FFTSVD: A Fast Multiscale Boundary-Element Method Solver Suitable for Bio-MEMS and Biomolecule Simulation

Michael D. Altman, Jaydeep P. Bardhan, *Student Member, IEEE*, Bruce Tidor, and Jacob K. White, *Associate Member, IEEE*

***Abstract*—This paper presents a fast boundary-element method (BEM) algorithm that is well suited for solving electrostatics problems that arise in traditional and bio-microelectromechanical systems (bio-MEMS) design. The algorithm, FFTSVD, is Green's-function-independent for low-frequency kernels and efficient for inhomogeneous problems. FFTSVD is a multiscale algorithm that decomposes the problem domain using an octree and uses sampling to calculate low-rank approximations to dominant source distributions and responses. Long-range interactions at each length scale are computed using the FFT. Computational results illustrate that the FFTSVD algorithm performs better than precorrected-FFT (pFFT)-style algorithms or the multipole-style algorithms in FastCap.**

***Index Terms*—Bio-MEMS, biomolecule, boundary element, electrostatic, fast solver, FFTSVD.**

I. Introduction

MICROELECTROMECHANICAL systems (MEMS) have recently become a popular platform for biological experiments because they offer new avenues for investigating the structure and function of biological systems. Their chief advantages over traditional *in vitro* methods are reduced sample requirements, potentially improved detection sensitivity, and structures of approximately the same dimensions as the systems under investigation [1]. Devices have been presented for sorting cells [2], separating and sequencing deoxyribonucleic acid (DNA) [3], and biomolecule detection [4]. Furthermore, because arrays of sensors can be batch-fabricated on a single device, parallel experiments and high-throughput analysis are readily performed. However, since microfabrication is relatively slow and expensive, numerical simulation of MEMS devices is an essential component of the design process [5], [6]. Design tools for integrated circuits cannot address multiphysics problems, and this has motivated the development of several computer-aided MEMS design software packages, most of which are based on the finite-element method (FEM) and the boundary-element method (BEM) [7].

Bio-microelectromechanical MEMS (bio-MEMS), when applied to such problems as biomolecule detection, are often functionalized with receptor molecules that bind targets of interest [8]. Molecular labels can also be used to aid in the detection process [9]. However, the interactions between these molecules, the MEMS device, and the solvent environment are often neglected during computational prototyping. In other fields, such as computational chemistry and chemical engineering, continuum models of solvation are often used to study the electrostatic component of these interactions [10]. These mean-field models permit the efficient calculation of many useful properties, including solvation energies and electrostatic fields [11], [12], and have been shown to correlate well with more expensive calculations that include explicit solvents [13]. However, continuum models are unable to resolve specific molecular interactions between solvent molecules and the solute. A variety of numerical techniques can be used to simulate the continuum models, including the finite difference method (FDM), FEM, and BEM [14]–[16].

The BEM has a number of advantages relative to FDM and FEM, such as requiring only surface discretizations and exactly treating boundary conditions at infinity. However, discretizing boundary integral equations produces dense linear systems whose memory costs scale as $O(n^2)$ and solution costs scale with $O(n^3)$, where n is the number of discretization unknowns. This rapid rise in cost with increasing problem complexity has motivated the development of accelerated BEM solvers. Preconditioned Krylov-subspace techniques, combined with fast algorithms for computing matrix–vector (MV) products, can require as little as $O(n)$ memory and time to solve BEM problems [17]. Many such algorithms have been presented, including the fast multipole method (FMM) [18], [19], $\mathcal{H}$-matrices [20]–[22], the precorrected-fast Fourier transform (pFFT) method [23], wavelet techniques [24], [25], FFT on multipoles [26], [27], kernel-independent multipole methods [28], [29], the hierarchical SVD method [30], [31], plane-wave expansion-based approaches [32], and the

Manuscript received March 5, 2005; revised June 10, 2005. This paper was supported in part by the National Institutes of Health under Grant GM065418 and Grant GM066524, in part by the National Science Foundation, and in part by grants from the Semiconductor Research Corporation, the MARCO Interconnect Focus Center, and the Singapore–MIT Alliance. The work of J. P. Bardhan was supported in part by a Department of Energy Computational Science Graduate Fellowship. This paper was recommended by Associate Editor J. Zeng.

M. D. Altman is with the Department of Chemistry, Massachusetts Institute of Technology, Cambridge, MA 02139-4307 USA.

J. P. Bardhan and J. K. White are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139-4307 USA.

B. Tidor is with the Department of Electrical Engineering and Computer Science and the Biological Engineering Division, Massachusetts Institute of Technology, Cambridge, MA 02139-4307 USA.

Digital Object Identifier 10.1109/TCAD.2005.855946

predetermined interaction list oct-tree (PILOT) algorithm [33]. Some algorithms, such as the original FMM, exploit the decay of the integral-equation kernel; the pFFT method makes use of kernel shift invariance. This paper introduces an algorithm that combines the benefits of both of these approaches, leading to a method that has excellent memory and time efficiency even on highly inhomogeneous problems.

Fast BEM algorithms whose structures depend on kernel decay suffer from a common, well-known problem: computing medium- and long-range interactions is still expensive, even when their numerical low rank is exploited. For instance, in the FMM, computing the multipole-to-local (M2L) products dominates the MV product time, because each cube can have as many as 124 or 189 interacting cubes, depending on the interaction-list definition, and the work per M2L multiplication scales as $O(p^4)$, where p is the expansion order and is related to accuracy [18], [19], [34]. Much work has focused on reducing this cost; for the FMM, plane-wave expansions [32], [35] and diagonalizing the M2L translation, but are typically only efficient for large p. The pFFT algorithm [23] does not rely on the kernel's decay, but rather its translation invariance to achieve high efficiency. The pFFT method is Green's-function independent, even for highly oscillatory kernels. Consequently, the method has been applied in a number of different fields, including wideband-impedance extraction [36], microfluidics [37]–[39], and biomolecule electrostatics [40]. One weakness of the pFFT method is that its efficiency decreases as the problem domain becomes increasingly inhomogeneous [23].

In this paper, we introduce a fast BEM algorithm called FFTSVD. The method is well suited to MEMS-device simulation because it is Green's-function independent and maintains high efficiency when solving inhomogeneous problems. The FFTSVD algorithm is similar to the PILOT algorithm introduced by Gope and Jandhyala [33], in that our algorithm is multiscale and based on an octree decomposition of the problem domain. Similar to PILOT and IES^3, our algorithm uses sampling and QR decomposition to calculate reduced representations for long-range interactions. The FFT is used to efficiently compute the interactions, as in the kernel-independent multipole method [29]. Numerical results from capacitance-extraction problems demonstrate that FFTSVD is more memory efficient than FastCap or pFFT and that the algorithm does not have the homogeneity problem. In addition, we illustrate electrostatic-force analysis by simulating a MEMS comb drive [39]. Finally, we demonstrate the method's kernel independence by calculating the electrostatic free energy of transferring a small fluorescent molecule from the gas phase to an aqueous solution, using an integral formulation of a popular continuum electrostatics model [16], [40].

The following section briefly describes a representative MEMS electrostatics problem, a BEM used to solve the problem, and a more complicated surface formulation for calculating the electrostatic component of the solvation energy of a biomolecule. Section III presents the FFTSVD algorithm. Computational results and performance comparisons appear in Section IV. Section V describes several algorithm variants and summarizes the paper.

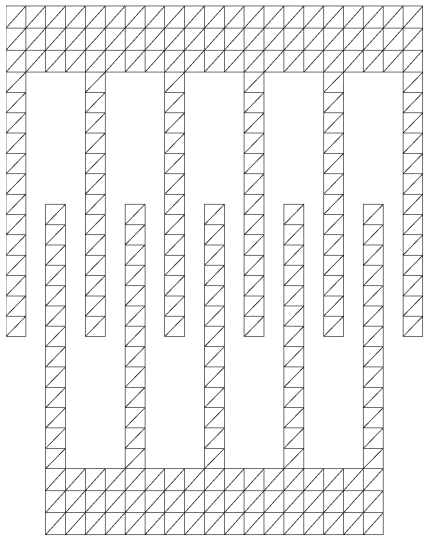


Fig. 1. Electrostatically actuated MEMS comb drive.

II. Background Examples

In this section, we describe two electrostatics problems that arise in Bio-MEMS design and describe how they can be addressed using BEM.

A. MEMS Electrostatic-Force Calculation

Consider the electrostatically actuated MEMS comb drive illustrated in Fig. 1. Two interdigitated polysilicon combs form the drive; one comb is fixed to the substrate and the other is attached to a flexible tether. Applying a voltage difference to the two combs results in an electrostatic force between the two structures, and the tethered comb moves in response [39]. The electrostatic response of the system to an applied voltage difference can be calculated by solving the first-kind integral equation

$$\int_S \sigma(r')G(r;r')dr' = V(r) \tag{1}$$

where S is the union of the comb surfaces, $V(r)$ is the applied potential on the comb surfaces, $G(r;r') = 1/\|r-r'\|$ is the free-space Green's function, and $\sigma(r)$ is the charge density on the comb surfaces. Note that this is a standard capacitance-extraction problem.

We can compute the axial electrostatic force between the combs by the relation

$$F(s) \;=\; -\frac{d}{ds}E = -\frac{d}{ds}\frac{1}{2}V^{\mathrm{T}}C(s)V \tag{2}$$

where $F(s)$ is the force in the axial direction, s is the separation between the combs, E is the electrostatic energy of the system, V is the vector of conductor potentials, and $C(s)$ is the capacitance matrix, written as a function of the comb separation.

To solve (1) numerically, we discretize the surfaces into n_{p} panels and represent $\sigma(r)$, the charge density on the surface, as

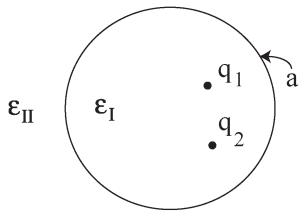


Fig. 2. Continuum model for calculating biomolecule solvation.

a weighted combination of compactly supported basis functions defined on the panels

$$\sigma(r) = \sum_{i=1}^{n_{\mathrm{p}}} x_i f_i(r). \tag{3}$$

Here, $f_i(r)$ is the ith basis function, and x_i, the corresponding weight. Forcing the integral over the discretized surface to match the known potential at a set of collocation points, we form the dense linear system

$$Gx = b. \tag{4}$$

The Green's function matrix G is defined by

$$G_{ij} = \int f_j(r')G(r_i, r')da' \tag{5}$$

where r_i is the ith collocation point and $b_i = V(r_i)$. Alternatively, one can use a Galerkin method, in which case

$$G_{ij} = \int\int f_i(r)f_j(r')G(r; r')drdr' \tag{6}$$

and

$$b_i = \int f_i(r)\psi(r)dr. \tag{7}$$

The linear system of (4) is solved using preconditioned GMRES [41].

B. BEM Simulation of Biomolecule Electrostatics

Electrostatic solvation energy, the cost of transferring a molecule from a nonpolar low-dielectric medium to an aqueous solution with mobile ions, plays an important role in understanding molecular interactions and properties. To calculate solvation energy, continuum electrostatic models are commonly employed. Fig. 2 illustrates one such model. The Richards molecular surface [42] is taken to define the boundary a that separates the biomolecule interior and the solvent exterior. The interior is modeled as a homogeneous region of low permittivity ε_I, where the potential $\varphi(r)$ is governed by the Poisson equation, and partial atomic charges on the biomolecule atoms are modeled as discrete-point charges at the atom centers

$$\nabla^2\varphi(r) = -\sum_{i=1}^{n_{\mathrm{c}}} \frac{q_i}{\varepsilon_I}\delta(r - r_i) \tag{8}$$

where n_{c} is the number of discrete-point charges and q_i and r_i are the ith charge's value and location, respectively. In the solvent region, the linearized Poisson–Boltzmann equation

$$\nabla^2\varphi(r) = \kappa^2\varphi(r) \tag{9}$$

governs the potential, where the inverse Debye screening length κ depends on the concentration of ions in the solution and a higher permittivity ε_{II}. We write Green's theorem in the interior and exterior regions and then enforce continuity conditions at the boundary to produce a pair of coupled integral equations

$$\frac{1}{2}\varphi(r_a) + \fint_a dr'\varphi(r')\frac{\partial G_1}{\partial n}(r_a; r') - \fint_a dr'\frac{\partial\varphi}{\partial n}(r')G_1(r_a; r') = \sum_{i=1}^{n_{\mathrm{c}}} \frac{q_i}{\varepsilon_I}G_1(r_a; r_i) \tag{10}$$

$$\frac{1}{2}\varphi(r_a) - \fint_a dr'\varphi(r')\frac{\partial G_2}{\partial n}(r_a; r') + \frac{\varepsilon_I}{\varepsilon_{II}}\fint_a dr'\frac{\partial\varphi}{\partial n}(r')G_2(r_a; r') = 0 \tag{11}$$

where r_a is a point on the surface, $\fint$ denotes the Cauchy principal-value integral, G_1 is the Laplace Green's function, G_2 is the real Helmholtz Green's function, $(\partial G_i/\partial n)$ denotes the appropriate double-layer Green's function, $\varphi(r)$ is the potential on the surface, and $(\partial\varphi/\partial n)(r)$ is the normal derivative of the potential on the surface. See [16] and [40] for detailed derivations of the formulation. To solve (10) and (11) numerically, we define a set of basis functions on the discretized surface and represent the surface potential and its normal derivative as weighted combinations of these basis functions

$$\varphi(r) \approx \sum_i x_i f_i(r) \tag{12}$$

$$\frac{\partial\varphi}{\partial n}(r) \approx \sum_i y_i f_i(r). \tag{13}$$

We force the discretized integrals to exactly match the known surface conditions at the panel centroids; this produces the dense linear system

$$\begin{bmatrix} \frac{1}{2}I + \frac{\partial G_1}{\partial n} & -G_1 \\ \frac{1}{2}I - \frac{\partial G_2}{\partial n} & +\frac{\varepsilon_I}{\varepsilon_{II}}G_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sum_k \frac{q_k}{\varepsilon_I}G_1(r; r_k) \\ 0 \end{bmatrix} \tag{14}$$

where, denoting the ith panel centroid as r_i, the block matrix entries are

$$G_{1,ij} = \fint f_j(r')G_1(r_i; r')dr' \tag{15}$$

$$\left(\frac{\partial G_1}{\partial n}\right)_{ij} = \fint f_j(r')\frac{\partial G_1}{\partial n(r')}(r_i; r')dr' \tag{16}$$

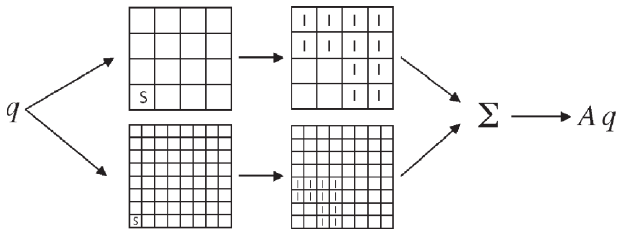


Fig. 3. Multiscale approach to fast matrix multiplication.

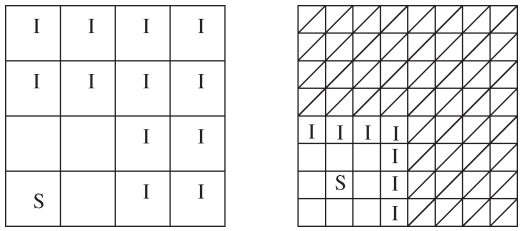


Fig. 4. Interacting squares at two levels of decomposition.

and the block matrices G_2 and $(\partial G_2/\partial n)$ are similarly defined. Note that BEM solution of this problem requires a Green's-function-independent fast algorithm.

III. FFTSVD Algorithm

The FFTSVD is a multiscale algorithm, like most fast algorithms for low-frequency applications: to compute the total action of the integral operator on a vector, we separate its actions at different length scales and compute them separately, combining them only at the end. In describing the FFTSVD algorithm, it is helpful to think of the basis functions as sources, $\int f_i(r')G(r;r')dr'$ as the potential produced by source i, and the collocation points r_i as destinations. Multiplying x by G in (4) is then computing potentials at all the destinations due to all sources. Fig. 3 illustrates the multiscale approach to fast matrix multiplication: the square S denotes a source, and the squares denoted I represent destinations.

A. Notation

Let d and s denote two sets of panels: then $G_{d,s}$ is the submatrix of G that maps sources in s to responses in d. The number of panels in set i is denoted by n_i.

B. Octree Decomposition

We first define the problem domain to be the union of all the sets of panels that comprise the discretized surfaces. We then place a bounding cube around the domain and recursively decompose the cube using octrees. Given a cube s at level i, the nearest neighbors N_s are those cubes at level i that share a face, edge, or vertex with s. The interaction list for s is denoted as I_s and defined to be the set of cubes at level i that are not nearest neighbors to s and not descended from any cube in an interaction list of an ancestor of s [43]. Fig. 4 illustrates the exclusion process for a two-dimensional (2-D) domain. At every level, each panel is assigned to the cube that contains its centroid. Where ambiguity will not result, s denotes either the cube itself or the set of panels assigned to it. This assignment rule ensures that each panel–panel interaction is treated exactly once.

The coarsest decomposition is termed level 0 and has 4^3 cubes; coarser decompositions have null interaction lists. We continue decomposing the domain until we reach a level l at which no cube is assigned more than $n_{p,\max}$ destinations. At each level i, every cube s has a set of interacting cubes I_s that are well separated from s with respect to the current cube size. Note that the definition of an interaction list is symmetric: $d \in I_s \rightarrow s \in I_d$.

C. Sampling Dominant Sources and Responses

One can compute the potential response φ_{I_s} in I_s due to a source q_s in s by the dense MV product

$$\varphi_{I_s} = G_{I_s,s}q_s \qquad G_{I_s,s} \in \Re^{n_{I_s}\times n_s}. \tag{17}$$

However, the separation between s and I_s motivates the approximation

$$\begin{aligned} G_{I_s,s} &\approx U_{I_s}V_{s,\mathrm{src}}^{\mathrm{T}} \\ U_{I_s} &\in \Re^{n_{I_s}\times k} \\ V_{s,\mathrm{src}}^{\mathrm{T}} &\in \Re^{k\times n_s} \\ k &\ll n_{I_s} \end{aligned} \tag{18}$$

where $V_{s,\mathrm{src}}$ has orthogonal columns [30]. The matrix $V_{s,\mathrm{src}}$ is small and represents the k source distributions in s that produce dominant effects in I_s. It is a reduced-row basis for $G_{I_s,s}$. The projection of q_s onto $V_{s,\mathrm{src}}$ loosely parallels the FMM's calculation of multipoles from sources, in the sense that both the multipole expansion and the product $V_{s,\mathrm{src}}^{\mathrm{T}}q_s$ capture the important pieces of q_s when calculating far-field interactions. We call $V_{s,\mathrm{src}}$ the source compression matrix.

A similar low-rank approximation can be made to find the response in a cube d given a source distribution in I_d

$$\begin{aligned} \varphi_d &= G_{d,I_d}q_{I_d} \\ &\approx U_{d,\mathrm{dest}}V_{I_d}^{\mathrm{T}}q_{I_d} \\ U_{d,\mathrm{dest}} &\in \Re^{n_d\times k} \\ V_{I_d}^{\mathrm{T}} &\in \Re^{k\times n_{I_d}} \\ k &\ll n_{I_d}. \end{aligned} \tag{19}$$

Here, $U_{d,\mathrm{dest}}$ is small and represents the k dominant potential responses in d, the destination cube, due to source distributions in I_d. We call $U_{d,\mathrm{dest}}$ the destination compression matrix; $U_{d,\mathrm{dest}}$ is a reduced-column basis for G_{d,I_d}.

Since it is impractical to compute $G_{I_s,s}$ and G_{s,I_s} for each cube s, we use a sampling procedure inspired by the Kapur and Long hierarchical SVD method [30]. Figs. 5 and 6 illustrate the process of finding a reduced-row basis $V_{s,\mathrm{src}}$. To determine the row basis, we begin by selecting one destination per interacting cube, computing the corresponding rows of $G_{I_s,s}$, and performing rank-revealing QR factorization with reorthogonalization

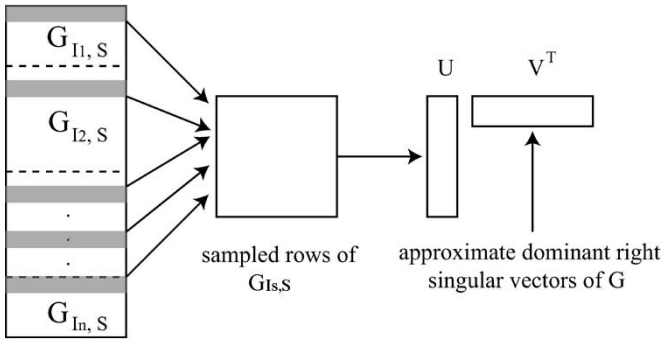


Fig. 5. Computing dominant-row basis for $G_{I_s,s}$ using sampling.

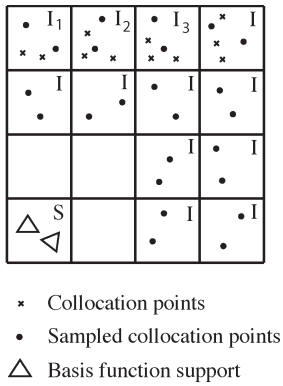


Fig. 6. Sampling a small set of long-range interactions.

on the transpose of the submatrix. If the submatrix rank is less than half the number of sampled destinations, the QR-determined row basis is considered to be adequate. Otherwise, an additional destination is sampled for each interacting cube; the extra destination is chosen to be well separated from the originally chosen destination. The transpose of the new submatrix is factorized and again required to have a rank less than half the total number of samples. The process of resampling is continued until the required rank threshold is met.

To compute the reduced-column basis $U_{d,\mathrm{dest}}$ for the matrix G_{d,I_d}, we select a set of well-separated panels in I_d, compute the corresponding columns of G_{d,I_d}, and QR-factorize the submatrix.

D. Computing Long-Range Interactions

Consider two well-separated cubes s and d. Because the cubes are well separated, we could find a low-rank approximation to $G_{d,s}$ by truncating its SVD

$$\varphi_d = G_{d,s}q_s \tag{20}$$

$$= U_{d,s}\Sigma_{d,s}V_{d,s}^{\mathrm{T}}q_s \tag{21}$$

$$\approx \hat{U}_{d,s}\hat{\Sigma}_{d,s}\hat{V}_{d,s}^{\mathrm{T}}q_s \tag{22}$$

where the hat denotes trunctation to k columns, $k < n_s$. Since the source compression matrix $V_{s,\mathrm{src}}$ finds an approximation to the dominant row space of $G_{I_s,s}$, we expect that it also approximates the dominant row space of $G_{d,s}$, which is a submatrix of $G_{I_s,s}$. Similarly, we expect that $U_{d,\mathrm{dest}}$ approximates the dominant column space of $G_{d,s}$. A small matrix $K_{d,s}$ maps source distributions in the reduced basis $V_{s,\mathrm{src}}$ to responses in the reduced basis $U_{d,\mathrm{dest}}$

Fig. 7. Schematic of the FFTSVD method for computing long-range interactions.

$$\varphi_d \approx U_{d,\mathrm{dest}}K_{d,s}V_{s,\mathrm{src}}^{\mathrm{T}}q_s \tag{23}$$

and it is easy to see that

$$K_{d,s} = U_{d,\mathrm{dest}}^{\mathrm{T}}G_{d,s}V_{s,\mathrm{src}}. \tag{24}$$

Note that $K_{d,s}$ is not diagonal because $U_{d,\mathrm{dest}}$ and $V_{s,\mathrm{src}}$ only approximate the singular vectors of $G_{d,s}$. If $V_{s,\mathrm{src}} \in \Re^{n_s \times k_s}$ and $U_{d,\mathrm{dest}} \in \Re^{n_d \times k_d}$, then $K_{d,s} \in \Re^{k_d \times k_s}$.

The action of the K matrices can be computed in a number of different ways: they can be computed explicitly, via multipoles, or via an FFT. Explicit storage is memory intensive, and multipole representations are Green's-function dependent. We have therefore chosen to implement the memory-efficient Green's-function-independent FFT translation method presented by Ying *et al.* [29].

E. Diagonalizing Long-Range Interactions With the FFT

Our method projects sources to a grid, uses an FFT convolution to accomplish translation between source and destination, and interpolates results back from the grid. Fig. 7 illustrates the approach. We introduce two matrices: $P_{g,j}$ projects sources in cube j to the cube grid, and $I_{j,g}$ interpolates from the grid in cube j to the evaluation points in j. We use an equivalent density scheme similar to those used by Phillips and White [23] and Biros *et al.* [28] to determine the projection and interpolation matrices.

1) Projection-Matrix Calculation: Given a cube s and the basis function weights q_s for panels in s, we wish to find a set of grid charges $q_{g,s}$ that reproduce the potential field far from s. We accomplish this by defining a sphere Γ bounding s and picking a set of quadrature points [44] on the sphere. Denoting quadrature point i on Γ by $r_{\Gamma,i}$, the mapping between q_s and the responses at the quadrature points can be written as $G_{\Gamma,s}$, where

$$G_{\Gamma,s,ij} = \int_{\text{panel j}} G(r_{\Gamma,i}; r')dr'. \tag{25}$$

The mapping between grid charges and responses at the quadrature points can be written as

$$G_{\Gamma,g,ij} = G(r_{\Gamma,i}; r_{g,j}) \tag{26}$$

where $r_{g,j}$ is the position of the jth grid point. If more quadrature points than grid points are used for the matching, solving a

least squares problem gives the desired projection $P_{g,s}$

$$P_{g,s} = G_{\Gamma,g}^{-1} G_{\Gamma,s}. \tag{27}$$

In practice, one uses the singular-value decomposition to solve for $P_{g,s}$.

2) Interpolation Matrix Calculation: Given grid potentials q_d in a cube d, we find the potentials φ_d at the panel centroids in d by interpolation. For problems in which centroid collocation is used to generate a linear system of equations, the interpolation matrix is calculated as

$$I_{d,g} = \left(G_{\Gamma,g}^{-1} G_{\Gamma,d} \right)^{\mathrm{T}} \tag{28}$$

where $G_{\Gamma,d}$ denotes the Green's-function matrix from the quadrature points on Γ to the panel centroids in d. If Galerkin methods are used rather than centroid collocation, the interpolation matrix is the transpose of the projection matrix.

3) Diagonal Translation: Once the grid charges in s are known, a spatial convolution with the Green's function produces the potentials at the grid points in the destination cube d. This spatial convolution is diagonalized by the Fourier transform; we write the transform matrix as $\mathcal{F}$, its inverse by $\mathcal{F}^{-1}$, and the transform of the Green's-function matrix by $\tilde{G}_{d,s}$. After calculating the grid potentials in d, interpolation produces the potentials at the desired evaluation points. The matrix $G_{d,s}$ is therefore written as

$$G_{d,s} = I_{d,g} \mathcal{F}^{-1} \tilde{G}_{d,s} \mathcal{F} P_{g,s}. \tag{29}$$

The products $I_{d,g}\mathcal{F}^{-1}$ and $\mathcal{F}P_{g,s}$ could be stored, but in our experience, this precomputation only marginally improves the MV product time while increasing memory use since $\mathcal{F}$ and $\mathcal{F}^{-1}$ are padded and complex.

In addition to diagonalizing the translation operation between cubes, the FFT significantly decreases memory requirements. Using explicit K matrices requires storing a small dense matrix for each pair of cubes; using FFT translation eliminates the expensive per-pair matrix cost. Instead, each cube has its own P_g and I_g matrices, which are used for all long-range interactions. In addition, because the Green's function is translationally invariant, we only need to store a small number of $\tilde{G}$ matrices for each octree level; each one represents a particular relative translation between source and destination cubes. Because these matrices are diagonal, storage requirements are minimal.

Since translation is the dominant cost in the FFTSVD MV product, efficient implementation of the translation procedure is essential to maximizing performance. The translation operation is simply an elementwise multiplication of two complex vectors, therefore, for g_{p} grid points per cube side, each translation vector is $(2g_{\mathrm{p}}-1)^2[(2g_{\mathrm{p}}-1)/2+1]$ complex numbers long when using the FFTW library [45]. This number takes into account padding and symmetry. For example, with $g_{\mathrm{p}} = 3$, 75 complex numbers are required, resulting in 250 individual multiplies during the translation operation. This number has been reduced by taking advantage of vectorization. Many modern CPUs include instructions that can assist in multiplying complex numbers within a register, effectively halving the number of required multiplies. For comparison, standard FMM translations require more multiplications since they are not diagonal, and cannot be vectorized as easily since they involve MV products. In addition, we have yet to exploit additional ways to accelerate the FFTSVD translation operation. These include using symmetries between related translation vectors $(\tilde{G})$, such as those that translate in opposite directions, and exploiting the fact that for axial translations, many $\tilde{G}$ elements are purely real.

F. Local Interactions

At the finest level of the decomposition, interactions between nearest neighbor cubes are computed directly by calculating the corresponding dense submatrices of G. These submatrices are denoted by $D_{i,j}$, where j is the source cube and i the destination. We bound the complexity of the local-interaction computation by continuing the octree decomposition until each cube has fewer than $n_{p,\max}$ panels.

G. Algorithm Detail

The mapping from source cube s to destination cube d can thus be written as

$$\varphi_d = U_d \left(U_d^{\mathrm{T}} I_{d,g} \right) \mathcal{F}^{-1} \tilde{G} \mathcal{F} (P_{g,s} V_s) V_s^{\mathrm{T}} q_s. \tag{30}$$

The computations are grouped to eliminate redundant multiplications; the matrix products $U_d^{\mathrm{T}} I_{d,g}$ and $P_{g,s}V_s$ are stored for each cube, rather than recomputed at every iteration. Below, we introduce the restriction operator $M_j^{(i)}$ that restricts a global vector to a local vector associated with cube j at level i; let the inverse operator map a local vector to the global by inserting appropriate zeros. Let L^i denote the set of cubes at level i. Given a charge vector q, the MV product is computed by the following procedure.

1) Downward Pass For Long-Range Interactions: For levels $i = 0, 1, \ldots, l$:
   a) Project Into Dominant Source Space: For each cube $j \in L^i$, compute

$$\zeta_j = \mathcal{F}(P_{g,j} V_{j,\mathrm{src}}) V_{j,\mathrm{src}}^{\mathrm{T}} M_j^{(i)} q. \tag{31}$$

   b) Compute Long-Range Interactions: For each cube $j \in L^i$, compute

$$\nu_j = \sum_{s \in I_j} \tilde{G} \zeta_s. \tag{32}$$

   c) Determine Total Dominant Response: For each cube $j \in L^i$, compute

$$\varphi = \varphi + M_j^{(i),-1} U_{j,\mathrm{dest}} (U_{j,\mathrm{dest}}^{\mathrm{T}} I_{j,g}) \mathcal{F}^{-1} \nu_j. \tag{33}$$

2) Sum Direct Interactions: For each cube d at level l, add the contributions from neighboring cubes N_d

$$\varphi = \varphi + M_d^{(l),-1} \sum_{s \in N_d} D_{d,s} M_s^{(l)} q. \tag{34}$$

IV. Computational Results

To demonstrate the accuracy, speed, and memory efficiency of the FFTSVD algorithm, we have used FFTSVD to solve for self and mutual capacitances in various geometries. A MEMS comb drive example [39] illustrates electrostatic-force calculation using FFTSVD. In addition, to show Green's-function independence and use of double-layer kernels, we have used FFTSVD to solve for the electrostatics of solvation for the highly charged dye-molecule fluorescein. Fluorescein is often used as a fluorescent label in Bio-MEMS applications [46], [47], and its electrostatic properties in aqueous solution modulate its interaction with other molecules and surfaces.

The FFTSVD algorithm has several adjustable parameters: $\varepsilon_{\mathrm{QR}}$ is the reduced-basis tolerance; g_{p} is the number of FFT grid points on each side of a finest level cube; $n_{p,\max}$ is the maximum number of panels in a finest level cube; n_{quad} is the number of quadrature points used on the equivalent density sphere, $\mathrm{tol}_{\mathrm{GMRES}}$ is the tolerance on the relative residual that the resulting linear equations are solved to. At the two finest levels, g_{p} FFT grid points per cube edge are used, and the number of grid points per edge increases by one for each successively coarser level; experience has shown that using different numbers of grid points per edge provides significant accuracy improvements for marginal memory and time costs. The parameters used for the following results are 10^{-4} for $\varepsilon_{\mathrm{QR}}$, 3 for g_{p}, 32 for $n_{p,\max}$, 25 for n_{quad}, and 10^{-4} for $\mathrm{tol}_{\mathrm{GMRES}}$, unless otherwise specified.

For capacitance calculations, we compare the performance to FastCap, based on the FMM [34], and fftcap++, based on the pFFT++ implementation of the pFFT method [48]. All programs were compiled with full optimizations using the Intel C++ compiler version 8.1 and benchmarked on an Intel Pentium 4 3.0-GHz desktop computer with 2 GB of RAM. All parameter settings in FastCap and fftcap++ were left at their defaults, except for the tolerance on solving the resulting linear equations, which was set to 10^{-4}, unless otherwise specified.

A. Self-Capacitance of a Sphere

In order to test the accuracy of the FFTSVD method, we have applied it to solving for the self-capacitance of a unit 1-meter (m)-radius sphere, a quantity known analytically. Fig. 8 shows the improvement in accuracy with increasing sphere discretization for FFTSVD with values of 3 and 5 for g_{p}, second- and fourth-order multipoles in FastCap, and default settings for fftcap++. A tolerance of 10^{-6} for the relative residual when solving the BEM equations was used in all programs. The analytical value for the self-capacitance of a 1-m-radius sphere is 0.111265 nF as computed by Gauss' law. The results show that FFTSVD with a value of 3 for g_{p} tends to be more accurate than second-order multipoles in FastCap. In addition, FFTSVD with low values of g_{p} tends to overshoot the analytical solution while FastCap tends to undershoot with a truncation of multipole order. These findings are consistent across many geometries when examining convergence behavior.

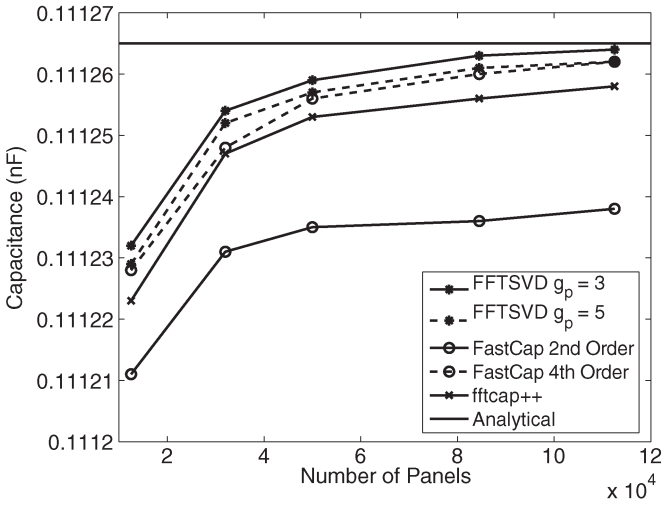


Fig. 8. Accuracy versus number of panels for FFTSVD, FastCap, and fftcap++ solving the unit-sphere self-capacitance problem.

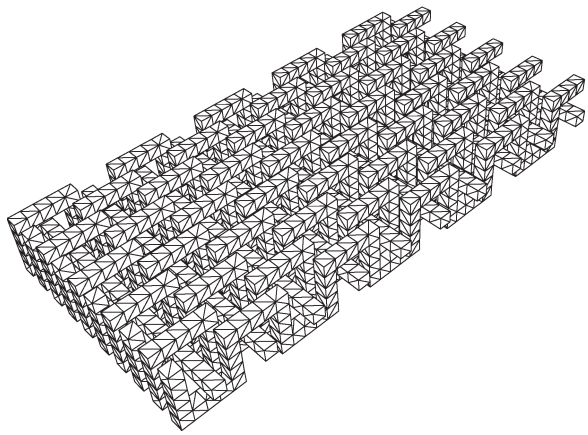


Fig. 9. Homogeneous woven-bus capacitance problem (woven10n01).

B. Woven-Bus Example (Homogeneous Problem)

As stated previously, one of the advantages of the FFTSVD method is its use of diagonal translation operators. This advantage becomes apparent in cases of homogeneous geometry, since a large number of translation operations are required. To examine performance in a problem with homogeneous geometry, we have applied FFTSVD to solving for the mutual capacitances between woven bus conductors as in Fig. 9. Table I summarizes the results for several woven-bus-capacitance problems. FFTSVD can achieve slightly better speed and memory performance than pFFT, which is expected to excel at problems with uniform distribution, and significantly better performance as compared to FastCap.

C. Inhomogeneous Capacitance Problem

One of the disadvantages of the pFFT method is that it lays down a uniform grid over the entire problem domain, and the simulation time grows roughly in proportion to the number of grid points. For simulations in which most of the domain is empty, therefore, the pFFT algorithm is inefficient. We have demonstrated this inefficiency, and FFTSVD's relative

TABLE I
COMPARISON OF FASTCAP (FC), FFTCAP++ (FFT++), AND FFTSVD (FS) PERFORMANCE IN TERMS OF MV PRODUCT TIME (MV) AND MEMORY USAGE (MEM) ON HOMOGENEOUS WOVEN-BUS-CAPACITANCE PROBLEMS WITH TWO, FIVE, AND TEN CROSSINGS (WOVEN02N03, WOVEN05N03, WOVEN10N03) AND TEN CROSSINGS WITH LOWER DISCRETIZATION (WOVEN10N01)

Problem	Panels	FC MV (s)	FC MEM (MB)	FFT++ MV (s)	FFT++ MEM (MB)	FS MV (s)	FS MEM (MB)
woven02n03	3168	0.03	30	0.02	23	0.01	11
woven05n03	18720	0.17	205	0.22	411	0.09	110
woven10n01	8160	0.08	89	0.04	69	0.04	41
woven10n03	73440	0.73	901	0.51	818	0.41	466

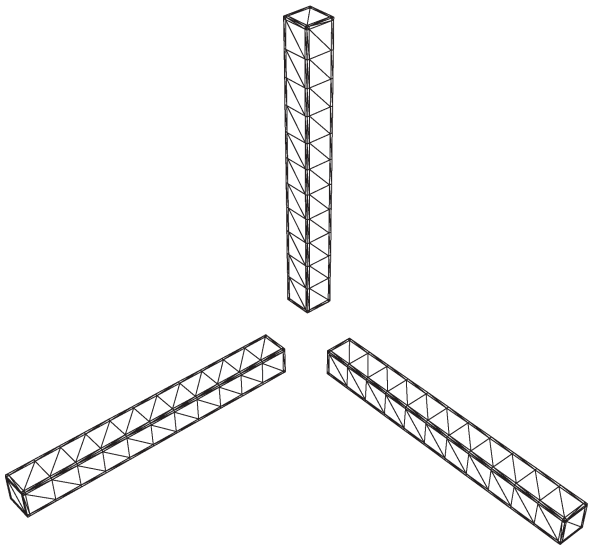


Fig. 10. Inhomogeneous capacitance problem.

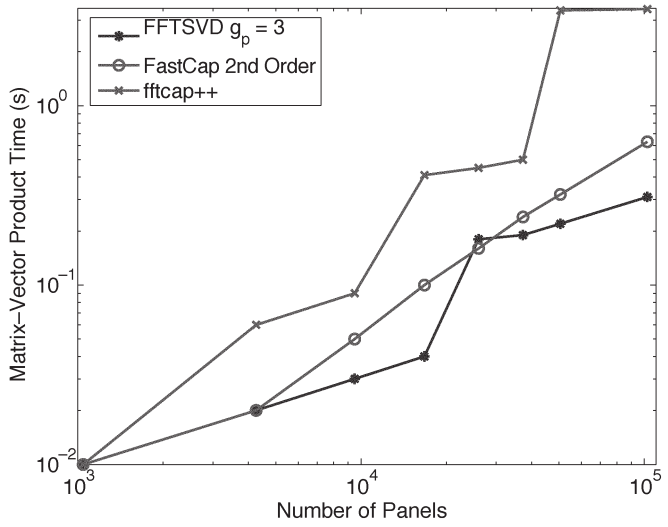


Fig. 11. MV product times for FFTSVD, FastCap, and fftcap++ codes solving the inhomogeneous capacitance problem.

advantage, by configuring a set of conductors as shown in Fig. 10. Almost all of the panels in this system are at the edges of a cube bounding the domain. Fig. 11 plots the MV product times for the FFTSVD, FastCap, and fftcap++ codes, and Fig. 12 plots the memory requirements. As expected, the pFFT-based fftcap++ code has poor performance, especially for fine discretizations of the inhomogeneous problem. FFTSVD performs consistently better than fftcap++ and generally better than FastCap. The sharp jumps in FFTSVD and fftcap++ MV product time with increasing panel count are due to a change in selection of the optimal octree decomposition depth or FFT grid size, respectively.

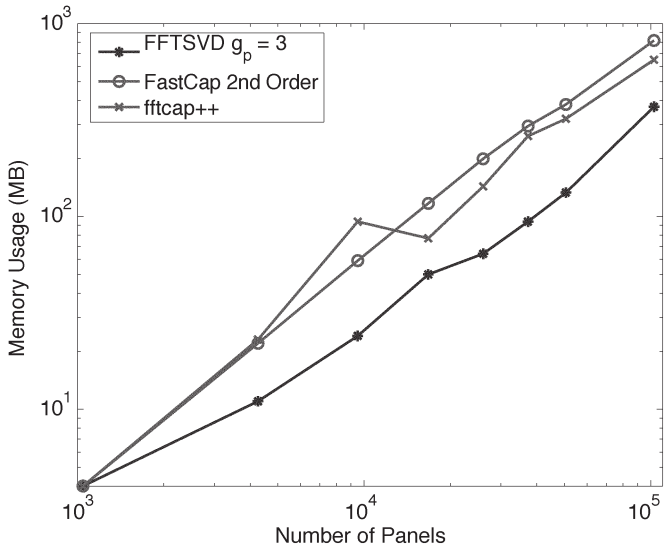


Fig. 12. Memory requirements for FFTSVD, FastCap, and fftcap++ codes solving the inhomogeneous capacitance problem.

D. MEMS Comb Drive

We have simulated the MEMS comb drive illustrated in Fig. 1 [39]. We applied a voltage difference of 1 V to the two structures and used a fourth-order finite-difference scheme to approximate the derivative in (2). Because the finite-difference scheme for force calculation requires high accuracy in the capacitance calculations, more stringent parameters are required for these simulations. We have used $\mathrm{tol}_{\mathrm{GMRES}} = 10^{-6}$, $\varepsilon_{\mathrm{QR}} = 10^{-6}$, $g_{\mathrm{p}} = 5$, $n_{\mathrm{QUAD}} = 64$, and for each discretization, we have fixed $n_{p,\max}$ such that the octree decomposition depth is equal for each of the four geometries.

The contribution of each panel to the axial force is plotted in Fig. 13 and the total axial electrostatic force is plotted in Fig. 14 as a function of the number of panels used to discretize the comb drive. We have used general triangles and note that the discretization scheme is poorly tuned for the calculation of electrostatic forces; nonuniform meshes achieve superior accuracy at reduced panel counts [49]. The force can also be calculated by integrating the squared charge density over the conductor surface, but this approach requires specialized treatment because the charge density becomes infinite at the edges and corners of the conductors [50], [51].

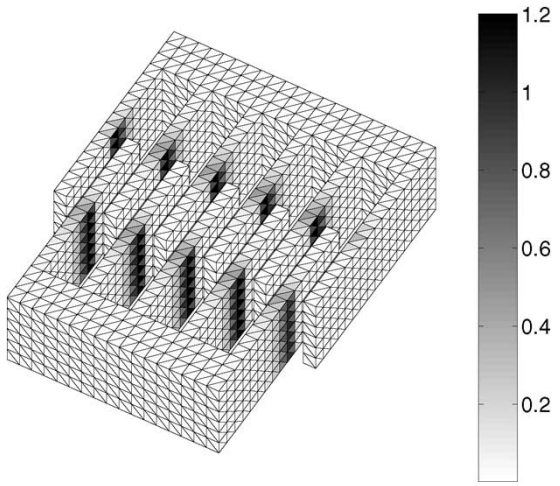


Fig. 13. Magnitudes of panel contributions to the axial electrostatic force. Units are in piconewtons.

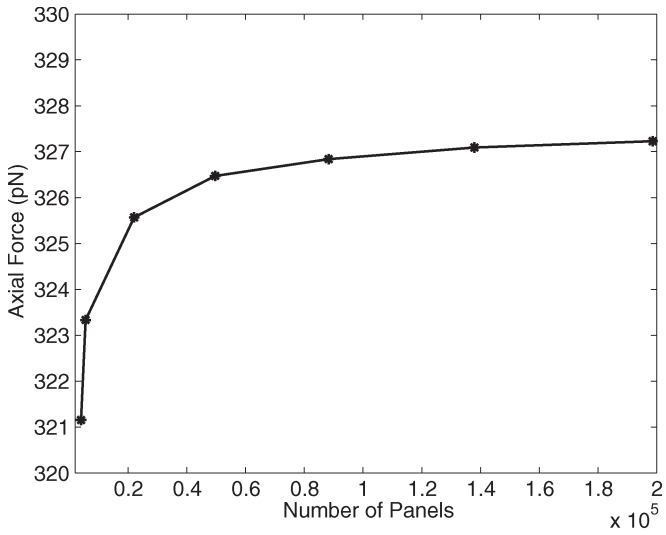


Fig. 14. Calculated total axial electrostatic force on one comb.

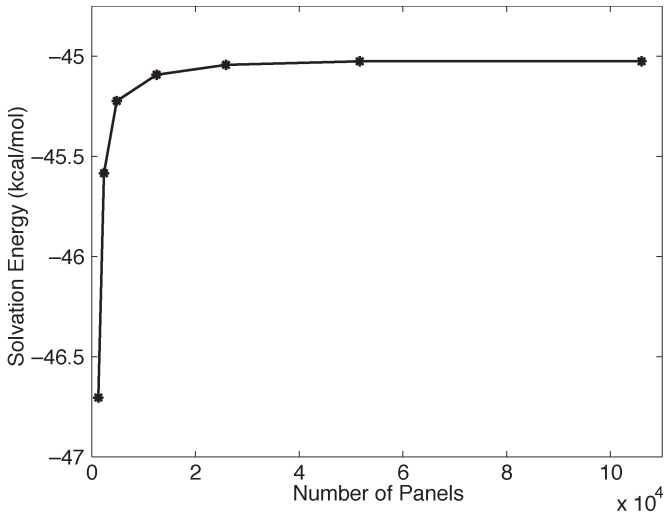


Fig. 15. Computed electrostatic solvation energy of fluorescein with increasing problem discretization.

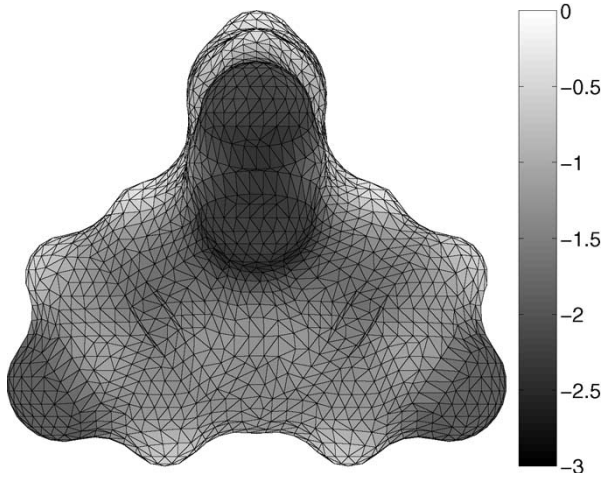


Fig. 16. Electrostatic solvation potentials on the molecular surface of fluorescein. Units are in kilocalories per mole per electron charge.

E. Solvation of Fluorescein

We have used the integral formulation in (10) and (11) to calculate the solvation energy of fluorescein. To prepare a model for solvation calculations, its structure and partial atomic charges were determined from quantum mechanical calculations. Radii were assigned to each atom and used to generate a triangulation of the molecular surface. The interior of the fluorescein molecule was assigned a dielectric constant of 4, and the exterior was assigned a dielectric constant of 80 (for water) with an ionic strength of 0.145 molar ($\kappa = 0.124$ Å^{-1}). FFTSVD was used to solve for both the electrostatic solvation energy (Fig. 15), as well as the total electrostatic potential on the surface of the fluorescein molecule (Fig. 16). We note that the long-range single- and double-layer integrals can be computed using only one set of translation operations. Different projection operators are used to find the corresponding grid charges due to monopole and dipole distributions, and the grid charges can then be summed for translation.

V. Discussion

A. Algorithm Variants

For problems with a small number of integral operators, memory constraints may not be a significant consideration. In these cases, the matrices $K_{d,s}$ can be stored explicitly. These $K_{d,s}$ matrices are computed using (24), but instead of computing $G_{d,s}$ explicitly, we project, translate, and interpolate an identity matrix using the methodology outlined in Section III-E. Although setup time and memory use increase when explicit K-matrices are used, the MV product time is significantly reduced. We have also implemented a parameter that allows a tradeoff between speed and memory use through K-matrices. Pairs of interacting octree cubes that contain fewer panels than the parameter are handled with explicit K-matrices, while all other cubes use the FFT-based translation. In this manner, the balance between speed and memory can be fine-tuned for the given application.

It is also straightforward to create an FFTSVD variant that runs in linear time; the same method used to generate the projection and interpolation matrices can be used to create "upward pass" and "downward pass" operators such as those found in multipole algorithms. This variant algorithm is essentially equivalent to the kernel-independent method by Ying *et al.* [29], except that we allow all the grid charges to be nonzero. The Ying method, in contrast, uses only grid charges on the surface of the cube.

The linear-time FFTSVD method requires a greater number of grid points per cube, due to the loss of degrees of freedom during each upward pass from child to parent cube. In addition, the SVD-based compression of dominant sources and responses is no longer computed, since these bases are now taken directly from child cubes. This method is extremely memory efficient since dominant source and response bases are no longer stored, but it trades off performance to achieve it due to the larger required grid sizes.

Finally, the multilevel structure of FFTSVD allows easy parallelization. Each processor can be assigned responsibility for a set of cubes on coarse levels, and the computation can proceed independently until the final potential responses are summed. We have implemented parallel FFTSVD using both OpenMP (Open specifications for MultiProcessing) and MPI (Message Passing Interface) libraries with good results.

B. Summary

We have developed a fast algorithm for computing the dense MV products required to solve boundary-element problems using Krylov subspace iterative methods. The FFTSVD method is a multiscale algorithm; an octree decomposes the matrix action into different length scales. For each length scale, we use sampling to calculate reduced bases for the interactions between well-separated groups of panels. The FFT is used to diagonalize the translation operation that computes the long-range interactions. The method described here relies on both kernel decay and translation invariance.

Numerical results illustrate that FFTSVD is much more memory efficient than FastCap or pFFT, and that it is generally faster than either technique on a variety of problems. In addition, FFTSVD is Green's-function independent, unlike FastCap, and the method performs well even when the problem domain is sparsely populated, unlike pFFT. Our implementation is well suited to solve problems with multiple dielectric regions. Finally, we note that the structure of the algorithm permits the treatment of kernels that are not translation invariant; for such problems, the K-matrix algorithm variant should be used rather than the FFT. Together, the algorithm's performance and flexibility make FFTSVD an excellent candidate for fast BEM solvers for microfluidic and microelectromechanical problems that appear in Bio-MEMS design.

ACKNOWLEDGMENT

The authors are grateful to Z. Zhu and D. Willis for numerous helpful discussions, and to S. Kuo for discussions about integral-equation formulations for biomolecule electrostatics.

REFERENCES

[1] J. Voldman, M. L. Gray, and M. A. Schmidt, "Microfabrication in biology and medicine," *Annu. Rev. Biomed. Eng.*, vol. 1, no. 1, pp. 401–425, Jan. 1999.

[2] D. S. Gray, T. L. Tan, J. Voldman, and C. S. Chen, "Dielectrophoretic registration of living cells to a microelectrode array," *Biosens. Bioelectron.*, vol. 19, no. 7, pp. 771–780, Feb. 2004.

[3] G.-B. Lee, S.-H. Chen, G.-R. Huang, W.-C. Sung, and Y.-H. Lin, "Microfabricated plastic chips by hot embossing methods and their applications for DNA separation and detection," *Sens. Actuators B, Chem.*, vol. 75, no. 1–2, pp. 142–148, Apr. 2001.

[4] T. P. Burg and S. R. Manalis, "Suspended microchannel resonators for biomolecular detection," *Appl. Phys. Lett.*, vol. 83, no. 13, pp. 2698–2700, Sep. 2003.

[5] T. Korsmeyer, "Design tools for bio MEMS," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 622–627.

[6] J. White, "CAD challenges in BioMEMS design," in *Proc. IEEE Design Automation Conf. (DAC)*, San Diego, CA, 2004, pp. 629–632.

[7] S. D. Senturia, R. M. Harris, B. P. Johnson, S. Kim, K. Nabors, M. A. Shulman, and J. K. White, "A computer-aided design system for microelectromechanical systems (MEMCAD)," *J. Microelectromech. Syst.*, vol. 1, no. 1, pp. 3–13, Mar. 1992.

[8] C. A. Savran, S. M. Knudsen, A. D. Ellington, and S. R. Manalis, "Micromechanical detection of proteins using aptamer-based receptor molecules," *Anal. Chem.*, vol. 76, no. 11, pp. 3194–3198, Jun. 2004.

[9] R. A. Potyrailo, R. C. Conrad, A. D. Ellington, and G. M. Hieftje, "Adapting selected nucleic acid ligands (aptamers) to biosensors," *Anal. Chem.*, vol. 70, no. 16, pp. 3419–3425, Aug. 1998.

[10] B. Honig and A. Nicholls, "Classical electrostatics in biology and chemistry," *Science*, vol. 268, no. 5214, pp. 1144–1149, May 1995.

[11] I. Klapper, R. Hagstrom, R. Fine, K. Sharp, and B. Honig, "Focusing of electric fields in the active site of Cu–Zn superoxide dismutase: Effects of ionic strength and amino-acid modification," *Proteins: Structure, Function, Genetics*, vol. 1, no. 1, pp. 47–59, Sep. 1986.

[12] M. K. Gilson and B. Honig, "Calculation of the total electrostatic energy of a macromolecular system: Solvation energies, binding energies, and conformational analysis," *Proteins: Structure, Function, Genetics*, vol. 4, no. 1, pp. 7–18, 1988.

[13] A. Jean-Charles, A. Nicholls, K. Sharp, B. Honig, A. Tempczyk, T. F. Hendrickson, and W.C. Still, "Electrostatic contributions to solvation energies: Comparison of free energy perturbation and continuum calculations," *J. Amer. Chem. Soc.*, vol. 113, no. 4, pp. 1454–1455, 1991.

[14] M. K. Gilson, K. A. Sharp, and B. H. Honig, "Calculating the electrostatic potential of molecules in solution: Method and error assessment," *J. Comput. Chem.*, vol. 9, no. 4, pp. 327–335, 1987.

[15] M. Holst, N. Baker, and F. Wang, "Adaptive multilevel finite element solution of the Poisson–Boltzmann equation. I. Algorithms and examples," *J. Comput. Chem.*, vol. 21, no. 15, pp. 1319–1342, 2000.

[16] B. J. Yoon and A. M. Lenhoff, "A boundary element method for molecular electrostatics with electrolyte effects," *J. Comput. Chem.*, vol. 11, no. 9, pp. 1080–1086, 1990.

[17] K. Nabors, F. T. Korsmeyer, F. T. Leighton, and J. K. White, "Preconditioned, adaptive, multipole-accelerated iterative methods for three-dimensional first-kind integral equations of potential theory," *SIAM J. Sci. Comput.*, vol. 15, no. 3, pp. 713–735, May 1994.

[18] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *J. Chem. Phys.*, vol. 73, no. 2, pp. 325–348, Dec. 1987.

[19] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. Cambridge, MA: MIT Press, 1988.

[20] W. Hackbusch, "A sparse matrix arithmetic based on H-matrices. I. Introduction to H-matrices," *Computing*, vol. 62, no. 2, pp. 89–108, 1999.

[21] W. Hackbusch and B.N. Khoromskij, "A sparse H-matrix arithmetic. II. Application to multi-dimensional problems," *Computing*, vol. 64, no. 1, pp. 21–47, 2000.

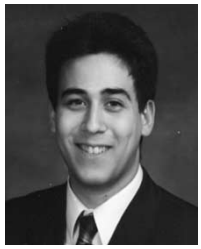
[22] S. Borm, L. Grasedyck, and W. Hackbusch, "Introduction to hierarchical matrices with applications," *Eng. Anal. Bound. Elem.*, vol. 27, no. 5, pp. 405–422, Jan. 2003.

[23] J. R. Phillips and J. K. White, "A precorrected-FFT method for electrostatic analysis of complicated 3-D structures," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 16, no. 10, pp. 1059–1072, Oct. 1997.

[24] W. Shi, J. Liu, N. Kakani, and T. Yu, "A fast hierarchical algorithm for 3-D capacitance extraction," in *Proc. Design Automation Conf.*, San Francisco, CA, 1998, pp. 212–217.

[25] J. Tausch and J. K. White, "A multiscale method for fast capacitance extraction," in *Proc. Design Automation Conf.*, New Orleans, LA, 1999, pp. 537–542.

[26] E. T. Ong, K. M. Lim, K. H. Lee, and H. P. Lee, "A fast algorithm for three-dimensional potential fields calculation: Fast Fourier transform on multipoles," *J. Comput. Phys.*, vol. 192, no. 1, pp. 244–261, Nov. 2003.
[27] E. T. Ong, H. P. Lee, and K. M. Lim, "A parallel fast Fourier transform on multipoles (FFTM) algorithm for electrostatics analysis of three-dimensional structures," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 7, pp. 1063–1072, Jul. 2004.
[28] G. Biros, L. Ying, and D. Zorin, "A fast solver for the Stokes equations with distributed forces in complex geometries," *J. Comput. Phys.*, vol. 193, no. 1, pp. 317–348, Jan. 2004.
[29] L. Ying, G. Biros, and D. Zorin, "A kernel-independent adaptive fast multipole algorithm in two and three dimensions," *J. Comput. Phys.*, vol. 196, no. 2, pp. 591–626, May 2004.
[30] S. Kapur and D. E. Long, "IES^3: A fast integral equation solver for efficient 3-dimensional extraction," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, San Jose, CA, 1997, pp. 448–455.
[31] ——, "IES^3: Efficient electrostatic and electromagnetic simulation," *IEEE Comput. Sci. Eng. Mag.*, vol. 5, no. 4, pp. 60–67, Oct.–Dec. 1998.
[32] L. Greengard, J. Huang, V. Rokhlin, and S. Wandzura, "Accelerating fast multipole methods for the Helmholtz equation at low frequencies," *IEEE Comput. Sci. Eng. Mag.*, vol. 5, no. 3, pp. 32–38, Jul.–Sep. 1998.
[33] D. Gope and V. Jandhyala, "PILOT: A fast algorithm for enhanced 3D parasitic extraction efficiency," in *Proc. IEEE Elec. Perform. Electron. Packag.*, 2003, pp. 337–340.
[34] K. Nabors and J. White, "FASTCAP: A multipole accelerated 3-D capacitance extraction program," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 10, no. 11, pp. 1447–1459, Nov. 1991.
[35] L. Greengard, J. F. Huang, V. Rokhlin, and S. Wandzura, "Accelerating fast multipole method for the Helmholtz equation at low frequencies," *IEEE Comput. Sci. Eng.*, vol. 5, no. 3, pp. 32–38, Jul.–Sep. 1998.
[36] Z. Zhu, B. Song, and J. White, "Algorithms in FastImp : A fast and wideband impedance extraction program for complicated 3D geometries," in *Proc. IEEE/ACM Design Automation Conf.*, Anaheim, CA, 2003, pp. 712–717.
[37] N. R. Aluru and J. White, "A fast integral equation technique for analysis of microflow sensors based on drag force calculations," *Modeling and Simulation of Microsystems*, pp. 283–286, 1998.
[38] W. Ye, X. Wang, and J. White, "A fast Stokes solver for generalized flow problems," *Modeling and Simulation of Microsystems*, pp. 524–527, 2000.
[39] X. Wang, "FastStokes : A fast 3-D fluid simulation program for micro-electro-mechanical systems," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Tech., Cambridge, 2002.
[40] S. S. Kuo, M. D. Altman, J. P. Bardhan, B. Tidor, and J. K. White, "Fast methods for simulation of biomolecule electrostatics," in *Proc. Int. Conf. Computer Aided Design (ICCAD)*, San Jose, CA, 2002, pp. 466–473.
[41] Y. Saad and M. Schultz, "GMRES : A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 3, pp. 856–869, Jul. 1986.
[42] F. M. Richards, "Areas, volumes, packing, and protein structure," *Annu. Rev. Biophys. Bioeng.*, vol. 6, pp. 151–176, 1977.
[43] J. Kanapka, J. Phillips, and J. White, "Fast methods for extraction and sparsification of substrate coupling," in *Proc. Design Automation Conf.*, Los Angeles, CA, 2000, pp. 738–743.
[44] J. Fliege and U. Maier, "The distribution of points on the sphere and corresponding cubature formulae," *IMA J. Numer. Anal.*, vol. 19, no. 2, pp. 317–334, Apr. 1999.
[45] M. Frigo and S. G. Johnson, "FFTW: An adaptive software architecture for the FFT," in *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, Seattle, WA, 1998, vol. 3, pp. 1381–1384.
[46] B. P. Mosier, J. I. Malho, and J. G. Santiago, "Photobleached-fluorescence imaging of microflows," *Exp. Fluids*, vol. 33, no. 4, pp. 545–554, 2002.
[47] S. I. Cho, S.-H. Lee, D. S. Chung, and Y.-K. Kim, "Bias-free pneumatic sample injection in microchip electrophoresis," *J. Chromatogr. A*, vol. 1063, no. 1–2, pp. 253–256, Jan. 2005.
[48] Z. Zhu, "Efficient techniques for wideband impedance extraction of complex 3-dimensional geometries," M.S. thesis, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, 2002.
[49] A. E. Ruehli and P. A. Brennan, "Efficient capacitance calculations for three-dimensional multiconductor systems," *IEEE Trans. Microw. Theory Tech.*, vol. MTT-21, no. 2, pp. 76–82, Feb. 1973.
[50] E. T. Ong, K. H. Lee, and K. M. Lim "Singular elements for electro-mechanical coupling analysis of micro-devices," *J. Micromech. Microeng.*, vol. 13, no. 3, pp. 482–490, May 2003.
[51] E. T. Ong and K. M. Lim, "Three-dimensional singular boundary elements for corner and edge singularities in potential problems," *Eng. Anal. Bound. Elem.*, vol. 29, no. 2, pp. 175–189, Feb. 2005.



Michael D. Altman received the S.B. degree in biology from the Massachusetts Institute of Technology, Cambridge, in 1999. He is currently pursuing the Ph.D. degree in the biological chemistry program in the Department of Chemistry, at the same university.

His research interests include computational drug design, the role of electrostatics in biomolecular recognition, and the application of numerical methods in biology.

Jaydeep P. Bardhan (S'01) received the S.B. and M.Eng. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, in 2000 and 2001, respectively. He is currently pursuing the Ph.D. degree in electrical engineering at the same university. He is the recipient of a Department of Energy Computational Science Graduate Fellowship. His current research focuses on the development of numerical methods for problems in biomolecule electrostatics and biological signaling networks.

Bruce Tidor received the A.B. degree in chemistry and physics from Harvard College, Cambridge, MA, and then received a Marshall Scholar Award to study at Oxford University's Wolfson College, Oxford, U.K., where he earned the M.Sc. degree in biochemistry. He received the Ph.D. degree in biophysics from Harvard.

He moved to the Whitehead Institute for Biomedical Research in 1990, where he started his independent research as a Whitehead Fellow. In 1994, he was appointed to the faculty at the Massachusetts Institute of Technology, where he is currently Professor of Biological Engineering and Computer Science and Codirector of the Computational and Systems Biology Initiative (CSBi). He has held the Leonard T. Skeggs Chair for Whitehead Fellows, the Paul M. Cook Career Development Chair, and the Cecil H. Green Professorship, and from 1999 to 2001, he was an Alfred P. Sloan Research Fellow. His research focuses on the analysis of complex biological systems at the molecular and cellular level. Using molecular modeling, theory, and computation, his work explores the structure, function, and interactions of proteins and nucleic acids. Using cell-level models, his group explores the relationship between network structure and biological function. He is actively involved in applying knowledge from modeling studies to rational design.

Jacob K. White (S'80–M'83–A'88) received the B.S. degree in electrical engineering and computer science from the Massachusetts Institute of Technology (M.I.T.), Cambridge, and the S.M. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley.

He worked at the IBM T. J. Watson Research Center from 1985 to 1987, and was the Analog Devices Career Development Assistant Professor at the M.I.T. from 1987 to 1989. He is currently at the M.I.T., where he is the C. H. Green Professor in Electrical Engineering and Computer Science and an Associate Director of the Research Laboratory of Electronics. His current research interests are in numerical algorithms for problems in circuits, interconnect, micromachining for biological applications, biomolecule design, and systems biology.

Dr. White was a 1988 Presidential Young Investigator, was an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN from 1992 until 1996, and was Chair of the International Conference on Computer-Aided Design in 1999.