Journal of Computational Physics 228 (2009) 8712–8725

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

The black-box fast multipole method

William Fong [a], Eric Darve [a,b,*]

[a] Institute for Computational and Mathematical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Stanford, CA 94305-4042, USA
[b] Department of Mechanical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Room 209, Stanford, CA 94305-4040, USA

ARTICLE INFO

Article history:
Received 31 March 2009
Received in revised form 17 August 2009
Accepted 26 August 2009
Available online 6 September 2009

Keywords:
Fast multipole method
Interpolation
Chebyshev polynomials
Singular value decomposition

ABSTRACT

A new $O(N)$ fast multipole formulation is proposed for non-oscillatory kernels. This algorithm is applicable to kernels $K(x,y)$ which are only known numerically, that is their numerical value can be obtained for any (x,y). This is quite different from many fast multipole methods which depend on analytical expansions of the far-field behavior of K, for $|x-y|$ large. Other "black-box" or "kernel-independent" fast multipole methods have been devised. Our approach has the advantage of requiring a small pre-computation time even for very large systems, and uses the minimal number of coefficients to represent the far-field, for a given L^2 tolerance error in the approximation. This technique can be very useful for problems where the kernel is known analytically but is quite complicated, or for kernels which are defined purely numerically.

1. Introduction

The fast multipole method (FMM) is a technique to calculate sums of the form

$$f(x_i) = \sum_{j=1}^{N} K(x_i, y_j)\ \sigma_j, \quad i = 1, \ldots, N$$

in $O(N)$ operations with a controllable error ε. Historically, Greengard and Rokhlin [1] first developed a technique for the kernel $K(x,y) = 1/r$ $(r = |x-y|)$ based on Legendre polynomials and spherical harmonics. The technique was later extended to the oscillatory kernel $e^{ikr}/|x-y|$. Both these approaches require approximations of $K(x,y)$ for $|x-y|$ sufficiently large (in a sense which can be made precise) which are typically obtained using known analytical expansions such as:

$$\frac{e^{ikr}}{ikr} = \sum_{m=0}^{\infty} (2m+1) h_m^{(1)}(k|u|) j_m(k|v|) P_m(\cos(\theta))$$

where $r = |x-y|, u - v = x - y$ with $|v| < |u|, h_m^{(1)}$ is a spherical Bessel function of the third kind, j_m is a spherical Bessel function of the first-kind, P_m is the Legendre polynomial of degree m, and θ is the angle between u and v. By truncating the infinite series and using other relations one can derive a fast $O(N)$ or $O(N \ln N)$ method.

Extensions to general kernels are possible as evidenced by kernel-independent FMMs, see for example [2,3]. Fewer methods exist which allow building a fast $O(N)$ method using only numerical values of K, that is without requiring approximations based on analytical expansions. These techniques are often based on wavelet decompositions [4,5], singular value decompositions [6,7], or other schemes [2,8].

* Corresponding author. Address: Institute for Computational and Mathematical Engineering, Stanford University, 496 Lomita Mall, Durand Building, Stanford, CA 94305-4042, USA. Tel.: +1 650 291 4946; fax: +1 650 725 2560.
E-mail address: darve@stanford.edu (E. Darve).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.08.031

In Gimbutas et al. [7], a scheme based on singular value decompositions (SVD) is used. Using the usual tree decomposition of the domain, they denote by Y^b a cluster at level l and by Z^b the union of Y^b and its nearest neighbor clusters at level l. Then X^b is defined as the complement of Z^b. The kernel $K(x,y)$ can then be decomposed using a continuous SVD:

$$K(x,y) \approx \sum_{l=1}^{n} s_l \; u_l(x) \; v_l(y)$$

where $y \in Y^b$ and $x \in X^b$. This low-rank approximation can be extended to produce multipole-to-multipole, local-to-local and multipole-to-local (M2L) operators. The advantage of this technique is that the SVD guarantees an optimal compression. Therefore the number of multipole coefficients that one operates with is minimal for a given approximation error in the L^2 norm. The drawback of this approach is the cost of pre-computing the SVD of K which can be very expensive for large 3-D domains.

Interpolation techniques can be used to construct fast multipole methods. This approach has not attracted a lot of attention but a few papers have used interpolation techniques (e.g. Chebyshev polynomials) in various ways as part of constructing fast methods [9–11]. The reference [12] discusses an idea similar to this paper, with some differences, including the fact that the multipole and local expansions are treated differently, whereas our scheme is more "symmetrical" and treats them in a similar way. In addition, [12] focuses on a 1 dimensional FMM with the kernel $1/x$, which is required by their fast algorithm for interpolation, differentiation and integration.

The basic idea of an interpolation-based FMM is as follows. If we let $w_l(x)$ denote the interpolating functions, then:

$$K(x,y) \approx \sum_{l} \sum_{m} K(x_l, y_m) w_l(x) w_m(y)$$

which is a low-rank approximation. This works for any interpolation scheme. The advantage of this type of approach is that it requires minimal pre-computing. In addition, the only input required is the ability to evaluate K at various points. No kernel-dependent analytical expansion is required. The drawback is that the number of expansion terms (indices l and m in the sum above) is in general sub-optimal for a given tolerance ε.

This paper proposes a new approach which essentially combines these two ideas. A Chebyshev interpolation scheme is used to approximate the far-field behavior of $K(x,y)$, i.e., when $|x-y|$ large. This leads to an efficient low-rank representation for non-oscillatory kernels. The multipole-to-local (M2L) operator then consists in evaluating the field due to particles located at Chebyshev nodes. This operation can be done efficiently using an SVD. This makes the scheme optimal since the M2L step is by far the most expensive. A key point is that the SVD needs to be computed only "locally". More specifically, given a tolerance ε, if the kernel is translation-invariant, i.e., of the form $K(x-y)$, then the cost of pre-computing the SVD is $O(\ln N)$; otherwise the pre-computing cost is $O(N)$. This is a true pre-computation since this calculation is independent of the location of the sources y_j and observation points x_i, and only depends on the desired accuracy.

This article starts by discussing interpolation methods and in particular Chebyshev polynomials, which have several desirable properties. Then we explain how one can construct an $O(N)$ fast method from this interpolation scheme, and how it can be further accelerated using singular value decompositions. Finally some numerical results illustrate the accuracy and efficiency of the method.

2. Using Chebyshev polynomials as an interpolation basis

Consider sums of the form

$$f(x_i) = \sum_{j=1}^{N} K(x_i, y_j)\sigma_j, \quad i = 1, \ldots, N \tag{1}$$

where $x_i \in [-1,1]$ are observation points, σ_j are the sources, $y_j \in [-1,1]$ are the locations of the sources, N is the number of observation points and sources, and the kernel $K(x,y)$ is continuous on $[-1,1] \times [-1,1]$. These sums appear in many applications such as N-body problems and integral equations in electromagnetics and acoustics. A direct calculation of this sum has a $O(N^2)$ complexity resulting from the multiplication of a $N \times N$ matrix $K_{ij} = K(x_i, y_j)$ with the N-vector of sources σ_j. Since the number of observation points and sources is often very large computing the sum directly is intractable. An approach to improve the efficiency of this computation consists in using a low-rank approximation of the kernel:

$$K(x,y) \approx \sum_{l=1}^{n} u_l(x) v_l(y). \tag{2}$$

A fast summation method can be created by substituting the low-rank approximation (2) into the sum (1):

$$f(x_i) \approx \sum_{l=1}^{n} u_l(x_i) \sum_{j=1}^{N} v_l(y_j)\sigma_j. \tag{3}$$

The outline for the method given by Eq. (3) is as follows:

1. First transform the sources using the basis functions v_l:

$$W_l = \sum_{j=1}^{N} v_l(y_j)\sigma_j, \quad l = 1, \ldots, n.$$

2. Then compute $f(x)$ at each observation point x_i using the basis functions u_l:

$$f(x_i) \approx \sum_{l=1}^{n} u_l(x_i)W_l, \quad i = 1, \ldots, N.$$

The computational cost of each step is $O(nN)$ hence the fast summation method proposed above scales as $O(2nN)$. When $n \ll N$ this is a significant reduction from the $O(N^2)$ scaling of the direct calculation.

A low-rank approximation of the kernel $K(x,y)$ can be constructed by introducing an interpolation scheme. To begin consider a function $g(x)$ on the closed interval $[-1,1]$. An n-point interpolant that approximates $g(x)$ can be expressed as

$$p_{n-1}(x) = \sum_{l=1}^{n} g(x_l)w_l(x) \tag{4}$$

where $\{x_l\}$ are the n interpolation nodes and $w_l(x)$ is the interpolating function corresponding to the node x_l. For example if the functions $w_l(x)$ are taken to be the Lagrange polynomials then $p_{n-1}(x)$ is a $(n-1)$-degree polynomial approximation of $g(x)$. Eq. (4) can be used to approximate the kernel $K(x,y)$ by first fixing the variable y and treating $K(x,y)$ as a function of x:

$$K(x,y) \approx \sum_{l=1}^{n} K(x_l,y)w_l(x).$$

Now noting that $K(x_l,y)$ is a function of y the interpolation formula (4) can be applied again to give

$$K(x,y) \approx \sum_{l=1}^{n}\sum_{m=1}^{n} K(x_l,y_m)w_l(x)w_m(y) \tag{5}$$

which is a low-rank representation of the kernel $K(x,y)$ with

$$u_l(x) = w_l(x)$$
$$v_l(y) = \sum_{m=1}^{n} K(x_l,y_m)w_m(y).$$

Although any interpolation scheme can be used to construct a low-rank approximation, the Chebyshev polynomials will serve as the interpolation basis along with their roots as the interpolation nodes. Before justifying this selection we begin by recalling some properties of Chebyshev polynomials.

The first-kind Chebyshev polynomial of degree n, denoted by $T_n(x)$, is defined by the relation

$$T_n(x) = \cos(n\theta), \quad \text{with } x = \cos\theta.$$

The domain of $T_n(x)$ is the closed interval $[-1,1]$. $T_n(x)$ has n roots located at

$$\bar{x}_m = \cos\theta_m = \cos\left(\frac{(2m-1)\pi}{2n}\right), \quad m = 1, \ldots, n$$

and $n+1$ extrema located at

$$\bar{x}'_m = \cos\theta'_m = \cos\left(\frac{m\pi}{n}\right), \quad \text{with } T_n(\bar{x}'_m) = (-1)^m, \quad m = 0, \ldots, n.$$

The set of roots $\{\bar{x}_m\}$ is commonly referred to as the Chebyshev nodes.

One advantage of using Chebyshev nodes is the stability of the interpolation scheme. While a scheme using equally-spaced nodes on the $[-1,1]$ interval to interpolate a function $g(x)$ suffers from Runge's phenomenon and does not converge uniformly as the number of nodes n becomes large, Chebyshev interpolation ensures uniform convergence with minimal restrictions on $g(x)$. Another benefit afforded by interpolating at the Chebyshev nodes is the near-minimax approximation of $g(x)$ which gives a uniform approximation error across the interval $[-1,1]$. This can be contrasted with the error behavior in the regular multipole expansion of the Laplacian kernel $1/r$ using spherical harmonics. Similar to a Taylor series expansion, the regular multipole expansion is very accurate around the center of the interval but suffers from larger errors near the endpoints. Therefore to ensure a specified accuracy across the entire interval a high-order interpolant is needed in order to adequately resolve the endpoints. The uniform error distribution of Chebyshev interpolation allows for the use of fewer interpolation nodes to achieve a given accuracy and is nearly optimal in the minimax sense.

Using the Chebyshev nodes of $T_n(x)$ as the interpolation nodes, the approximating polynomial $p_{n-1}(x)$ to the function $g(x)$ can be expressed as a sum of Chebyshev polynomials

$$p_{n-1}(x) = \sum_{k=0}^{n-1} c_k T_k(x)$$

where

$$c_k = \begin{cases} \frac{2}{n}\sum_{l=1}^{n} g(\bar{x}_l)T_k(\bar{x}_l) & \text{if } k > 0 \\ \frac{1}{n}\sum_{l=1}^{n} g(\bar{x}_l) & \text{if } k = 0 \end{cases}$$

and $\bar{x}_l$ are the roots of $T_n(x)$. By rearranging the terms in the sum, $p_{n-1}(x)$ can be written in the form of (4):

$$p_{n-1}(x) = \sum_{l=1}^{n} g(\bar{x}_l) S_n(\bar{x}_l, x)$$

where

$$S_n(x,y) = \frac{1}{n} + \frac{2}{n}\sum_{k=1}^{n-1} T_k(x)T_k(y).$$

The rate of convergence of $p_n(x)$ to $g(x)$ is given by two results from Chebyshev approximation theory [13]. First if $g(x)$ has $m+1$ continuous derivatives on $[-1,1]$, then the pointwise approximation error for all $x \in [-1,1]$ is

$$|g(x) - p_n(x)| = O(n^{-m}).$$

Second if $g(x)$ can be extended to a function $g(z)$, where z is a complex variable, which is analytic within a simple closed contour C that encloses the point x and all the roots of the Chebyshev polynomial $T_{n+1}(x)$ then the interpolating polynomial $p_n(x)$ can be written as

$$p_n(x) = \frac{1}{2\pi i}\int_C \frac{[T_{n+1}(z) - T_{n+1}(x)]g(z)}{T_{n+1}(z)(z-x)}dz$$

and its error is

$$g(x) - p_n(x) = \frac{1}{2\pi i}\int_C \frac{T_{n+1}(x)g(z)}{T_{n+1}(z)(z-x)}dz.$$

Moreover if $g(x)$ extends to an analytic function within the ellipse E_r given by the locus of points $\frac{1}{2}(r\exp(i\theta) + r^{-1}\exp(-i\theta))$ (for some $r > 1$ and as θ varies from 0 to 2π and $|g(z)| \leqslant M$ at every point z on E_r then for every real $x \in [-1,1]$ the approximating polynomial $p_n(x)$ exhibits spectral convergence:

$$|g(x) - p_n(x)| \leqslant \frac{(r + r^{-1})M}{(r^{n+1} + r^{-(n+1)})(r + r^{-1} - 2)}.$$

This exponential accuracy is yet another desirable aspect of using Chebyshev polynomials for the interpolation basis.

Identifying $S_n(\bar{x}_l, x)$ as the interpolating function for the node $\bar{x}_l$, it follows from Eq. (5) that a low-rank approximation of the kernel $K(x,y)$ using Chebyshev polynomials is given by

$$K(x,y) \approx \sum_{l=1}^{n}\sum_{m=1}^{n} K(\bar{x}_l, \bar{y}_m) S_n(\bar{x}_l, x) S_n(\bar{y}_m, y). \tag{6}$$

Substituting this expression into Eq. (1) and changing the order of summation we have

$$f(x_i) = \sum_{j=1}^{N} K(x_i, y_j)\sigma_j \approx \sum_{j=1}^{N}\left[\sum_{l=1}^{n}\sum_{m=1}^{n} K(\bar{x}_l, \bar{y}_m) S_n(\bar{x}_l, x_i) S_n(\bar{y}_m, y_j)\right]\sigma_j = \sum_{l=1}^{n} S_n(\bar{x}_l, x_i)\sum_{m=1}^{n} K(\bar{x}_l, \bar{y}_m)\sum_{j=1}^{N}\sigma_j S_n(\bar{y}_m, y_j).$$

From this decomposition a fast summation method using Chebyshev interpolation can be constructed.

1. First compute the weights at the Chebyshev nodes $\bar{y}_m$ by anterpolation:

$$W_m = \sum_{j=1}^{N}\sigma_j S_n(\bar{y}_m, y_j), \quad m = 1, \ldots, n$$

2. Next compute $f(x)$ at the Chebyshev nodes $\bar{x}_l$:

$$f(\bar{x}_l) = \sum_{m=1}^{n} W_m K(\bar{x}_l, \bar{y}_m), \quad l = 1, \ldots, n$$

3. Last compute $f(x)$ at the observation points x_i by interpolation:

$$f(x_i) = \sum_{l=1}^{n} f(\bar{x}_l) S_n(\bar{x}_l, x_i), \quad i = 1, \ldots, N$$

The computational cost of steps 1 and 3 are both $O(nN)$ while step 2 is $O(n^2)$, hence for $n \ll N$ the algorithm scales like $O(2nN)$.

The decomposition above can be extended to include kernels $K(x,y)$ that are defined over arbitrary rectangular domains $[a,b] \times [c,d]$ by mapping back to the square $[-1,1] \times [-1,1]$ via linear transformation. In addition this fast summation method can be extended to higher dimensions by taking a tensor product of the interpolating functions S_n, one for each dimension. For example consider a 3-D kernel $K(\boldsymbol{x},\boldsymbol{y})$ where $\boldsymbol{x} = (x_1, x_2, x_3)$ and $\boldsymbol{y} = (y_1, y_2, y_3)$. The low-rank approximation of the kernel $K(\boldsymbol{x},\boldsymbol{y})$ using Chebyshev polynomials can be expressed as

$$K(\boldsymbol{x},\boldsymbol{y}) \approx \sum_{\boldsymbol{l}} \sum_{\boldsymbol{m}} K(\bar{\boldsymbol{x}}_{\boldsymbol{l}}, \bar{\boldsymbol{y}}_{\boldsymbol{m}}) R_n(\bar{\boldsymbol{x}}_{\boldsymbol{l}}, \boldsymbol{x}) R_n(\bar{\boldsymbol{y}}_{\boldsymbol{m}}, \boldsymbol{y}) \tag{7}$$

where

$$R_n(\boldsymbol{x},\boldsymbol{y}) = S_n(x_1, y_1) S_n(x_2, y_2) S_n(x_3, y_3)$$

and $\bar{\boldsymbol{x}}_{\boldsymbol{l}} = (\bar{\boldsymbol{x}}_{\boldsymbol{l}_1}, \bar{\boldsymbol{x}}_{\boldsymbol{l}_2}, \bar{\boldsymbol{x}}_{\boldsymbol{l}_3})$ and $\bar{\boldsymbol{y}}_{\boldsymbol{m}} = (\bar{y}_{m_1}, \bar{y}_{m_2}, \bar{y}_{m_3})$ are 3-vectors of Chebyshev nodes with $l_i, m_i \in \{1, \ldots, n\}$.

3. A black-box FMM with Chebyshev interpolation

In the previous section a fast summation method was constructed for continuous kernels based on a low-rank approximation using Chebyshev polynomials. However if the kernel contains discontinuities in its domain, e.g. the Laplacian kernel $1/|x-y|$, this low-rank representation is not applicable. In order for the low-rank approximation (6) to accurately represent the kernel, the observation and source intervals need to be well-separated, i.e., the two intervals are non-overlapping. Hence (6) can be thought of as a far-field approximation of the kernel $K(x,y)$. Local interactions involving observation points and sources in non-well-separated intervals can also be computed with the far-field approximation by subdividing the intervals. On this refined scale, interactions between well-separated observation points and sources can be treated by (6). Applying this refinement recursively produces a multilevel fast summation method. A black-box fast multipole method (bbFMM) can be constructed by combining this multilevel scheme with the FMM tree structure as detailed in [1]. Our method is a black-box in the sense that the functional form of the low-rank approximation (6) is independent of the kernel. Let the root level of the tree (level 0) be the computational interval containing all observation points and sources. The algorithm for a κ-level 1-D bbFMM is as follows:

1. For all subintervals I on level κ compute the weights at the Chebyshev nodes $\bar{y}_m^I$ by anterpolation:

$$W_m^I = \sum_{y_j \in I} \sigma_j \, S_n(\bar{y}_m^I, y_j), \quad m = 1, \ldots, n$$

2. For all subintervals I on level k compute the weights at the Chebyshev nodes $\bar{y}_m^I$ by recursion, $\kappa - 1 \geqslant k \geqslant 0$ (M2M):

$$W_m^I = \sum_{\substack{J,\ \text{child} \\ \text{interval of } I}} \sum_{m'} W_{m'}^J \, S_n\left(\bar{y}_m^I, \bar{y}_{m'}^J\right), \quad m = 1, \ldots, n$$

3. Calculate the far-field contribution at the Chebyshev nodes $\bar{x}_l^I$ for all subintervals J in the interaction list of I on level k, $0 \leqslant k \leqslant \kappa$ (M2L):

$$g_l^I = \sum_{\substack{J \text{ in interaction} \\ \text{list of } I}} \sum_{m} W_m^J \, K(\bar{x}_l^I, \bar{y}_m^J), \quad l = 1, \ldots, n$$

4. Letting $f_l^I = g_l^I$ for all subintervals I on level 0, then for each subinterval I on level k, $1 \leqslant k \leqslant \kappa$, add the effect of the far-field sources by interpolating the field from the parent interval on level $k-1$ (L2L):

$$f_l^I = g_l^I + \sum_{l'} f_{l'}^J \, S_n\left(\bar{x}_l^I, \bar{x}_{l'}^J\right), \quad l = 1, \ldots, n$$

where J is the parent of I.

5. Finally compute $f(x_i)$, where x_i is in subinterval I on level κ, by interpolating the far-field approximation and adding the nearby interactions:

$$f(x_i) = \sum_l f_l^I \, S_n(\bar{x}_l^I, x_i) + \sum_{\substack{J,\ \text{nearest neighbor} \\ \text{interval of } I}} \sum_{y_j \in J} \sigma_j \, K(x_i, y_j), \quad i = 1, \ldots, N$$

An analogous algorithm can be written for the 3-D bbFMM using (7).

4. Fast convolution using SVD compression

In the FMM the largest contribution to the computational cost is the multipole-to-local (M2L) operation described in step 3 of the bbFMM algorithm. As such the optimization of this operation is important for an efficient fast summation method. One way to reduce the cost is to produce a more compact multipole and local expansion. Here we propose using the singular value decomposition to compress the low-rank approximation generated by Chebyshev interpolation.

To find such a low-rank approximation of the kernel we look to minimize the approximation error with respect to a specified norm. For sums of the form (1) where the distribution of observation points and sources is assumed to be uniform, a natural estimate of the error introduced by replacing the kernel $K(x,y)$ with a low-rank approximation $\widetilde{K}(x,y)$ is

$$\varepsilon = \left[\int_{-1}^{1} \int_{-1}^{1} [K(x,y) - \widetilde{K}(x,y)]^2 \, dxdy \right]^{1/2}$$

where the domain of $K(x,y)$ is $[-1,1] \times [-1,1]$. This expression can be approximated using a Chebyshev quadrature for the double integral

$$\varepsilon' = \left[\sum_{l=1}^{n} \sum_{m=1}^{n} \omega_l^{\mathrm{x}} \omega_m^{\mathrm{y}} [K(\bar{x}_l, \bar{y}_m) - \widetilde{K}(\bar{x}_l, \bar{y}_m)]^2 \right]^{1/2}$$

where $\{\bar{x}_l\}$ and $\{\bar{y}_m\}$ are the Chebyshev nodes and

$$\omega_l^{\mathrm{x}} = \frac{\pi}{n} \sqrt{1 - \bar{x}_l^2}$$
$$\omega_m^{\mathrm{y}} = \frac{\pi}{n} \sqrt{1 - \bar{y}_m^2}$$

are the corresponding weights. Defining the matrices $\mathbf{K}_{lm} = K(\bar{x}_l, \bar{y}_m)$, $\widetilde{\mathbf{K}}_{\mathbf{lm}} = \widetilde{K}(\bar{x}_l, \bar{y}_m)$, $(\Omega^{\mathrm{x}})_{ll} = \omega_l^{\mathrm{x}}$, and $(\Omega^{\mathrm{y}})_{mm} = \omega_m^{\mathrm{y}}$, the error ε' can be expressed in terms of the Frobenius norm:

$$\varepsilon' = \| (\Omega^{\mathrm{x}})^{\frac{1}{2}} \mathbf{K} (\Omega^{\mathrm{y}})^{\frac{1}{2}} - (\Omega^{\mathrm{x}})^{\frac{1}{2}} \widetilde{\mathbf{K}} (\Omega^{\mathrm{y}})^{\frac{1}{2}} \|_F \tag{8}$$

Observe that for the low-rank approximation (6), we have $\widetilde{\mathbf{K}} = \mathbf{K}$ which gives $\varepsilon' = 0$. However, we are interested in obtaining a compressed low-rank approximation, so we look for $\widetilde{\mathbf{K}}$ such that: $\mathrm{rank}(\widetilde{\mathbf{K}}) < \mathrm{rank}(\mathbf{K})$. The solution to this constrained minimization aof (8) is given by a theorem from numerical linear algebra which states that the best rank-r approximation of an n-by-n matrix $\mathbf{A}$, where $r \leqslant n$, with respect to the Frobenius norm corresponds to picking the r left and right singular vectors of the SVD of $\mathbf{A}$ with the largest singular values [14]. Let the SVD of $(\Omega^{\mathrm{x}})^{\frac{1}{2}} \mathbf{K} (\Omega^{\mathrm{y}})^{\frac{1}{2}}$ be denoted by

$$(\Omega^{\mathrm{x}})^{\frac{1}{2}} \mathbf{K} (\Omega^{\mathrm{y}})^{\frac{1}{2}} = \mathbf{U} \Sigma \mathbf{V}^T$$

where the columns of $\mathbf{U}$ are the left singular vectors, the columns of $\mathbf{V}$ are the right singular vectors, and Σ is a diagonal matrix whose entries are the singular values of $(\Omega^{\mathrm{x}})^{\frac{1}{2}} \mathbf{K} (\Omega^{\mathrm{y}})^{\frac{1}{2}}$ in order of decreasing magnitude. The optimal rank-r approximation of $\mathbf{K}$ for Eq. (8) is then

$$\widetilde{\mathbf{K}} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}} \mathbf{U}_r \Sigma_r \mathbf{V}_r^T (\Omega^{\mathrm{y}})^{-\frac{1}{2}} \tag{9}$$

where $\mathbf{U}_r$ is the first r columns of $\mathbf{U}, \mathbf{V}_r$ is the first r columns of $\mathbf{V}$, and Σ_r is a diagonal matrix containing the first r singular values. It should be noted that if the compression was done by computing the SVD of $\mathbf{K}$ instead of $(\Omega^{\mathrm{x}})^{\frac{1}{2}} \mathbf{K} (\Omega^{\mathrm{y}})^{\frac{1}{2}}$ then, using a similar argument as above, it can be shown that the error in the low-rank approximation is minimized for a distribution of sources and observation points concentrated on the boundaries. However for a uniform distribution, the reduced-rank matrix in (9) gives the optimal compression.

We now proceed to show how to compress the M2L operator using the SVD compression described above. With the same notation as in the previous section, the M2L operation between observation points $\{\bar{x}_l\}$ and sources located at $\{\bar{y}_m\}$ can be expressed as the matrix-vector product

$$\boldsymbol{g} = \mathbf{K}\boldsymbol{w}$$

where $\boldsymbol{g}_l = g(\bar{x}_l)$, $\mathbf{K}_{lm} = K(\bar{x}_l, \bar{y}_m)$, and $\boldsymbol{w}_m = W_m$. Then $\boldsymbol{g}$ can be approximated by

$$\tilde{\mathbf{g}} = \widetilde{\mathbf{K}}\mathbf{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{U}_r\Sigma_r\mathbf{V}_r^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w}.$$

The compressed low-rank approximation reduces the cost from an n-by-n matrix-vector product to two n-by-r matrix-vector products. This calculation can be further streamlined so as to involve mainly r-by-r matrices. To begin consider the M2L operation in 3-D for an observation cell on level k of the FMM tree. Each observation cell interacts with up to $6^3 - 3^3 = 189$ source cells, with each interaction indexed by its transfer vector. The union of transfer vectors over all observation cells on level k forms a set of $7^3 - 3^3 = 316$ vectors. Assuming a translational invariant kernel, there are 316 unique M2L operators on level k, each one corresponding to a particular transfer vector. Let $\mathbf{K}^{(i)}$ denote the 3-D M2L operator for the i-th transfer vector. Then this collection of M2L operators, with the appropriate weighting in 3-D, can be expressed either as a fat matrix

$$\mathbf{K}_{\mathrm{fat}} = \left[(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(1)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(2)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\ \cdots\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(316)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\right]$$

with the $(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}$ blocks arranged in a single row, or as a thin matrix

$$\mathbf{K}_{\mathrm{thin}} = \left[(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(1)}(\Omega^{\mathrm{y}})^{\frac{1}{2}};\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(2)}(\Omega^{\mathrm{y}})^{\frac{1}{2}};\ \cdots;\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(316)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\right]$$

with the $(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}$ blocks arranged in a single column. Here Ω^{x} and Ω^{y} are the 3-D analogs of the weighting matrices used in (8).

To construct compact multipole and local expansions we perform two SVDs, one on $\mathbf{K}_{\mathrm{fat}}$ and the other on $\mathbf{K}_{\mathrm{thin}}$:

$$\mathbf{K}_{\mathrm{fat}} = \left[(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(1)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(2)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\ \cdots\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(316)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\right] = \mathbf{U}\Sigma\left[{\mathbf{V}^{(1)}}^T{\mathbf{V}^{(2)}}^T\ \cdots\ {\mathbf{V}^{(316)}}^T\right]$$

$$\mathbf{K}_{\mathrm{thin}} = \left[(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(1)}(\Omega^{\mathrm{y}})^{\frac{1}{2}};(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(2)}(\Omega^{\mathrm{y}})^{\frac{1}{2}};\ \cdots;\ (\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(316)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\right] = \left[\mathbf{R}^{(1)};\mathbf{R}^{(2)};\ \cdots;\mathbf{R}^{(316)}\right]\Lambda\mathbf{S}^T.$$

Observe that if the kernel is symmetric then $\mathbf{K}_{\mathrm{thin}} = \mathbf{K}_{\mathrm{fat}}^T$ so the two SVDs are just transposes of each other. The pre-computation cost for these SVDs is $O(\kappa)$ since the dimensions of these matrices are independent of the problem size.

The cost of the convolution for the i-th transfer vector between $\mathbf{K}^{(i)}$ and a vector of sources $\mathbf{w}$ can be reduced by employing these two SVDs as follows. First begin by introducing the weighting matrices Ω^{x} and Ω^{y} and substituting in the i-th block of $\mathbf{K}_{\mathrm{thin}}$.

$$\mathbf{K}^{(i)}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{R}^{(i)}\Lambda\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w}$$

Next the identity matrix $\mathbf{S}^T\mathbf{S}$ can be inserted between Λ and $\mathbf{S}^T$ after which the i-th block of $\mathbf{K}_{\mathrm{thin}}$ is replaced.

$$\mathbf{K}^{(i)}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{R}^{(i)}\Lambda\mathbf{S}^T\mathbf{S}\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\mathbf{S}\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w}$$

Now substituting in the i-th block of $\mathbf{K}_{\mathrm{fat}}$ and inserting the identity matrix $\mathbf{U}^T\mathbf{U}$ between $\mathbf{U}$ and Σ we have

$$\mathbf{K}^{(i)}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{U}\Sigma{\mathbf{V}^{(\mathbf{i})}}^T\mathbf{S}\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{U}\mathbf{U}^T\mathbf{U}\Sigma{\mathbf{V}^{(\mathbf{i})}}^T\mathbf{S}\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w} = (\Omega^{\mathrm{x}})^{-\frac{1}{2}}\mathbf{U}\left[\mathbf{U}^T(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\mathbf{S}\right]\mathbf{S}^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\boldsymbol{w}.$$

Consider the term inside the square brackets:

$$\mathbf{U}^T(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\mathbf{S} = \Sigma{\mathbf{V}^{(i)}}^T\mathbf{S} = \mathbf{U}^T\mathbf{R}^{(i)}\Lambda.$$

This shows that the rows and columns of $\mathbf{U}^T(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{(i)}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\mathbf{S}$ decay as quickly as the singular values found in Σ and Λ. Hence the product $\mathbf{K}^{(i)}\boldsymbol{w}$ can be approximated by keeping only the first r singular vectors in each of the SVDs. Let $\mathbf{U}_r$ and $\mathbf{S}_r$ denote the r left singular vectors and r right singular vectors, respectively. Using these reduced SVDs a fast convolution method involving compressed multipole and local expansions can be formulated. The M2L operation, step 3 in the bbFMM algorithm, can now be done as follows. Using the notation adopted in the bbFMM algorithm we have:

0. Pre-computation: compute the compressed M2L operators for all transfer vectors $i = 1,\ldots,316$ on level $k, 0 \leqslant k \leqslant \kappa$

$$\mathbf{C}^{i,k} = (\mathbf{U}_r^k)^T(\Omega^{\mathrm{x}})^{\frac{1}{2}}\mathbf{K}^{i,k}(\Omega^{\mathrm{y}})^{\frac{1}{2}}\mathbf{S}_r^k$$

3a. Pre-processing: compute the compressed multipole coefficients for all source cells I on level $k, 0 \leqslant k \leqslant \kappa$

$$\boldsymbol{w}_{\mathrm{c}}^I = (\mathbf{S}_r^k)^T(\Omega^{\mathrm{y}})^{-\frac{1}{2}}\ \boldsymbol{w}^I$$

3b. Convolution: calculate the compressed local coefficients for all observation cells I on level k, $0 \leqslant k \leqslant \kappa$; let $i(I,J)$ denote the index of the transfer vector representing the interaction between cells I and J

$$\boldsymbol{g}_{\mathrm{c}}^I = \sum_{\substack{J \text{ in interaction} \\ \text{list of } I}} \mathbf{C}^{i(I,J),k}\ \boldsymbol{w}_{\mathrm{c}}^J$$

Table 1
Computational cost of pre-computation. The pre-computation includes all steps which depend only on the boxes in the tree and are independent of the particles' location and σ_j. The kernel is $K(x,y)$. Notations: N: number of particles, κ: number of levels.

Kernel type	Cost
General case	$O(N)$
Translational invariant	$O(\kappa)$
Homogeneous	$O(1)$
Symmetric	Cost is reduced by 2

3c. Post-processing: compute the coefficients of the local expansion for all observation cells I on level $k, 0 \leqslant k \leqslant \kappa$

$$\boldsymbol{g}^I = (\Omega^{\mathrm{x}})^{-\frac{1}{2}} \mathbf{U}_r^k \, \boldsymbol{g}_{\mathrm{c}}^I$$

Most of the computational cost in the fast convolution algorithm is concentrated in step 3b which involves **reduced-rank** matrix-vector products. Without the SVD compression, the cost of the M2L operation corresponds to **full-rank** matrix-vector products.

The cost and memory requirements of the pre-computation step can be reduced for the case of homogeneous kernels. Recall that a function $K(\boldsymbol{x},\boldsymbol{y})$ is homogeneous of degree m if $K(\alpha\boldsymbol{x},\alpha\boldsymbol{y}) = \alpha^m K(\boldsymbol{x},\boldsymbol{y})$ for any nonzero real α. In this case the M2L operators can be determined for interactions between observation and source cubes with unit volume and two SVDs are performed. Letting $\mathbf{D}^{(i)}$ represent the compressed M2L operators constructed from these two SVDS then, for a cubic computational cell with sides of length L, the operators on each level of the FMM tree are scaled versions of $\mathbf{D}^{(i)}$:

$$\mathbf{C}^{(i),k} = \left(\frac{L}{2^k}\right)^m \mathbf{D}^{(i)}.$$

Hence only one set of operators, $\{\mathbf{D}^{(i)}\}$, needs to be computed and stored because $\mathbf{C}^{(i),k}$ can be easily computed from $\mathbf{D}^{(i)}$. In addition only the singular vectors from the two SVDs are needed for the fast convolution algorithm because singular vectors are invariant under multiplicative scaling of the matrix. In that case the cost of the pre-computation is $O(1)$ for any problem size.

This is summarized in Table 1. In the general case, the SVD provides only limited savings because a new SVD has to be computed for every cluster. If the method is applied many times for a given tree, this is still computationally advantageous.

5. Numerical results

In this section we will present numerical results for the bbFMM. The accuracy of the method will be examined as well as the computational cost. Results for five kernels will be detailed: (1) the Laplacian kernel $1/r$, (2) the kernel $1/r^4$, (3) the Stokes kernel $I_{3\times 3}/r + (\vec{r}\otimes\vec{r})/r^3$ where $\vec{r}$ is the 3-dimensional position vector, (4) the 3-D isotropic multiquadric radial basis function $\sqrt{(r/a)^2+1}$ where a is a scaling constant, and (5) the isotropic Gaussian function $\exp[-(r/a)^2]$ where a is a scaling constant.

5.1. Compression using SVD

We start by examining the amount of compression that can be achieved in the M2L operation by using the SVDs of the kernel matrices $\mathbf{K}_{\mathrm{fat}}$ and $\mathbf{K}_{\mathrm{thin}}$. In Fig. 1 the singular values of the Laplacian kernel matrices are plotted for various number of

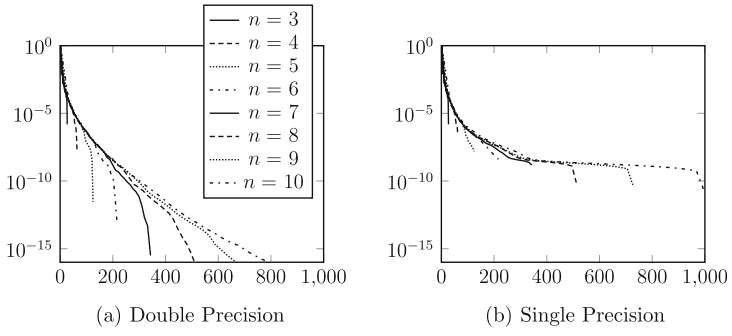


Fig. 1. Singular values for the Laplacian kernel. The relative singular value magnitude (vertical axis) is plotted as a function of the singular value index (shown on the horizontal axis). The subsequent plots (Figs. 2–5) show the decay of the singular values for four other kernels. The legend is the same for all plots.

Chebyshev nodes n. Since the Laplacian kernel is symmetric the singular values of $\mathbf{K}_{\text{fat}}$ and $\mathbf{K}_{\text{thin}}$ are identical. For each n the singular values are scaled such that the largest singular value is normalized to 1. The index of the singular values $(1,\ldots,n^3)$ is represented on the horizontal axis. The left subfigure shows the singular values obtained by setting up the kernel matrices and performing the SVD in double precision while the right subfigure corresponds to single precision. Taking the curve for $n = 10$ as the best approximation to the continuous SVD, observe that the double-precision singular values are accurate up to approximately the index $n^3/2$ [12]. This suggests that the kernel matrices can be compressed by a factor of 2 without adversely affecting the accuracy of the method. In the single-precision plot, we see that the amount of compression is less as the curves deviate at an index greater than $n^3/2$. The leveling of the curves around 10^{-8} reflects the roundoff error incurred by using single precision.

Figs. 2 and 3 are similar plots for the $1/r^4$ and Stokes kernels, respectively. For the Stokes kernel the indices of the singular values are $1,\ldots,3n^3$. We stopped at $n = 7$ because we ran out of computer memory. The 9 components of the 3×3 Stokes kernel were treated simultaneously. The memory requirement can be reduced by treating each component one at a time, or by using a parallel computer with distributed memory.

While the rate of decay of the singular values for homogeneous kernels is scale-invariant, this is not the case for inhomogeneous kernels such as the 3-D isotropic multiquadric radial basis function $\sqrt{(r/a)^2+1}$. In Fig. 4 the double-precision singular values for two radial basis functions, $a = 1$ and $a = 8$, are shown. When $a \ll 1$ or $a = O(1)$ (e.g. Fig. 4(a)) the profile is similar to that obtained for the Laplacian kernel and hence the kernel matrices can be compressed by keeping only half of the

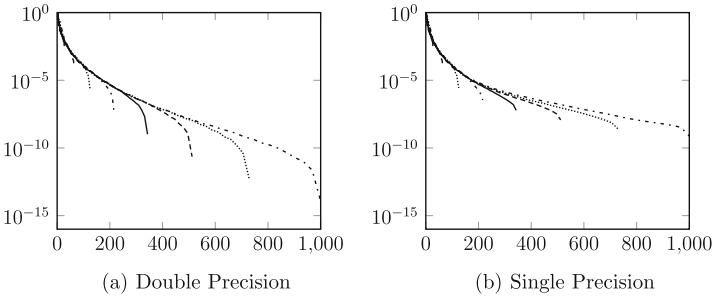


Fig. 2. Singular values for the $1/r^4$ kernel.

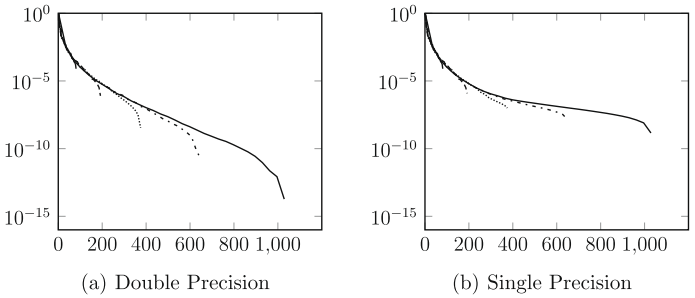


Fig. 3. Singular values for the Stokes kernel. In this plot, n ranges from 3 to 7.

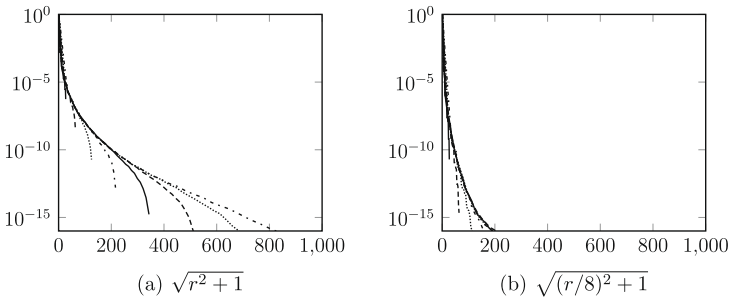


Fig. 4. Singular values for two 3-D isotropic multiquadric radial basis functions. Double precision was used.

singular values. However for $a \gg 1$ (e.g. Fig. 4(b)) the decay is much more rapid since the radial basis function is well-approximated by the constant 1 around the origin. The implication of this behavior for the multilevel FMM scheme is that fewer singular values can be retained for deeper levels in the tree, thereby reducing the computational cost. This illustrates that in order to achieve the best compression for a particular kernel, the decay behavior of the singular values needs to be studied on a case-by-case basis.

Fig. 5 shows the double-precision singular values for the isotropic Gaussian function $\exp[-(r/a)^2]$ with $a = 1$ and $a = 8$. For $a \gg 1$ the Gaussian is well-approximated by the constant 1 around the origin. Therefore Fig. 5(b) is very similar to Fig. 4(b).

In all subsequent benchmarks, the kernel matrices were compressed by retaining only the largest $n^3/2$ singular values except for the Stokes kernel where $3n^3/2$ were kept.

5.2. Interpolation error

To investigate the Chebyshev interpolation error we looked at a system of 10,000 uniformly distributed sources in a cubic computational domain with edge length 1. Each source was assigned a strength of +1 or −1 such that the computational cell has zero net source strength. The observation points were chosen to be identical to the source locations and the pairwise interactions between these points were computed for each of the five kernels. To compute the error a subset of 100 observation points was used. The relative error in the observation values was measured with respect to the L^2 and L^∞ norms for various n by using the values obtained by direct calculation as the reference solution. Letting $f^{\text{FMM}}(x_i)$ and $f^{\text{dir}}(x_i)$ be the observation values computed with bbFMM and direct calculation, respectively, the errors are given by

$$\varepsilon_2 = \left[\frac{\sum_{i=1}^{100}\left(f^{\text{FMM}}(x_i) - f^{\text{dir}}(x_i)\right)^2}{\sum_{i=1}^{100}\left(f^{\text{dir}}(x_i)\right)^2}\right]^{1/2} \tag{10}$$

and

$$\varepsilon_\infty = \frac{\max\limits_{1\leqslant i\leqslant 100}\left|f^{\text{FMM}}(x_i) - f^{\text{dir}}(x_i)\right|}{\max\limits_{1\leqslant i\leqslant 100}\left|f^{\text{dir}}(x_i)\right|}. \tag{11}$$

Fig. 6 shows that for the Laplacian kernel the error in both norms exhibits spectral convergence when double precision is used. However for single-precision the error levels off for large n as roundoff error degrades the solution. Similar results were observed for the $1/r^4$ (Fig. 7) and Stokes (Fig. 8) kernels. It should be noted that for the Stokes kernel we do not see the single-precision error curve level off because the roundoff error is minimal for the values of n tested.

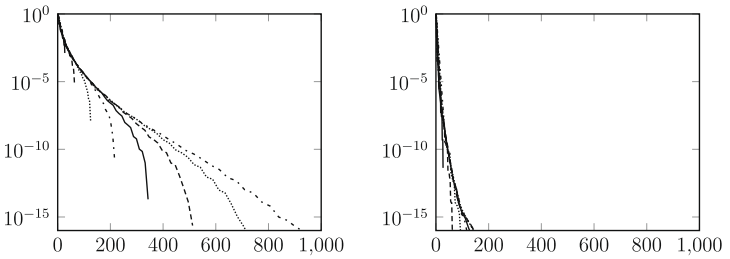


Fig. 5. Singular values for two isotropic Gaussian functions. Double precision was used.

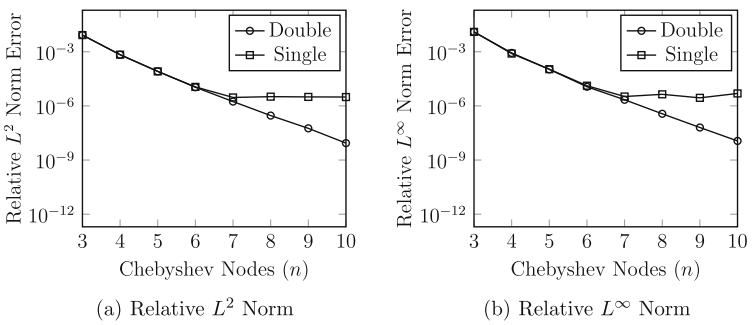


Fig. 6. Interpolation error for the Laplacian kernel. The relative L^2- and L^∞-norm errors are defined by Eqs. (10) and (11), respectively.

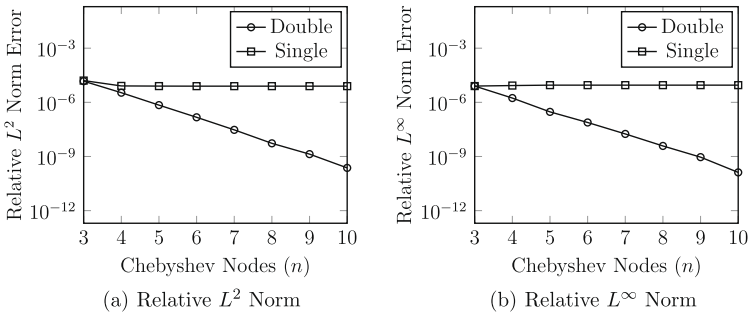


Fig. 7. Interpolation error for the $1/r^4$ kernel.

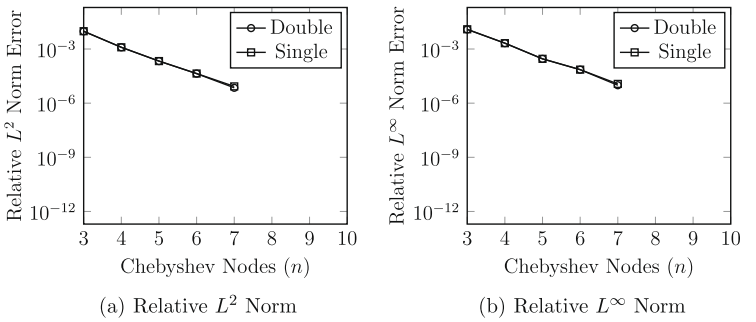


Fig. 8. Interpolation error for the Stokes kernel.

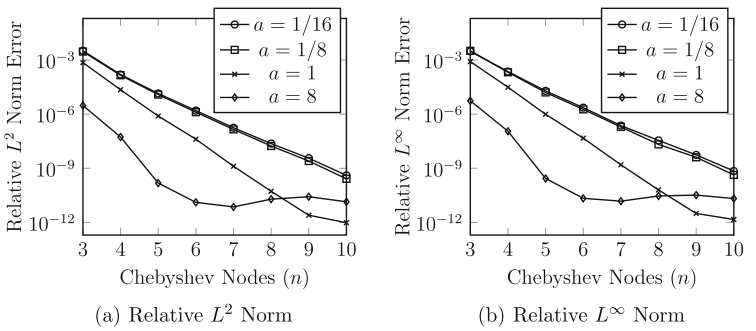


Fig. 9. Interpolation error for various 3-D isotropic multiquadric radial basis functions $\sqrt{(r/a)^2+1}$. Double precision was used.

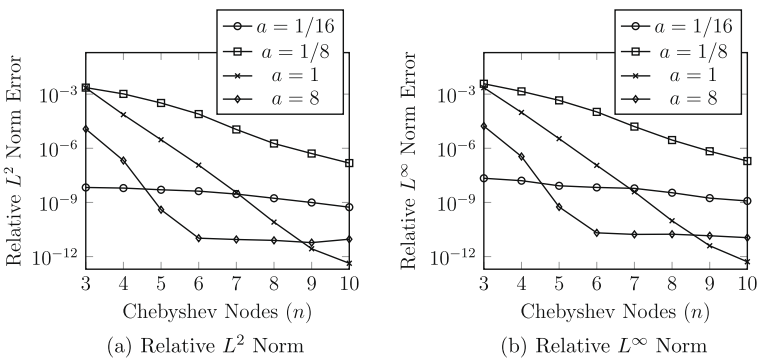
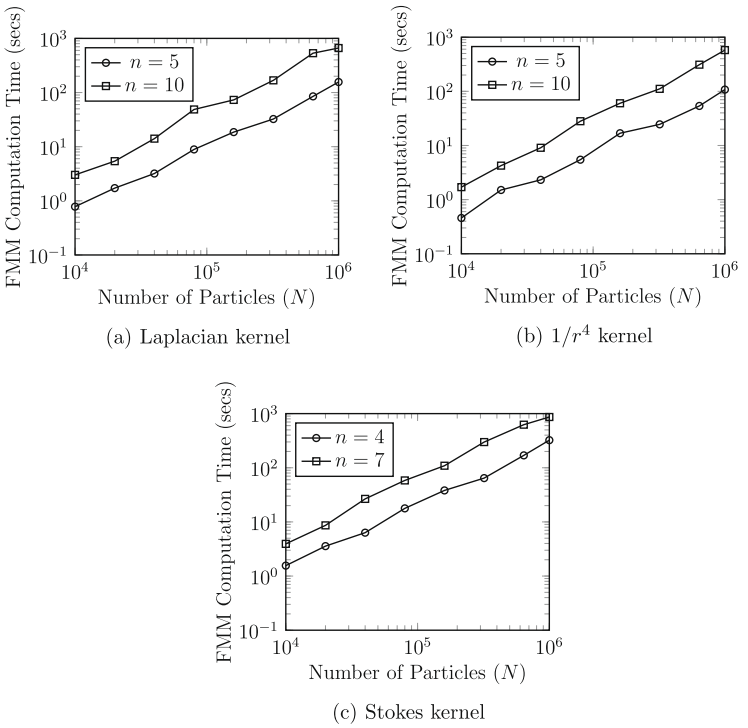


Fig. 10. Interpolation error for various isotropic Gaussian functions $\exp[-(r/a)^2]$. Double precision was used.



(a) Laplacian kernel (b) $1/r^4$ kernel

(c) Stokes kernel

Fig. 11. Computational cost for the Laplacian, $1/r^4$, and Stokes kernels. Two choices for the number of Chebyshev nodes (n) were made for each kernel.

In Fig. 9 the error is plotted for various isotropic radial basis functions. The $a = 8$ curve displays a higher rate of convergence than for $a = 1$ due to the rapid decay of singular values observed in Fig. 4(b). For large n the curve levels off due to roundoff error. When $a \ll 1$ the radial basis function $\sqrt{(r/a)^2 + 1}$ approaches the homogeneous function r/a. As a result the error curves for $a = 1/16$ and $a = 1/8$ are nearly identical.

For the isotropic Gaussian function (Fig. 10), the $a = 8$ curve displays a faster rate of convergence than for $a = 1$ as predicted by the rapid decay of singular values seen in Fig. 5(b). Roundoff error is responsible for the curve leveling off at large values of n. The convergence rate is slower for small a (e.g. $a = 1/16$ and $a = 1/8$) because the function $\exp[-(r/a)^2]$ becomes less smooth. In addition when $a \ll 1$ the contributions from the far-field are negligible compared to those from the near field. As a result the error for the $a = 1/16$ case is small despite the poor rate of convergence for the far-field expansion.

5.3. Computational cost

To examine the computational complexity of bbFMM, a system of N uniformly distributed sources in a cubic computational domain with edge length 1 was used. Each source was assigned a strength of +1 or −1 such that the computational cell has zero net source strength. The observation points were chosen to be identical to the source locations and the sources interacted according to the Laplacian, $1/r^4$, and Stokes kernels. As the number of sources N was varied from 10^4 to 10^6 the FMM computation time (cost of the M2M, M2L, and L2L operations and cost of direct interactions) was measured for $n = 5$ and $n = 10$ and plotted in Fig. 11. For a given number of sources the number of levels in the FMM tree was selected to achieve a computational balance between the M2L operations and the direct interactions, the two most expensive steps in the method. We observed the correct $O(N)$ complexity for the three kernels.[1] The kinks in the curves result from increases in the number of levels in the tree.

5.4. Comparison with analytic multipole expansion

To study the efficiency of the SVD compression, a comparison was done with the analytic multipole expansion of the Laplacian kernel using Legendre polynomials for a system consisting of two well-separated cubes. The source cube was centered at the origin while the observation cube was centered at $(1/2, 0, 0)$. Both cubes have edge length (1/4). Letting $\vec{r}_i$ denote

[1] For homogeneous distributions of points, this complexity can be achieved using a tree with a constant depth. This is the algorithm we implemented. If the distribution of points becomes too inhomogeneous, an adaptive tree [15] with a varying number of levels must be used in order to achieve an $O(N)$ complexity.

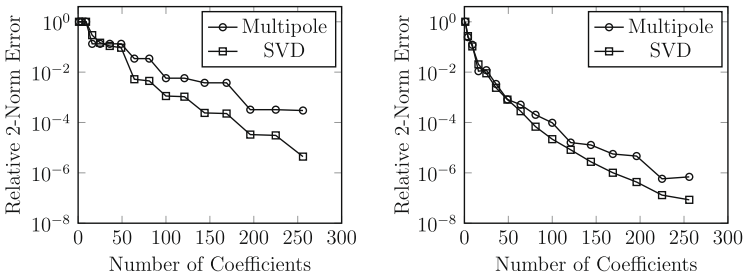


Fig. 12. Error comparison between the SVD compression and the analytic multipole expansion. Left: charges distributed on the surface of the cubes. Right: random uniform charge distribution.

the position vector of the i-th observation point and $\vec{r}_j$ the position of the j-th source, the p-th order analytic multipole expansion of the observational value is given by

$$f(\vec{r}_i) = \frac{1}{r_i}\sum_{j=1}^{N} q_j \sum_{l=0}^{p-1}\left(\frac{r_j}{r_i}\right)^l P_l(\cos\theta_{ij}) \tag{12}$$

where P_l is the Legendre polynomial of degree l and

$$\cos\theta_{ij} = \frac{\vec{r}_i^T\vec{r}_j}{r_i\, r_j}$$

To derive a separable expansion, i.e., $\vec{r}_i$ and $\vec{r}_j$ appear in separate factors, P_l can be replaced by $2l+1$ spherical harmonics. Then this expansion can be rewritten in terms of a finite number of spherical harmonics. For a p-th order multipole expansion the number of spherical harmonics coefficients is

$$\sum_{l=0}^{p-1}(2l+1) = p + 2\sum_{l=0}^{p-1} l = p + (p-1)p = p^2.$$

This comparison was carried out for two systems. The first system is constructed by placing $6^3 - 4^3 = 152$ charges on the faces of the cubes, such that, on each face, we have $6^2 = 36$ charges distributed on a regular grid. Charge strengths were alternated between +1 and −1 in a checkerboard fashion. In the second system the 152 charges were uniformly distributed within each cube. For various values of p, the L^2-norm error in the observational values was computed using the values obtained by direct calculation as the reference solution. The error was also determined when retaining p^2 singular values in the bbFMM. We used $n = 10$ Chebyshev nodes in each direction for both test problems. Fig. 12 shows the errors for the two systems. The values of p^2, which were varied from $1^2 = 1$ to $16^2 = 256$, are plotted on the horizontal axis. From the plots, the SVD compression is at least as good as the analytic multipole expansion with respect to the L^2-norm error. For the case on the left (charges confined to the surface) the difference between the two methods is more substantial for large p. This is because the multipole expansion is similar to a Taylor series and therefore does not do a good job of approximating charges near the boundaries of the computational domain. The SVD compression with the Chebyshev-based expansion, however, is able to resolve those charges more accurately.

6. Conclusion

We have presented a new black-box or kernel-independent fast multipole method. The method requires as input only a user defined routine to numerically evaluate $K(x,y)$ at a given point (x,y). This is very convenient for complex kernels, for which analytical expansions might be difficult to obtain. This method relies on Chebyshev polynomials for the interpolation part and on singular value decompositions to further reduce the computational cost. Because of the SVD, we can prove that the scheme uses the minimal number of coefficients given a tolerance ε. The pre-computing time of the method was analyzed and was found to be small for most practical cases. The numerical scheme was tested on various problems. The accuracy was confirmed and spectral convergence was observed. The linear complexity was also confirmed by numerical experiments.

A limitation of the current approach is that the M2L operator is dense. This, probably, cannot be avoided if one uses an SVD. However, if one agrees to choose a different expansion, involving more terms, it is sometimes possible to design methods with diagonal M2L operators. Even though more coefficients are used, one may end up with a faster algorithm overall. This is the case for example with the plane wave version of the FMM for $1/r$ [16]. In that case, a specialized algorithm can be faster than the proposed scheme. It remains to be seen if a black-box method can be derived using diagonal operators.

Another issue is the extension to periodic boundary conditions, in particular to the case of conditionally convergent series like $K(x,y) = 1/r$, which converge only for a cell with zero net charge.

References

[1] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, J. Comp. Phys. 73 (2) (1987) 325–348.
[2] P.G. Martinsson, V. Rokhlin, An accelerated kernel-independent fast multipole method in one dimension, SIAM J. Sci. Comput. 29 (3) (2007) 1160–1178.
[3] L. Ying, G. Biros, D. Zorin, A kernel-independent adaptive fast multipole algorithm in two and three dimensions, J. Comp. Phys. 196 (2) (2004) 591–626.
[4] W. Dahmen, H. Harbrecht, R. Schneider, Compression techniques for boundary integral equations – asymptotically optimal complexity estimates, SIAM J. Numer. Anal. 43 (6) (2006) 2251–2271.
[5] B. Alpert, G. Beylkin, R. Coifman, V. Rokhlin, Wavelet-like bases for the fast solution of second-kind integral equations, SIAM J. Sci. Comput. 14 (1) (1993) 159–184.
[6] Z. Gimbutas, M. Minion, L. Greengard, Coulomb interactions on planar structures: inverting the square root of the Laplacian, SIAM J. Sci. Comput. 22 (6) (2001) 2093–2108.
[7] Z. Gimbutas, V. Rokhlin, A generalized fast multipole method for nonoscillatory kernels, SIAM J. Sci. Comput. 24 (3) (2003) 796–817.
[8] H. Cheng, Z. Gimbutas, P.G. Martinsson, V. Rokhlin, On the compression of low-rank matrices, SIAM J. Sci. Comput. 26 (4) (2005) 1389–1404.
[9] F. Ethridge, L. Greengard, A new fast-multipole accelerated poisson solver in two dimensions, SIAM J. Sci. Comput. 23 (3) (2001) 741–760.
[10] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data, SIAM J. Sci. Comput. 14 (6) (1993) 1368–1393.
[11] A. Edelman, P. McCorquodale, S. Toledo, The future fast Fourier transform?, SIAM J Sci. Comput. 20 (3) (1999) 1094–1114.
[12] A. Dutt, M. Gu, V. Rokhlin, Fast algorithms for polynomial interpolation, integration, and differentiation, SIAM J. Numer. Anal. 33 (5) (1996) 1689–1711.
[13] J. Mason, D. Handscomb, Chebyshev Polynomials, Chapman and Hall/CRC, 2003.
[14] L. Trefethen, D. Bau III, Numerical Linear Algebra, Society for Industrial and Applied Mathematics, 1997.
[15] H. Cheng, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm in three dimensions, J. Comp. Phys. 155 (2) (1999) 468–498.
[16] L. Greengard, V. Rokhlin, A new version of the fast multipole method for the laplace equation in three dimensions, Acta Numer. 6 (1997) 229–270.