MULTISCALE MODEL. SIMUL.
Vol. 13, No. 1, pp. 423–439

FAST DIRECTIONAL COMPUTATION OF HIGH FREQUENCY BOUNDARY INTEGRALS VIA LOCAL FFTs*

LEXING YING†

Abstract. The boundary integral method is an efficient approach for solving time-harmonic acoustic obstacle scattering problems. The main computational task is the evaluation of an oscillatory boundary integral at each discretization point of the boundary. This paper presents a new fast algorithm for this task in two dimensions. This algorithm is built on top of directional low-rank approximations of the scattering kernel and uses oscillatory Chebyshev interpolation and local FFTs to achieve quasi-linear complexity. The algorithm is simple, fast, and kernel-independent. Numerical results are provided to demonstrate the effectiveness of the proposed algorithm.

Key words. boundary integral method, scattering, high frequency waves, directional algorithm, low-rank approximation, Chebyshev interpolation, fast Fourier transforms

AMS subject classifications. 65N38, 65R20, 78A45

DOI. 10.1137/140985123

1. Introduction.

This paper is concerned with the solution of time-harmonic acoustic obstacle scattering problems. Let ω be the frequency of the wave field and Ω the scatterer with boundary $\partial\Omega$. The boundary integral method is an appealing approach for this problem since it typically requires fewer unknowns and provides better accuracy when compared to volumetric-type methods. The main computational task of the boundary integral method is the evaluation of the integral

$$u(x) = \int_{\partial\Omega} G(x,y) f(y) dy, \quad x \in \partial\Omega, \tag{1}$$

where $G(x,y)$ is the Green's function of the Helmholtz operator at frequency ω, or a similar integral where the kernel is replaced with one of the derivatives of the Green's function.

The numerical treatment of $u(x)$ requires boundary discretization. For a typical scattering problem, at least a couple of points per wavelength $\lambda = 2\pi/\omega$ are required in order for the solution method to achieve reasonable accuracy. If we assume that both the diameter and the boundary length of Ω are of order $\Theta(1)$, then the boundary $\partial\Omega$ shall be discretized with a set P of $n = \Theta(1/\lambda) = \Theta(\omega)$ points. Discretizing (1) with an appropriate numerical quadrature scheme results in the following discrete summation problem: for each $x \in P$, evaluate

$$u(x) = \sum_{y \in P} G(x,y) f(y). \tag{2}$$

Direct evaluation of (2) takes $\Theta(n^2)$ steps. This can be computationally intensive for high frequency problems where ω and n are large. Hence, there is a practical need for algorithms that can evaluate (2) rapidly and accurately.

*Received by the editors September 3, 2014; accepted for publication January 20, 2015; published electronically March 26, 2015. This work was partially supported by the National Science Foundation under award DMS-1328230 and by the U.S. Department of Energy's Advanced Scientific Computing Research program under award DE-FC02-13ER26134/DE-SC0009409.
http://www.siam.org/journals/mms/13-1/98512.html
†Department of Mathematics and Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA (lexing@stanford.edu).

In the past thirty years, a lot of research has been devoted to this problem and several successful algorithms have been proposed. The first and probably most well-known approach is the high frequency fast multipole method initiated by Rokhlin in [20, 21] and further developed in [7, 9, 14, 24, 23]. This method follows the structure of the classical fast multipole method for the Laplace equation and uses fast spherical harmonic transforms and diagonal forms to construct the multipole and local expansions efficiently. This method has a $\Theta(n \log n)$ complexity but is rather technical.

The second method is the butterfly algorithm proposed by Michielssen and Boag in [17], where the main observation is that the interaction between two regions under certain geometric configuration is numerically low rank even for high frequency scattering problems. The complexity of this algorithm is $\Theta(n \log^2 n)$ with a relatively large prefactor.

A third method is the directional FMM-type algorithm proposed by Engquist and Ying in [12, 13], where the concept of a directional parabolic separation condition is introduced to construct directional low-rank approximations between certain square regions and wedges with narrow opening angle. This algorithm has a $\Theta(n \log n)$ complexity and the advantage of being kernel-independent. The main observation of this method is closely related to an earlier work by Rokhlin [22] on sparse diagonal forms of the Helmholtz kernel. Other algorithms that rely on this directional low-rank idea can be found in [2, 16].

While all the above methods are based on hierarchical partitioning of the boundary, methods based on the fast Fourier transform (FFT) [4, 5] are also widely used. These methods are relatively easy to implement, but have superlinear complexity like $\Theta(n^\alpha \log n)$ with $\alpha = 1.5$, for example.

In this paper, we introduce a new algorithm for two-dimensional problems. This algorithm leverages the directional idea of [12, 13] and uses oscillatory Chebyshev interpolation and local FFTs to speed up the computation. The unique feature of this new algorithm is that the low-rank separated approximations are computed *directly* from input data, as opposed to *recursively* via multipole/local expansions in other FMM-type algorithms. Though the local FFTs have also been used in the computation of multipole/local expansions in earlier work [6, 10], they are used here to totally bypass these expansions. This new method has a $\Theta(n \log^2 n)$ complexity and is conceptually much simpler.

The rest of this paper is organized as follows. Section 2 explained the algorithm in detail. Section 3 provides numerical results, and discussions are given in section 4.

2. Algorithm.

2.1. Notation.

In two dimensions, the Green's function of the Helmholtz equation is given by

$$G(x, y) = \frac{i}{4} H_0^1(\omega |x - y|)$$

and $\lambda = 2\pi/\omega$ is the wavelength. To simplify the problem, we assume that the domain Ω has a C^2 boundary and that both its diameter and its boundary length L are of order $\Theta(1)$, and let $\rho : \partial\Omega \to [0, L]$ be the arclength parameterization of the boundary. The extension to piecewise C^2 boundary is quite straightforward and requires little extra effort.

We also assume without loss of generality that the length L of $\partial\Omega$ is equal to $4^q\lambda$, where $q > 0$ is an integer. The actual number 4^q is not important but it makes

the presentation of the algorithm simpler. Under these assumptions, it is clear that $\lambda = \Theta(1/4^q)$ and $\omega = \Theta(4^q)$.

Following the typical practice of using a constant number p of points per wavelength, we discretize the boundary $\partial\Omega$ with a set P of $n = 4^q p$ equally spaced sampling points.

2.2. Existence of low-rank approximation. A function $H(x, y)$ with $x \in X$ and $y \in Y$ is said to have a separated approximation of n terms with relative accuracy ϵ if there exist functions $\{\alpha_i(x)\}_{1\le i\le n}$ and $\{\beta_i(y)\}_{1\le i\le n}$ such that

$$\left| H(x,y) - \sum_{i=1}^{n} \alpha_i(x)\beta_i(y) \right| \le \epsilon |H(x,y)|.$$

The following result is the theoretical foundation of the new algorithm.

THEOREM 2.1. *Consider two rectangles,*

$$R_1 = \left[\frac{d}{2}\lambda, \frac{3d}{2}\lambda\right] \times \left[-\frac{h_1}{2}\lambda, \frac{h_1}{2}\lambda\right],$$
$$R_2 = \left[-\frac{3d}{2}\lambda, -\frac{d}{2}\lambda\right] \times \left[-\frac{h_2}{2}\lambda, \frac{h_2}{2}\lambda\right].$$

Suppose that $d > 1$ and $d \ge 2\max(h_1, h_2)^2$. Then, for each $\epsilon > 0$ sufficiently small, there exists an (ω, d)-independent constant $C(\epsilon)$ such that $G(x, y) = \frac{i}{4}H_0^1(\omega|x - y|)$, with $x \in R_1$ and $y \in R_2$, has a separated approximation of $C(\epsilon)$ terms with relative accuracy ϵ.

The main point of the theorem is that the number of terms $C(\epsilon)$ is independent of ω and d. Therefore, even if ω and d grow, the number of terms remains uniformly bounded. For this reason, such a separated approximation is called low rank.

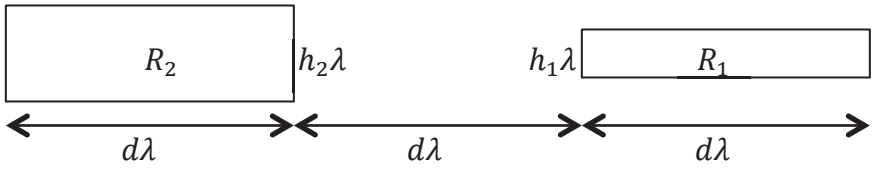

FIG. 1. *The geometric setup of Theorem* 2.1.

Proof. The geometric setting is illustrated in Figure 1. We first observe that $G(x, y)$ can be written as

$$G(x,y) = \exp(i\omega|x-y|)G_0(|x-y|),$$

where $G_0(|x - y|)$ is a nonoscillatory real analytic function (see [1], for example). A low-rank separation approximation for $G_0(|x - y|)$ with relative error ϵ can be constructed via Taylor expansion or Chebyshev interpolation. Hence, the remaining task is to find a low-rank separated approximation for $\exp(i\omega|x - y|)$ with relative accuracy ϵ. For $x = (x_1, x_2) \in R_1$ and $y = (y_1, y_2) \in R_2$, we write

$$\exp(i\omega|x-y|) = \exp(i\omega(x_1-y_1))\exp(i\omega(|x-y| - (x_1-y_1))).$$

As the first term on the right-hand side is obviously separated, we only focus on the second term, which can be rewritten as

$$\exp(i\omega(|x-y|-(x_1-y_1))) = \exp\left(i\omega(x_1-y_1)\left(\sqrt{1+\frac{|x_2-y_2|^2}{|x_1-y_1|^2}}-1\right)\right).$$

We denote the last term by η and decompose its calculation into multiple steps by introducing

$$\xi = \frac{1}{|x_1-y_1|^2}, \quad \alpha = |x_2-y_2|^2\xi, \quad \beta = \sqrt{1+\alpha}-1,$$
$$\gamma = i\omega(x_1-y_1)\beta, \quad \eta = \exp(i\gamma).$$

For a quantity δ, we use the notation $\mathrm{lrsa}(\delta)$ to stand for a separated approximation of δ with a number of terms that is (ω, d)-independent and with a relative error bounded by a constant multiple of ϵ.

For $\xi = 1/|x_1-y_1|^2$, applying a truncated Taylor expansion of x_1 at the center of R_1 with relative error ϵ gives a low-rank separated approximation $\mathrm{lrsa}(\xi)$ for which the number of terms is (ω, d)-independent due to scale invariance:

$$\mathrm{lrsa}(\xi) = \xi(1+\epsilon_\xi), \quad |\epsilon_\xi| \lesssim \epsilon.$$

For $\alpha = |x_2-y_2|^2\xi$, we define $\mathrm{lrsa}(\alpha) = |x_2-y_2|^2\mathrm{lrsa}(\xi)$. Clearly, by letting $\epsilon_\alpha = \epsilon_\xi$, we have

$$\mathrm{lrsa}(\alpha) = \alpha(1+\epsilon_\alpha), \quad |\epsilon_\alpha| \lesssim \epsilon.$$

From the relation $\alpha = |x_2-y_2|^2/|x_1-y_1|^2$ and the geometric setup in Figure 1, it is clear that $\alpha \in (0, 1/4)$. Hence, by choosing ϵ sufficiently small, one can easily guarantee that $\mathrm{lrsa}(\alpha) \in (0, 1/3)$.

Next, we consider $\beta = \sqrt{1+\alpha}-1$. Since both α and $\mathrm{lrsa}(\alpha)$ are between 0 and $1/3$, expanding $\sqrt{1+\alpha}-1$ with truncated Taylor series at $\alpha = 0$ with relative error ϵ gives a polynomial $p(\alpha)$ for which the number of terms is independent of ω and d. We then define $\mathrm{lrsa}(\beta) = p(\mathrm{lrsa}(\alpha))$. Notice that

$$\begin{aligned}
\mathrm{lrsa}(\beta) &= (\sqrt{1+\mathrm{lrsa}(\alpha)}-1)(1+\epsilon)\\
&= (\sqrt{1+\alpha(1+\epsilon_\alpha)}-1)(1+\epsilon)\\
&= (\sqrt{1+\alpha}-1)(1+C\epsilon_\alpha)(1+\epsilon)\\
&= (\sqrt{1+\alpha}-1)(1+C\epsilon_\alpha)(1+\epsilon)\\
&= \beta(1+C\epsilon_\alpha)(1+\epsilon),
\end{aligned}$$

where C is a uniformly bounded constant. By defining $\epsilon_\beta = (1+C\epsilon_\alpha)(1+\epsilon)-1$, we have

$$\mathrm{lrsa}(\beta) = \beta(1+\epsilon_\beta), \quad |\epsilon_\beta| \lesssim \epsilon.$$

For $\gamma = i\omega(x_1-y_1)\beta$, we define $\mathrm{lrsa}(\gamma) = i\omega(x_1-y_1)\mathrm{lrsa}(\beta)$ and $\epsilon_\gamma = \epsilon_\beta$. Clearly,

$$\mathrm{lrsa}(\gamma) = \gamma(1+\epsilon_\gamma), \quad |\epsilon_\gamma| \lesssim \epsilon.$$

An easy but essential calculation also shows that

$$\gamma = i\omega(x_1 - y_1)\left(\sqrt{1 + \frac{|x_2 - y_2|^2}{|x_1 - y_1|^2}} - 1\right)$$

is of order $O(1)$ for $x \in R_1$ and $y \in R_2$. Then it follows that $\mathrm{lrsa}(\gamma)$ is also of order $O(1)$.

Finally, consider $\eta = \exp(i\gamma)$. As both γ and $\mathrm{lrsa}(\gamma)$ are of order $O(1)$, applying truncated Taylor expansion to $\exp(i\gamma)$ near $\gamma = 0$ with relative error ϵ gives a polynomial $q(\gamma)$ with an (ω, d)-independent number of terms. We then define $\mathrm{lrsa}(\eta) = q(\mathrm{lrsa}(\gamma))$ and have

$$\begin{aligned}\mathrm{lrsa}(\eta) &= q(\mathrm{lrsa}(\gamma)) = \exp(i\mathrm{lrsa}(\gamma))(1+\epsilon)\\ &= \exp(i\gamma)\exp(i(\mathrm{lrsa}(\gamma) - \gamma))(1+\epsilon)\\ &= \exp(i\gamma)\exp(i\gamma\epsilon_\gamma)(1+\epsilon)\\ &= \exp(i\gamma)(1 + C\gamma\epsilon_\gamma)(1+\epsilon),\end{aligned}$$

where C is uniformly bounded from the fact that $\gamma = O(1)$ and $\mathrm{lrsa}(\gamma) = O(1)$. By defining $\epsilon_\eta = (1 + C\gamma\epsilon_\gamma)(1+\epsilon) - 1$, one has

$$\mathrm{lrsa}(\eta) = \eta(1 + \epsilon_\eta), \quad |\epsilon_\eta| \lesssim \epsilon.$$

This shows that $\exp(i\omega|x-y|)$ has a low-rank separated approximation with an (ω, d)-independent number of terms and relative error $O(\epsilon)$. Combined with the approximation for $G_0(|x-y|)$, we conclude that $G(x, y)$ has a low-rank separated approximation with an (ω, d)-independent number of terms and relative error $O(\epsilon)$.

At this point, it is clear that in order to get a relative error strictly bounded by ϵ, we only need to repeat the above argument by pre-dividing ϵ by a uniform constant factor. ∎

The above proof does not give the explicit dependence of the number of terms $C(\epsilon)$ on the relative accuracy ϵ. Numerical results show that in fact the number of terms is bounded by a universal constant times $O(\log(1/\epsilon))$.

2.3. Setup algorithm. We recall that the length of $\partial\Omega$ is $4^q\lambda$. The setup algorithm first generates a complete binary tree structure of $\partial\Omega$ by bisecting it into segments of equal length recursively. The process is stopped when a segment is of length $m_\ell\lambda$, where m_ℓ is a constant typically set to 2 or 4. A segment at the final level is called a *leaf* and it contains $m_\ell p$ discretization points. A segment T is called *almost planar* if its length is bounded by $2^q\lambda/\sqrt{\kappa_T}$, where κ_T is the maximum of the absolute value of the curvature within the segment T.

The key component of the setup algorithm is how to compress the interaction between two segments. Consider two segments T and S and

- let c_T and c_S be the centers of T and S;
- let t_T and t_S be the tangent directions at c_T and c_S; and
- let $a_{TS} = (c_T - c_S)/|c_T - c_S|$ be the unit vector pointing from c_S to c_T (see Figure 2).

The pair (T, S) is called *parabolically separated* if T and S can be enclosed in two rectangles R_1 and R_2 that satisfy the condition of Theorem 2.1. More precisely, we define

- ℓ_{TS} to be the line passing through the centers c_T and c_S;

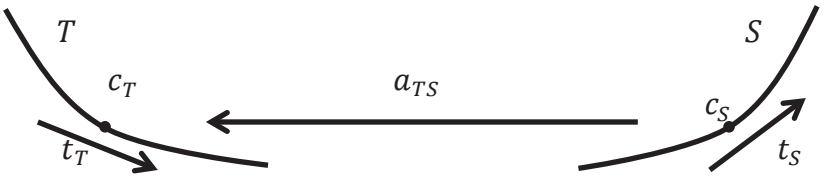

FIG. 2. *The geometric setting for constructing low-rank approximation between two segments T and S.*

- $w_T\lambda$ and $w_S\lambda$ to be the lengths of the projections of T and S onto ℓ_{TS};
- $d_{TS}\lambda$ to be the distance between the projections; and
- $h_T\lambda$ and $h_S\lambda$ to be the lengths of the projections of T and S, respectively, onto the direction $a_{TS}^{\perp}$.

With these definitions, (T, S) is parabolically separated if $d_{TS} > 1$, $d_{TS} > \max(w_T, w_S)$, and $d_{TS} > 2\max(h_T, h_S)^2$. Theorem 2.1 states that for a parabolically separated pair (T, S), the interaction between T and S is numerically low rank. As we shall see, the setup algorithm constructs a low-rank approximation for such a pair when at least one of T and S is also almost planar.

Let $\mathcal{G}$ be an empty set initially. The algorithm first traverses the tree in a breadth-first search. Whenever it reaches an almost-planar segment, the algorithm inserts it into $\mathcal{G}$ and skips the whole subtree starting from it. When it reaches a leaf, the algorithm also inserts it into $\mathcal{G}$. At the end of this step, $\mathcal{G}$ consists of almost-planar segments and nonplanar leaves, and they form a disjoint union of the boundary $\partial\Omega$.

Next, the algorithm starts with a queue $\mathcal{Q}$ of segment pairs, initialized to be $\{(T, S), T, S \in \mathcal{G}\}$, as well as a set $\mathcal{F}$, initialized to be empty. When the queue $\mathcal{Q}$ is not empty, the top element is popped out and denoted by (T, S). The algorithm treats the pair (T, S) depending on the types of T and S.

- If both T and S are nonplanar leaves, we mark (T, S) as a *dense pair* and insert it into $\mathcal{F}$.
- Suppose that T is a nonplanar leaf but S is almost planar. If (T, S) is parabolically separated, then we mark (T, S) as a *low-rank pair* and insert it into $\mathcal{F}$. If it is not, one of the following two options is taken.
  - If S is not at the leaf level, we partition S evenly into two segments S_1 and S_2 and insert (T, S_1) and (T, S_2) into $\mathcal{Q}$.
  - If S is at the leaf level, we mark (T, S) as a *dense pair* and insert it into $\mathcal{F}$.
- Suppose that T is almost planar but S is a nonplanar leaf. If (T, S) is parabolically separated, then we mark (T, S) as a *low-rank pair* and insert it into $\mathcal{F}$. If it is not, one of the following two options is taken.
  - If T is not at the leaf level, we partition T evenly into two segments T_1 and T_2 and insert (T_1, S) and (T_2, S) into $\mathcal{Q}$.
  - If T is at the leaf level, we mark (T, S) as a *dense pair* and insert it into $\mathcal{F}$.
- Suppose that both T and S are almost planar. If (T, S) is parabolically separated, then we mark (T, S) as a *low-rank pair* and insert it into $\mathcal{F}$. If it is not, one of the following options is taken.
  - If T is longer than S, partition T evenly into two segments T_1 and T_2 and insert (T_1, S) and (T_2, S) into $\mathcal{Q}$.

- If S is longer than T, partition S evenly into two segments S_1 and S_2 and insert (T, S_1) and (T, S_2) into $\mathcal{Q}$.
- If T and S are of the same length but are not at the leaf level, then partition T evenly into T_1 and T_2 and S evenly into S_1 and S_2 and insert (T_1, S_1), (T_2, S_1), (T_1, S_2), and (T_2, S_2) into $\mathcal{Q}$.
- Finally if T and S are of the same length and at the leaf level, then mark (T, S) as a *dense pair* and insert it into $\mathcal{F}$.

This process is repeated as long as the queue $\mathcal{Q}$ is not empty. When Q is empty, the set $\mathcal{F}$ provides a hierarchical compressed approximation of (2).

2.3.1. Construction of directional low-rank approximation. What is missing at this point is the procedure for constructing an actual low-rank separated approximation. In the setup algorithm, this procedure is used when (T, S) is parabolically separated and at least one of T and S is almost planar.

Let us first consider a low-rank pair $(T, S) \in \mathcal{F}$ where both T and S are almost planar. It is instructive to consider the function $\exp(i\omega|x-y|)$ instead of $G(x, y)$ as this is where the oscillations come from. In what follows, we shall use $\sim$ to denote an approximation up to a nonoscillatory multiplicative term.

Let $2^{\ell_T}\lambda$ and $2^{\ell_S}\lambda$ be the lengths of T and S, respectively. For a point $x \in T$ and $y \in S$,

$$\begin{aligned}
&\exp(i\omega|x-y|) \sim \exp(i\omega a_{TS}\cdot(x-y))\\
&=\exp(i\omega a_{TS}\cdot((x-c_T)+(c_T-c_S)+(c_S-y)))\\
&=\exp(i\omega a_{TS}\cdot(x-c_T))\cdot\exp(i\omega a_{TS}\cdot(c_T-c_S))\cdot\exp(-i\omega a_{TS}\cdot(y-c_S)),
\end{aligned} \tag{3}$$

where the approximation in the first line is based on the fact that (T, S) is parabolically separated.

Let us recall that $\rho : \partial\Omega \to [0, L]$ is the arclength parameterization of the boundary (see section 2.1). To further approximate the first term in (3), we perform a Taylor expansion for $\rho^{-1}(t)$ near $t = \rho(c_T)$ and evaluate it at $\rho(x)$:

$$\begin{aligned}
&\left|\rho^{-1}(\rho(x)) - \left(\rho^{-1}(\rho(c_T)) + t_T(\rho(x)-\rho(c_T))\right)\right|\\
&\lesssim \frac{1}{2}|\rho(x)-\rho(c_T)|^2\kappa_T \le \frac{1}{2}(2^q\lambda/\sqrt{\kappa_T})^2\kappa_T = O(4^q\lambda^2) = O(\lambda),
\end{aligned}$$

where κ_T is the maximum of the absolute value of the curvature in T, and here we also use the fact that T is almost planar. This is equivalent to

$$(x-c_T) = (\rho(x)-\rho(c_T))\cdot t_T + O(\lambda).$$

Multiplying it with $i\omega a_{TS}$ and taking exponential gives the approximation

$$\exp(i\omega a_{TS}\cdot(x-c_T)) \sim \exp(i\omega a_{TS}\cdot t_T(\rho(x)-\rho(c_T))). \tag{4}$$

For the last term of (3), the same argument works for $(y-c_S)$. Since S is almost planar, we get

$$(y-c_S) = (\rho(y)-\rho(c_S))\cdot t_S + O(\lambda)$$

and

$$\exp(-i\omega a_{TS}\cdot(y-c_S)) \sim \exp(-i\omega a_{TS}\cdot t_S(\rho(y)-\rho(c_S))). \tag{5}$$

Noticing that $\omega a_{TS} \cdot t_T \in [-\omega, \omega]$, we partition the interval $[-\omega, \omega]$ uniformly into $2^{\ell_T+1} m_f$ intervals with a set K_T of $2^{\ell_T+1} m_f + 1$ gridpoints. Here m_f is a parameter that is typically set to 2 or 4. A key step of the algorithm is to approximate $\omega a_{TS} \cdot t_T$ with a nearby gridpoint. More precisely, we define $[k]_T$ to be the value of rounding k to the nearest gridpoint in K_T. Since

$$|(\omega a_{TS} \cdot t_T - [\omega a_{TS} \cdot t_T]_T)(\rho(x) - \rho(c_T))| \le \frac{2\omega}{2^{\ell_T+1} m_f} \cdot \frac{1}{2} \cdot \frac{2^{\ell_T} \lambda}{2} = \frac{2\pi}{4 m_f} = O(1),$$

replacing $\omega a_{TS} \cdot t_T$ with $[\omega a_{TS} \cdot t_T]_T$ in (4) only introduces a nonoscillatory multiplicative term. Therefore,

$$\exp(i\omega a_{TS} \cdot (x - c_T)) \sim \exp(i[\omega a_{TS} \cdot t_T]_T(\rho(x) - \rho(c_T))). \tag{6}$$

Similarly for $-\omega a_{TS} \cdot t_S$, we partition the interval $[-\omega, \omega]$ uniformly into $2^{\ell_S+1} m_f$ intervals with a set K_S of $2^{\ell_S+1} m_f + 1$ gridpoints and define $[k]_S$ to be the value of rounding k to the nearest gridpoint in K_S. Again since

$$|(-\omega a_{TS} \cdot t_S - [-\omega a_{TS} \cdot t_S]_S)(\rho(y) - \rho(c_S))| \le \frac{2\omega}{2^{\ell_S+1} m_f} \cdot \frac{1}{2} \cdot \frac{2^{\ell_S} \lambda}{2} = \frac{2\pi}{4 m_f} = O(1),$$

replacing $-\omega a_{TS} \cdot t_S$ with $[-\omega a_{TS} \cdot t_S]_S$ in (5) introduces an extra nonoscillatory multiplicative term. Thus

$$\exp(-i\omega a_{TS} \cdot (y - c_S)) \sim \exp(i[-\omega a_{TS} \cdot t_S]_S(\rho(y) - \rho(c_S))). \tag{7}$$

The reason for introducing the rounding is that

- $\exp(ik(\rho(x) - \rho(c_T))$ for $x \in T$ and $k \in K_T$ is a partial Fourier matrix since both $\rho(x)$ and k are on a uniform grid.

The same also holds for $\exp(ik(\rho(y) - \rho(c_S))$ for $y \in S$ and $k \in K_S$. As we shall see, this has an enormous impact on the efficiency of applying the low-rank approximations.

By introducing

$$\begin{aligned}
k_{TS}^T &= [\omega a_{TS} t_T]_T, \\
k_{TS}^S &= [-\omega a_{TS} t_S]_S, \\
F_T(x,k) &= \exp(ik(\rho(x) - \rho(c_T))), \quad x \in T, k \in K_T, \\
F_S(y,k) &= \exp(ik(\rho(y) - \rho(c_S))), \quad y \in S, k \in K_S,
\end{aligned}$$

and applying (6) and (7) to (3), we have

$$\exp(i\omega|x - y|) \sim F_T(x, k_{TS}^T) \cdot \exp(i\omega(c_T - c_S) \cdot a_{TS}) \cdot F_S(y, k_{TS}^S).$$

So far, we have been considering the kernel $\exp(i\omega|x - y|)$. Let us now replace $\exp(i\omega|x - y|)$ back with the kernel $G(x, y)$. Since the difference $G_0(|x - y|)$ is a nonoscillatory term, the above discussion applies without any change and we have the following representation for $G(x, y)$:

$$G(x,y) = F_T(x, k_{TS}^T) \cdot G_{TS}^{mm}(x,y) \cdot F_S(y, k_{TS}^S), \quad x \in T, y \in S, \tag{8}$$

where the term $G_{TS}^{mm}(x, y)$ defined via this equation is nonoscillatory in $x \in T$ and $y \in S$. The superscript mm of G_{TS}^{mm} indicates that the complex exponential modulation is done both for T and for S. In a more compact operator form, we can write (8) as

$$G(T,S) = \mathrm{diag}(F_T(:, k_{TS}^T)) \cdot G_{TS}^{mm}(T,S) \cdot \mathrm{diag}(F_S(:, k_{TS}^S)). \tag{9}$$

Since $G_{TS}^{mm}(x, y)$ is nonoscillatory, it can be approximated with Chebyshev interpolation in the parametric domain of the boundary. More specifically, we define

- m_c to be the size of the Chebyshev grid used;
- R_T to be the image (under ρ^{-1}) of the Chebyshev grid in T, and R_S to be the image of the Chebyshev grid in S;
- I_T and I_S to be the interpolation operator for T and S associated with the Chebyshev grids R_T and R_S, respectively.

In the matrix notation, I_T is a matrix with entries given by $I_T(x, j)$ for $x \in T$ and $j \in R_T$, and I_S is a matrix with entries given by $I_S(y, j)$ for $y \in S$ and $j \in R_S$. In an operator form, this approximation reads

$$G_{TS}^{mm}(T, S) \approx I_T \cdot G_{TS}^{mm}(R_T, R_S) \cdot I_S^t.$$

Putting it together with (9) gives us

$$G(T, S) \approx \operatorname{diag}(F_T(:, k_{TS}^T)) \cdot I_T \cdot G_{TS}^{mm}(R_T, R_S) \cdot I_S^t \cdot \operatorname{diag}(F_S(:, k_{TS}^S)).$$

Notice that for a pair (T, S), representing this low-rank approximation only requires storing k_{TS}^T, k_{TS}^S, and $G_{TS}^{mm}(R_T, R_S)$.

The above discussion is for the case in which both T and S are almost planar. However, the setup algorithm also needs to construct low-rank approximations for low-rank pairs in which only one of T and S is almost planar. For those cases, the oscillatory Chebyshev interpolation is only used for the almost-planar segment while a dense evaluation is used at the other nonplanar segment. More precisely, if S is almost planar, we form the representation

$$G(T, S) = G_{TS}^{\cdot m}(T, S) \cdot \operatorname{diag}(F_S(:, k_{TS}^S))$$

and the approximation

$$G(T, S) \approx G_{TS}^{\cdot m}(T, R_S) \cdot I_S^t \cdot \operatorname{diag}(F_S(:, k_{TS}^S)),$$

where the superscript $\cdot m$ means that the exponential modulation is done only for S. Similarly if T is almost planar, we construct the representation

$$G(T, S) = \operatorname{diag}(F_T(:, k_{TS}^T)) \cdot G_{TS}^{m\cdot}(T, S)$$

and the approximation

$$G(T, S) \approx \operatorname{diag}(F_T(:, k_{TS}^T)) \cdot I_T \cdot G_{TS}^{m\cdot}(R_T, S),$$

where the superscript $m\cdot$ means that the exponential modulation is done only for T.

2.3.2. Complexity of the setup algorithm. Since Ω has a C^2 boundary, the curvature is uniformly bounded. Let us first estimate the number of segments in $\mathcal{G}$. From the setup algorithm, it is clear that the length of a segment T in $\mathcal{G}$ is bounded by $\min(2^q\lambda, 2^q\lambda/\sqrt{\kappa_T}) = \Theta(2^q\lambda) = \Theta(\sqrt{\omega}\lambda)$. As the total length of the boundary is $\Theta(1)$, the number of segments in $\mathcal{G}$ is bounded by $\Theta(1/(\sqrt{\omega}\lambda)) = \Theta(\sqrt{\omega})$.

Next, let us consider the number of pairs in $\mathcal{F}$. Since the boundary is C^2, for almost all pairs (T, S) in $\mathcal{F}$ the segments T and S have the same length. Consider first the segments of length $2^q\lambda$. As there are 2^q segments of this length, the total number of pairs is bounded by $2^{2q} = \Theta(\omega)$. For length equal to smaller values, a segment T only appears together with a constant number of segments S in its neighborhood. Therefore, the total number of pairs in $\mathcal{F}$ is again $\Theta(\omega)$.

The complexity of the setup algorithm contains three parts:

- The generation of the set $\mathcal{F}$. It takes $O(\omega \log \omega)$ steps.
- The evaluation of the dense pairs. Clearly there are at most $\Theta(\omega)$ dense pairs. Since each leaf segment contains at most $O(1)$ points, the dense matrix $G(T,S)$ has $O(1)$ entries. Therefore, the evaluation cost of all dense pairs is $O(\omega)$.
- The evaluation of the low-rank pairs. Again, the number of low-rank pairs is at most $\Theta(\omega)$. In all cases of the low-rank pairs, the matrices $G^{mm}_{T,S}$, $G^{\cdot m}_{T,S}$, and $G^{m\cdot}_{T,S}$ have $O(1)$ entries. Therefore, the evaluation cost of all low-rank pairs is also $O(\omega)$.

Summing these contributions together gives a $\Theta(\omega \log \omega) = \Theta(n \log n)$ complexity for the setup algorithm.

2.4. Evaluation algorithm. Now we are ready to describe how to evaluate the sums

$$u(x) = \sum_{y \in P} G(x,y) f(y), \quad x \in P,$$

efficiently. In the following discussion, we slightly abuse notation by using u to denote the vector $(u(x))_{x \in P}$ and f to denote the vector $(f(y))_{y \in P}$. For a segment T, $u(T)$ stands for the subvector obtained by restricting u to the points in T. Similarly, $f(S)$ is the restriction of f to the points in S.

Initially, we set $u = 0$. The algorithm visits all pairs $(T,S) \in \mathcal{F}$. If (T,S) is dense, then

$$u(T) \Leftarrow u(T) + G(T,S) f(S),$$

where $\Leftarrow$ stands for assignment.

If (T,S) is low rank and both T and S are almost planar,

$$u(T) \Leftarrow u(T) + \operatorname{diag}(F_T(:, k^T_{TS})) \cdot I_T \cdot G^{mm}_{TS}(R_T, R_S) \cdot I^t_S \cdot \operatorname{diag}(F_S(:, k^S_{TS})) \cdot f(S). \tag{10}$$

If (T,S) is low rank and only S is almost planar,

$$u(T) \Leftarrow u(T) + G^{\cdot m}_{TS}(T, R_S) \cdot I^t_S \cdot \operatorname{diag}(F_S(:, k^S_{TS})) \cdot f(S). \tag{11}$$

If (T,S) is low rank and only T is almost planar,

$$u(T) \Leftarrow u(T) + \operatorname{diag}(F_T(:, k^T_{TS})) \cdot I_T \cdot G^{m\cdot}_{TS}(R_T, S) \cdot f(S). \tag{12}$$

Direct evaluation of these three formulas results in a $\Theta(\omega^{3/2})$ complexity. In order speed up this calculation, the work is split into multiple steps. Let us consider (10), i.e., the situation where both T and S are almost planar.

First, at segment S, one needs to compute

$$I^t_S \cdot \operatorname{diag}(F_S(:, k^S_{TS})) \cdot f(S).$$

It turns out to be extremely useful to consider all $k \in K_S$ altogether instead of one by one:

$$I^t_S \cdot \operatorname{diag}(F_S(:,k)) \cdot f(S).$$

This is equivalent to evaluating for each $j \in R_S$

$$\langle I_S(:,j), F_S(:,k) \odot f(S)\rangle = \langle F_S(:,k), I_S(:,j) \odot f(S)\rangle := \hat{f}_S(j,k), \tag{13}$$

where the symbol $\odot$ stands for the entrywise product of two vectors and $\langle \cdot, \cdot \rangle$ is the bilinear product, i.e., $\langle v, w\rangle = \sum_i v_i w_i$. An essential observation from the last equation is the following:

- Computing $\hat{f}_S(j,:)$ (i.e., all $k \in K_S$ with a fixed $j \in R_S$) simply requires entrywise product of $f(S)$ with $I_S(:,k)$, followed by an FFT.

Looping through each $j \in R_S$ provides all entries in the $|R_S| \times |K_S|$ matrix $\hat{f}_S(:,:)$.

Second, instead of considering one pair (T,S) at a time, we fix T and consider all pairs for which the first component is T and both T and S are almost planar. More precisely,

$$\begin{aligned} u(T) &\Leftarrow u(T) + \sum_S \operatorname{diag}(F_T(:,k^T_{TS})) \cdot I_T \cdot G^{mm}_{TS}(R_T,R_S) \cdot \hat{f}_S(:,k^S_{TS}) \\ &\Leftarrow u(T) + \sum_{k\in K_T} \sum_{S:k^T_{TS}=k} \operatorname{diag}(F_T(:,k)) \cdot I_T \cdot G^{mm}_{TS}(R_T,R_S) \cdot \hat{f}_S(:,k^S_{TS}) \\ &\Leftarrow u(T) + \sum_{k\in K_T} \operatorname{diag}(F_T(:,k)) \cdot I_T \cdot \left(\sum_{S:k^T_{TS}=k} G^{mm}_{TS}(R_T,R_S) \cdot \hat{f}_S(:,k^S_{TS}) \right). \end{aligned}$$

Motivated by the last equation, we introduce an $|R_T| \times |K_T|$ matrix $\hat{u}_T(:,:)$ defined via

$$\hat{u}_T(:,k) = \sum_{S:k^T_{TS}=k} G^{mm}_{TS}(R_T,R_S)\hat{f}_S(:,k^S_{TS}).$$

Third, we can now write

$$u(T) \Leftarrow u(T) + \sum_{k\in K_T} \operatorname{diag}(F_T(:,k)) \cdot I_T \cdot \hat{u}_T(:,k).$$

To perform this computation efficiently, notice that the last term is

$$\sum_{k\in K_T} \sum_{j\in R_T} \operatorname{diag}(F_T(:,k)) I_T(:,j) \cdot \hat{u}_T(j,k) = \sum_{j\in R_T} I_T(:,j) \odot \left(\sum_{k\in K_T} F_T(:,k)\hat{u}_T(j,k) \right). \tag{14}$$

An essential observation is that

- for a fixed $j \in R_T$, the sum over $k \in K_T$ can be carried out by an FFT.

Therefore, for each $j \in R_T$, the work is reduced to an FFT followed by an entrywise product.

The above discussion addresses the case where both T and S are almost planar. For the cases where only one of them is almost planar, i.e., (11) and (12), the procedure is similar except that the oscillatory Chebyshev interpolation and local FFTs are done only on one side.

To summarize, the evaluation algorithm takes the following steps:

1. Set $u = 0$ and $\hat{u}_T(:,:) = 0$.
2. For each segment S and for each $j \in R_S$, form $\hat{f}_S(j,:)$ by an entrywise product followed by an FFT, as shown in (13).

3. For each pair (T, S) in $\mathcal{F}$,
   (a) if (T, S) is dense,
   $$u(T) \Leftarrow u(T) + G(T, S) f(S);$$
   (b) if (T, S) is low rank and both are almost planar,
   $$\hat{u}_T(:, k_{TS}^T) \Leftarrow \hat{u}_T(:, k_{TS}^T) + G_{TS}^{mm}(R_T, R_S) \hat{f}_S(:, k_{TS}^S);$$
   (c) if (T, S) is low rank and only S is almost planar,
   $$u(T) \Leftarrow u(T) + G_{TS}^{\cdot m}(T, R_S) \hat{f}_S(:, k_{TS}^S);$$
   (d) if (T, S) is low rank and only T is almost planar,
   $$\hat{u}_T(:, k_{TS}^T) \Leftarrow \hat{u}_T(:, k_{TS}^T) + G_{TS}^{m\cdot}(R_T, S) f(S).$$
4. For each segment T and for each $j \in R_T$, apply an FFT and then an entrywise product to $\hat{u}_T(j, :)$ and add the result to $u(T)$, as shown in (14).

2.4.1. Complexity of the evaluation algorithm. The complexity of the four steps of the evaluation algorithm is estimated as follows.

1. The cost of the first step is clearly $O(\omega)$.
2. The FFT for a segment of length $t\lambda$ and tp points take $\Theta(t \log t)$ steps as p is constant. Since there are $O(4^q/t)$ segments of this length, the overall cost is $O(4^q \log t) = O(\omega \log t)$. Now summing over all possible values of t results in a complexity of $O(\omega \log^2 \omega)$ for the second step of the algorithm.
3. Now consider the third step. The matrices $G(T, S)$, $G_{TS}^{mm}(R_T, R_S)$, $G_{TS}^{\cdot m}(T, R_S)$, and $G_{TS}^{m\cdot}(R_T, S)$ that appear in all four scenarios all have size $O(1)$. Taking into consideration that there are at most $O(\omega)$ pairs in $\mathcal{F}$, the cost of this step is $O(\omega)$.
4. The last step is very similar to the second one. Its cost is $O(\omega \log^2 \omega)$ as well.

Putting these together shows that the complexity of the evaluation algorithm is $O(\omega \log^2 \omega) = O(n \log^2 n)$.

3. Numerical results. The proposed algorithms are implemented in MATLAB and the numerical results in this section are obtained on a desktop computer with a 3.60GHz CPU. The numerical experiments are performed for two domains: (a) an ellipse and (b) a bean-shaped object (shown in Figure 3).

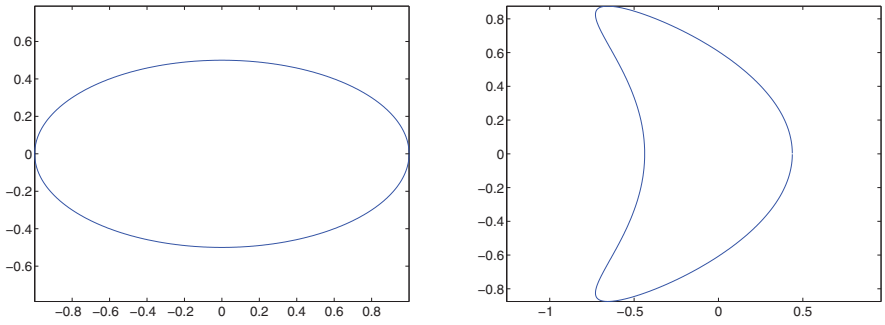

FIG. 3. *The two scatterers used in the numerical tests:* (a) *an ellipse;* (b) *a bean-shaped object.*

The constants in the algorithm are set as follows:

- The number of points p per wavelength is 8.
- The length of the leaf segment is $m_\ell \lambda$ with $m_\ell = 4$.
- The constant m_f in defining the frequency grid is equal to 2.
- The size m_c of the Chebyshev grid is equal to 6, 8, 10, or 12. This number controls the accuracy of the evaluation algorithm.

We denote the exact values and the numerical approximations by $u_e(x)$ and $u_a(x)$, respectively. To check the error, the exact solution is evaluated at a set S of 100 points and the relative error is estimated by

$$\left(\frac{\sum_{x \in S} |u_e(x) - u_a(x)|^2}{\sum_{x \in S} |u_e(x)|^2} \right)^{1/2}. \tag{15}$$

The experiments are performed for the four operators

$$u(x) = (Sf)(x) := \int_{\partial\Omega} G(x,y) f(y) dy,$$
$$u(x) = (Df)(x) := \int_{\partial\Omega} \frac{\partial G(x,y)}{\partial n(y)} f(y) dy,$$
$$u(x) = (D'f)(x) := \int_{\partial\Omega} \frac{\partial G(x,y)}{\partial n(x)} f(y) dy,$$
$$u(x) = (Nf)(x) := \int_{\partial\Omega} \frac{\partial^2 G(x,y)}{\partial n(x) \partial n(y)} f(y) dy,$$

which are typically referred as single layer (S), double layer (D), normal derivative of single layer (D'), and normal derivative of double layer (N) [8, 18]. These operators are the basic building blocks of the boundary integral methods for solving the Dirichlet and Neumann problems of acoustic wave scattering.

The numerical results for the single-layer operator S on the two domains are given in Tables 1 and 2. In these tables,

- m_c is the size of the Chebyshev grid that controls the accuracy of the method;
- ω is the frequency of the wave field;
- n is the number of boundary discretization points;
- T_s is the running time for the setup algorithm;
- T_a is the running times of the application algorithm; and
- e is the relative error estimated via (15).

The numerical results for the double-layer operator D on the two domains are given in Tables 3 and 4. The numerical results for the normal derivative of the single-layer operator D' on the two domains are given in Tables 5 and 6. Finally, the numerical results for the normal derivative of the double-layer operator N on the two domains are given in Tables 7 and 8.

The numerical results show that both the setup and the evaluation algorithms scale linearly with respect to ω and n. The log-square factor $\log^2 \omega$ of the application algorithm is not significant. The running times reported here are obtained with our current MATLAB implementation. We expect a careful C/C++ or Fortran implementation to exhibit much lower absolute running times. The relative error is clearly controlled by the size of the Chebyshev grid m_c. Even for relative small Chebyshev grids, the algorithms achieve good accuracy. An attractive feature of the new algorithms is that once the kernel function is provided, the algorithms are fully kernel-independent.

TABLE 1
Numerical results of operator S for the ellipse.

m_c	ω	n	T_s	T_a	e
6	5.3e+03	3.3e+04	1.9e+01	1.3e+00	2.4e-04
6	2.1e+04	1.3e+05	8.0e+01	5.5e+00	3.5e-04
6	8.5e+04	5.2e+05	3.3e+02	2.2e+01	2.9e-04
8	5.3e+03	3.3e+04	2.0e+01	1.9e+00	1.0e-05
8	2.1e+04	1.3e+05	8.2e+01	5.6e+00	8.8e-06
8	8.5e+04	5.2e+05	3.3e+02	2.4e+01	1.3e-05
10	5.3e+03	3.3e+04	2.1e+01	2.0e+00	9.9e-07
10	2.1e+04	1.3e+05	8.4e+01	6.1e+00	7.8e-07
10	8.5e+04	5.2e+05	3.4e+02	2.6e+01	5.3e-07
12	5.3e+03	3.3e+04	2.2e+01	2.2e+00	2.0e-08
12	2.1e+04	1.3e+05	8.8e+01	6.6e+00	2.2e-08
12	8.5e+04	5.2e+05	3.6e+02	2.9e+01	2.7e-08

TABLE 2
Numerical results of operator S for the bean-shaped object.

m_c	ω	n	T_s	T_a	e
6	5.2e+03	3.3e+04	2.6e+01	2.0e+00	7.6e-04
6	2.1e+04	1.3e+05	1.0e+02	6.6e+00	9.1e-04
6	8.3e+04	5.2e+05	4.2e+02	3.0e+01	9.4e-04
8	5.2e+03	3.3e+04	2.7e+01	2.4e+00	5.6e-05
8	2.1e+04	1.3e+05	1.1e+02	7.0e+00	5.4e-05
8	8.3e+04	5.2e+05	4.3e+02	3.0e+01	7.5e-05
10	5.2e+03	3.3e+04	2.8e+01	2.5e+00	2.2e-06
10	2.1e+04	1.3e+05	1.1e+02	7.4e+00	2.3e-06
10	8.3e+04	5.2e+05	4.4e+02	3.2e+01	2.7e-06
12	5.2e+03	3.3e+04	2.9e+01	2.7e+00	1.9e-07
12	2.1e+04	1.3e+05	1.2e+02	7.9e+00	3.1e-07
12	8.3e+04	5.2e+05	4.7e+02	3.4e+01	3.5e-07

TABLE 3
Numerical results of operator D for the ellipse.

m_c	ω	n	T_s	T_a	e
6	5.3e+03	3.3e+04	2.2e+01	1.4e+00	3.3e-04
6	2.1e+04	1.3e+05	8.6e+01	5.3e+00	5.1e-04
6	8.5e+04	5.2e+05	3.5e+02	2.3e+01	4.7e-04
8	5.3e+03	3.3e+04	2.2e+01	1.9e+00	2.1e-05
8	2.1e+04	1.3e+05	8.9e+01	5.8e+00	1.9e-05
8	8.5e+04	5.2e+05	3.6e+02	2.4e+01	2.6e-05
10	5.3e+03	3.3e+04	2.2e+01	2.0e+00	6.8e-07
10	2.1e+04	1.3e+05	9.1e+01	6.2e+00	4.8e-07
10	8.5e+04	5.2e+05	3.7e+02	2.6e+01	8.3e-07
12	5.3e+03	3.3e+04	2.3e+01	2.2e+00	2.3e-08
12	2.1e+04	1.3e+05	9.5e+01	6.7e+00	2.7e-08
12	8.5e+04	5.2e+05	3.9e+02	2.9e+01	2.6e-08

TABLE 4
Numerical results of operator D for the bean-shaped object.

m_c	ω	n	T_s	T_a	e
6	5.2e+03	3.3e+04	2.8e+01	2.5e+00	1.2e-03
6	2.1e+04	1.3e+05	1.1e+02	6.6e+00	2.0e-03
6	8.3e+04	5.2e+05	4.5e+02	3.0e+01	1.0e-03
8	5.2e+03	3.3e+04	2.9e+01	2.3e+00	7.8e-05
8	2.1e+04	1.3e+05	1.1e+02	7.0e+00	6.3e-05
8	8.3e+04	5.2e+05	4.6e+02	3.0e+01	7.3e-05
10	5.2e+03	3.3e+04	3.0e+01	2.5e+00	1.6e-06
10	2.1e+04	1.3e+05	1.2e+02	7.4e+00	4.1e-06
10	8.3e+04	5.2e+05	4.7e+02	3.2e+01	4.0e-06
12	5.2e+03	3.3e+04	3.1e+01	2.7e+00	3.2e-07
12	2.1e+04	1.3e+05	1.2e+02	8.0e+00	4.5e-07
12	8.3e+04	5.2e+05	5.0e+02	3.4e+01	5.4e-07

TABLE 5
Numerical results of operator D' for the ellipse.

m_c	ω	n	T_s	T_a	e
6	5.3e+03	3.3e+04	2.1e+01	2.3e+00	3.1e-04
6	2.1e+04	1.3e+05	8.6e+01	5.4e+00	3.2e-04
6	8.5e+04	5.2e+05	3.5e+02	2.3e+01	4.3e-04
8	5.3e+03	3.3e+04	2.1e+01	1.9e+00	1.5e-05
8	2.1e+04	1.3e+05	8.8e+01	5.8e+00	1.7e-05
8	8.5e+04	5.2e+05	3.5e+02	2.4e+01	1.0e-05
10	5.3e+03	3.3e+04	2.2e+01	2.0e+00	3.8e-07
10	2.1e+04	1.3e+05	9.1e+01	6.2e+00	4.1e-07
10	8.5e+04	5.2e+05	3.7e+02	2.6e+01	4.1e-07
12	5.3e+03	3.3e+04	2.3e+01	2.2e+00	1.6e-08
12	2.1e+04	1.3e+05	9.6e+01	6.7e+00	3.3e-08
12	8.5e+04	5.2e+05	3.9e+02	2.9e+01	1.8e-08

TABLE 6
Numerical results of operator D' for the bean-shaped object.

m_c	ω	n	T_s	T_a	e
6	5.2e+03	3.3e+04	2.8e+01	2.5e+00	1.6e-03
6	2.1e+04	1.3e+05	1.1e+02	6.6e+00	1.5e-03
6	8.3e+04	5.2e+05	4.5e+02	3.1e+01	7.6e-04
8	5.2e+03	3.3e+04	2.9e+01	2.3e+00	9.3e-05
8	2.1e+04	1.3e+05	1.1e+02	7.0e+00	1.3e-04
8	8.3e+04	5.2e+05	4.6e+02	3.0e+01	1.0e-04
10	5.2e+03	3.3e+04	3.0e+01	2.5e+00	6.3e-06
10	2.1e+04	1.3e+05	1.2e+02	7.4e+00	5.0e-06
10	8.3e+04	5.2e+05	4.7e+02	3.2e+01	3.6e-06
12	5.2e+03	3.3e+04	3.1e+01	2.7e+00	2.5e-07
12	2.1e+04	1.3e+05	1.2e+02	7.9e+00	3.7e-07
12	8.3e+04	5.2e+05	5.0e+02	3.5e+01	3.8e-07

TABLE 7
Numerical results of operator N for the ellipse.

m_c	ω	n	T_s	T_a	e
6	5.3e+03	3.3e+04	2.4e+01	2.3e+00	6.9e-04
6	2.1e+04	1.3e+05	1.0e+02	5.3e+00	4.2e-04
6	8.5e+04	5.2e+05	4.0e+02	2.3e+01	3.1e-04
8	5.3e+03	3.3e+04	2.5e+01	1.9e+00	2.0e-05
8	2.1e+04	1.3e+05	1.0e+02	5.8e+00	1.3e-05
8	8.5e+04	5.2e+05	4.2e+02	2.4e+01	1.6e-05
10	5.3e+03	3.3e+04	2.7e+01	2.0e+00	4.5e-07
10	2.1e+04	1.3e+05	1.1e+02	6.2e+00	1.3e-06
10	8.5e+04	5.2e+05	4.4e+02	2.6e+01	7.4e-07
12	5.3e+03	3.3e+04	2.9e+01	2.2e+00	3.0e-08
12	2.1e+04	1.3e+05	1.2e+02	6.7e+00	2.9e-08
12	8.5e+04	5.2e+05	4.7e+02	2.8e+01	2.0e-08

TABLE 8
Numerical results of operator N for the bean-shaped object.

m_c	ω	n	T_s	T_a	e
6	5.2e+03	3.3e+04	3.2e+01	2.6e+00	1.2e-03
6	2.1e+04	1.3e+05	1.3e+02	6.6e+00	1.1e-03
6	8.3e+04	5.2e+05	5.1e+02	3.0e+01	1.4e-03
8	5.2e+03	3.3e+04	3.3e+01	2.4e+00	1.0e-04
8	2.1e+04	1.3e+05	1.3e+02	7.0e+00	1.2e-04
8	8.3e+04	5.2e+05	5.3e+02	3.0e+01	1.2e-04
10	5.2e+03	3.3e+04	3.5e+01	2.5e+00	4.2e-06
10	2.1e+04	1.3e+05	1.4e+02	7.4e+00	4.6e-06
10	8.3e+04	5.2e+05	5.6e+02	3.2e+01	3.4e-06
12	5.2e+03	3.3e+04	3.8e+01	2.8e+00	5.1e-07
12	2.1e+04	1.3e+05	1.5e+02	8.0e+00	3.4e-07
12	8.3e+04	5.2e+05	6.1e+02	3.4e+01	5.9e-07

4. Discussions. This paper presented a new directional algorithm for rapid evaluation of high frequency boundary integrals via oscillatory Chebyshev interpolation and local FFTs. This algorithm is conceptually simple, fast, and kernel-independent.

Among the previously presented algorithms, this algorithm shares with the algorithm in [12, 13] the idea of using directional low-rank separated approximations. However, in [12, 13] the directional upward and downward equivalent sources are computed in upward and downward passes in a recursive fashion. In this algorithm, the low-rank separated approximations are computed instead using local FFTs combined with oscillatory Chebyshev interpolation. This makes the current algorithm much simpler to implement.

Though the paper presented the algorithms in the case of smooth boundaries, they work for piecewise smooth boundaries without much modification. One assumption is that the discretization points are equally spaced according to the arclength parameterization. This assumption can be relaxed if the parameterization is a smooth function of the arclength. For an irregular discretization, the nonuniform FFTs [3, 11, 15, 19] can be used instead. The main topic for future work is the extension of this approach to the three-dimensional boundary integral formulations of the scattering problems.

Acknowledgment. The author thanks Anil Damle and the anonymous referees for comments and suggestions.

REFERENCES

[1] M. Abramowitz and I. A. Stegun, eds., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, New York, 1992. Reprint of the 1972 edition.

[2] M. Bebendorf, C. Kuske, and R. Venn, *Wideband nested cross approximation for Helmholtz problems*, Numer. Math., to appear.

[3] G. Beylkin, *On the fast Fourier transform of functions with singularities*, Appl. Comput. Harmon. Anal., 2 (1995), pp. 363–381.

[4] E. Bleszynski, M. Bleszynski, and T. Jaroszewicz, *Aim: Adaptive integral method for solving large-scale electromagnetic scattering and radiation problems*, Radio Science, 31 (1996), pp. 1225–1251.

[5] O. P. Bruno and L. A. Kunyansky, *A fast, high-order algorithm for the solution of surface scattering problems: Basic implementation, tests, and applications*, J. Comput. Phys., 169 (2001), pp. 80–110.

[6] C. Cecka and E. Darve, *Fourier-based fast multipole method for the Helmholtz equation*, SIAM J. Sci. Comput., 35 (2013), pp. A79–A103.

[7] R. Coifman, V. Rokhlin, and S. Wandzura, *The fast multipole method for the wave equation: A pedestrian prescription*, IEEE Antennas and Propagation Mag., 35 (1993), pp. 7–12.

[8] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, 3rd ed., Appl. Math. Sci. 93, Springer, New York, 2013.

[9] E. Darve, *The fast multipole method* I: *Error analysis and asymptotic complexity*, SIAM J. Numer. Anal., 38 (2000), pp. 98–128.

[10] E. Darve and P. Havé, *A fast multipole method for Maxwell equations stable at all frequencies*, Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 362 (2004), pp. 603–628.

[11] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data*, SIAM J. Sci. Comput., 14 (1993), pp. 1368–1393.

[12] B. Engquist and L. Ying, *Fast directional multilevel algorithms for oscillatory kernels*, SIAM J. Sci. Comput., 29 (2007), pp. 1710–1737.

[13] B. Engquist and L. Ying, *A fast directional algorithm for high frequency acoustic scattering in two dimensions*, Commun. Math. Sci., 7 (2009), pp. 327–345.

[14] M. A. Epton and B. Dembart, *Multipole translation theory for the three-dimensional Laplace and Helmholtz equations*, SIAM J. Sci. Comput., 16 (1995), pp. 865–897.

[15] L. Greengard and J.-Y. Lee, *Accelerating the nonuniform fast Fourier transform*, SIAM Rev., 46 (2004), pp. 443–454.

[16] M. Messner, M. Schanz, and E. Darve, *Fast directional multilevel summation for oscillatory kernels based on Chebyshev interpolation*, J. Comput. Phys., 231 (2012), pp. 1175–1196.

[17] E. Michielssen and A. Boag, *A multilevel matrix decomposition algorithm for analyzing scattering from large structures*, IEEE Trans. Antennas and Propagation, 44 (1996), pp. 1086–1093.

[18] J.-C. Nédélec, *Acoustic and Electromagnetic Equations: Integral Representations for Harmonic Problems*, Appl. Math. Sci. 144, Springer-Verlag, New York, 2001.

[19] D. Potts, G. Steidl, and M. Tasche, *Fast Fourier transforms for nonequispaced data: A tutorial*, in Modern Sampling Theory, Appl. Numer. Harmon. Anal., Birkhäuser Boston, Boston, MA, 2001, pp. 247–270.

[20] V. Rokhlin, *Rapid solution of integral equations of scattering theory in two dimensions*, J. Comput. Phys., 86 (1990), pp. 414–439.

[21] V. Rokhlin, *Diagonal forms of translation operators for the Helmholtz equation in three dimensions*, Appl. Comput. Harmon. Anal., 1 (1993), pp. 82–93.

[22] V. Rokhlin, *Sparse diagonal forms for translation operators for the Helmholtz equation in two dimensions*, Appl. Comput. Harmon. Anal., 5 (1998), pp. 36–67.

[23] J. Song, C.-C. Lu, and W. C. Chew, *Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects*, IEEE Trans. Antennas and Propagation, 45 (1997), pp. 1488–1493.

[24] J. M. Song and W. C. Chew, *Multilevel fast-multipole algorithm for solving combined field integral equations of electromagnetic scattering*, Microwave Opt. Tech. Lett., 10 (1995), pp. 15–19.