

A Fast N-Body Solver for the Poisson(-Boltzmann) Equation

Robert D. Skeel

Departments of Computer Science (and Mathematics)
Purdue University

<http://bionum.cs.purdue.edu/2008December.pdf>

Thesis

- Poisson(-Boltzmann) equation can be solved much faster.
- Using a fast N-body solver as preconditioner is key.
Especially suited for modern computers.
Work with D. Yershov and S. Bond.
- **Multilevel summation is the method of choice**
for molecular biophysics / structural biology.
Work with D. Hardy.

Poisson(-Boltzmann) equation

Generalized Poisson equation with explicit ions.:

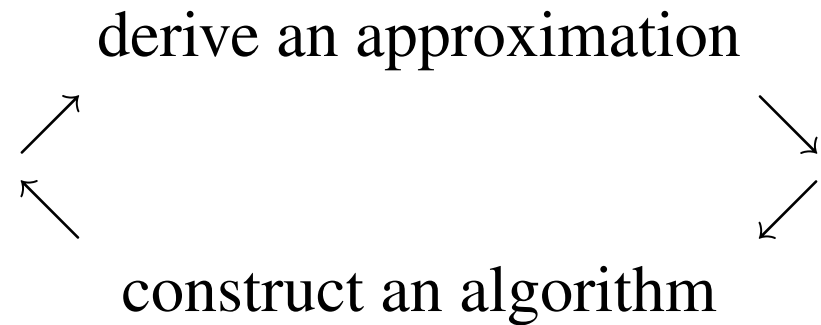
$$-\nabla \cdot (\varepsilon(\vec{r}) \nabla \Phi(\vec{r})) = 4\pi \sum_{i=1}^N q_i \delta(\vec{r} - \vec{r}_i).$$

Extension to other equations possible.

Outline

- I. **Multilevel summation: an approximation**
- II. Multilevel summation: an algorithm
- III. Types of N-body solvers
- IV. Comparison of N-body solvers

Creating a fast N-body solver



Please defer any thoughts about an algorithm until later.

Interactions in 1 dimension

An N-body solver calculates $\mathcal{O}(N^2)$ 2-body nonbonded interactions,

$$U^{\text{el}} = \frac{1}{2} \sum_{i=1}^N \sum_{j=1, j \neq i}^N q_i q_j k(x_i, x_j), \quad k(x, x') = \frac{1}{|x - x'|}$$

where x_i are particle positions and q_i are partial charges.

Matrix form

$$U^{\text{el}} = \frac{1}{2} q^{\text{T}} K q,$$

$$q = \text{charges}, \quad e = K q = \text{potentials}$$

where

$$K_{ij} = \begin{cases} k(x_i, x_j), & j \neq i \\ 0, & j = i \end{cases}$$

Matrix form best reveals structure of algorithms.

Kernel splitting

Separate length scales by splitting

$$k = k_0 + k_1 + \cdots + k_\nu$$

where

(i) the range of k_μ increases with μ ,

but also

(ii) the smoothness of k_μ increases with μ .

(k_1, \dots, k_ν are approximated on grids of increasing coarseness.)

A corresponding matrix splitting:

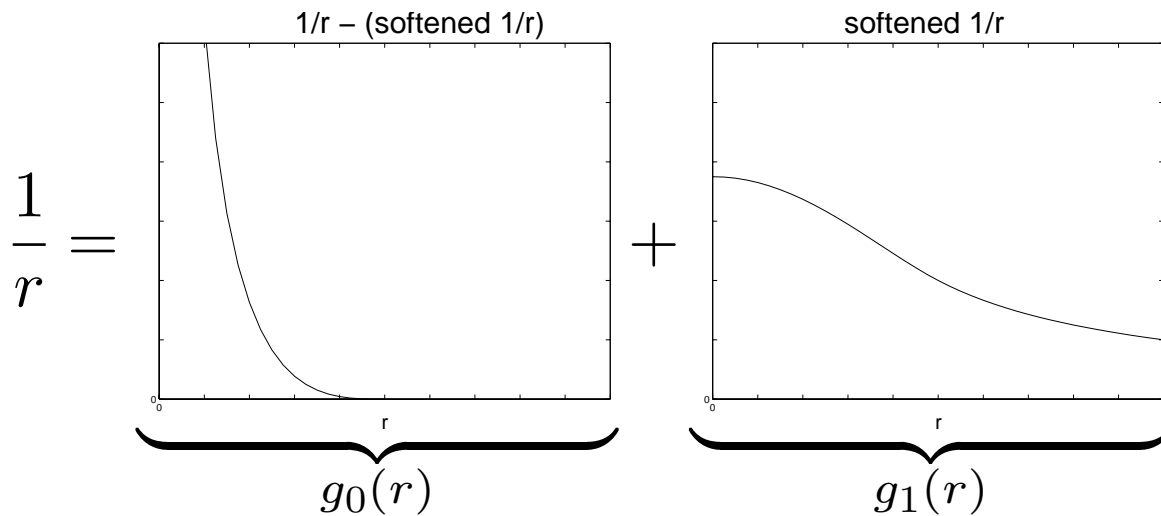
$$K = K_0 + K_1 + \cdots + K_\nu$$

Distant-dependent kernel

If $k(x, x') = g(|x - x'|)$, define $k_\mu(x, x') = g_\mu(|x - x'|)$ where

$$g(r) = g_0(r) + g_1(r) + \cdots + g_\nu(r)$$

e.g.,



Banded matrices

Also, if $0 \leq x_1 < x_2 < \dots < x_n \leq L$,
the values $k_\mu(x_i, x_j) = 0$ for large enough $|i - j|$,
so the matrices K_μ are **banded** with increasing bandwidth.

The slowly varying parts $k_1(x, x'), \dots, k_\nu(x, x')$
are *approximated* ...

Digression: approximation in 1D

Consider $f(x) = k_\mu(x, x')$ with x' an arbitrary parameter.

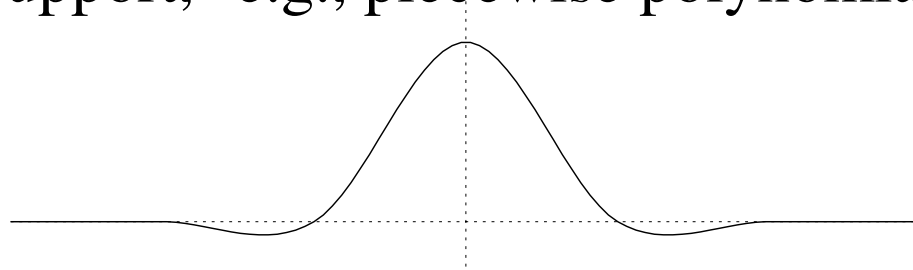
Goal is to approximate $f(x)$ in terms of its values

(and derivs.) on a uniform grid $0 = x_0^h < x_1^h < \dots < x_M^h = L$:

$$f(x) \approx \phi_0(x)f(x_0^h) + \phi_1(x)f(x_1^h) + \dots + \phi_M(x)f(x_M^h)$$

where the $\phi_m(x)$ are basis functions

having “local support,” e.g., piecewise polynomials:



Matrix form of approximation

$$f(x) \approx \phi(x)^\top f^h$$

where

$$\phi(x) = [\phi_0(x), \phi_1(x), \dots, \phi_M(x)]^\top$$

and

$$f^h = [f(x_0^h), f(x_1^h), \dots, f(x_M^h)]^\top.$$

At particle positions

$$\begin{aligned} f(x_1) &\approx \phi(x_1)^\top f^h \\ &\vdots \\ f(x_N) &\approx \phi(x_N)^\top f^h \end{aligned}$$

Approximation operator

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{bmatrix} \approx \begin{bmatrix} \phi(x_1)^\top \\ \vdots \\ \phi(x_N)^\top \end{bmatrix} f^h \stackrel{\text{def}}{=} I_h f^h$$

The operator I_h maps gridded values to particle values.

Only a fixed number, e.g., 4, of nonzeros for each row of I_h .

Approximation in 2D

Analogous to $f(x) \approx \phi(x)^\top f^h$, it can be “shown” that with

$$K_\mu^h = \begin{bmatrix} k_\mu(x_0^h, x_0^h) & k_\mu(x_0^h, x_1^h) & \cdots & k_\mu(x_0^h, x_M^h) \\ k_\mu(x_1^h, x_0^h) & k_\mu(x_1^h, x_1^h) & \cdots & k_\mu(x_1^h, x_M^h) \\ \vdots & \vdots & \ddots & \vdots \\ k_\mu(x_M^h, x_0^h) & k_\mu(x_M^h, x_1^h) & \cdots & k_\mu(x_M^h, x_M^h) \end{bmatrix},$$

we have

$$k_\mu(x, x') \approx \phi(x)^\top K_\mu^h \phi(x')$$

—a **separable approximation**.

Therefore,

tabulation of $k_\mu(x, x') \approx$ tabulation of $\phi(x)^\top K_\mu^h \phi(x')$

and

$$K_\mu = \begin{bmatrix} k_\mu(x_1, x_1) & \cdots & k_\mu(x_1, x_N) \\ \vdots & \ddots & \vdots \\ k_\mu(x_N, x_1) & \cdots & k_\mu(x_N, x_N) \end{bmatrix}$$
$$\approx \begin{bmatrix} \phi(x_1)^\top \\ \vdots \\ \phi(x_N)^\top \end{bmatrix} K_\mu^h \begin{bmatrix} \phi(x_1) & \cdots & \phi(x_N) \end{bmatrix} = I_h K_\mu^h I_h^\top.$$

Multilevel summation approximation

A 4-level summation method splits the kernel as

$$k = k_0 + k_1 + k_2 + k_3,$$

where each term has twice the range of its predecessor.

With interpolation from successively coarser grids,

the corresponding matrix decomposition is

$$K \approx K_0 + I_h K_1^h I_h^\top + I_{2h} K_2^{2h} I_{2h}^\top + I_{4h} K_3^{4h} I_{4h}^\top.$$

Outline

I. Multilevel summation: an approximation

II. Multilevel summation: an algorithm

III. Types of N-body solvers

IV. Comparison of N-body solvers

An $\mathcal{O}(N \log N)$ method

interpolation:

$$q^h = I_h^\top q, \quad q^{2h} = I_{2h}^\top q, \quad q^{4h} = I_{4h}^\top q$$

$$U \approx \frac{1}{2} (q^\top K_0 q + (q^h)^\top K_1^h q^h + (q^{2h})^\top K_2^{2h} q^{2h} + (q^{4h})^\top K_3^{4h} q^{4h}).$$

With $\mathcal{O}(\log N)$ terms

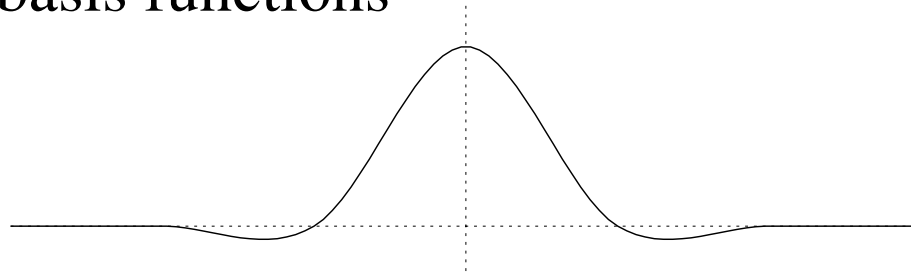
the number of operations per term is only $\mathcal{O}(N)$.

An $\mathcal{O}(N)$ method

For finite element-style interpolation

$$I_{2h} = I_h I_{2h}^h \text{ and } I_{4h} = I_h I_{2h}^h I_{4h}^{2h}.$$

Not quite true for finite difference-style interpolation,
e.g., for nodal basis functions



Substituting and factoring out common factors gives ...

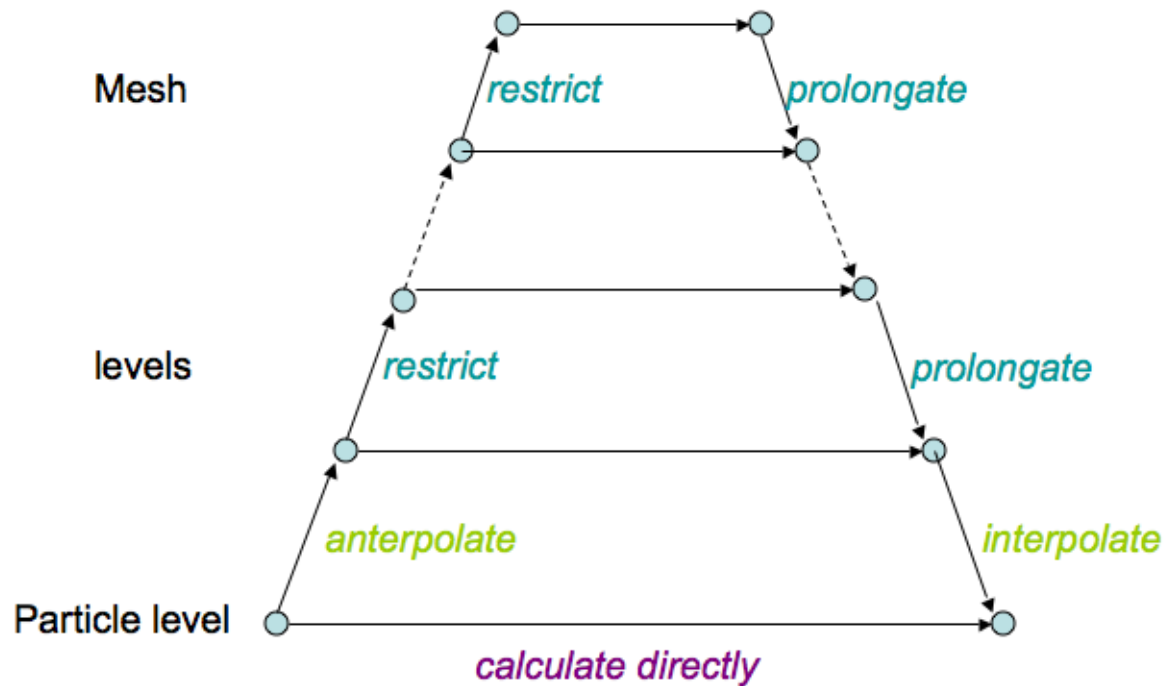
Nested multilevel summation method

$$K \approx K_0 + I_h \left(K_1^h + I_{2h}^h \left(K_2^{2h} + I_{4h}^{2h} K_3^{4h} (I_{4h}^{2h})^\top \right) (I_{2h}^h)^\top \right) I_h^\top.$$

$e \approx Kq$ is computed as follows for a 5-level method:

$$\begin{aligned}
q^h &= I_h^\top q && \mathcal{O}(N) \text{ ops} \\
q^{2h} &= (I_{2h}^h)^\top q^h && \mathcal{O}(M) \text{ ops} \\
q^{4h} &= (I_{4h}^{2h})^\top q^{2h} && \mathcal{O}(M/2) \text{ ops} \\
q^{8h} &= (I_{8h}^{4h})^\top q^{4h} && \mathcal{O}(M/4) \text{ ops} \\
e^{8h} &= K_4^{8h} q^{8h} && \mathcal{O}((M/8)^2) \text{ ops} \\
e^{4h} &= K_3^{4h} q^{4h} + I_{8h}^{4h} e^{8h} && \mathcal{O}(M/4) \text{ ops} \\
e^{2h} &= K_2^{2h} q^{2h} + I_{4h}^{2h} e^{4h} && \mathcal{O}(M/2) \text{ ops} \\
e^h &= K_1^h q^h + I_{2h}^h e^{2h} && \mathcal{O}(M) \text{ ops} \\
e &= K_0 q + I_h e^h && \mathcal{O}(N) \text{ ops}
\end{aligned}$$

Nested multilevel summation method, diagram



Basis for all fast N-body solvers

- separable approximation for kernel $k(x, x')$,
- associativity (and distributivity) of matrix operations.

Outline

- I. Multilevel summation: an approximation
- II. Multilevel summation: an algorithm
- III. Types of N-body solvers
- IV. Comparison of N-body solvers

Nonbonded 2-body interactions

More generally,

$$U^{\text{nb}} = U^{\text{nb}}(\vec{r}_1, \vec{r}_2, \dots, \vec{r}_N) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N{}' q_i q_j k(\vec{r}_i, \vec{r}_j)$$

where \vec{r}_i are particle positions,
the primed sum omits excluded pairs,
kernels other than $|\vec{r} - \vec{r}'|^{-1}$ possible,
periodicity in any dimension possible,
extension to dipoles, etc. possible.

Forces

$$\vec{F}_i^{\text{nb}} = -\nabla_i U^{\text{nb}}(\vec{r}_1, \vec{r}_2, \dots, \vec{r}_N)$$

are best obtained by

differentiating the approximation to $U^{\text{nb}}(\vec{r}_1, \vec{r}_2, \dots, \vec{r}_N)$.

Types of N-body solvers

I. kernel splitting method (**KSM**):
multilevel summation, PME, P³M

II. hierarchical clustering method (**HCM**):
FMM, tree methods

Another classification:

- 2-level: uses FFT for gridded values
- multilevel: uses $\mathcal{O}(\log N)$ levels of grids/cells
- nested multilevel: reduces cost to $\mathcal{O}(N)$

History of multilevel summation

- integral transforms, Brandt & Lubrecht (1990)
- particle monopoles and dipoles in 2D, Sandak (2001)
- C^1 potentials for particles in 3D, Skeel, Tezcan & Hardy (2002)
- generalized Born potentials, Lee, Salsbury & Olson (2004)
- higher order accuracy, analysis, periodicity, Hardy (2006)

Two-level method w/ real space FFT

In $K_1 \approx I_h K_1^h I_h^\top$

$$K_1^h = \begin{bmatrix} k_1(0) & k_1(-h) & \cdots & k_1(-Mh) \\ k_1(h) & k_1(0) & \cdots & k_1(h - Mh) \\ \vdots & \vdots & \ddots & \vdots \\ k_1(Mh) & k_1(Mh - h) & \cdots & k_1(0) \end{bmatrix},$$

is a nonperiodic convolution computable by an FFT
—if the dimension of the matrix is doubled.

Two-level method w/ reciprocal space FFT

Modify $g_1(r)$ to be constant for $r \geq L + \Delta L$ and let $\kappa_1(r)$ be a periodic extension of $g_1(r)$ with period $2(L + \Delta L)$. Truncate Fourier series:

$$k_1(x, x') = \kappa_1(x - x') \approx \overline{\psi(x')}^\top D \psi(x)$$

where D is diagonal matrix and $\psi(x)$ is Fourier basis.

Interpolating from gridded values F of the Fourier basis

$$\psi(x)^\top \approx \phi(x)^\top F$$

hence

$$\kappa_1(x - x') \approx \phi(x')^\top \bar{F}^\top DF \phi(x)$$

Tabulating at particle positions

$$K_1 \approx I_h \bar{F}^\top DF I_h^\top.$$

This is **particle–mesh Ewald** (PME), a variant of an older approach, viz., particle–particle particle–mesh Ewald (P³M).

Hierarchical clustering methods

employ an oct-tree decomposition of space to *partition* the interactions into different levels:

$$K = K_0 + K_1 + K_2 + K_3,$$

with higher level interactions for more separated particles.

Short range directly.

Longer range are partitioned into pairs of particle clusters, each using a different localized (polynomial) approximation for the kernel $k(\vec{r}, \vec{r}')$.

Truncated Taylor expansions

are used in practice.

For $|\vec{r} - \vec{r}'|^{-1}$, each polynomial is harmonic and can be expressed in terms of p^2 spherical harmonics, where $p - 1$ is the degree of the polynomial, instead of $\frac{1}{6}p^3 + \frac{1}{2}p^2 + \frac{1}{3}p$ monomials. The entries of K_1^h, K_2^{2h}, \dots are then multipoles.

Outline

- I. Multilevel summation: an approximation
- II. Multilevel summation: an algorithm
- III. Types of N-body solvers
- IV. Comparison of N-body solvers

KSMs vs. HCMs

Advantages of KSMs over HCMs:

- HCMs yield a U that is a discontinuous function of \vec{r}, \vec{r}' ;
KSMs can attain any degree of continuity.
- KSMs are simpler—they do not need interaction lists.
- KSMs are faster in comparisons I have seen.

Advantage of HCMs over KSMs:

- HCMs can exploit special properties of kernel.
However, this depends on the kernel,
and it can greatly complicate the algorithm.

Nonissue:

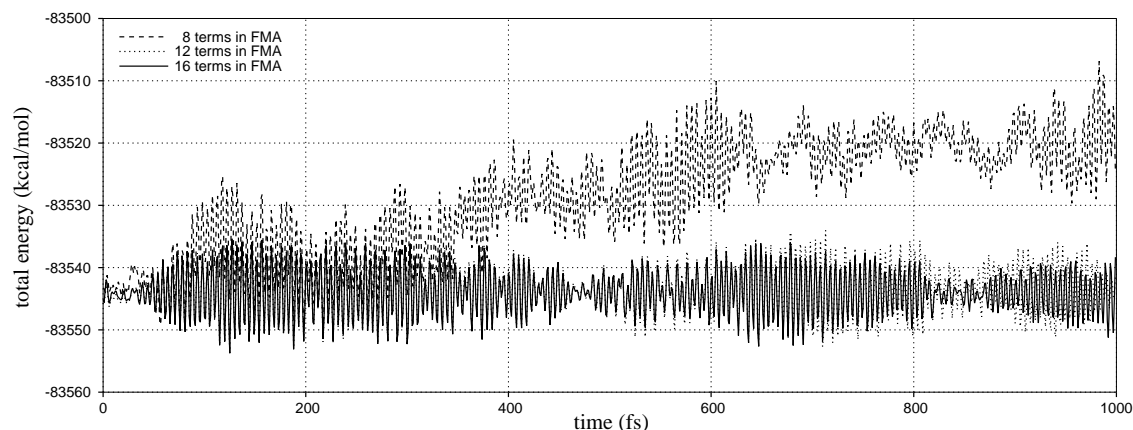
- Only HCMs honor Newton's 3rd Law.

Unknown:

- Is adaptivity as efficient for KSMs as it is for HCMs?

Energy drift for HCMs

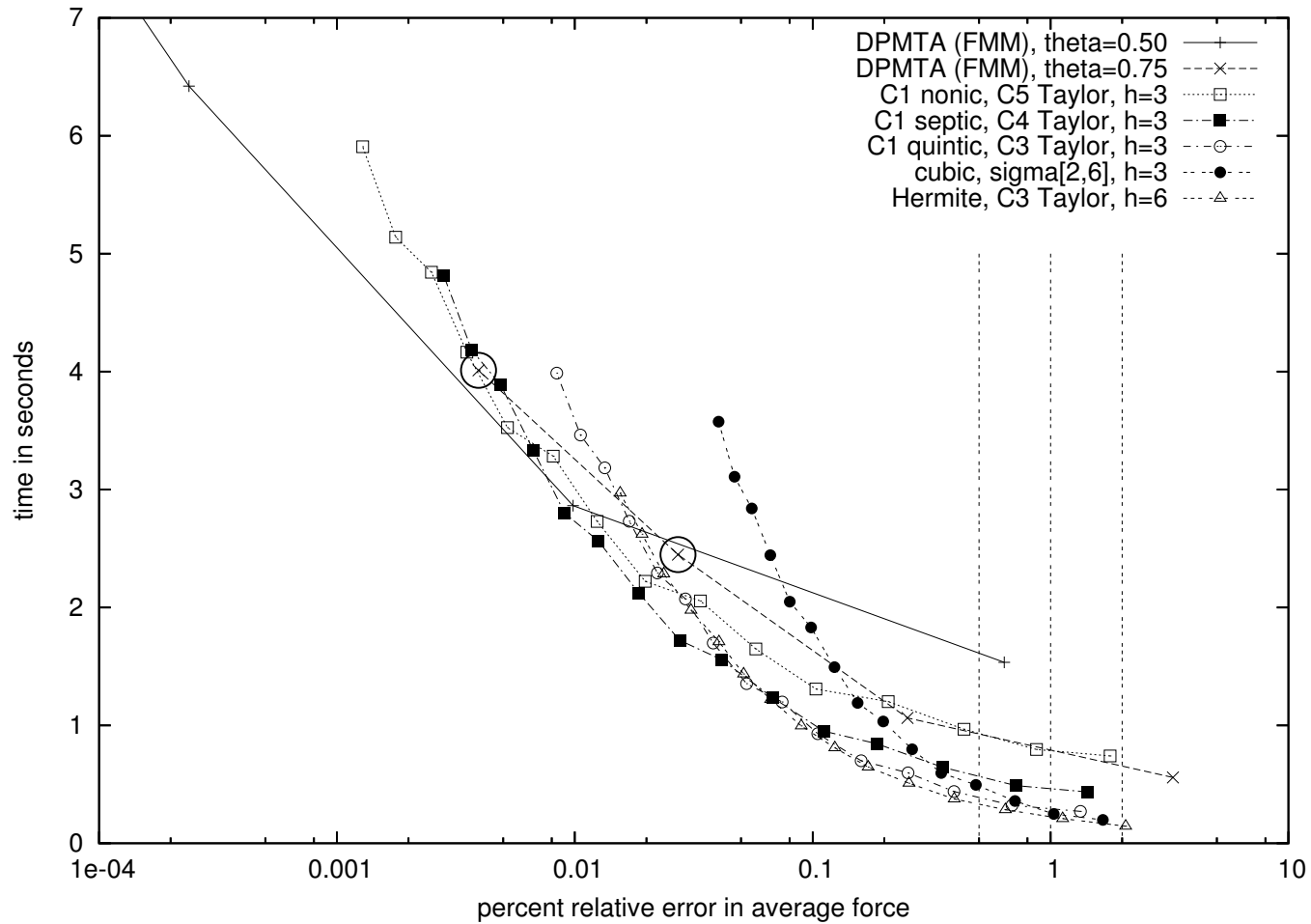
HCMs compute discontinuous forces, so are not usable unless high accuracy is requested.



Our **multisummation method is 11 times faster** than a fast multipole method for MD, partly because the latter needs 12 terms to avoid rapid energy growth.

Multisummation vs. multipole

CPU time vs. error for 10,002 waters



Conservation of linear momentum

HCM approximations yield forces that obey Newton's 3rd Law.

KSM approximations lack this property.

(And produce self forces!)

The usual remedy $\vec{F}_i - (1/N) \sum_j \vec{F}_j$

yields nonconservative forces.

However, the mass-weighted correction

$$\vec{F}_i(\dots) - \frac{m_i}{m_{\text{tot}}} \sum_j \vec{F}_j(\dots)$$

is the gradient of a potential. Skeel, Hardy & Phillips (2007)

Multilevel vs. 2-level

Advantages of multilevel over 2-level methods:

- FFT-based methods require doubling the dimension in each direction that is nonperiodic.
- FFT-based methods are very difficult to parallelize.
- adaptivity not possible for 2-level methods
- multilevel methods are faster in comparisons we have done.

Our multisummation method is **twice as fast** as PME.

Multisummation vs. particle-mesh Ewald

CPU time vs. error for 21,950 waters

