

Particle-Particle, Particle-Scaling function (P3S) algorithm for electrostatic problems in free boundary conditions.

Alexey Neelov, S. Alireza Ghasemi, Stefan Goedecker

Institute of Physics, University of Basel

Abstract

An algorithm for fast calculation of the Coulombic forces and energies of point particles with free boundary conditions is proposed. Its calculation time scales as $N \log N$ for N particles. This novel method has lower crossover point with the full $O(N^2)$ direct summation than the Fast Multipole Method. The forces obtained by our algorithm are analytical derivatives of the energy which guarantees energy conservation during a molecular dynamics simulation.

1. Introduction

The computation of the electrostatic or gravitational interaction of a large number N of point particles

$$U = \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N \frac{Q_i Q_j}{r_{ij}}, \quad r_{ij} \equiv |\mathbf{r}_i - \mathbf{r}_j|. \quad (1)$$

is a central problem in many fields of physics, such as molecular dynamics and astrophysics.

We are interested in the case of free boundary conditions.

The systems of interest nowadays typically contain $10^3 - 10^6$ particles. The direct summation approach requires $N^2/2$ force computations, which becomes prohibitively large already for moderate N . With periodic BC one should in addition take the infinite number of periodic images into account; this is clearly impossible.

Therefore one needs an approximate solver.

The table below shows how different algorithms satisfy the criteria of a good approximate coulombic interaction solver. The P3S algorithm of our group will be described below.

Energy is conserved by an algorithm if the approximate forces are equal to (minus) derivatives of the approximate energy. If this is true then the algorithm in question can be used for MD simulation in the microcanonical ensemble. The otherwise very good Fast Multipole Method (FMM) does not fulfill this criterion except in the limit of very high precision.

The P3S algorithm developed in our group will be described in this poster. The M&T (Martina and Tuckerman, [1]) method is similar to ours in that it uses FFTs. However, its accuracy decreases near the border of the cell; therefore bigger cells are needed. In other words, the "uniform error distribution" criterion is not satisfied.

	Cutoff	FMM	M&T	P3S
Scaling	N	N	$N \log N$	$N \log N$
Small prefactor	✓	×	×	✓
Absence of artifacts	×	×	✓	✓
Energy conservation	✓	×	✓	✓
Uniform error distrib.	✓	✓	×	✓
High accuracy	×	✓	✓	✓

2. The Ewald-type methods

In the periodic BC the Ewald method has been used for many years. Following [2], one can also use the Ewald method in free BC, by rewriting (1) as

$$U = E_{\text{short}} + E_{\text{long}} - E_{\text{self}}, \quad (2)$$

$$E_{\text{short}} = \frac{1}{2} \sum_{i=1}^N \sum_{j \neq i}^N \frac{Q_i Q_j}{r_{ij}} \text{erfc} \left(\frac{G r_{ij}}{\sqrt{2}} \right);$$

$$E_{\text{long}} = \frac{1}{2} \int \frac{\rho(\mathbf{r}) \rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r} d\mathbf{r}'; \quad \rho(\mathbf{r}) \equiv \sum_{i=1}^N \rho_i(\mathbf{r});$$

$$\rho_i(\mathbf{r}) = Q_i (G^2/\pi)^{3/2} \exp[-G^2 \mathbf{r}^2]$$

where E_{self} is a known constant dependent only on G .

In the above equations, the terms in E_{short} are short-ranged and thus appropriate for a cutoff approximation; therefore the calculation time of E_{short} scales linearly with N . The short range part can also be efficiently parallelised.

One can consider E_{long} as the energy of a smooth charge distribution $\rho(\mathbf{r})$. Finding this energy in free BC is a nontrivial task.

3. The P3S method.

One can use Poisson solvers appropriate to free BC to calculate the potential $\int \frac{\rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r}'$ in E_{long} . In particular we used a parallel FFT-based solver [3] developed in our group. This solver uses the following interpolation of the density:

$$\rho(\mathbf{r}) \approx \sum_i \rho(ih) \Phi_i(\mathbf{r}), \quad \mathbf{i} \equiv (i_1, i_2, i_3), \quad (3)$$

$$\Phi_i(\mathbf{r}) = \Phi(x/h - i_1) \Phi(y/h - i_2) \Phi(z/h - i_3),$$

where the $\Phi(x)$ is a smooth function called Deslaurier-Dubuc (lazy) scaling function. The basis of lazy scaling functions of order L can interpolate exactly a polynomial of degree $L - 1$. We are using scaling functions of orders up to 100 that interpolate Gaussians very accurately. The scaling

function of order 100 is shown on Fig. 1. Note that the lazy scaling functions are cardinal, i.e., $\Phi(0) = 1$ and $\Phi(i) = 0$ for nonzero integer i .

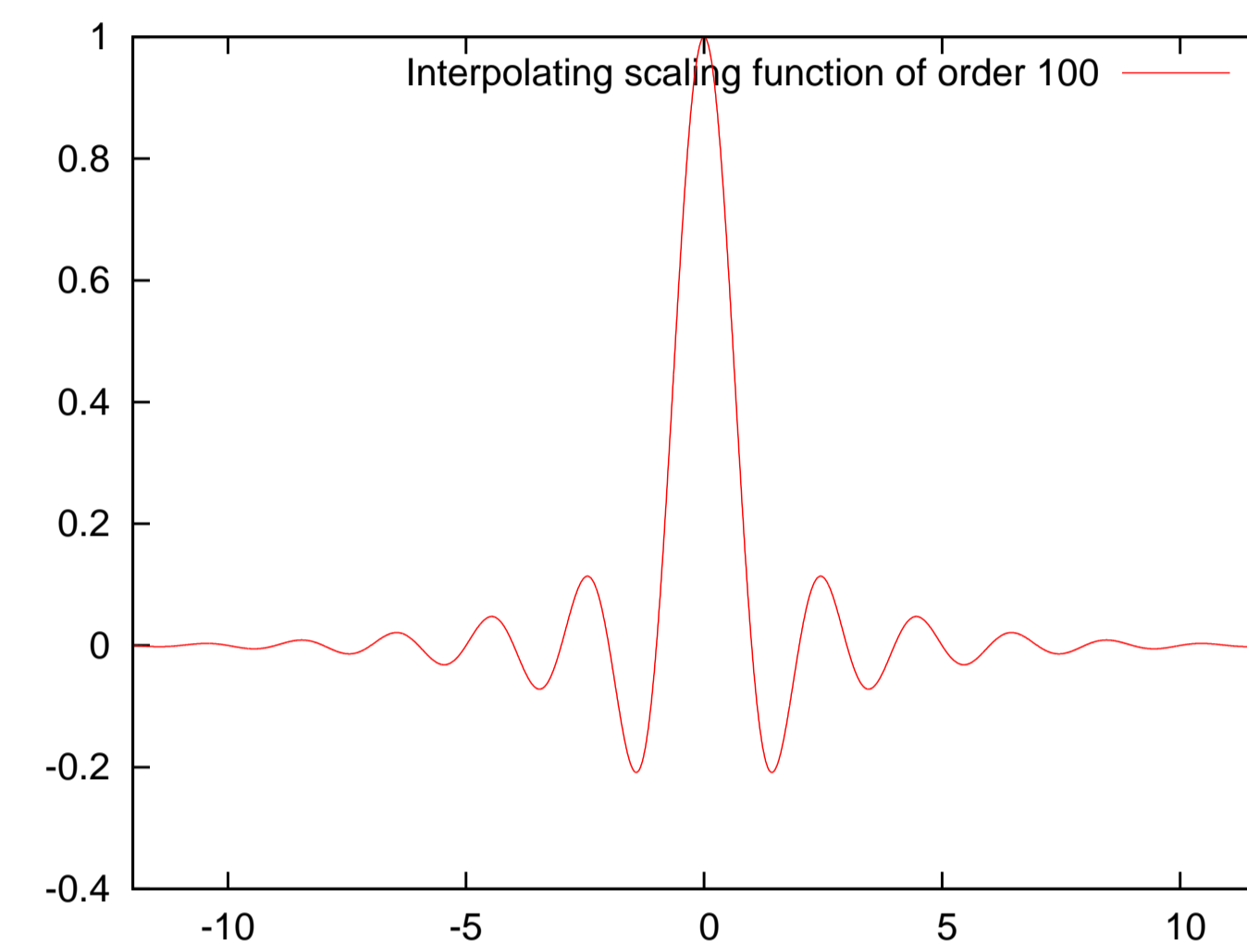


Figure 1: Interpolating scaling function of the order 100.

Consider the potential at grid point ih of a single scaling function centered at the origin:

$$K_i \equiv \int \frac{\Phi_i(\mathbf{r}) d\mathbf{r}}{|\mathbf{r}|} \quad (4)$$

One can show that if the density is approximated by (3), the energy integral E_{long} can be approximated by uniform quadrature integration on the grid. The details can be found in [3]; the result is:

$$E_{\text{long}} \approx \frac{h^5}{2} \sum_{ij} \rho_i \rho_j K_{i-j}. \quad (5)$$

The above expression is a convolution; hence, it can be calculated by FFT-ing the density values from the real space grid, and calculating the product in the reciprocal space. Under most circumstances such as MD simulations the kernel (4) is calculated once in the beginning of the simulation and then only reused.

Our method is thus defined by the relations (3) and (5). Note that Eq. (3) describes putting Gaussians onto a grid, just as it is done in the Fast Fourier Poisson (FFP) algorithm of York and Yang [5]. This can be done fast since the Gaussians are

- separable in Cartesian coordinates, thus we just apply a 1D filter three times.
- well localized, so the filters are short
- spherically symmetric, so we can truncate the charge assignment spherically for optimal performance.

The expressions for the forces can be obtained by differentiating Eq. (2). The short range ones are

standard for the Ewald method. For the long range force on the particle i we get:

$$\mathbf{F}_i^{\text{long}} \approx h^5 \sum_{lj} \mathbf{q}(lh - \mathbf{r}_i) \rho_j K_{l-j};$$

$$\mathbf{q}(\mathbf{r}) = -\frac{2Q_i G^5}{\pi^{3/2}} \mathbf{r} \exp[-G^2 \mathbf{r}^2]$$

Similarly to the FFP method, the above force can be calculated very fast because it only involves the derivative Gaussians that have the same advantages as the Gaussians proper.

Thus, one can obtain the long range forces by our algorithm in the following way:

- Put the Gaussians surrounding the charges to a grid
- FFT the resulting charge density
- Multiply the charge density in the reciprocal space by the Fourier transform of the kernel (4) to get the potential
- FFT the potential back into real space
- Compute the long range forces on particles from the potential by convolution with derivative Gaussians

These steps are in fact similar to the FFP method, with the difference that we have free boundary conditions and calculate the kernel using scaling functions.

4. Results of the serial code

Among others, we chose the following test system: N particles in the unit cube with random coordinates and charges equal to ± 1 , the total charge being zero.

The cutoff radii, grid constant and the value of G were optimized for optimal CPU time at given accuracy.

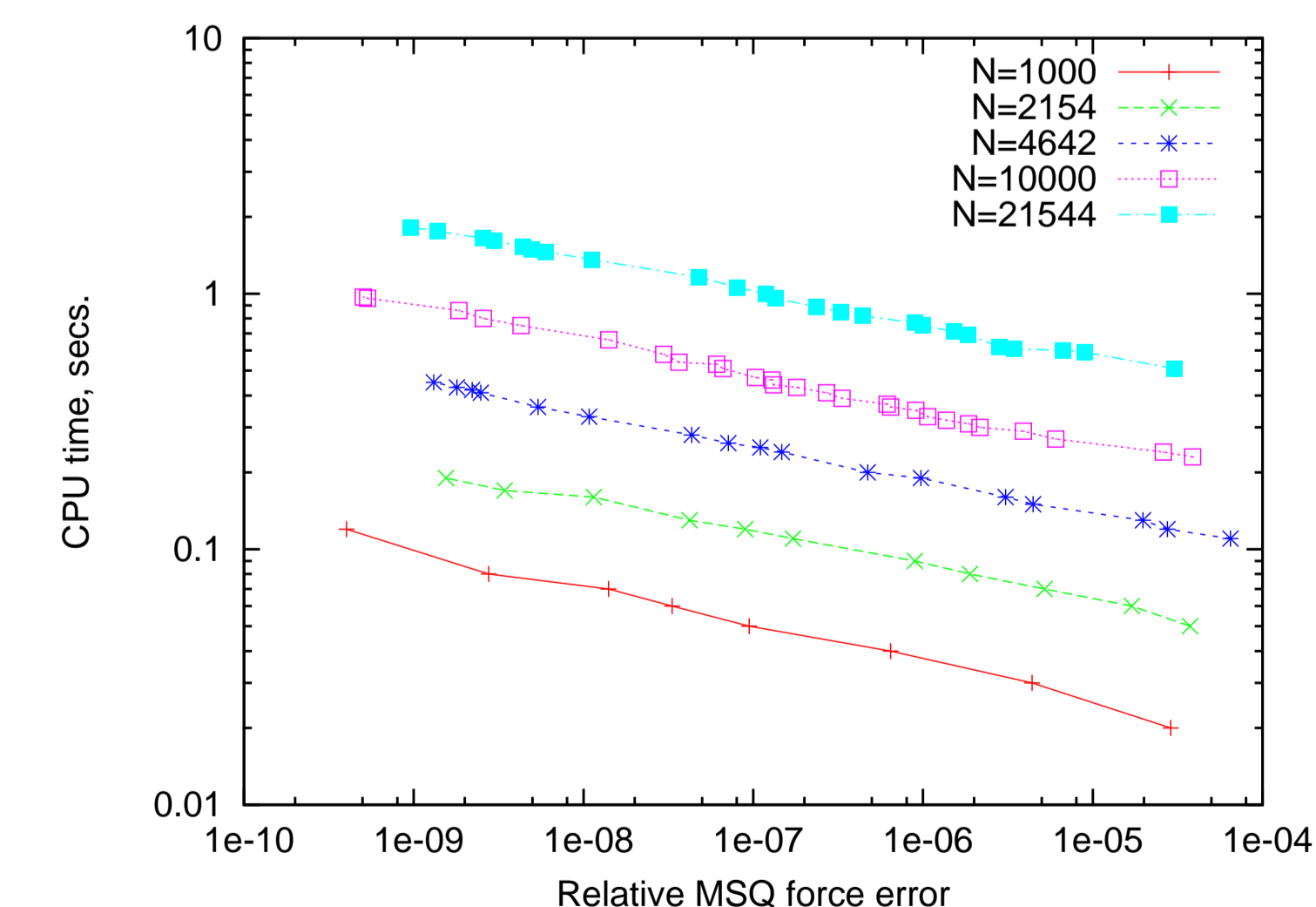


Figure 2: CPU time for combined energy and force calculation vs. accuracy for optimal values of parameters

The fact that the curves for different N are nearly equidistant shows near linear scaling of our algorithm at a broad range of accuracies. From Fig. 2 one can also compute the crossover points with the full direct calculation for a given accuracy, presented on Fig. 3.

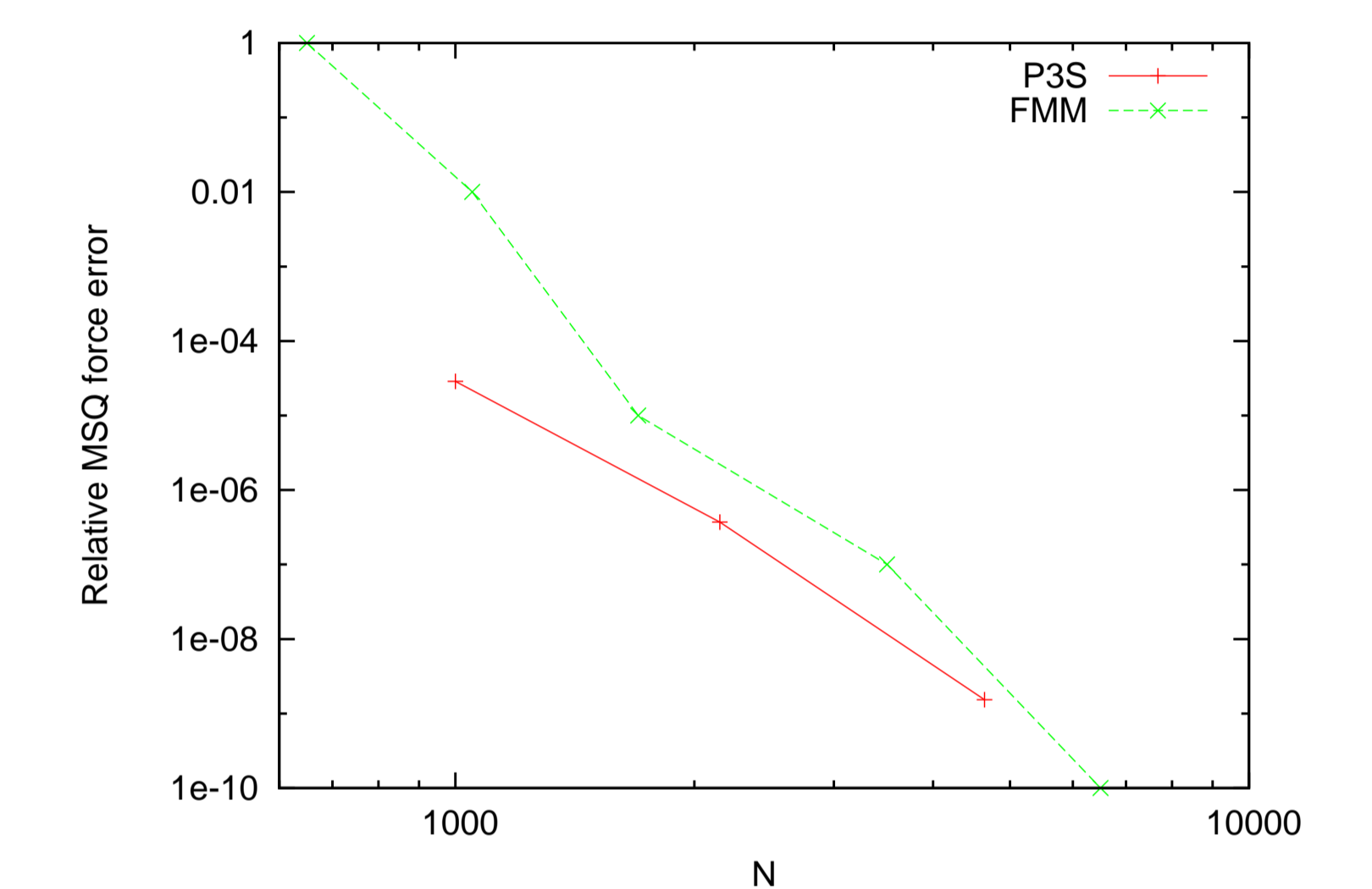


Figure 3: Crossover points with the full direct calculation.

One can see from Fig. 3 that our algorithm has lower crossover points than FMM. The graph of the FMM crossover points is taken from [6]; to our knowledge it is the only paper that provides one.

The parallelisation of our algorithm is similar to that of the P3M or PME. Our code has been parallelised with MPI, and speedups of up to 16 for 32 processors were observed.

References

- [1] G. J. Martyna and M. E. Tuckerman, J. Chem. Phys. **110**, 2810 (1999).
- [2] E. L. Pollock and J. N. Glosli, Comp. Phys. Comm. **95**, 93 (1996).
- [3] L. Genovese, T. Deutsch, A. Neelov, S. Goedecker, and G. Beylkin, J. Chem. Phys. **125**, 074105 (2006).
- [4] A. Neelov, S. A. Ghasemi, S. Goedecker, J. Chem. Phys. **127**, 024109 (2007).
- [5] D. York and W. Yang, J. Chem. Phys. **101**, 3298 (1994)
- [6] C. A. White and M. Head-Gordon, J. Chem. Phys. **101**, 6593 (1994).