text{SPME}}$  computations. First, the use of a cutoff for the short range real space part of the interactions makes the broadcast of all the dipoles at each iteration unnecessary: at each iteration each process only has to receive the values of the induced dipoles that are at a distance inferior to the real space cutoff.

To take advantage of this, we choose to use a spatial decomposition load balancing where each process is assigned to a region of the elementary cell. This implies that each process computes the induced dipoles arising on every atomic site lying in its part of the elementary cell. For now our algorithm only considers a spatial decomposition into slabs using a 1D-decomposition of one of the three directions and an iterative procedure has been implemented in order to adapt the size of these domains in the case of non homogeneous systems. More advanced load balancing procedures have been proposed in the literature (Ref. 51). Further developments and the tuning of these techniques for the polarization problem is under active investigation in our team and they will be included in future implementations.

To reduce the total number of messages, a good strategy is to send the induced dipoles by block: a process receives the whole part of the induced dipoles vector treated by another process as long as it needs to receive at least one of its values.

Also, we have observed that using Newton's third law, that is to compute the distance involving a pair of atom sites only once, is an important obstacle to a scalable implementation. Indeed, although it allows one to reduce the computations of such distances by half, one has to make additional communication between the processes when using this technique, which has a great impact on the parallel efficiency of the algorithm for large systems and/or when a large number of cores is employed. Furthermore, we observed that the loss in efficiency for a few cores is quickly compensated by the gain in parallel efficiency when a larger number of cores is used.

## 4.2 Reciprocal Space Part

The computation of the reciprocal polarization matrix/induced dipoles matrix vector product can be divided in four steps as explained in section 2.

1. The first one is to fill the multipolar array  $Q^R$ : this can be easily distributed among the processes.
2. The second one is to compute the  $_{\text{FFT}}$  of this array. We use the parallel version of the FFTW library<sup>52</sup> to perform this task. As all the other parallel  $_{\text{FFT}}$  libraries, the large number of communications that it requires limits its parallel scaling to a relatively low number of cores for the 3D grids normally used for the PME method (up to

128×128×128 for example). The impact of this can be limited by increasing the real space cutoff while decreasing the size of the FFT grid in order to keep the same accuracy for the PME while limiting the proportional time spent doing FFTS.

Furthermore, as a 1D decomposition is used again to split the work among the processors (but this time the reciprocal space is divided into slabs), no performance gain can be obtained by using slabs smaller than one.

Note also that the AMOEBA force field<sup>53</sup> requires one to compute two sets of induced dipoles, that correspond to two sets of electric fields created by the permanent multipoles on the atom sites with different scaling parameters.<sup>37</sup> However, the arrays involved in the FFT being all real, it is possible to get all the results by calling only one complex to complex FFT calculation.

As mentioned above, the grid points are distributed among the processes with a 1D decomposition when using the MPI version of the FFTW library. Suppose that  $K_1$ ,  $K_2$  and  $K_3$  are the sizes of the grid axis on the three dimensions the FFT has to be done on. Then each MPI process treats slabs of data of size  $K_1 \times K_2 \times K_{3loc}$ , where the variation of the values of  $K_{3loc}$  is as small as possible between processes in order for the computational load to be well distributed. We require each process to compute the contribution to the grid of the dipoles whose scaled fractional coordinate  $u_{3i}$  belongs to the portion of the grid treated by this process. Depending on how the spatial decomposition is made initially, this may require a few communications between neighboring processes.

As the B-splines are non zero only on a small portion of the grid around the scaled fractional coordinates ( $u_{ci}$ ) of the corresponding atoms, only processes treating a neighboring portion of the grid may have contributions to the array representing the dipoles that overlap each other grids domain. In terms of communication, this means that before calling the FFT routine, each process needs to receive the contributions of its neighboring processes to its part of the grid, and send its contribution to the part of the grid of its neighboring processes. Thus, these local communications are not a limiting factor in terms of parallel scaling.

3. Then the convolution in Fourier space is easily distributed among the processes and the backward FFT can be directly computed in parallel as each process has already its whole portion of the grid.
4. Finally, for the same reasons as above, processes have to communicate their part of the grid to neighboring processes in order to extract the reciprocal potentials by multiplying the grid values with the appropriate B-spline values.

As explained above, reductions have to be done during the iterative procedure. This can be done without affecting the parallel scaling by using the non blocking collective operations from the MPI3 standard that allow one to cover efficiently communications with computations.

### 4.3 Forces

Things are less complicated for the computation of the forces associated with the polarization energy: no iterative procedure is necessary and one can make use of previous computations in order to limit the number of calls to the  $\text{FFT}$  routines. Indeed, as the first step of the computation of the induced dipoles is to compute the electric fields due to the permanent multipoles of the system  $\mathbf{E}$ , the associated multipolar grid  $G^R \star Q^R(\vec{r})$  can be stored and reused to get the derivatives of the reciprocal part of this electric fields by just multiplying this grid by the appropriate derivative of the B-splines at the appropriate grid points. This cannot be done to get the electric fields created by the induced dipoles as the only associated grid that could be stored would be the one associated with the dipoles one iteration before convergence. Thus, it needs to be computed.

## 5 Numerical results

### 5.1 Computational Details

The new solvers were included in our Tinker-HP code that is a stand alone module based on elements of Tinker 6.3 and 7.0 that includes new developments introduced by our lab. The parallel implementation was tested on the Stampede supercomputer of the TACC (Texas Advanced Computing Center) whose architecture consists, for the part we did our tests on, in 6400 nodes with two intel Xeon E5-2680 CPUs with eight cores at 2.7Ghz and 32 GB of DDR3 DIMM RAM. These nodes communicate within a 56 GB/s InfiniBand network. The following numerical results are all based on computations made on this supercomputer. To benchmark our algorithms, we considered three water molecule clusters of different sizes: one of 20,000 molecules (60,000 atoms), one of 32,000 molecules (96,000 atoms) and one of 96,000 molecules (288,000 atoms). In all three cases, the convergence threshold was set to  $10^{-5}$  for the dipoles increment which guarantees for energy conservation during a MD;<sup>34</sup> we used B-splines of degree 5 and a real space cutoff parameter of 1nm. The AMOEBA09 force field was used for every computation. The characteristics of these test systems, together with the PME grid sizes used and the size of the elementary cubic cell, are given in table 1. In view of using many processors, notice that the PME parameters are chosen in order to increase the computational cost of the direct part of the interactions (with better parallel scaling) and decreasing the importance of the reciprocal part, while keeping a good accuracy of the global results. From our tests, we observed that with these parameters, the computation of the polarization energy and of the associated forces represents more than 70% of a full force field single point calculation. We tested our implementation of both solvers and of the forces on the Stampede cluster and verified that we obtained the same results as the public version of TINKER up to the machine precision when the convergence criterion was pushed down to extremely small values. A first implementation showed that the poor parallel scaling of the 3D  $\text{FFT}$  routines limited the global parallel scaling of the algorithm: no gain could be obtained by using more than 64–128 cores. To limit and compensate the impact of the poor parallel scaling of the  $\text{FFT}$  routines, a popular strategy is to take advantage of the independence of the direct and reciprocal contributions by assigning their treatment to separate group of processes,<sup>51,54</sup> in a heterogeneous scheme. The reciprocal space part is faster than the direct space one (by a factor that depends on the parameter  $\beta$  used); however the latter has a better parallel scaling: it is therefore possible to use more



processors for the better scaling part by balancing the distribution of cores in such a way that the reciprocal and direct space computations take the same time doing each their part of the matrix/vector products. The heterogeneous strategy allows us to increase the scalability of our code up to 512 cores.

The scaling of the PCG and JI/DIIS solvers is illustrated in figures 2–4.

The absolute best timings for the solvers are reported in table 2. The best timings obtained with the 6.3 version of TINKER which is parallelized with OpenMP routines and uses a different implementation of the PCG solver are also reported in supporting information for comparison. Both the JI/DIIS and the PCG solvers share a similar parallel behavior; the JI/DIIS solver exhibits a small edge in terms of parallel efficiency, which is explained in section 4, due to the additional communications happening after convergence in order to compute the forces. Nevertheless, the absolute best timings of the two solvers are comparable.

The parallel computation of the forces associated with the polarization energy also naturally benefits from the separate calculation of the direct and reciprocal contributions. The scaling of our implementation is reported in figure 5. The absolute best timings are also reported in table 3, and a comparison with the best timings obtained with the 6.3 version of TINKER can be found in supporting information.

## 5.2 Further performance improvements

All the loops involved in the solvers and in the computation of the forces can be parallelized with OpenMP within each MPI process in order to limit the number of necessary communications for a given number of cores and to take advantage of the shared memory architecture within each node of the supercomputer. Furthermore, the FFTW library also enables to use such a computational paradigm. We found experimentally that the best results using this hybrid OpenMP/MPI paradigm were obtained by using 2 OpenMP threads per MPI process, allowing to improve the overall best timing of the solvers and of the computation of the forces. Notice that this hybrid paradigm can be used to extend the range of scalability of the MPI implementation; however, whenever the MPI implementation has not reached its scaling plateau, the pure MPI code is always more efficient than the hybrid one. Nevertheless, the hybrid implementation allows one to take advantage of a larger number of processors, when available.

The best overall timings obtained with the hybrid implementation are given in table 4 for both solvers and for the forces. The total best timings (energy + forces) are also reported. The numbers in the last column of table 4 give a measure of the overall computational overhead introduced by the use of polarizable force field in a PBC MD simulation: even for a system as large as S3 (almost 300,000 atoms) such an overhead can be reduced to less than a second of wall time by using the JI/DIIS solver together with the new parallel implementation. However, such numbers were computed by using a guess for the dipoles that does not take advantage of previous information, for instance, the values of the dipoles at previous steps of the simulation. As discussed in a recent publication,<sup>34</sup> the results can be further improved by using as a guess the predictor step of Kolafa's always stable predictor

corrector:<sup>55</sup> we report in table 5 the results obtained by using such a guess, which represents our default choice for MD simulations.

In conclusion, our new implementation allows one to treat systems as large as S3 introducing an overhead as little as 0.7s on 1024 cores, which can be further reduced to 0.4s per time step during a MD simulation by using better initial guesses, allowing polarizable MD simulations in PBC to be performed on large and very large systems.

## 6 Conclusion

In this paper, we presented a new implementation of the polarization energy and associated forces for the AMOEBA force field which is suited to perform polarizable MD simulations using periodic boundary conditions on parallel computers. To do so, we studied the coupling of the PCG and JI/DIIS iterative solvers for polarization to SPME for distributed multipoles and discussed their optimal parallel implementation. We stated in the introduction that three ingredients are needed in order to achieve an efficient implementation of a polarizable force field: a fast convergent iterative solver, a fast matrix-vector multiplication technique and an efficient parallel setup. The three points have been addressed in this paper and the resulting implementation tested on large to very large systems. Let us recapitulate the main results of this work. The first two points have been addressed by extending PCG and JI/DIIS<sup>34</sup> to the context of PBC simulations with SPME. Indeed, SPME transforms the involved quantities avoiding the  $\mathcal{O}(N^2)$  computational bottleneck and replacing it by a  $\mathcal{O}(N \log N)$  calculation.

Finally, the paper addresses the third point with an extended discussion of the various technical issues that affect the specific polarization problems. The two iterative solvers have been analyzed in a real-life context, i.e., the one of MD simulations, when the forces have to be computed after the linear equations have been solved. Both solvers are well suited for a parallel implementation, however, JI/DIIS seems slightly superior to PCG for very scalable implementations. Finally, the use of only a portion of the available CPUs to do the various reciprocal space computations, while employing the other processors for the more scalable direct space ones in order to compensate the non-optimal parallel scaling of the FFTs has been addressed. This non-optimal parallel behavior is the main bottleneck of our parallel implementation which is the first to address such issues in a production code for polarizable MD.

The scaling of our implementation is overall quite good up to as many as 512–1024 cores: we would like to point out that a parallel implementation of polarizable force fields is much more challenging than for additive force fields, as solving the polarization linear systems implies a large number of synchronization barriers, one per iteration, in particular, the effect of which can be only partially mitigated by covering communication with computation.

Overall, we illustrated how our new implementation is well suited to treat large and very large systems, extending thus the range of applicability of dipole-based polarizable force fields such as AMOEBA to very large systems: we remark that, by using 1024 cores, as little as 0.4 second per time step is needed to handle the polarization energy and forces for a system

composed of as much as 288,000 atoms, when Kolafa's predictor step is used to provide a guess for the iterative solver.

There are still various open challenges and improvements that need to be addressed for polarizable MD simulations. The use of more advanced load-balancing techniques, in the spirit of what was done for example in GROMACS,<sup>51,56</sup> certainly deserves future investigation. The extension of the presented machinery to other, advanced force fields, such as SIBFA<sup>8</sup> and GEM<sup>42</sup> is another interesting development which is being actively investigated in our groups. On a different side, we have recently proposed a complementary approach, that is, the use of polarizable continuum solvation methods instead of PBC: a comparison between the two approaches, especially for difficult systems such as highly charged metal ions or biological molecules, as well as a comparison of the performances of the two approaches, is in progress. The presented methods will also be included in the FFX package.<sup>57</sup> We will present in an upcoming paper an improved global scaling of the AMOEBA force field within the massively parallel Tinker-HP package based on Tinker 7.

## Acknowledgements

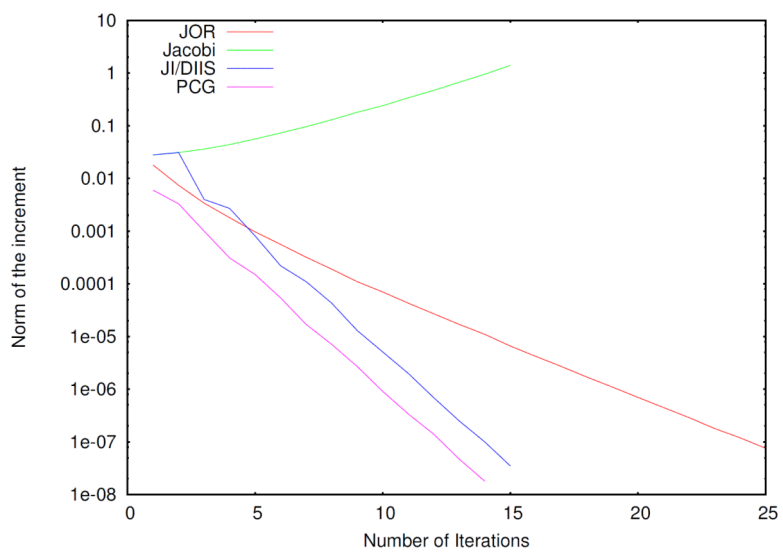
We wish to thank Marie-France Couret for technical support. This work was supported in part by French state funds managed by CalSimLab and the ANR within the Investissements d'Avenir program under reference ANR-11-IDEX-0004-02. Pengyu Ren is grateful for support by the Robert A. Welch Foundation (F-1691) and the National Institutes of Health (GM106137), as well as the high performance computing resources provided by TACC and XSEDE (TG-MCB100057). Funding from French CNRS through a PICS grant between UPMC and UT Austin is acknowledged. Filippo Lipparini gratefully acknowledges the Alexander von Humboldt foundation for funding.

## References

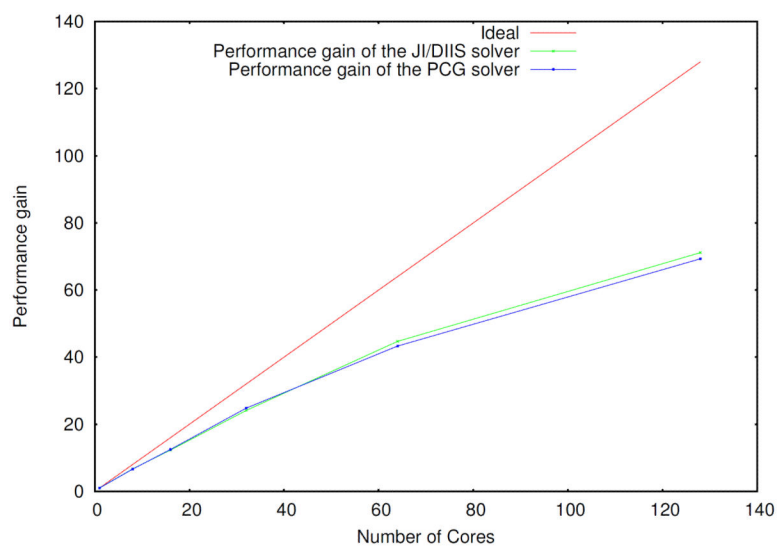
- (1). Jorgensen WL. *J. Chem. Theory Comput.* 2007; 3:1877–1877. and references therein.
- (2). Warshel A, Kato M, Pislakov AV. *J. Chem. Theory Comput.* 2007; 3:2034–2045.
- (3). Cieplak P, Dupradeau F-Y, Duan Y, Wang J. *J. Phys. Condens. Mat.* 2009; 21:333102.
- (4). Lopes PE, Roux B, MacKerell J, Alexander D. *Theor. Chem. Acc.* 2009; 124:11–28. [PubMed: 20577578]
- (5). Piquemal J-P, Jordan KD. *Theor. Chem. Acc.* 2012; 131:1–2.
- (6). Ji C, Mei Y. *Acc. Chem. Res.* 2014; 47:2795–2803. [PubMed: 24883956]
- (7). Cisneros GA, Karttunen M, Ren P, Sagui C. *Chem. Rev.* 2014; 114:779–814. [PubMed: 23981057]
- (8). Gresh N, Cisneros GA, Darden TA, Piquemal J-P. *J. Chem. Theory Comput.* 2007; 3:1960–1986. [PubMed: 18978934]
- (9). Freddolino PL, Harrison CB, Liu Y, Schulten K. *Nature Physics.* 2010; 6:751–758. [PubMed: 21297873]
- (10). Tong Y, Mei Y, Li YL, Ji CG, Zhang JZH. *J. Am. Chem. Soc.* 2010; 132:5137–5142. [PubMed: 20302307]
- (11). Luo Y, Jiang W, Yu H, MacKerell AD, Roux B. *Faraday Discuss.* 2013; 160:135–149. [PubMed: 23795497]
- (12). Huang J, Lopes PEM, Roux B, MacKerell AD. *J. Phys. Chem. Lett.* 2014; 5:3144–3150. [PubMed: 25247054]
- (13). Nielsen CB, Christiansen O, Mikkelsen KV, Kongsted J. *J. Chem. Phys.* 2007; 126:154112. [PubMed: 17461619]
- (14). Kongsted J, Osted A, Mikkelsen KV, Christiansen O. *J. Chem. Phys.* 2003; 118:1620–1633.
- (15). Steindal AH, Olsen JMH, Ruud K, Frediani L, Kongsted J. *Phys. Chem. Chem. Phys.* 2012; 14:5440–5451. [PubMed: 22407300]

- (16). Marini A, Muñoz-Losa A, Biancardi A, Mennucci B. *J. Phys. Chem. B.* 2010; 114:17128–17135. [PubMed: 21128657]
- (17). Jacobson LD, Herbert JM. *J. Chem. Phys.* 2010; 133:154506. [PubMed: 20969402]
- (18). Arora P, Slipchenko LV, Webb SP, DeFusco A, Gordon MS. *J. Phys. Chem. A.* 2010; 114:6742–6750. [PubMed: 20527868]
- (19). Caprasecca S, Curutchet C, Mennucci B. *J. Chem. Theory Comput.* 2012; 8:4462–4473.
- (20). Lipparini F, Barone V. *J. Chem. Theory Comput.* 2011; 7:3711–3724.
- (21). Lipparini F, Cappelli C, Barone V. *J. Chem. Theory Comput.* 2012; 8:4153–4165.
- (22). Lipparini F, Cappelli C, Barone V. *J. Chem. Phys.* 2013; 138:234108. [PubMed: 23802952]
- (23). Boulanger E, Thiel W. *J. Chem. Theory Comput.* 2012; 8:4527–4538.
- (24). Rick SW, Stuart SJ, Berne BJ. *J. Chem. Phys.* 1994; 101:6141–6156.
- (25). Verstraelen T, Speybroeck VV, Waroquier M. *J. Chem. Phys.* 2009; 131:044127. [PubMed: 19655857]
- (26). Piquemal J-P, Chelli R, Procacci P, Gresh N. *J. Phys. Chem. A.* 2007; 111:8170–8176. [PubMed: 17665882]
- (27). Lamoureux G, Roux B. *J. Chem. Phys.* 2003; 119:3025–3039.
- (28). Mills MJ, Popelier PL. *Comp. Theor. Chem.* 2011; 975:42–51.
- (29). Applequist J, Carl JR, Fung K-K. *J. Am. Chem. Soc.* 1972; 94:2952–2960.
- (30). Thole B. *Chem. Phys.* 1981; 59:341–350.
- (31). Wang J, Cieplak P, Li J, Wang J, Cai Q, Hsieh M, Lei H, Luo R, Duan Y. *J. Phys. Chem. B.* 2011; 115:3100–3111. [PubMed: 21391583]
- (32). Ponder JW, Wu C, Ren P, Pande VS, Chodera JD, Schnieders MJ, Haque I, Mobley DL, Lambrecht DS, DiStasio RA, Head-Gordon M, Clark GNI, Johnson ME, Head-Gordon T. *J. Phys. Chem. B.* 2010; 114:2549–2564. [PubMed: 20136072]
- (33). Gordon MS, Slipchenko L, Li H, Jensen JH. *Annu. Rep. Comput. Chem.* 2007; 3:177–193.
- (34). Lipparini F, Lagardère L, Stamm B, Cancès E, Schnieders M, Ren P, Maday Y, Piquemal J-P. *J. Chem. Theory Comput.* 2014; 10:1638–1651.
- (35). Wang W, Skeel RD. *J. Chem. Phys.* 2005; 123:164107. [PubMed: 16268681]
- (36). Sagui C, Pedersen LG, Darden TA. *J. Chem. Phys.* 2004; 120:73–87. [PubMed: 15267263]
- (37). Lipparini F, Lagardère L, Raynaud C, Stamm B, Cancès E, Mennucci B, Schnieders M, Ren P, Maday Y, Piquemal J-P. *J. Chem. Theory Comput.* 2015; 11:623–634.
- (38). Caprasecca S, Jurinovich S, Lagardère L, Stamm B, Lipparini F. *J. Chem. Theory Comput.* 2015; 11:694–704.
- (39). Ewald PP. *Ann. Phys.* 1921; 369:253.
- (40). Darden T, York D, Pedersen L. *J. Chem. Phys.* 1993; 98:10089–10092.
- (41). Essmann U, Perera L, Berkowitz ML, Darden T, Lee H, Pedersen LG. *J. Chem. Phys.* 1995; 103:8577–8593.
- (42). Cisneros GA, Piquemal J-P, Darden TA. *J. Chem. Phys.* 2006; 125:184101. [PubMed: 17115732]
- (43). Smith ER. *Proc. R. Soc. A.* 1981; 375:475–505.
- (44). Aguado A, Madden PA. *J. Chem. Phys.* 2003; 119:7471–7483.
- (45). Toukmaji A, Sagui C, Board J, Darden T. *J. Chem. Phys.* 2000; 113:10913–10927.
- (46). Nyman TM, Linse P. *J. Chem. Phys.* 2000; 112:6152–6160.
- (47). Sala J, Guàrdia E, Masia M. *J. Chem. Phys.* 2010; 133:234101. [PubMed: 21186852]
- (48). Sagui C, Pedersen LG, Darden TA. *J. Chem. Phys.* 2004; 120:73–87. [PubMed: 15267263]
- (49). Smith W. *CCP5 Newsletter.* 1998; 46:18–30.
- (50). Rohwedder T, Schneider R. *J. Math. Chem.* 2011; 49:1889–1914.
- (51). Hess B, Kutzner C, Van Der Spoel D, Lindahl E. *J. Chem. Theory Comput.* 2008; 4:435–447.
- (52). Frigo, M.; Johnson, SG. *FFTW: An adaptive software architecture for the FFT.* 1998.
- (53). Ponder JW, Wu C, Ren P, Pande VS, Chodera JD, Schnieders MJ, Haque I, Mobley DL, Lambrecht DS, DiStasio RA, Head-Gordon M, Clark GNI, Johnson ME, Head-Gordon T. *J. Phys. Chem. B.* 2010; 114:2549–2564. [PubMed: 20136072]

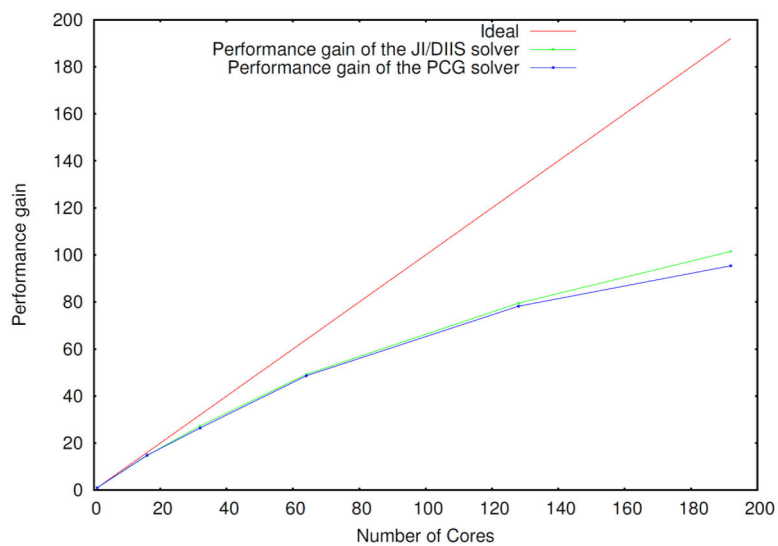
- (54). Phillips JC, Braun R, Wang W, Gumbart J, Tajkhorshid E, Villa E, Chipot C, Skeel RD, Kale L, Schulten K. *J. Comput. Chem.* 2005; 26:1781–1802. [PubMed: 16222654]
- (55). Kolafa J. *J. Comput. Chem.* 2004; 25:335–342. [PubMed: 14696069]
- (56). Pronk S, Páll S, Schulz R, Larsson P, Bjelkmar P, Apostolov R, Shirts MR, Smith JC, Kasson PM, van der Spoel D, Hess B, Lindahl E. *Bioinformatics.* 2013; 29:845–854. [PubMed: 23407358]
- (57). Schnieders MJ, Fenn TD, Pande VS. *J. Chem. Theory Comput.* 2011; 7:1141–1156.



**Figure 1.** Norm of the increment as a function of the number of iterations for Ubiquitin and different iterative methods.

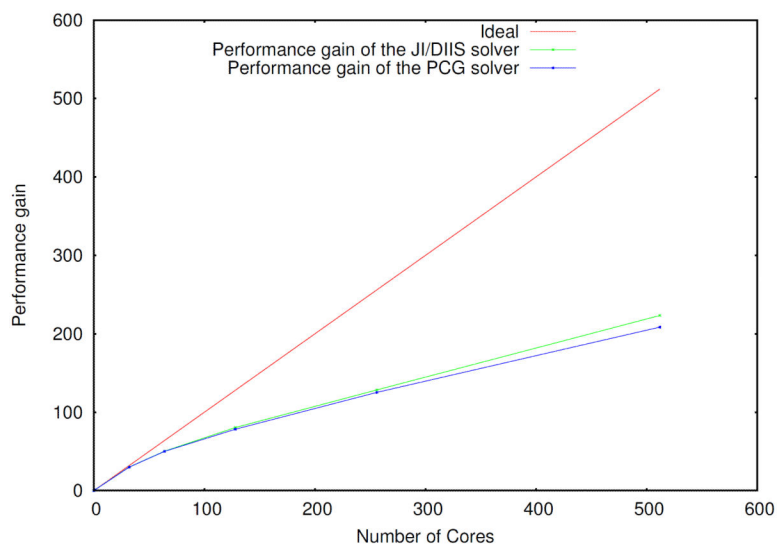


**Figure 2.** Parallel scaling of the induced dipoles calculation (PCG and JI/DIIS) for the S1 system, using a separate group of processes to compute the reciprocal space contribution

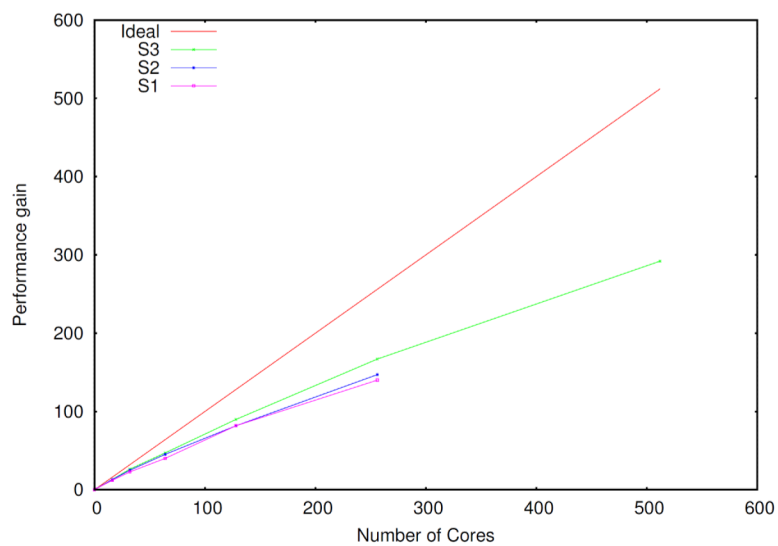


**Figure 3.** Parallel scaling of the induced dipoles calculation (PCG and JI/DIIS) for the S2 system using a separate group of processes to compute the reciprocal space contribution





**Figure 4.** Parallel scaling of the induced dipoles calculation (PCG and JI/DIIS) for the S3 system using a separate group of processes to compute the reciprocal space contribution



**Figure 5.** Parallel scaling of the computation of the forces associated to the polarization energy for the S1, S2 and S3 systems using a separate group of processes to compute the reciprocal space contribution. Notice that negligible performance gains were observed for S1 and S2 when increasing the number of threads beyond 256.

**Table 1**

Characteristics of the test systems used to benchmark the parallel implementation and sizes of the FFT grids. Box size is the size of the edge of the cubic box, in nanometers

System	Number of atoms	Box size	Grid Size
S1	60,000	8.40	$72 \times 72 \times 72$
S2	96,000	9.85	$80 \times 80 \times 80$
S3	288,000	14.30	$128 \times 128 \times 128$

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

**Table 2**

Absolute timings (in seconds) and number of cores used (in parentheses) for the PCG and JI/DIIS parallel solvers in our implementation (TINKER HP).

System	PCG	JI/DIIS
S1	0.45 (128)	0.44 (128)
S2	0.50 (192)	0.47 (192)
S3	0.92 (512)	0.88 (512)

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

**Table 3**

Absolute best timings (in seconds) and number of cores used (in parentheses) for the computation of the forces associated to the  $_{\text{AMOEBa}}$  polarization energy in our (TINKER HP) implementation

System	TINKER HP
S1	0.07 (256)
S2	0.10 (256)
S3	0.20 (512)

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

**Table 4**

Best Hybrid MPI/OpenMP timings (in seconds) and maximum number of CPU cores for the solvers and for the polarization forces. The overall dipoles plus forces best timings is also reported.

System	Cores	Solver		Forces	Total
		PCG	J1/DIIS		
S1	256	0.32	0.30	0.04	0.34
S2	512	0.35	0.35	0.08	0.43
S3	1024	0.61	0.60	0.10	0.70

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

**Table 5**

Best Hybrid MPI/OpenMP timings (in seconds) and maximum number of CPU cores for the solvers and for the polarization forces by using the predictor step of Kolafa's always stable predictor corrector integrator as an initial guess. The overall dipoles plus forces best timings is also reported.

System	Cores	Solver		Forces	Total
		PCG	J1/DIIS		
S1	256	0.16	0.15	0.04	0.19
S2	414	0.18	0.18	0.08	0.26
S3	1024	0.32	0.30	0.10	0.40

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript