Hierarchical Interpolative Factorization for Elliptic Operators: Integral Equations

KENNETH L. HO
Stanford University

LEXING YING
Stanford University

Abstract

This paper introduces the hierarchical interpolative factorization for integral equations (HIF-IE) associated with elliptic problems in two and three dimensions. This factorization takes the form of an approximate generalized LU decomposition that permits the efficient application of the discretized operator and its inverse. HIF-IE is based on the recursive skeletonization algorithm but incorporates a novel combination of two key features: (1) a matrix factorization framework for sparsifying structured dense matrices and (2) a recursive dimensional reduction strategy to decrease the cost. Thus, higher-dimensional problems are effectively mapped to one dimension, and we conjecture that constructing, applying, and inverting the factorization all have linear or quasilinear complexity. Numerical experiments support this claim and further demonstrate the performance of our algorithm as a generalized fast multipole method, direct solver, and preconditioner. HIF-IE is compatible with geometric adaptivity and can handle both boundary and volume problems. MATLAB® codes are freely available.

1 Introduction

This paper considers integral equations (IEs) of the form

$$a(x)u(x) + b(x)\int_{\Omega} K(\|x-y\|)c(y)u(y)\,d\Omega(y) = f(x), \quad x \in \Omega \subset \mathbb{R}^d, \tag{1.1}$$

associated with elliptic partial differential equations (PDEs), where $a(x)$, $b(x)$, $c(x)$, and $f(x)$ are given functions; the integral kernel $K(r)$ is related to the fundamental solution of the underlying PDE; and $d = 2$ or 3. Such equations encompass both boundary and volume problems and can be derived from PDEs in various ways. We give two prototypical examples below:

(1) Consider the interior Dirichlet Laplace problem

$$\Delta u(x) = 0, \qquad x \in \mathscr{D} \subset \mathbb{R}^d, \tag{1.2a}$$

$$u(x) = f(x), \quad x \in \partial\mathscr{D} \equiv \Gamma, \tag{1.2b}$$

Communications on Pure and Applied Mathematics, Vol. LXIX, 1314–1353 (2016)

in a smooth, simply connected domain, which can be solved by writing $u(x)$ as the double-layer potential

$$u(x) = \int_\Gamma \frac{\partial G}{\partial \nu_y}(\|x-y\|)\sigma(y)d\Gamma(y), \quad x \in \mathscr{D}, \tag{1.3}$$

over an unknown surface density $\sigma(x)$, where

$$G(r) = \begin{cases} -\frac{1}{2\pi}\log r, & d = 2, \\ \frac{1}{4\pi r}, & d = 3, \end{cases} \tag{1.4}$$

is the fundamental solution of the free-space PDE and ν_y is the unit outer normal at $y \in \Gamma$. By construction, (1.3) satisfies (1.2a). To enforce the boundary condition (1.2b), take the limit as $x \to \Gamma$ and use standard results from potential theory [31] to obtain

$$-\frac{1}{2}\sigma(x) + \int_\Gamma \frac{\partial G}{\partial \nu_y}(\|x-y\|)\sigma(y)d\Gamma(y) = f(x), \quad x \in \Gamma, \tag{1.5}$$

where the integral is defined in the principal value sense. This is a boundary IE for $\sigma(x)$ of the form (1.1) (up to a straightforward generalization to matrix-valued kernels).

Alternatively, one could use the single-layer potential representation

$$u(x) = \int_\Gamma G(\|x-y\|)\sigma(y)d\Gamma(y), \quad x \in \mathscr{D},$$

which immediately gives the IE

$$\int_\Gamma G(\|x-y\|)\sigma(y)d\Gamma(y) = f(x), \quad x \in \Gamma,$$

upon taking the limit as $x \to \Gamma$ since the integral is well-defined. Note that this has $a(x) \equiv 0$ in (1.1). Such equations are called first-kind Fredholm IEs and are generally ill-conditioned. Second-kind Fredholm IEs such as (1.5), on the other hand, have $a(x) \neq 0$ for all x and are usually well-conditioned.

(2) Consider the divergence-form PDE

$$\nabla \cdot (a(x)\nabla u(x)) = f(x), \quad x \in \Omega \subset \mathbb{R}^d,$$

and let

$$u(x) = \int_\Omega G(\|x-y\|)\sigma(y)d\Omega(y),$$

where $G(r)$ is as defined in (1.4). Then the PDE becomes the volume IE

$$a(x)\sigma(x) + \nabla a(x) \cdot \int_\Omega \nabla_x G(\|x-y\|)\sigma(y)d\Omega(y) = f(x), \quad x \in \Omega,$$

upon substitution, which again has the form (1.1).

IEs can similarly be derived for many of the PDEs of classical physics including the Laplace, Helmholtz, Stokes, and time-harmonic Maxwell equations. In such cases, the kernel function $K(r)$ is typically singular near 0 but otherwise smooth with noncompact support. For this paper, we will also require that $K(r)$ not be too oscillatory.

Discretization of (1.1) using, e.g., the Nyström, collocation, or Galerkin method leads to a linear system

$$Au = f, \tag{1.6}$$

where $A \in \mathbb{C}^{N\times N}$ is dense with u and f the discrete analogues of $u(x)$ and $f(x)$, respectively. This paper is concerned with the efficient factorization and solution of such systems.

1.1 Previous Work

Numerical methods for solving (1.6) can be classified into several groups. The first consists of classical direct methods like Gaussian elimination or other standard matrix factorizations [26], which compute the solution exactly (in principle, to machine precision, up to conditioning) without iteration. These methods are useful when N is small. However, since A is dense, such algorithms generally have $O(N^3)$ complexity, which quickly makes them infeasible as N increases.

The second group is that of iterative methods, among the most popular of which are Krylov subspace methods such as conjugate gradient [38, 49] or GMRES [47]. The number of iterations required depends on the problem and is typically small for second-kind IEs but can grow rapidly for first-kind ones. The main computational cost is the calculation of matrix-vector products at each iteration. Combined with fast multipole methods (FMMs) [22, 28, 29, 54] or other accelerated matrix multiplication schemes [5, 36], such techniques can yield asymptotically optimal or near-optimal solvers with $O(N)$ or $O(N \log N)$ complexity. However, iterative methods are not as robust as their direct counterparts, especially when $a(x)$, $b(x)$, or $c(x)$ lacks regularity or has high contrast. In such cases, convergence can be slow and specialized preconditioners are often needed. Furthermore, iterative methods can be inefficient for systems involving multiple right-hand sides or low-rank updates, which is an important setting for many applications of increasing interest, including time stepping, inverse problems, and design.

The third group covers rank-structured direct solvers, which exploit the observation that certain off-diagonal blocks of A are numerically low-rank in order to dramatically lower the cost. The seminal work in this area is due to Hackbusch et al. [33–35], whose $\mathscr{H}$- and $\mathscr{H}^2$-matrices have been shown to achieve linear or quasilinear complexity. Although their work has had significant theoretical impact, in practice, the constants implicit in the asymptotic scalings tend to be large due to the recursive nature of the inversion algorithms and the use of expensive hierarchical matrix-matrix multiplication.

More recent developments aimed at improving practical performance include solvers for hierarchically semiseparable (HSS) matrices [10, 11, 52] and methods based on recursive skeletonization (RS) [25, 27, 39, 43], among other related schemes [2, 8, 13]. These can be viewed as special cases of $\mathscr{H}^2$-matrices and are optimal in one dimension (1D) (e.g., boundary IEs on curves) but have superlinear complexities in higher dimensions. In particular, RS proceeds analogously to the nested dissection multifrontal method (MF) for sparse linear systems [19,23], with the so-called skeletons characterizing the off-diagonal blocks corresponding to the separator fronts. These grow as $O(N^{1/2})$ in two dimensions (2D) and $O(N^{2/3})$ in three dimensions (3D), resulting in solver complexities of $O(N^{3/2})$ and $O(N^2)$, respectively.

Recently, Corona, Martinsson, and Zorin [16] constructed an $O(N)$ RS solver in 2D by exploiting further structure among the skeletons and using hierarchical matrix algebra. The principal observation is that for a broad class of integral kernels, the generic behavior of RS is to retain degrees of freedom (DOFs) only along the boundary of each cell in a domain partitioning. Thus, 2D problems are reduced to 1D, and the large skeleton matrices accumulated throughout the algorithm can be handled efficiently using 1D HSS techniques. However, this approach is quite involved and has yet to be realized in 3D or in complicated geometries.

1.2 Contributions

In this paper, we introduce the hierarchical interpolative factorization for IEs (HIF-IE), which produces an approximate generalized LU decomposition of A with linear or quasilinear complexity estimates. HIF-IE is based on RS but augments it with a novel combination of two key features: (1) a matrix factorization formulation via a sparsification framework similar to that developed in [11, 50, 52] and (2) a recursive dimensional reduction scheme as pioneered in [16]. Unlike [16], however, which keeps large skeleton sets but works with them implicitly using fast structured methods, our sparsification approach allows us to reduce the skeletons explicitly. This obviates the need for internal hierarchical matrix representations, which substantially simplifies the algorithm and enables it to extend naturally to 3D and to complex geometries, in addition to promoting a more direct view of the dimensional reduction process.

Figure 1.1 shows a schematic of HIF-IE as compared to RS in 2D. In RS (top), the domain is partitioned into a set of square cells at each level of a tree hierarchy. Each cell is skeletonized from the finest level to the coarsest, leaving DOFs only along cell interfaces. The size of these interfaces evidently grows as we march up the tree, which ultimately leads to the observed $O(N^{3/2})$ complexity.

In contrast, in HIF-IE (bottom), we start by skeletonizing the cells at the finest level as in RS but, before proceeding further, perform an additional level of edge skeletonization by grouping the remaining DOFs by cell edge. This respects the 1D structure of the interface geometry and allows more DOFs to be eliminated. The combination of cell and edge compression is then repeated up the tree, with

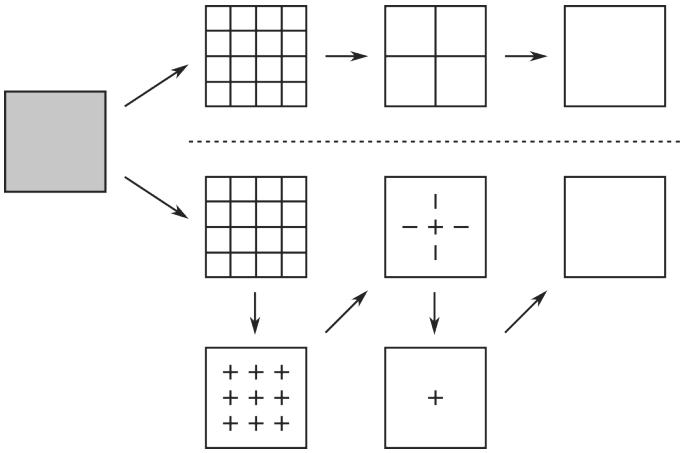

FIGURE 1.1. Schematic of RS (top) and HIF-IE (bottom) in 2D. The gray box (left) represents a uniformly discretized square; the lines in the interior of the boxes (right) denote the remaining DOFs after each level of skeletonization.

the result that the skeleton growth is now suppressed. The reduction from 2D (square cells) to 1D (edges) to zero dimensions (0D) (points) is completely explicit. Extension to 3D is immediate by skeletonizing cubic cells, then faces, then edges at each level to execute a reduction from 3D to 2D to 1D to 0D. This tight control of the skeleton size is essential for achieving near-optimal scaling.

Once the factorization has been constructed, it can be used to rapidly apply both A and A^{-1}, thereby serving as a generalized FMM, direct solver, or preconditioner (depending on the accuracy). Other capabilities are possible, too, though they will not be pursued here. As such, HIF-IE is considerably more general than many previous non-factorization-based fast direct solvers [10,16,25,39,43], which facilitate only the application of the inverse.

Extensive numerical experiments reveal strong evidence for quasilinear complexity and demonstrate that HIF-IE can accurately approximate various integral operators in both boundary and volume settings with high practical efficiency.

1.3 Outline

The remainder of this paper is organized as follows. In Section 2, we introduce the basic tools needed for our algorithm, including an efficient matrix sparsification operation that we call skeletonization. In Section 3, we describe the recursive skeletonization factorization (RSF), a reformulation of RS using our new factorization approach. This will serve to familiarize the reader with our sparsification framework as well as to highlight the fundamental difficulty associated with RS methods in 2D and 3D. In Section 4, we present HIF-IE as an extension of RSF with additional levels of skeletonization corresponding to recursive dimensional

reductions. Although we cannot yet provide a rigorous complexity analysis, estimates based on well-supported rank assumptions suggest that HIF-IE achieves linear or quasilinear complexity. This conjecture is borne out by numerical experiments, which we detail in Section 5. Finally, Section 6 concludes with some discussion and future directions.

2 Preliminaries

In this section, we first list our notational conventions and then describe the basic elements of our algorithm.

Uppercase letters will generally denote matrices, while the lowercase letters c, p, q, r, and s denote ordered sets of indices, each of which is associated with a DOF in the problem. For a given index set c, its cardinality is written $|c|$. The (unordered) complement of c is given by c^{c}, with the parent set to be understood from the context. The uppercase letter C is reserved to denote a collection of disjoint index sets.

Given a matrix A, A_{pq} is the submatrix with rows and columns restricted to the index sets p and q, respectively. We also use the MATLAB® notation $A_{:,q}$ to denote the submatrix with columns restricted to q.

Throughout, $\|\cdot\|$ refers to the 2-norm.

2.1 Sparse Elimination

Let

$$A = \begin{bmatrix} A_{pp} & A_{pq} & \\ A_{qp} & A_{qq} & A_{qr} \\ & A_{rq} & A_{rr} \end{bmatrix} \tag{2.1}$$

be a matrix defined over the indices (p, q, r). This matrix structure often appears in sparse PDE problems, where, for example, p corresponds to the interior DOFs of a region $\mathscr{D}$, q to the DOFs on the boundary $\partial\mathscr{D}$, and r to the external region $\Omega \setminus \bar{\mathscr{D}}$, which should be thought of as large. In this setting, the DOFs p and r are separated by q and hence do not directly interact, resulting in the form (2.1).

Our first tool is quite standard and concerns the efficient elimination of DOFs from such sparse matrices.

LEMMA 2.1. *Let A be given by* (2.1) *and write $A_{pp} = L_p D_p U_p$ in factored form, where L_p and U_p are unit triangular matrices* (*up to permutation*). *If A_{pp} is nonsingular, then*

$$R_p^* A S_p = \begin{bmatrix} D_p & & \\ & B_{qq} & A_{qr} \\ & A_{rq} & A_{rr} \end{bmatrix}, \tag{2.2}$$

where

$$R_p^* = \begin{bmatrix} I & & \\ -A_{qp}U_p^{-1}D_p^{-1} & I & \\ & & I \end{bmatrix} \begin{bmatrix} L_p^{-1} & & \\ & I & \\ & & I \end{bmatrix},$$
$$S_p = \begin{bmatrix} U_p^{-1} & & \\ & I & \\ & & I \end{bmatrix} \begin{bmatrix} I & -D_p^{-1}L_p^{-1}A_{pq} & \\ & I & \\ & & I \end{bmatrix}$$

and $B_{qq} = A_{qq} - A_{qp}A_{pp}^{-1}A_{pq}$ *is the associated Schur complement.*

Note that the indices p have been decoupled from the rest. Regarding the subsystem in (2.2) over the indices (q, r) only, we may therefore say that the DOFs p have been eliminated. The operators R_p and S_p carry out this elimination, which furthermore is particularly efficient since the interactions involving the large index set r are unchanged.

2.2 Interpolative Decomposition

Our next tool is the interpolative decomposition (ID) [14] for low-rank matrices, which we present in a somewhat nonstandard form below.

LEMMA 2.2. *Let* $A = A_{:,q} \in \mathbb{C}^{m\times n}$ *with rank* $k \leq \min(m, n)$. *Then there exist a partitioning* $q = \hat{q} \cup \check{q}$ *with* $|\hat{q}| = k$ *and a matrix* $T_q \in \mathbb{C}^{k\times n}$ *such that* $A_{:,\check{q}} = A_{:,\hat{q}}T_q$.

PROOF. Let

$$A\Pi = QR = Q\begin{bmatrix} R_1 & R_2 \end{bmatrix}$$

be a so-called thin pivoted QR decomposition of A, where $Q \in \mathbb{C}^{m\times k}$ is unitary, $R \in \mathbb{C}^{k\times n}$ is upper triangular, and the permutation matrix $\Pi \in \{0, 1\}^{n\times n}$ has been chosen so that $R_1 \in \mathbb{C}^{k\times k}$ is nonsingular. Then identifying the first k pivots with $\hat{q}$ and the remainder with $\check{q}$,

$$A_{:,\check{q}} = QR_2 = (QR_1)\big(R_1^{-1}R_2\big) \equiv A_{:,\hat{q}}T_q$$

for $T_q = R_1^{-1}R_2$. □

The ID can also be written more traditionally as

$$A = A_{:,\hat{q}}\begin{bmatrix} I & T_q \end{bmatrix}\Pi$$

where Π is the permutation matrix associated with the ordering $(\hat{q}, \check{q})$. We call $\hat{q}$ and $\check{q}$ the *skeleton* and *redundant* indices, respectively. Lemma 2.2 states that the redundant columns of A can be interpolated from its skeleton columns. The following shows that the ID can also be viewed as a sparsification operator.

COROLLARY 2.3. *Let $A = A_{:,q}$ be a low-rank matrix. If $q = \hat{q} \cup \check{q}$ and T_q are such that $A_{:,\check{q}} = A_{:,\hat{q}} T_q$, then*

$$\begin{bmatrix} A_{:,\check{q}} & A_{:,\hat{q}} \end{bmatrix} \begin{bmatrix} I & \\ -T_q & I \end{bmatrix} = \begin{bmatrix} 0 & A_{:,\hat{q}} \end{bmatrix}.$$

In general, let $A_{:,\check{q}} = A_{:,\hat{q}} T_q + E$ for some error matrix E and characterize the ID by the functions $\alpha(n,k)$ and $\beta(n,k)$ such that

$$\|T_q\| \le \alpha(n,k), \quad \|E\| \le \beta(n,k)\sigma_{k+1}(A), \tag{2.3}$$

where $\sigma_{k+1}(A)$ is the $(k+1)^{\text{st}}$ largest singular value of A. If $|\alpha(n,k)|$ and $|\beta(n,k)|$ are not too large, then (2.3) implies that the reconstruction of $A_{:,\check{q}}$ is stable and accurate. There exists an ID with

$$\alpha(n,k) = \sqrt{f^2 k(n-k)}, \quad \beta(n,k) = \sqrt{1 + f^2 k(n-k)} \tag{2.4}$$

for $f = 1$, but computing it can take exponential time, requiring the combinatorial maximization of a submatrix determinant. However, an ID satisfying (2.4) with any $f > 1$ can be computed in polynomial time [30]. In this paper, we use the algorithm of [14] based on a simple pivoted QR decomposition, which has a possibility of failure but seems to consistently achieve (2.4) with $f = 2$ in practice at a cost of $O(kmn)$ operations. Fast algorithms based on random sampling are also available [37], but these can incur some loss of accuracy (see also Section 4.3).

The ID can be applied in both fixed and adaptive rank settings. In the former, the rank k is specified, while, in the latter, the approximation error is specified and the rank adjusted to achieve (an estimate of) it. Hereafter, we consider the ID only in the adaptive sense, using the relative magnitudes of the pivots to adaptively select k such that $\|E\| \lesssim \varepsilon \|A\|$ for any specified relative precision $\varepsilon > 0$.

2.3 Skeletonization

We now combine Lemmas 2.1 and 2.2 to efficiently eliminate redundant DOFs from dense matrices with low-rank off-diagonal blocks.

LEMMA 2.4. *Let*

$$A = \begin{bmatrix} A_{pp} & A_{pq} \\ A_{qp} & A_{qq} \end{bmatrix}$$

with A_{pq} and A_{qp} low-rank, and let $p = \hat{p} \cup \check{p}$ and T_p be such that

$$\begin{bmatrix} A_{q\check{p}} \\ A_{\check{p}q}^* \end{bmatrix} = \begin{bmatrix} A_{q\hat{p}} \\ A_{\hat{p}q}^* \end{bmatrix} T_p;$$

i.e., $A_{q\check{p}} = A_{q\hat{p}} T_p$ and $A_{\check{p}q} = T_p^* A_{\hat{p}q}$. Without loss of generality, write

$$A = \begin{bmatrix} A_{\check{p}\check{p}} & A_{\check{p}\hat{p}} & A_{\check{p}q} \\ A_{\hat{p}\check{p}} & A_{\hat{p}\hat{p}} & A_{\hat{p}q} \\ A_{q\check{p}} & A_{q\hat{p}} & A_{qq} \end{bmatrix}$$

and define

$$Q_p = \begin{bmatrix} I & & \\ -T_p & I & \\ & & I \end{bmatrix}.$$

Then

$$Q_p^* A Q_p = \begin{bmatrix} B_{\check{p}\check{p}} & B_{\check{p}\hat{p}} & \\ B_{\hat{p}\check{p}} & A_{\hat{p}\hat{p}} & A_{\hat{p}q} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix}, \tag{2.5}$$

where

$$\begin{aligned} B_{\check{p}\check{p}} &= A_{\check{p}\check{p}} - T_p^* A_{\hat{p}\check{p}} - A_{\check{p}\hat{p}} T_p + T_p^* A_{\hat{p}\hat{p}} T_p, \\ B_{\check{p}\hat{p}} &= A_{\check{p}\hat{p}} - T_p^* A_{\hat{p}\hat{p}}, \\ B_{\hat{p}\check{p}} &= A_{\hat{p}\check{p}} - A_{\hat{p}\hat{p}} T_p, \end{aligned}$$

so

$$R_{\check{p}}^* Q_p^* A Q_p S_{\check{p}} = \begin{bmatrix} D_{\check{p}} & & \\ & B_{\hat{p}\hat{p}} & A_{\hat{p}q} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix} \equiv \mathscr{Z}_p(A), \tag{2.6}$$

where $R_{\check{p}}$ and $S_{\check{p}}$ are the elimination operators of Lemma 2.1 *associated with $\check{p}$ and $B_{\hat{p}\hat{p}} = A_{\hat{p}\hat{p}} - B_{\hat{p}\check{p}} B_{\check{p}\check{p}}^{-1} B_{\check{p}\hat{p}}$, assuming that $B_{\check{p}\check{p}}$ is nonsingular.*

In essence, the ID sparsifies A by decoupling $\check{p}$ from q, thereby allowing it to be eliminated using efficient sparse techniques. We refer to this procedure as *skeletonization* since only the skeletons $\hat{p}$ remain. Note that the interactions involving $q = p^{\mathsf{c}}$ are unchanged. A very similar approach has previously been described in the context of HSS ULV decompositions [11] by combining the structure-preserving rank-revealing factorization [53] with reduced matrices [50].

In general, the ID often only approximately sparsifies A (for example, if its off-diagonal blocks are low-rank only to a specified numerical precision) so that (2.5) and consequently (2.6) need not hold exactly. In such cases, the skeletonization operator $\mathscr{Z}_p(\cdot)$ should be interpreted as also including an intermediate truncation step that enforces sparsity explicitly. For notational convenience, however, we will continue to identify the left- and right-hand sides of (2.6) by writing $\mathscr{Z}_p(A) \approx R_{\check{p}}^* Q_p^* A Q_p S_{\check{p}}$, with the truncation to be understood implicitly.

In this paper, we often work with a collection C of disjoint index sets, where $A_{c,c^{\mathsf{c}}}$ and $A_{c^{\mathsf{c}},c}$ are numerically low-rank for all $c \in C$. Applying Lemma 2.4 to all $c \in C$ gives

$$\mathscr{Z}_C(A) \approx U^* A V, \qquad U = \prod_{c \in C} Q_c R_{\check{c}}, \quad V = \prod_{c \in C} Q_c S_{\check{c}},$$

where the redundant DOFs $\check{c}$ for each $c \in C$ have been decoupled from the rest and the matrix products over C can be taken in any order. The resulting skeletonized

matrix $\mathscr{Z}_C(A)$ is significantly sparsified and has a block diagonal structure over the index groups

$$\theta = \Big(\bigcup_{c\in C}\{\check{c}\}\Big) \cup \Big\{s \setminus \bigcup_{c\in C}\check{c}\Big\},$$

where the outer union is to be understood as acting on collections of index sets and $s = \{1, \dots, N\}$ is the set of all indices.

3 Recursive Skeletonization Factorization

In this section, we present RSF, a reformulation of RS [25, 27, 39, 43] as a matrix factorization using the sparsification view of skeletonization as developed in Lemma 2.4. Mathematically, RSF is identical to RS but expresses the matrix A as a (multiplicative) multilevel generalized LU decomposition instead of as an additive hierarchical low-rank update. This representation enables much simpler algorithms for applying A and A^{-1} as well as establishes a direct connection with MF [19,23] for sparse matrices, which produces a (strict) LU decomposition using Lemma 2.1. Indeed, RSF is essentially just MF with presparsification via the ID at each level. This point of view places methods for structured dense and sparse matrices within a common framework, which provides a potential means to transfer techniques from one class to the other.

Note that because RSF is based on elimination, it requires that certain intermediate matrices be invertible, which in general means that A must be square. This is a slight limitation when compared to RS, which can be used, for example, as a generalized FMM [25, 39] or least squares solver [40] for rectangular matrices.

We begin with a detailed description of RSF in 2D before extending to 3D in the natural way (the 1D case will not be treated but should be obvious from the discussion). The same presentation framework will also be used for HIF-IE in Section 4, which we hope will help make clear the specific innovations responsible for its improved complexity estimates.

3.1 Two Dimensions

Consider the IE (1.1) on $\Omega = (0,1)^2$, discretized using a piecewise constant collocation method over a uniform $n \times n$ grid for simplicity. More general domains and discretizations can be handled without difficulty, but the current setting will serve to illustrate the main ideas.

Let h be the step size in each direction and assume that $n = 1/h = 2^L m$, where $m = O(1)$ is a small integer. Integer pairs $j = (j_1, j_2)$ index the elements $\Omega_j = h(j_1 - 1, j_1) \times h(j_2 - 1, j_2)$ and their centers $x_j = h(j_1 - \frac{1}{2}, j_2 - \frac{1}{2})$ for $1 \le j_1, j_2 \le n$. With $\{x_j\}$ as the collocation points, the discrete system (1.6) reads

$$a_i u_i + b_i \sum_j K_{ij} c_j u_j = f_i$$

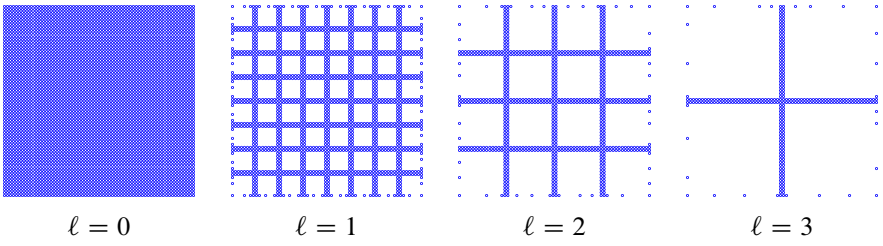

FIGURE 3.1. Active DOFs at each level ℓ of RSF in 2D.

at each x_i, where $a_j = a(x_j)$, $b_j = b(x_j)$, $c_j = c(x_j)$, and $f_j = f(x_j)$; u_j is the approximation to $u(x_j)$; and

$$K_{ij} = \int_{\Omega_j} K(\|x_i - y\|) d\Omega(y). \tag{3.1}$$

Note that A is not stored since it is dense; rather, its entries are generated as needed. The total number of DOFs is $N = n^2$, each of which is associated with a point x_j and an index in s.

The algorithm proceeds by eliminating DOFs level by level. At each level ℓ, the set of DOFs that have not been eliminated are called *active* with indices s_ℓ. Initially, we set $A_0 = A$ and $s_0 = s$. Figure 3.1 shows the active DOFs at each level for a representative example.

Level 0

Defined at this stage are A_0 and s_0. Partition Ω into the Voronoi cells [4] $mh(j_1 - 1, j_1) \times mh(j_2 - 1, j_2)$ of width $mh = n/2^L$ about the centers $mh(j_1 - \frac{1}{2}, j_2 - \frac{1}{2})$ for $1 \le j_1, j_2 \le 2^L$. Let C_0 be the collection of index sets corresponding to the active DOFs of each cell. Clearly, $\bigcup_{c \in C_0} c = s_0$. Then skeletonization with respect to C_0 gives

$$A_1 = \mathscr{Z}_{C_0}(A_0) \approx U_0^* A_0 V_0, \qquad U_0 = \prod_{c \in C_0} Q_c R_{\check{c}}, \quad V_0 = \prod_{c \in C_0} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_0} \check{c}$ have been eliminated (and marked inactive). Let $s_1 = s_0 \setminus \bigcup_{c \in C_0} \check{c} = \bigcup_{c \in C_0} \hat{c}$ be the remaining active DOFs. The matrix A_1 is block diagonal with block partitioning

$$\theta_1 = \Big(\bigcup_{c \in C_0} \{\check{c}\} \Big) \cup \{s_1\}.$$

Level ℓ

Defined at this stage are A_ℓ and s_ℓ. Partition Ω into the Voronoi cells $2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2)$ of width $2^\ell mh = n/2^{L-\ell}$ about the centers $2^\ell mh(j_1 -$

$\frac{1}{2}, j_2 - \frac{1}{2})$ for $1 \le j_1, j_2 \le 2^{L-\ell}$. Let C_ℓ be the collection of index sets corresponding to the active DOFs of each cell. Clearly, $\bigcup_{c \in C_\ell} c = s_\ell$. Skeletonization with respect to C_ℓ then gives

$$A_{\ell+1} = \mathscr{Z}_{C_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell, \qquad U_\ell = \prod_{c \in C_\ell} Q_c R_{\check{c}}, \quad V_\ell = \prod_{c \in C_\ell} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_\ell} \check{c}$ have been eliminated. The matrix $A_{\ell+1}$ is block diagonal with block partitioning

$$\theta_{\ell+1} = \Big(\bigcup_{c \in C_0} \{\check{c}\} \Big) \cup \cdots \cup \Big(\bigcup_{c \in C_\ell} \{\check{c}\} \Big) \cup \{s_{\ell+1}\},$$

where $s_{\ell+1} = s_\ell \setminus \bigcup_{c \in C_\ell} \check{c} = \bigcup_{c \in C_\ell} \hat{c}$.

Level L

Finally, we have A_L and s_L, where $D \equiv A_L$ is block diagonal with block partitioning

$$\theta_L = \Big(\bigcup_{c \in C_0} \{\check{c}\} \Big) \cup \cdots \cup \Big(\bigcup_{c \in C_{L-1}} \{\check{c}\} \Big) \cup \{s_L\}.$$

Combining the approximation over all levels gives

$$D \approx U_{L-1}^* \cdots U_0^* A V_0 \cdots V_{L-1},$$

where each U_ℓ and V_ℓ are products of unit triangular matrices, each of which can be inverted simply by negating its off-diagonal entries. Therefore,

$$A \approx U_0^{-*} \cdots U_{L-1}^{-*} D V_{L-1}^{-1} \cdots V_0^{-1} \equiv F, \tag{3.2a}$$

$$A^{-1} \approx V_0 \cdots V_{L-1} D^{-1} U_{L-1}^* \cdots U_0^* = F^{-1}. \tag{3.2b}$$

The factorization F permits fast multiplication and can be used as a generalized FMM. Its inverse F^{-1} can be used as a direct solver at high accuracy or as a preconditioner otherwise. If D is stored in factored form, e.g., as an LU decomposition, then the same factorization can readily be used for both tasks. We call (3.2) an (approximate) generalized LU decomposition since while each U_ℓ and V_ℓ are composed of triangular factors, they are not themselves triangular, being the product of both upper and lower triangular matrices. We emphasize that F and F^{-1} are not assembled explicitly and are applied only in factored form.

The entire procedure is summarized compactly as Algorithm 3.1. In general, we construct the cell partitioning at each level using an adaptive quadtree [48], which recursively subdivides the domain until each node contains only $O(1)$ DOFs.

3.2 Three Dimensions

Consider now the analogous setting in 3D, where $\Omega = (0,1)^3$ is discretized using a uniform $n \times n \times n$ grid with $\Omega_j = h(j_1 - 1, j_1) \times h(j_2 - 1, j_2) \times h(j_3 - 1, j_3)$ and $x_j = h(j_1 - \frac{1}{2}, j_2 - \frac{1}{2}, j_3 - \frac{1}{2})$ for $j = (j_1, j_2, j_3)$. The total number of DOFs is $N = n^3$.

Algorithm 3.1 RSF.

$A_0 = A$ ▷ initialize
for $\ell = 0, 1, \ldots, L-1$ **do** ▷ loop from finest to coarsest level
  $A_{\ell+1} = \mathscr{Z}_{C_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell$ ▷ skeletonize cells
end for
$A \approx U_0^{-*} \cdots U_{L-1}^{-*} A_L V_{L-1}^{-1} \cdots V_0^{-1}$ ▷ generalized LU decomposition

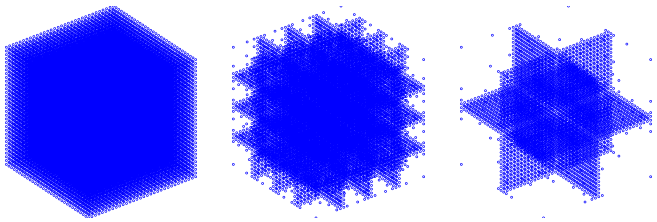

FIGURE 3.2. Active DOFs at each level ℓ of RSF in 3D.

The algorithm extends in the natural way with cubic cells $2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2) \times 2^\ell mh(j_3 - 1, j_3)$ about the centers $2^\ell mh(j_1 - \frac{1}{2}, j_2 - \frac{1}{2}, j_3 - \frac{1}{2})$ replacing the square cells in 2D at level ℓ for $1 \le j_1, j_2, j_3 \le 2^{L-\ell}$. With this modification, the rest of the algorithm remains unchanged. Figure 3.2 shows the active DOFs at each level for a representative example. The output is again a factorization of the form (3.2). General geometries can be treated using an adaptive octree.

3.3 Accelerated Compression

A dominant contribution to the cost of RSF is computing IDs for skeletonization. The basic operation required is the construction of an ID of

$$W_{\ell,c} = \begin{bmatrix} (A_\ell)_{c^{\mathsf{c}},c} \\ (A_\ell)^*_{c,c^{\mathsf{c}}} \end{bmatrix},$$

where $c \in C_\ell$ and $c^{\mathsf{c}} = s_\ell \setminus c$, following Lemma 2.4. We hereafter drop the dependence on ℓ for notational convenience. Observe that W_c is a tall-and-skinny matrix of size $O(N) \times |c|$, so forming its ID takes at least $O(N|c|)$ work. The total number of index sets $c \in C_\ell$ for all ℓ is $O(N)$, so considering all W_c yields a lower bound of $O(N^2)$ on the total work and hence on the complexity of RSF.

In principle, it is straightforward to substantially accelerate the algorithm by reconstructing an ID of W_c from that of a much smaller matrix Y_c. All that is needed is that the rows of Y_c span those of W_c, i.e., $\mathscr{R}(W_c^*) \subseteq \mathscr{R}(Y_c^*)$, where $\mathscr{R}(\cdot)$ denotes the matrix range.

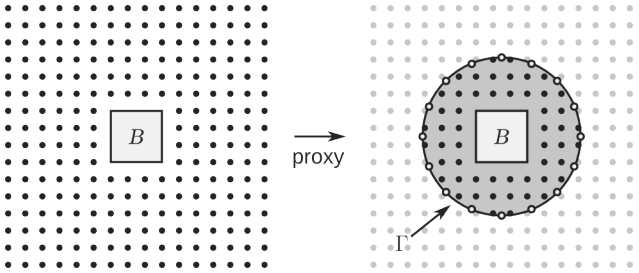

FIGURE 3.3. Accelerated compression using equivalent interactions. By Green's theorem, all off-diagonal interactions with a given box B can be represented by its interactions with an artificial local proxy surface Γ and with all DOFs interior to Γ.

LEMMA 3.1. *Let $W = XY$ with column indices q. If $q = \hat{q} \cup \check{q}$ and T_q are such that $Y_{:,\check{q}} = Y_{:,\hat{q}} T_q$, then*

$$W_{:,\check{q}} = XY_{:,\check{q}} = XY_{:,\hat{q}} T_q = W_{:,\hat{q}} T_q.$$

In other words, an ID of Y_c gives an ID of $W_c = X_c Y_c$. Note that we make no explicit reference to X_c; only its existence is assumed. Of course, such a small matrix Y_c always exists since $\operatorname{rank}(W_c) \leq |c|$; the difficulty lies in finding Y_c a priori.

For elliptic problems, the integral kernel $K(r)$ typically satisfies some form of Green's theorem, in which its values inside a region $\mathscr{D} \in \Omega$ can be recovered from its values on the boundary $\Gamma = \partial\mathscr{D}$. Consider, for example, the Laplace kernel (1.4) and let $\varphi(x) = G(\|x - x_0\|)$ be the harmonic field in $\mathscr{D}$ due to an exterior source $x_0 \in \Omega \setminus \bar{\mathscr{D}}$. Then

$$\varphi(x) = \int_\Gamma \left[\varphi(y) \frac{\partial G}{\partial \nu_y}(\|x - y\|) - \frac{\partial \varphi}{\partial \nu_y}(y) G(\|x - y\|) \right] d\Gamma(y), \quad x \in \mathscr{D},$$

i.e., the "incoming" field $\varphi(x)$ lives in the span of single- and double-layer interactions with Γ. In practice, we will use this fact only when $x \in \mathscr{D}$ is sufficiently separated from Γ (see below), in which case the double-layer term can often even be omitted since the corresponding discrete spaces are equal to high precision. Outgoing interactions can essentially be treated in the same way using the "transpose" of this idea.

In such cases, a suitable Y_c can readily be constructed. To see this, let B denote the cell containing the DOFs c and draw a local "proxy" surface Γ around B (Figure 3.3). This partitions c^{c} as $c^{\mathsf{c}} = c^{\mathsf{N}} \cup c^{\mathsf{F}}$, where c^{N} consists of all DOFs interior to Γ (the near field) and c^{F} consists of the rest (the far field). By Green's theorem, the interactions involving c^{F} can be represented by artificial "equivalent" interactions with Γ. Therefore, discretizing Γ with equivalent DOFs c^{E}, we assert the following:

LEMMA 3.2. *Consider* (1.1) *with* $b(x) \equiv c(x) \equiv 1$ *and let all quantities be as defined in the preceding discussion. Then, up to discretization error* (*see* [45]), $\mathscr{R}(A^*_{c^{\mathsf{F}},c}) \subseteq \mathscr{R}(Y^*_{c^{\mathsf{E}},c})$, *where* $(Y_{c^{\mathsf{E}},c})_{ij} = K(\|x_i^{\mathsf{E}} - x_j\|)$ *for* $\{x_j\}$ *and* $\{x_j^{\mathsf{E}}\}$ *the points identified with the DOFs* c *and* c^{E}, *respectively.*

PROOF. This immediately follows from Green's theorem upon recognizing that $A_{c^{\mathsf{F}},c}$ contains interactions involving only the original kernel function $K(r)$. This must be checked because $A_{:,c}$ may have Schur complement interactions (SCIs), i.e., those corresponding to the matrix $B_{\hat{p}\hat{p}}$ in (2.6), accumulated from skeletonization at previous levels, over which we do not have analytic control. However, due to the hierarchical nature of the domain partitioning, any such SCIs must be restricted to the diagonal block A_{cc}. Thus, Green's theorem applies. □

LEMMA 3.3. *Consider* (1.1) *with general* $b(x)$ *and* $c(x)$. *Then, up to discretization error,* $\mathscr{R}(A^*_{c^{\mathsf{F}},c}) \subseteq \mathscr{R}(Y^*_{c^{\mathsf{E}},c})$ *and* $\mathscr{R}(A_{c,c^{\mathsf{F}}}) \subseteq \mathscr{R}(Y_{c,c^{\mathsf{E}}})$, *where*

$$(Y_{c^{\mathsf{E}},c})_{ij} = K\big(\big\|x_i^{\mathsf{E}} - x_j\big\|\big)c(x_j), \quad (Y_{c,c^{\mathsf{E}}})_{ij} = b(x_i)K\big(\big\|x_i - x_j^{\mathsf{E}}\big\|\big).$$

PROOF. The functions $b(x)$ and $c(x)$ act as diagonal multipliers, so $A_{c^{\mathsf{F}},c} = B_{c^{\mathsf{F}}}\widetilde{A}_{c^{\mathsf{F}},c}C_c$, where $\widetilde{A}_{c^{\mathsf{F}},c}$ is the corresponding interaction matrix with $b(x) \equiv c(x) \equiv 1$ (i.e., that in Lemma 3.2), and $B_{c^{\mathsf{F}}} = \operatorname{diag}(b(x_i^{\mathsf{F}}))$ and $C_c = \operatorname{diag}(c(x_i))$ for $\{x_j^{\mathsf{F}}\}$ the points attached to c^{F}. By Lemma 3.2, $\widetilde{A}_{c^{\mathsf{F}},c} = \widetilde{X}_{c^{\mathsf{E}},c}\widetilde{Y}_{c^{\mathsf{E}},c}$ for some $\widetilde{X}_{c^{\mathsf{E}},c}$, so

$$A_{c^{\mathsf{F}},c} = B_{c^{\mathsf{F}}}\widetilde{X}_{c^{\mathsf{E}},c}\widetilde{Y}_{c^{\mathsf{E}},c}C_c = \big(B_{c^{\mathsf{F}}}\widetilde{X}_{c^{\mathsf{E}},c}\big)\big(\widetilde{Y}_{c^{\mathsf{E}},c}C_c\big) \equiv X_{c^{\mathsf{E}},c}Y_{c^{\mathsf{E}},c}.$$

A similar argument with $A_{c,c^{\mathsf{F}}} = B_c\widetilde{A}_{c,c^{\mathsf{F}}}C_{c^{\mathsf{F}}}$ analogously defined and

$$\widetilde{A}_{c,c^{\mathsf{F}}} = \widetilde{A}^{\mathsf{T}}_{c^{\mathsf{F}},c} = \widetilde{Y}^{\mathsf{T}}_{c^{\mathsf{E}},c}\widetilde{X}^{\mathsf{T}}_{c^{\mathsf{E}},c} \equiv \widetilde{Y}_{c,c^{\mathsf{E}}}\widetilde{X}_{c,c^{\mathsf{E}}}$$

proves that $A_{c,c^{\mathsf{F}}} = Y_{c,c^{\mathsf{E}}}X_{c,c^{\mathsf{E}}}$ for some $X_{c,c^{\mathsf{E}}}$. □

If Γ is separated from B, for example as in Figure 3.3, then standard multipole estimates [28, 29] show that we only need $|c^{\mathsf{E}}| = O(\log^{d-1}(1/\varepsilon))$ to satisfy Green's theorem to any precision ε. In particular, for fixed ε, we can choose $|c^{\mathsf{E}}|$ to be constant. Therefore, Lemma 3.3 gives

$$W_c \approx X_cY_c \equiv X_c\begin{bmatrix} A_{c^{\mathsf{N}},c} \\ A^*_{c,c^{\mathsf{N}}} \\ Y_{c^{\mathsf{E}},c} \\ Y^*_{c,c^{\mathsf{E}}} \end{bmatrix} \tag{3.3}$$

for some X_c, where Y_c has size $O(|c^{\mathsf{N}}| + 1) \times |c|$ with $|c^{\mathsf{N}}| = O(|c|)$ typically. Lemma 3.1 then reduces the global compression of W_c to the local compression of Y_c. This so-called proxy trick has also been employed by [14, 16, 25, 27, 39, 43, 44, 46, 54] and is crucial for reducing the asymptotic complexity. For numerical stability, we include the quadrature weights for the integral (3.1) in $Y_{c^{\mathsf{E}},c}$ and $Y_{c,c^{\mathsf{E}}}$ so that the various components of Y_c are all of the same order.

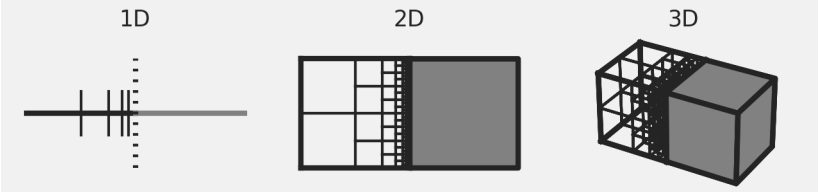

FIGURE 3.4. Recursive subdivision of source domain (white) into well-separated subdomains from the target (gray), each of which has constant interaction rank.

In this paper, for a cell B with scaled width 1 centered at the origin, we take as Γ the circle of radius $\frac{3}{2}$ in 2D, uniformly discretized with 64 points, and the sphere of radius $\frac{3}{2}$ in 3D, uniformly sampled (by projecting Gaussian random vectors) with 512 points. These values of $|c^{\mathsf{E}}|$ have been experimentally validated to reproduce interactions via the Laplace kernel (1.4) with $\varepsilon \sim 10^{-15}$. This approach is more efficient than the "supercell" proxy of [27, 39] by factors of $4/\pi = 1.2732\ldots$ in 2D and $6/\pi = 1.9099\ldots$ in 3D (volume ratio of the cube to the sphere of equal diameter), which takes as Γ the outer boundary of the 3×3 ($\times3$) cell block centered at B.

3.4 Complexity Estimates

We now investigate the computational complexity of RSF. For this, we need to estimate the skeleton size $|\hat{c}|$ for a typical index set $c \in C_\ell$ at level ℓ. Denote this quantity by k_ℓ and let $n_\ell = (2^\ell m)^d = O(2^{d\ell})$ be the number of DOFs (both active and inactive) in each cell. From Figures 3.1 and 3.2, it is clear that skeletons tend to cluster around cell interfaces, which can again be justified by Green's theorem, so $k_\ell = O(n_\ell^{1/2}) = O(2^\ell)$ in 2D and $k_\ell = O(n_\ell^{2/3}) = O(2^{2\ell})$ in 3D. Indeed, this can be verified using standard multipole estimates by noting that k_ℓ is on the order of the interaction rank between two adjacent cells at level ℓ, which can be analyzed via recursive subdivision to expose well-separated structures (Figure 3.4). This yields the more detailed result

$$k_\ell = \begin{cases} O(\ell), & d = 1, \\ O(2^{(d-1)\ell}), & d \geq 2, \end{cases} \tag{3.4}$$

which, in fact, holds for d equal to the intrinsic dimension rather than the ambient dimension.

THEOREM 3.4 ([39, 43]). *Assume that* (3.4) *holds. Then the cost of constructing the factorization* F *in* (3.2) *using RSF with accelerated compression is*

$$t_f = O(2^{dL}m^{3d}) + \sum_{\ell=0}^{L} 2^{d(L-\ell)}O(k_\ell^3) = \begin{cases} O(N), & d = 1, \\ O(N^{3(1-1/d)}), & d \geq 2, \end{cases} \tag{3.5}$$

while that of applying F *or* F^{-1} *is*

$$t_{a/s} = O(2^{dL}m^{2d}) + \sum_{\ell=0}^{L} 2^{d(L-\ell)}O(k_\ell^2) = \begin{cases} O(N), & d = 1, \\ O(N \log N), & d = 2, \\ O(N^{2(1-1/d)}), & d \geq 3. \end{cases} \tag{3.6}$$

PROOF. Consider first the factorization cost t_f. There are $2^{d(L-\ell)}$ cells at level ℓ, where each cell $c \in C_\ell$ requires the calculation of an ID of Y_c in (3.3) as well as various local matrix operations at a total cost of $O(|c|^3)$, assuming that $|c^{\mathsf{N}}| = O(|c|)$. But $|c| = m^d$ for $\ell = 0$, while $|c| = O(k_{\ell-1}) = O(k_\ell)$ for $\ell \geq 1$ since the active DOFs c are obtained by merging the skeletons of 2^d cells at level $\ell - 1$. Hence (3.5) follows.

A similar derivation holds for $t_{a/s}$ by observing that each $c \in C_\ell$ requires local matrix-vector products with cost $O(|c|^2)$. □

Remark 3.5. If a tree is used, then there is also a cost of $O(N \log N)$ for tree construction, but the associated constant is tiny, and so we can ignore it for all practical purposes.

The memory cost to store F itself is clearly $m_f = O(t_{a/s})$ and so is also given by (3.6). From Theorem 3.4, it is immediate that RSF behaves just like MF, with the geometric growth of k_ℓ in 2D and 3D leading to suboptimal complexities.

COROLLARY 3.6. *If*

$$k_\ell = O(k\ell) \tag{3.7}$$

for some constant k*, then* $t_f = O(Nk^2)$ *and* $t_{a/s} = O(Nk)$.

PROOF. From (3.5), $t_f = O(2^{dL}(m^d + k)^3)$, so choosing $m^d = O(k)$ gives $N = n^d = (2^L m)^d = O(2^{dL}k)$ and $t_f = O(2^{dL}k^3) = O(Nk^2)$. Similarly, $t_{a/s} = O(2^{dL}(m^d + k)^2) = O(2^{dL}k^2) = O(Nk)$. □

This is a more precise version of the 1D result that will be useful later when discussing HIF-IE.

4 Hierarchical Interpolative Factorization

In this section, we present HIF-IE, which builds upon RSF by introducing additional levels of skeletonization in order to effectively reduce all problems to 1D. Considering the 2D case for concreteness, the main idea is simply to employ an additional level $\ell + \frac{1}{2}$ after each level ℓ by partitioning Ω according to the cell

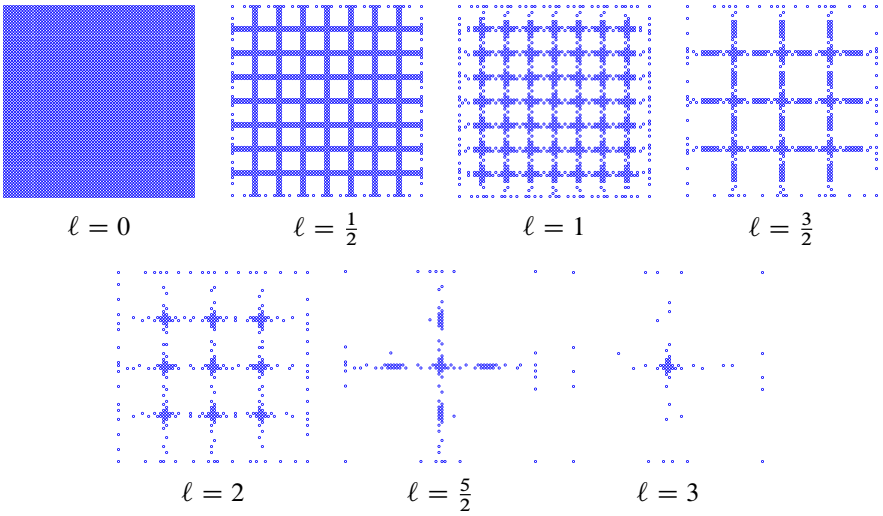

FIGURE 4.1. Active DOFs at each level ℓ of HIF-IE in 2D.

edges near which the surviving active DOFs cluster. This fully exploits the 1D geometry of the active DOFs. However, the algorithm is complicated by the fact that the cell and edge partitions are nonnested, so different index groups may now interact via SCIs. Such SCIs do not lend themselves easily to analysis, and we have yet to prove a statement like (3.4) on their ranks. Nevertheless, extensive numerical experiments by ourselves (Section 5) and others [16] reveal that very similar bounds appear to be obeyed. This suggests that SCIs do not need to be treated in any significantly different way, and we hereafter assume that interaction rank is completely determined by geometry.

The overall approach of HIF-IE is closely related to that of [16], but our sparsification framework permits a much simpler implementation and analysis. As with RSF, we begin first in 2D before extending to 3D.

4.1 Two Dimensions

Assume the same setup as in Section 3.1. HIF-IE supplements cell skeletonization (2D to 1D) at level ℓ with edge skeletonization (1D to 0D) at level $\ell + \frac{1}{2}$ for each $\ell = 0, 1, \ldots, L-1$. Figure 4.1 shows the active DOFs at each level for a representative example.

Level ℓ

Partition Ω into Voronoi cells about the cell centers $2^\ell mh(j_1 - \frac{1}{2}, j_2 - \frac{1}{2})$ for $1 \le j_1, j_2 \le 2^{L-\ell}$. Let $\mathcal{C}_\ell$ be the collection of index sets corresponding to the active DOFs of each cell. Skeletonization with respect to $\mathcal{C}_\ell$ then gives

$$A_{\ell+\frac{1}{2}} = \mathscr{Z}_{\mathcal{C}_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell, \qquad U_\ell = \prod_{c \in \mathcal{C}_\ell} Q_c R_{\check{c}}, \quad V_\ell = \prod_{c \in \mathcal{C}_\ell} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in \mathcal{C}_\ell} \check{c}$ have been eliminated.

Level $\ell + \frac{1}{2}$

Partition Ω into Voronoi cells about the edge centers $2^\ell mh(j_1, j_2 - \frac{1}{2})$ for $1 \le j_1 \le 2^{L-\ell} - 1$, $1 \le j_2 \le 2^{L-\ell}$, and $2^\ell mh(j_1 - \frac{1}{2}, j_2)$ for $1 \le j_1 \le 2^{L-\ell}$, $1 \le j_2 \le 2^{L-\ell} - 1$. Let $C_{\ell+1/2}$ be the collection of index sets corresponding to the active DOFs of each cell. Skeletonization with respect to $C_{\ell+1/2}$ then gives

$$A_{\ell+1} = \mathscr{Z}_{C_{\ell+1/2}}(A_{\ell+1/2}) \approx U_{\ell+1/2}^* A_{\ell+1/2} V_{\ell+1/2},$$
$$U_{\ell+1/2} = \prod_{c \in \boldsymbol{C}_{\ell+1/2}} Q_c R_{\check{c}}, \quad V_{\ell+1/2} = \prod_{c \in \boldsymbol{C}_{\ell+1/2}} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in \boldsymbol{C}_{\ell+1/2}} \check{c}$ have been eliminated.

Level L

Combining the approximation over all levels gives

$$D \equiv A_L \approx U_{L-1/2}^* \cdots U_{1/2}^* U_0^* A V_0 V_{1/2} \cdots V_{L-1/2},$$

so

$$A \approx U_0^{-*} U_{1/2}^{-*} \cdots U_{L-1/2}^{-*} D V_{L-1/2}^{-1} \cdots V_{1/2}^{-1} V_0^{-1} \equiv F, \tag{4.1a}$$

$$A^{-1} \approx V_0 V_{1/2} \cdots V_{L-1/2} D^{-1} U_{L-1/2}^* \cdots U_{1/2}^* U_0^* = F^{-1}. \tag{4.1b}$$

This is a factorization of exactly the same type as that in (3.2) (but with twice the number of factors). The entire procedure is summarized as Algorithm 4.1.

Algorithm 4.1 HIF-IE in 2D.

$A_0 = A$ ▷ initialize
for $\ell = 0, 1, \ldots, L-1$ **do** ▷ loop from finest to coarsest level
  $A_{\ell+1/2} = \mathscr{Z}_{C_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell$ ▷ skeletonize cells
  $A_{\ell+1} = \mathscr{Z}_{C_{\ell+1/2}}(A_{\ell+1/2}) \approx U_{\ell+1/2}^* A_{\ell+1/2} V_{\ell+1/2}$ ▷ skeletonize edges
end for
$A \approx U_0^{-*} U_{1/2}^{-*} \cdots U_{L-1/2}^{-*} A_L V_{L-1/2}^{-1} \cdots V_{1/2}^{-1} V_0^{-1}$ ▷ generalized LU decomposition

4.2 Three Dimensions

Assume the same setup as in Section 3.2. HIF-IE now performs two rounds of additional dimensional reduction over RSF by supplementing cell skeletonization (3D to 2D) at level ℓ with face skeletonization (2D to 1D) at level $\ell + \frac{1}{3}$ and edge skeletonization (1D to 0D) at level $\ell + \frac{2}{3}$. Figure 4.2 shows the active DOFs at each level for a representative example.

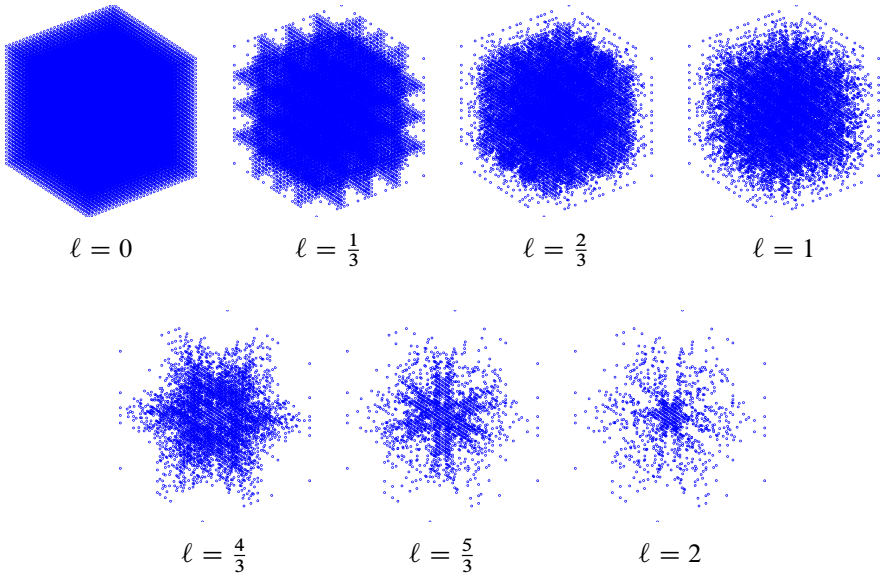

FIGURE 4.2. Active DOFs at each level ℓ of HIF-IE in 3D.

Level ℓ

Partition Ω into Voronoi cells about the cell centers $2^\ell mh(j_1-\frac{1}{2}, j_2-\frac{1}{2}, j_3-\frac{1}{2})$ for $1 \le j_1, j_2, j_3 \le 2^{L-\ell}$. Let C_ℓ be the collection of index sets corresponding to the active DOFs of each cell. Skeletonization with respect to C_ℓ then gives

$$A_{\ell+1/3} = \mathscr{Z}_{C_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell, \qquad U_\ell = \prod_{c \in C_\ell} Q_c R_{\check{c}}, \quad V_\ell = \prod_{c \in C_\ell} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_\ell} \check{c}$ have been eliminated.

Level $\ell + \frac{1}{3}$

Partition Ω into Voronoi cells about the face centers

$$\begin{aligned}
&2^\ell mh\left(j_1, j_2 - \frac{1}{2}, j_3 - \frac{1}{2}\right), && 1 \le j_1 \le 2^{L-\ell} - 1, \quad 1 \le j_2, j_3 \le 2^{L-\ell},\\
&2^\ell mh\left(j_1 - \frac{1}{2}, j_2, j_3 - \frac{1}{2}\right), && 1 \le j_2 \le 2^{L-\ell} - 1, \quad 1 \le j_1, j_3 \le 2^{L-\ell},\\
&2^\ell mh\left(j_1 - \frac{1}{2}, j_2 - \frac{1}{2}, j_3\right), && 1 \le j_3 \le 2^{L-\ell} - 1, \quad 1 \le j_1, j_2 \le 2^{L-\ell}.
\end{aligned}$$

Let $C_{\ell+1/3}$ be the collection of index sets corresponding to the active DOFs of each cell. Skeletonization with respect to $C_{\ell+1/3}$ then gives

$$A_{\ell+2/3} = \mathscr{Z}_{C_{\ell+1/3}}(A_{\ell+1/3}) \approx U_{\ell+1/3}^* A_{\ell+1/3} V_{\ell+1/3},$$
$$U_{\ell+1/3} = \prod_{c \in C_{\ell+1/3}} Q_c R_{\check{c}}, \quad V_{\ell+1/3} = \prod_{c \in C_{\ell+1/3}} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_{\ell+1/3}} \check{c}$ have been eliminated.

Level $\ell + \frac{2}{3}$

Partition Ω into Voronoi cells about the edge centers

$$2^\ell mh\left(j_1, j_2, j_3 - \frac{1}{2}\right), \qquad 1 \le j_1, j_2 \le 2^{L-\ell} - 1, \qquad 1 \le j_3 \le 2^{L-\ell},$$
$$2^\ell mh\left(j_1, j_2 - \frac{1}{2}, j_3\right), \qquad 1 \le j_1, j_3 \le 2^{L-\ell} - 1, \qquad 1 \le j_2 \le 2^{L-\ell},$$
$$2^\ell mh\left(j_1 - \frac{1}{2}, j_2, j_3\right), \qquad 1 \le j_2, j_3 \le 2^{L-\ell} - 1, \qquad 1 \le j_1 \le 2^{L-\ell}.$$

Let $C_{\ell+2/3}$ be the collection of index sets corresponding to the active DOFs of each cell. Skeletonization with respect to $C_{\ell+2/3}$ then gives

$$A_{\ell+1} = \mathscr{Z}_{C_{\ell+2/3}}(A_{\ell+2/3}) \approx U_{\ell+2/3}^* A_{\ell+2/3} V_{\ell+2/3},$$
$$U_{\ell+2/3} = \prod_{c \in C_{\ell+2/3}} Q_c R_{\check{c}}, \quad V_{\ell+2/3} = \prod_{c \in C_{\ell+2/3}} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_{\ell+2/3}} \check{c}$ have been eliminated.

Level L

Combining the approximation over all levels gives

$$D \equiv A_L \approx U_{L-1/3}^* \cdots U_{2/3}^* U_{1/3}^* U_0^* A V_0 V_{1/3} V_{2/3} \cdots V_{L-1/3},$$

so

$$A \approx U_0^{-*} U_{1/3}^{-*} U_{2/3}^{-*} \cdots U_{L-1/3}^{-*} D V_{L-1/3}^{-1} \cdots V_{2/3}^{-1} V_{1/3}^{-1} V_0^{-1} \equiv F, \tag{4.2a}$$

$$A^{-1} \approx V_0 V_{1/3} V_{2/3} \cdots V_{L-1/3} D^{-1} U_{L-1/3}^* \cdots U_{2/3}^* U_{1/3}^* U_0^* = F^{-1}. \tag{4.2b}$$

This procedure is summarized as Algorithm 4.2.

4.3 Accelerated Compression

Proxy compression still applies, provided that we make some minor modifications to account for SCIs, which we generally have access to only numerically and so cannot evaluate at arbitrary points as needed in Lemma 3.3. Specifically, for a given index set c, we now expand c^{N} by including all DOFs that interact with c via SCIs in addition to those interior to Γ as in Section 3.3. The far field $c^{\mathsf{F}} = c^{\mathsf{c}} \setminus c^{\mathsf{N}}$ then consists only of original kernel interactions, so Lemma 3.3 holds.

Algorithm 4.2 HIF-IE in 3D.

$A_0 = A$ ▷ initialize
for $\ell = 0, 1, \ldots, L-1$ **do** ▷ loop from finest to coarsest level
  $A_{\ell+1/3} = \mathscr{Z}_{C_\ell}(A_\ell) \approx U_\ell^* A_\ell V_\ell$ ▷ skeletonize cells
  $A_{\ell+2/3} = \mathscr{Z}_{C_{\ell+1/3}}(A_{\ell+1/3}) \approx U_{\ell+1/3}^* A_{\ell+1/3} V_{\ell+1/3}$ ▷ skeletonize faces
  $A_{\ell+1} = \mathscr{Z}_{C_{\ell+2/3}}(A_{\ell+2/3}) \approx U_{\ell+2/3}^* A_{\ell+2/3} V_{\ell+2/3}$ ▷ skeletonize edges
end for
$A \approx U_0^{-*} U_{1/3}^{-*} \cdots U_{L-1/3}^{-*} A_L V_{L-1/3}^{-1} \cdots V_{1/3}^{-1} V_0^{-1}$ ▷ generalized LU decomposition

It remains to observe that SCIs are local due to the domain partitioning strategy. Thus, all c^{N} reside in an immediate neighborhood of c and we again conclude that $|c^{\mathsf{N}}| = O(|c|)$.

Even with this acceleration, however, the ID still manifests as a computational bottleneck. To combat this, we also tried fast randomized methods [37] based on compressing $\Phi_c Y_c$, where Φ_c is a small Gaussian random sampling matrix. We found that the resulting ID was inaccurate when Y_c contained SCIs. This could be remedied by considering instead $\Phi_c (Y_c Y_c^*)^\gamma Y_c$ for some small integer $\gamma = 1, 2, \ldots$, but the expense of the extra multiplications usually outweighed any efficiency gains.

4.4 Modifications for Second-Kind Integral Equations

The algorithms presented so far are highly accurate for first-kind IEs in that $\|A - F\|/\|A\| = O(\varepsilon)$, where ε is the input precision to the ID (Section 5). For second-kind IEs, however, we see a systematic deterioration of the relative error roughly as $O(N\varepsilon)$ as $N \to \infty$. This instability can be explained as follows. Let A be a typical second-kind IE matrix discretization. Then the diagonal entries of A are $O(1)$, while its off-diagonal entries are $O(1/N)$. Since the interpolation matrix, say, T_p, from the ID has entries of order $O(1)$, the same is true of $B_{\check{p}\check{p}}$, $B_{\check{p}\hat{p}}$, and $B_{\hat{p}\check{p}}$ in (2.5). Therefore, the entries of the Schur complement $B_{\hat{p}\hat{p}}$ in (2.6) are $O(1)$; i.e., SCIs dominate kernel interactions by a factor of $O(N)$.

LEMMA 4.1. *Assume the setting of the discussion above and let $c \in C_\ell$ be such that Y_c in* (3.3) *contains SCIs. Then $\|Y_c\| = O(1)$, so the ID of Y_c has absolute error $\|E_c\| = O(\varepsilon)$.*

Consider now the process of "unfolding" the factorization F from the middle matrix $D \equiv A_L$ outward. This is accomplished by undoing the skeletonization operation for each $c \in C_\ell$ in reverse order, at each step reconstructing $(A_\ell)_{:,\check{c}}$ and $(A_\ell)_{\check{c},:}$ from $(A_{\ell+1/d})_{:,\hat{c}}$ and $(A_{\ell+1/d})_{\hat{c},:}$. Restricting attention to 2D for concreteness, we start at level L with interactions between the DOFs s_L as depicted in Figure 4.3 (left). By Lemma 4.1, unskeletonizing each edge $c \in C_{L-1/2}$ induces an error in the interactions between the edges e_1 and e_2 as labeled in the figure

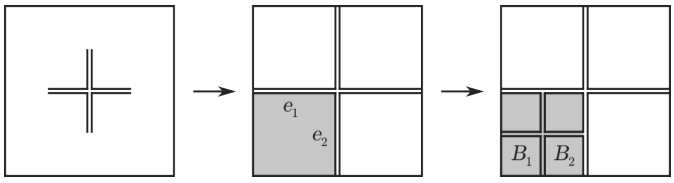

FIGURE 4.3. Matrix reconstruction from skeleton-skeleton interactions.

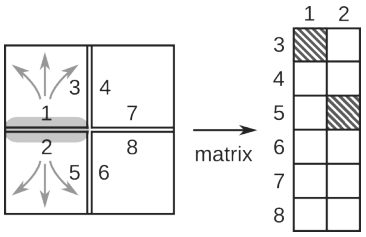

FIGURE 4.4. Sparsity pattern of SCIs. A reference domain configuration (left) is shown with each half-edge labeled from 1 to 8. The edge of interest (1 and 2) is outlined in gray along with all outgoing SCIs. The corresponding matrix view (right) shows these interactions (hatched) indexed by half-edge.

(center) of absolute magnitude $O(\varepsilon)$. At the next level, unskeletonizing the shaded cell $c \in C_{L-1}$ that they bound then relies on the approximate interactions between e_1 and e_2. This spreads the $O(\varepsilon)$ error over the reconstructed cell interactions, which is small for SCIs acting internally to each cell $c \in C_{L-2}$ (omitting level $L - \frac{3}{2}$ for simplicity) but not for kernel interactions between any two distinct cells B_1 and B_2 (right); indeed, the relative error for the latter is $O(N\varepsilon)$. These corrupted interactions are then used for reconstruction at the next level and are eventually spread throughout the whole matrix. The same argument clearly holds in 3D.

This analysis suggests that the only fix is to skeletonize at effective precision $O(\varepsilon/N)$ so that kernel interactions are accurately reconstructed. This is equivalent to ensuring that both scales in Y_c are well approximated by the ID. Following this intuition, we decompose Y_c as $Y_c = Y_c^{\mathsf{K}} + Y_c^{\mathsf{S}}$, where Y_c^{K} consists purely of kernel interactions, and set $\rho_c\varepsilon$ for $\rho_c = \min(1, \|Y_c^{\mathsf{K}}\|/\|Y_c^{\mathsf{S}}\|)$ as the local compression tolerance, which we note uses increased precision only when necessary.

The two-scale structure of Y_c also enables an additional optimization as can be seen by studying the sparsity patterns of SCIs. Figure 4.4 shows an example configuration in 2D after cell skeletonization at level ℓ, which leaves a collection of edges at level $\ell + \frac{1}{2}$, each composed of two half-edges consisting of skeletons from the two cells on either side (left). Let $c = g_1 \cup g_2 \in C_{\ell+1/2}$ be a given edge with indices partitioned by half-edge, and let Y_{g_j} be the submatrix of Y_c corresponding to g_j. Then $Y_{g_1}^{\mathsf{S}}$ and $Y_{g_2}^{\mathsf{S}}$ (analogously defined) have different nonzero structures, so Y_{g_1} and Y_{g_2} have large entries in different row blocks (right). The

stable interpolation of Y_c hence requires that all interpolation coefficients from one half-edge to the other be $O(1/N)$ since otherwise the reconstruction of, say, Y_{g_1} will have large errors in rows where $Y^{\mathsf{S}}_{g_2}$ is nonzero. As $N \to \infty$, these cross-interpolation coefficients must therefore vanish and the compression of Y_c decouples into the compression of Y_{g_1} and Y_{g_2} separately. We enforce this asymptotic decoupling explicitly, which moreover provides an acceleration due to the cubic cost of the ID. The ID of Y_c is then given by $\hat{c} = (\hat{g}_1, \hat{g}_2)$, $\check{c} = (\check{g}_1, \check{g}_2)$, and $T_c = \operatorname{diag}(T_{g_1}, T_{g_2})$, where $g_j = \hat{g}_j \cup \check{g}_j$ and T_{g_j} define the ID of Y_{g_j}. We use the compression tolerance $\rho_{g_j}\varepsilon$ with $\rho_{g_j} = \min(1, \|Y^{\mathsf{K}}_{g_j}\|/\|Y^{\mathsf{S}}_{g_j}\|)$ locally for each g_j.

In general, we define the subsets $\{g_j\}$ algebraically according to the sparsity pattern of Y^{S}_c, which can be done using the matrix indicator function

$$(\mathscr{S}(A))_{ij} = \begin{cases} 0, & A_{ij} = 0, \\ 1, & A_{ij} \neq 0. \end{cases}$$

LEMMA 4.2. *Let $B = \mathscr{S}(A)^*\mathscr{S}(A)$ for some matrix A. Then $A_{:,i}$ and $A_{:,j}$ have the same sparsity pattern if and only $B_{ij} = \max(\|A_{:,i}\|_0, \|A_{:,j}\|_0)$.*

4.5 Complexity Estimates

Analysis of HIF-IE is impeded by the compression of SCIs, for which we do not have rigorous bounds on the interaction rank. Nonetheless, ample numerical evidence suggests that SCIs behave very similarly to standard kernel interactions. For the sake of analysis, we hence assume that the same rank estimates apply, from which we have (3.7) for all $\ell \geq 1$ by reduction to 1D. We emphasize that this has yet to be proven, so all following results should formally be understood as conjectures, albeit ones with strong experimental support (Section 5).

THEOREM 4.3. *Assume that* (3.7) *holds. Then the cost of constructing the factorization F in* (4.1) *or* (4.2) *using HIF-IE with accelerated compression is $t_f = O(N)$, while that of applying F or F^{-1} is $t_{a/s} = O(N)$.*

PROOF. This is essentially just a restatement of Corollary 3.6 (but with the sum now taken also over fractional levels). □

COROLLARY 4.4. *For second-kind IEs,*

$$t_f = \begin{cases} O(N \log N), & d = 2, \\ O(N \log^6 N), & d = 3, \end{cases} \qquad t_{a/s} = \begin{cases} O(N \log\log N), & d = 2, \\ O(N \log^2 N), & d = 3. \end{cases}$$

PROOF. According to the modifications of Section 4.4, there are now two effective ID tolerances: ε for all $c \in C_\ell$ such that $Y^{\mathsf{S}}_c = 0$ and $O(\varepsilon/N)$ otherwise. The former is used for all initial levels $\ell \leq \lambda$ before SCIs have become widespread (i.e., before any meaningful dimensional reduction has occurred), and the latter for all $\ell > \lambda$. But using precision $O(\varepsilon/N)$ yields a rank estimate with constant of

proportionality $O(\log^\delta N)$, where δ is the intrinsic dimension of the DOF cluster c [28, 29], so the amount of compression depends on N. Thus, $\lambda = \lambda(N)$ and our first task is to determine its form.

The crossover level λ can be obtained by balancing the typical size $|c|$ of an edge (2D and 3D) or face (3D only) with its skeleton size $|\widehat{c}|$. In 2D, this is $2^\lambda \sim \lambda \log N$, where the left-hand side gives the size of an edge at level λ, and the right-hand side the estimated rank for SCI compression. Therefore, $\lambda \sim \log\log N$.

In 3D, there are two crossover levels λ_1 and λ_2 corresponding to face and edge compression, respectively, with $\lambda = \max(\lambda_1, \lambda_2)$:

$$2^{2\lambda_1} \sim 2^{\lambda_1} \log^2 N, \quad 2^{\lambda_2} \sim \lambda_2 \log N.$$

Hence, $\lambda_1 \sim 2\log\log N$ and $\lambda_2 \sim \log\log N$, so $\lambda \sim 2\log\log N$.

The cost of constructing F for second-kind IEs is then

$$t_f = O(2^{dL} m^{3d}) + \sum_{\ell=0}^{\lambda} 2^{d(L-\ell)} O(2^{3(d-1)\ell}) + {\sum_{\ell=\lambda}^{L}}' O(2^{d(L-\ell)} k_\ell^3),$$

where prime notation denotes summation over all levels, both integer and fractional, and k_ℓ is as given in (3.7) with $k = O(\log N)$. The first sum corresponds to running RSF on the initial levels and reduces to

$$\sum_{\ell=0}^{\lambda} 2^{d(L-\ell)} O(2^{3(d-1)\ell}) = \begin{cases} O(N \log N), & d = 2, \\ O(N \log^6 N), & d = 3, \end{cases}$$

while the second can be interpreted as the cost of the standard HIF-IE (without modification) applied to the remaining

$$O(2^{-\lambda} N) = \begin{cases} O(N/\log N), & d = 2, \\ O(N/\log^2 N), & d = 3, \end{cases}$$

DOFs at uniform precision $O(\varepsilon/N)$. By Corollary 3.6, this is

$${\sum_{\ell=\lambda}^{L}}' O(2^{d(L-\ell)} k_\ell^3) = \begin{cases} O(N \log N), & d = 2, \\ O(N), & d = 3, \end{cases}$$

so, adding all terms, we derive t_f as claimed.

A similar argument for

$$t_{a/s} = O(2^{dL} m^{2d}) + \sum_{\ell=0}^{\lambda} 2^{d(L-\ell)} O(2^{2(d-1)\ell}) + {\sum_{\ell=\lambda}^{L}}' O(2^{d(L-\ell)} k_\ell^2)$$

completes the proof. □

Remark 4.5. Like Theorem 3.4 for RSF, the parameter d in Corollary 4.4 can also be regarded as the intrinsic dimension.

5 Numerical Results

In this section, we demonstrate the efficiency of HIF-IE by reporting numerical results for some benchmark problems in 2D and 3D. All algorithms and examples were implemented in MATLAB® and are freely available at `https://github.com/klho/FLAM/`. In what follows, we refer to RSF as rskelf2 in 2D and rskelf3 in 3D. Similarly, we call HIF-IE hifie2 and hifie3, respectively, with hifie2x and hifie3x denoting their second-kind IE counterparts. All codes are fully adaptive and built on quadtrees in 2D and octrees in 3D. The average block size $|c|$ at level 0 (and hence the tree depth L) was chosen so that $|c| \sim 2|\widehat{c}|$. In select cases, the first few fractional levels of HIF-IE were skipped to optimize the running time. Symmetry was exploited wherever possible by compressing

$$Y_c' = \begin{bmatrix} A_{c^{\mathsf{N}},c} \\ Y_{c^{\mathsf{E}},c} \end{bmatrix}$$

instead of the full matrix Y_c in (3.3), which reduces the cost by about a factor of 2. Diagonal blocks, i.e., A_{pp} in Lemma 2.1, were factored using the (partially pivoted) LDL decomposition if A is symmetric and the LU decomposition otherwise.

For each example, the following, if applicable, are given:

- ε: base relative precision of the ID;
- N: total number of DOFs in the problem;
- $|s_L|$: number of active DOFs remaining at the highest level;
- t_f: wall clock time for constructing the factorization F in seconds;
- m_f: memory required to store F in GB;
- $t_{a/s}$: wall clock time for applying F or F^{-1} in seconds;
- e_a: a posteriori estimate of $\|A - F\|/\|A\|$ (see below);
- e_s: a posteriori estimate of $\|I - AF^{-1}\| \geq \|A^{-1} - F^{-1}\|/\|A^{-1}\|$;
- n_i: number of iterations to solve (1.6) using GMRES with preconditioner F^{-1} to a tolerance of 10^{-12}, where f is a standard uniform random vector (ill-conditioned systems only).

The operator errors e_a and e_s were estimated using power iteration with a standard uniform random start vector [18, 42] and a convergence criterion of 10^{-2} relative precision in the matrix norm. This procedure requires the application of both A and A^*, which for translation-invariant kernels was done using fast Fourier convolution [9] and for non-translation-invariant kernels using an ID-based kernel-independent FMM [44, 46] at precision 10^{-15}. The same methods were also used to apply A when solving (1.6) iteratively.

For simplicity, all IEs were discretized using a piecewise-constant collocation method as in Section 3. Certain near-field interactions (to be defined for each case) were computed using adaptive quadrature, while all other interactions were handled as simple one-point approximations, e.g., $K_{ij} = K(\|x_i - x_j\|)h^d$ in (3.1).

TABLE 5.1. Factorization results for Example 1.

		rskelf2			hifie2		
ε	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-3}	512^2	2058	1.9e+2	7.7e−1	67	6.2e+1	3.0e−1
	1024^2	4106	1.4e+3	3.6e+0	67	2.5e+2	1.2e+0
	2048^2	6270	6.6e+3	1.4e+1	70	1.0e+3	4.7e+0
10^{-6}	512^2	3430	7.7e+2	1.8e+0	373	2.7e+2	8.5e−1
	1024^2	5857	4.6e+3	7.7e+0	428	1.2e+3	3.5e+0
	2048^2	11317	3.0e+4	3.3e+1	455	4.8e+3	1.4e+1
10^{-9}	512^2	4162	1.2e+3	2.3e+0	564	4.3e+2	1.2e+0
	1024^2	8264	1.0e+4	1.1e+1	686	2.1e+3	4.8e+0
	2048^2	16462	8.3e+4	5.2e+1	837	9.1e+3	1.9e+1

TABLE 5.2. Matrix application results for Example 1.

		rskelf2	hifie2			
ε	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i
10^{-3}	512^2	7.2e−1	5.2e−1	3.4e−04	1.2e−1	9
	1024^2	3.2e+0	2.1e+0	3.8e−04	1.6e−1	10
	2048^2	1.3e+1	1.2e+1	4.3e−04	1.6e−1	10
10^{-6}	512^2	9.2e−1	9.7e−1	3.8e−07	5.0e−4	3
	1024^2	4.2e+0	4.1e+0	3.3e−07	6.5e−4	4
	2048^2	2.1e+1	1.5e+1	5.0e−07	4.1e−4	4
10^{-9}	512^2	1.1e+0	8.1e−1	2.8e−10	4.3e−7	2
	1024^2	4.9e+0	3.5e+0	2.7e−10	6.8e−7	2
	2048^2	2.8e+1	1.4e+1	5.7e−10	1.1e−6	2

All computations were performed in MATLAB® R2010b on a single core (without parallelization) of an Intel Xeon E7-4820 CPU at 2.0 GHz on a 64-bit Linux server with 256 GB of RAM.

5.1 Two Dimensions

We begin first in 2D, where we present three examples.

Example 1. Consider (1.1) with $a(x) \equiv 0$, $b(x) \equiv c(x) \equiv 1$, $K(r) = -\frac{1}{2\pi}\log r$, and $\Omega = (0,1)^2$, i.e., a first-kind volume IE in the unit square, discretized over a uniform $n \times n$ grid. The diagonal entries K_{ii} are computed adaptively, while all K_{ij} for $i \neq j$ are approximated using one-point quadratures. We factored the resulting matrix A using both rskelf2 and hifie2 at $\varepsilon = 10^{-3}$, 10^{-6}, and 10^{-9}. The data are summarized in Tables 5.1 and 5.2 with scaling results shown in Figure 5.1.

It is evident that $|s_L| \sim k_L$ behaves as predicted, with HIF-IE achieving significant compression over RSF. Consequently, we find strong support for asymptotic complexities consistent with Theorems 3.4 and 4.3. For all problem sizes tested, t_f and m_f are always smaller for HIF-IE, though $t_{a/s}$ is quite comparable. This is

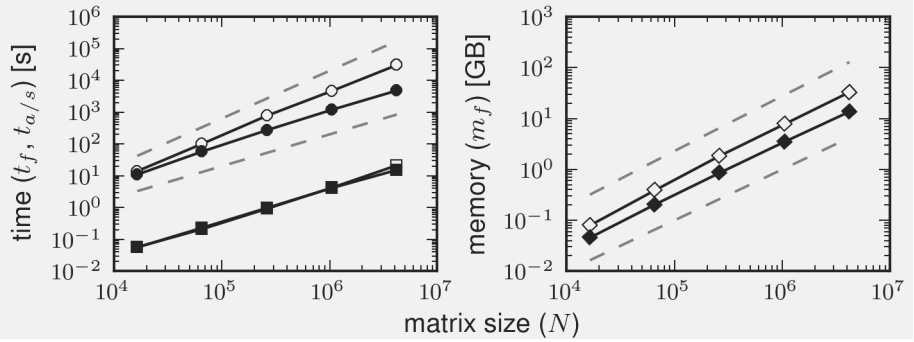

FIGURE 5.1. Scaling results for Example 1. Wall clock times t_f (○) and $t_{a/s}$ (□) and storage requirements m_f (◇) are shown for rskelf2 (white) and hifie2 (black) at precision $\varepsilon = 10^{-6}$. Included also are reference scalings (gray dashed lines) of $O(N)$ and $O(N^{3/2})$ (left, from bottom to top), and $O(N)$ and $O(N \log N)$ (right). The lines for $t_{a/s}$ (bottom left) lie nearly on top of each other.

because $t_{a/s}$ is dominated by memory access (at least in our current implementation), which also explains its relative insensitivity to ε. Furthermore, we observe that $t_{a/s} \ll t_f$ for both methods, which makes them ideally suited to systems involving multiple right-hand sides.

The forward approximation error $e_a = O(\varepsilon)$ for all N and seems to increase only very mildly, if at all, with N. This indicates that the local accuracy of the ID provides a good estimate of the overall accuracy of the algorithm, which is not easy to prove since the multilevel matrix factors constituting F are not unitary. On the other hand, we expect the inverse approximation error to scale as $e_s = O(\kappa(A)e_a)$, where $\kappa(A) = \|A\|\|A^{-1}\|$ is the condition number of A, and indeed we see that e_s is much larger due to the ill-conditioning of the first-kind system. When using F^{-1} to precondition GMRES, however, the number of iterations required is always very small. This indicates that F^{-1} is a highly effective preconditioner.

Example 2. Consider now the same setup as in Example 1 but with $a(x) \equiv 1$. This gives a well-conditioned second-kind IE, which we factored using rskelf2, hifie2, and hifie2x. The data are summarized in Tables 5.3 and 5.4 with scaling results in Figure 5.2.

As expected, results for rskelf2 are essentially the same as those in Example 1 since the off-diagonal interactions at each level are identical. We also see the breakdown of hifie2, which still has linear complexity but fails to properly approximate A as predicted in Section 4.4. This is remedied by hifie2x, which achieves $e_a, e_s = O(\varepsilon)$ but with a slight increase in cost. In particular, it appears to scale somewhat faster than linearly but remains consistent with Corollary 4.4.

TABLE 5.3. Factorization results for Example 2.

		rskelf2			hifie2			hifie2x		
ε	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-3}	512^2	2058	1.9e+2	7.7e−1	108	6.8e+1	3.5e−1	376	1.1e+2	5.1e−1
	1024^2	4106	1.4e+3	3.6e+0	135	2.8e+2	1.4e+0	456	5.3e+2	2.2e+0
	2048^2	6270	6.6e+3	1.4e+1	172	1.2e+3	5.7e+0	522	2.4e+3	9.4e+0
10^{-6}	512^2	3430	7.7e+2	1.8e+0	475	2.2e+2	8.8e−1	804	4.7e+2	1.4e+0
	1024^2	5857	4.7e+3	7.7e+0	580	9.1e+2	3.4e+0	962	2.2e+3	5.7e+0
	2048^2	11317	3.0e+4	3.3e+1	614	3.6e+3	1.4e+1	1115	9.6e+3	2.3e+1
10^{-9}	512^2	4162	1.2e+3	2.3e+0	1030	6.4e+2	1.5e+0	1087	6.7e+2	1.7e+0
	1024^2	8264	1.0e+4	1.1e+1	1241	3.2e+3	6.3e+0	1381	3.6e+3	7.2e+0
	2048^2	16462	8.2e+4	5.2e+1	1583	1.5e+4	2.6e+1	1697	1.8e+4	3.1e+1

TABLE 5.4. Matrix application results for Example 2.

		rskelf2	hifie2			hifie2x		
ε	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	$t_{a/s}$	e_a	e_s
10^{-3}	512^2	7.2e−1	5.4e−1	7.8e−2	8.5e−2	5.3e−1	2.6e−04	2.9e−4
	1024^2	3.3e+0	2.3e+0	8.3e−2	9.1e−2	2.4e+0	2.7e−04	3.0e−4
	2048^2	1.1e+1	1.2e+1	9.8e−2	1.1e−1	1.2e+1	8.0e−04	8.7e−4
10^{-6}	512^2	1.2e+0	9.6e−1	4.1e−4	4.4e−4	1.0e+0	5.9e−07	6.7e−7
	1024^2	5.1e+0	3.3e+0	8.2e−4	9.0e−4	4.5e+0	9.3e−07	1.0e−6
	2048^2	1.8e+1	1.2e+1	3.7e−3	4.1e−3	1.7e+1	1.6e−06	1.8e−6
10^{-9}	512^2	1.4e+0	8.9e−1	3.0e−7	3.4e−7	1.2e+0	2.8e−10	3.2e−10
	1024^2	5.4e+0	3.7e+0	8.4e−7	9.6e−7	5.0e+0	3.5e−10	3.9e−10
	2048^2	2.5e+1	1.5e+1	1.8e−6	2.0e−6	1.8e+1	1.1e−09	1.2e−09

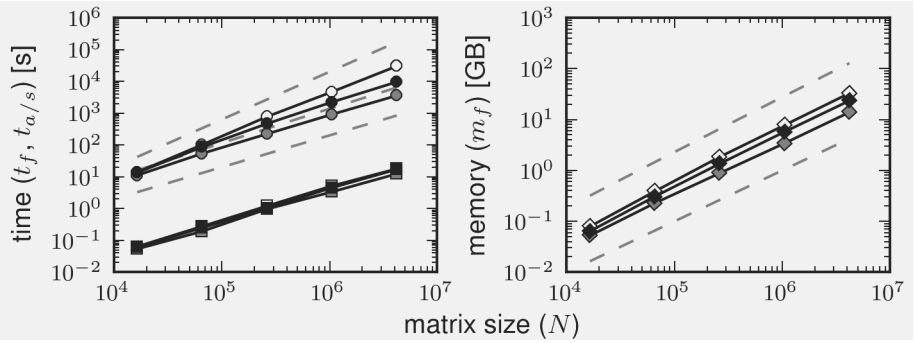

FIGURE 5.2. Scaling results for Example 2, comparing rskelf2 (white), hifie2 (gray), and hifie2x (black) at precision $\varepsilon = 10^{-6}$. Included also are reference scalings of $O(N)$, $O(N \log N)$, and $O(N^{3/2})$ (left); and $O(N)$ and $O(N \log N)$ (right). All other notation as in Figure 5.1.

TABLE 5.5. Factorization results for Example 3.

			rskelf2			hifie2			hifie2x		
ε	N	κ	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
	256^2	8	1522	8.3e+2	8.5e−1	551	7.8e+2	6.8e−1	592	8.4e+2	7.2e−1
10^{-6}	512^2	16	2995	5.0e+3	4.4e+0	860	4.0e+3	3.0e+0	825	4.3e+3	3.4e+0
	1024^2	32	5918	3.0e+4	2.2e+1	1331	1.8e+4	1.3e+1	1229	2.0e+4	1.5e+1

TABLE 5.6. Matrix application results for Example 3.

			rskelf2	hifie2				hifie2x			
ε	N	κ	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i	$t_{a/s}$	e_a	e_s	n_i
	256^2	8	4.1e−1	3.5e−1	1.8e−4	8.5e−4	3	4.6e−1	7.7e−6	3.9e−5	3
10^{-6}	512^2	16	2.4e+0	1.6e+0	8.8e−4	5.8e−3	6	2.1e+0	1.8e−5	1.7e−4	3
	1024^2	32	1.2e+1	8.3e+0	5.5e−3	5.7e−2	9	9.3e+0	6.5e−5	9.6e−4	3

Example 3. We then turn to the Lippmann-Schwinger equation

$$\sigma(x) + k^2 \int_\Omega K(\|x-y\|)\omega(y)\sigma(y)d\Omega(y) = f(x), \quad x \in \Omega = (0,1)^2,$$

for Helmholtz scattering, where $k = 2\pi\kappa$ is the frequency of the incoming wave with κ the number of wavelengths in Ω; $K(r) = (i/4)H_0^{(1)}(kr)$ is the fundamental solution of the associated Helmholtz equation satisfying the Sommerfeld radiation condition, where i is the imaginary unit and $H_0^{(1)}(\cdot)$ is the zeroth-order Hankel function of the first kind; and $\omega(x)$ is a continuous function representing the scatterer. We refer the interested reader to [15] for details. Assuming that $\omega(x) \geq 0$, this can be symmetrized by the change of variables $u(x) = \sqrt{\omega(x)}\sigma(x)$ as

$$u(x) + k\sqrt{\omega(x)} \int_\Omega K(\|x-y\|)[k\sqrt{\omega(y)}]u(y)d\Omega(y) = \sqrt{\omega(x)}\, f(x), \quad x \in \Omega, \tag{5.1}$$

i.e., (1.1) with $a(x) \equiv 1$ and $b(x) \equiv c(x) = k\sqrt{\omega(x)}$. We took a Gaussian bump $\omega(x) = \exp(-32(x-x_0)^2)$ for $x_0 = (\frac{1}{2}, \frac{1}{2})$ as the scatterer and discretized (5.1) using a uniform grid with quadratures as computed in Example 1. The frequency k was increased with $n = \sqrt{N}$ to keep the number of DOFs per wavelength fixed at 32. Data for rskelf2, hifie2, and hifie2x with $\kappa = 8$, 16, and 32 at $\varepsilon = 10^{-6}$ are shown in Tables 5.5 and 5.6.

Overall, the results are similar to those in Example 2 but with added computational expense due to working over $\mathbb{C}$ and computing $H_0^{(1)}(kr)$. Moreover, although (5.1) is formally a second-kind IE, it becomes increasingly first-kind as $k \to \infty$. Thus, the problem is somewhat ill-conditioned, as reflected in the deterioration of e_a and e_s even for hifie2x. Nevertheless, F^{-1} remains a very good preconditioner, with $n_i = O(1)$ for hifie2x. Interestingly, despite its inaccuracy,

hifie2 is also quite effective for preconditioning: experimentally, we observe that $n_i = O(\log N)$, which can be justified as follows.

LEMMA 5.1. *If $A = I + E$ with $\varepsilon = \|E\|$, then the number of iterations for GMRES to solve* (1.6) *to any target precision $\varepsilon_0 > 0$ is $n_i \leq \log_\varepsilon \varepsilon_0$.*

PROOF. Let u_k be the k^{th} iterate with residual $r_k = Au_k - f$. Then the relative residual satisfies

$$\frac{\|r_k\|}{\|f\|} \leq \min_{p \in \mathscr{P}_k} \|p(A)\|,$$

where $\mathscr{P}_k$ is the set of all polynomials p of degree no greater than k such that $p(0) = 1$ [47]. Consider, in particular, the choice $p(z) = (1 - z)^k$. Then $\|p(A)\| \leq \|I - A\|^k = \|E\|^k = \varepsilon^k$, so $\|r_k\|/\|f\| \leq \varepsilon^k$. Setting the left-hand side equal to ε_0 yields $n_i \equiv k \leq \log_\varepsilon \varepsilon_0$. □

COROLLARY 5.2. *Let $F = A + E$ and $F^{-1} = A^{-1} + G$ with $\|E\| \leq CN\varepsilon\|A\|$ and $\|G\| \leq CN\varepsilon\kappa(A)\|A^{-1}\|$ for some constant C such that $CN\varepsilon\kappa(A) \ll 1$. Then the number of iterations for GMRES to solve* (1.6) *with preconditioner F^{-1} is*

$$n_i \sim \left(1 + \log_{1/\varepsilon} CN\kappa(A)\right) \log_\varepsilon \varepsilon_0.$$

PROOF. The preconditioned matrix is $F^{-1}A = F^{-1}(F - E) = I - F^{-1}E$, where

$$\|F^{-1}E\| \leq (\|A^{-1}\| + \|G\|)\|E\| \leq CN\varepsilon\kappa(A)(1 + CN\varepsilon\kappa(A)) \sim CN\varepsilon\kappa(A),$$

so Lemma 5.1 gives

$$\begin{aligned} n_i \sim \log_{CN\varepsilon\kappa(A)} \varepsilon_0 = \frac{\log \varepsilon_0}{\log CN\varepsilon\kappa(A)} &= \left(\frac{1}{1 + \log_\varepsilon CN\kappa(A)}\right) \frac{\log \varepsilon_0}{\log \varepsilon} \\ &= \left(\frac{1}{1 - \log_{1/\varepsilon} CN\kappa(A)}\right) \log_\varepsilon \varepsilon_0. \end{aligned}$$

But $CN\kappa(A) \ll 1/\varepsilon$, so $\log_{1/\varepsilon} CN\kappa(A) \ll 1$. The claim now follows by first-order expansion of the term in parentheses. □

We remark that HIF-IE is effective only at low to moderate frequency since the rank structures employed break down as $k \to \infty$. In the limit, the only compression possible is due to Green's theorem, with HIF-IE reducing to RSF for volume IEs. The situation is yet worse for boundary IEs, for which no compression at all is available in general, and both RSF and HIF-IE revert to having $O(N^3)$ complexity.

5.2 Three Dimensions

We next present three examples in 3D: a boundary IE and two volume IEs as in Examples 1 and 2.

TABLE 5.7. Factorization results for Example 4.

		rskelf3			hifie3			hifie3x		
ε	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-3}	20480	3843	3.3e+2	4.7e−1	1143	2.2e+2	2.2e−1	2533	3.5e+2	3.4e−1
	81920	7659	2.7e+3	2.2e+0	1247	7.3e+2	7.2e−1	3456	1.7e+3	1.3e+0
	327680	15091	2.0e+4	1.0e+1	1300	3.0e+3	2.9e+0	2875	7.4e+3	5.2e+0
	1310720	27862	1.4e+5	4.2e+1	1380	1.1e+4	1.1e+1	2934	2.6e+4	1.8e+1
10^{-6}	20480	6939	1.3e+3	1.2e+0	4976	1.2e+3	8.0e−1	6256	1.4e+3	1.1e+0
	81920	14295	1.5e+4	6.2e+0	8619	8.4e+3	3.2e+0	10748	9.5e+3	4.7e+0
	327680	28952	1.3e+5	3.1e+1	13782	5.0e+4	1.2e+1	13625	5.4e+4	1.9e+1

TABLE 5.8. Matrix application results for Example 4.

		rskelf3	hifie3			hifie3x		
ε	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	$t_{a/s}$	e_a	e_s
10^{-3}	20480	2.6e−1	1.8e−1	6.4e−3	1.0e−2	2.1e−1	3.8e−4	7.0e−4
	81920	1.2e+0	5.3e−1	4.0e−2	5.1e−2	6.7e−1	1.0e−3	1.8e−3
	327680	4.7e+0	1.9e+0	8.8e−2	1.1e−1	3.3e+0	4.2e−4	8.1e−4
	1310720	2.2e+1	7.2e+0	2.4e−1	3.3e−1	1.1e+1	6.0e−4	7.1e−4
10^{-6}	20480	5.6e−1	4.3e−1	3.7e−6	6.8e−6	4.9e−1	4.1e−7	8.0e−7
	81920	2.9e+0	1.8e+0	1.3e−5	2.4e−5	2.1e+0	3.7e−7	6.1e−7
	327680	1.5e+1	6.5e+0	5.6e−5	1.0e−4	1.1e+1	5.9e−7	1.0e−6

Example 4. Consider the second-kind boundary IE (1.5) on the unit sphere $\Gamma = S^2$, where $G(r)$ is as defined in (1.4). It is possible to reparametrize Γ in 2D and then use 2D algorithms, but we ran the full 3D solvers here. We represented Γ as a collection of flat triangles and discretized via a centroid collocation scheme. Near-field interactions for all centroids within a local neighborhood of radius h about each triangle, where h is the average triangle diameter, were computed using fourth-order tensor-product Gauss-Legendre quadrature. This gives a linear system (1.6) with *unsymmetric* A. Data for rskelf3, hifie3, and hifie3x at $\varepsilon = 10^{-3}$ and 10^{-6} are shown in Tables 5.7 and 5.8 with scaling results in Figure 5.3.

Since Γ is a 2D surface, $d = 2$ in Theorem 3.4, so we can expect RSF to have $O(N^{3/2})$ complexity, as observed. However, the skeleton size is substantially larger than in 2D, so the corresponding costs are much higher. The same is true for HIF-IE, which achieves quasilinear complexity as predicted in Theorem 4.3 and Corollary 4.4. As before, $e_a, e_s = O(\varepsilon)$ for hifie3x but suffer for hifie3.

We also tested the accuracy of our algorithms in solving the associated PDE (1.2) by constructing an interior harmonic field

$$v(x) = \sum_j G(\|x - y_j\|)q_j, \quad x \in \mathscr{D},$$

due to 16 random exterior sources $\{y_j\}$ with $\|y_j\| = 2$, where the "charge" strengths q_j were drawn from the standard uniform distribution. This induces the boundary data $f(x) = v(x)|_{x\in\Gamma}$, which returns the charge density $\sigma(x)$ upon solving (1.5).

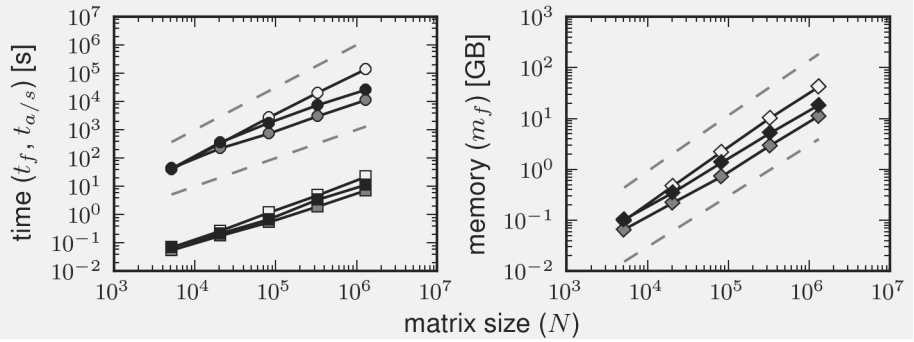

FIGURE 5.3. Scaling results for Example 4, comparing rskelf3 (white), hifie3 (gray), and hifie3x (black) at precision $\varepsilon = 10^{-3}$; all other notation is as in Figure 5.1.

TABLE 5.9. Relative errors against exact solutions for the PDE in Example 4.

ε	N	rskelf3	hifie3	hifie3x
10^{-3}	20480	7.6e−4	2.8e−3	7.8e−4
	81920	3.0e−4	3.0e−2	4.2e−4
	327680	1.2e−4	8.1e−2	2.1e−4
	1310720	4.8e−4	3.1e−1	2.0e−4
10^{-6}	20480	7.9e−4	7.9e−4	7.8e−4
	81920	3.7e−4	3.7e−4	3.7e−4
	327680	1.8e−4	1.8e−4	1.8e−4

The field $u(x)$ due to $\sigma(x)$ via the double-layer potential (1.3) is then, in principle, identical to $v(x)$ by uniqueness of the boundary value problem. This equality was assessed by evaluating both $u(x)$ and $v(x)$ at 16 random interior targets $\{z_j\}$ with $\|z_j\| = \frac{1}{2}$. The relative error between $\{u(z_j)\}$ and $\{v(z_j)\}$ is shown in Table 5.9, from which we observe that rskelf3 and hifie3x are both able to solve the PDE up to the discretization or approximation error.

Example 5. Now consider the 3D analogue of Example 1, i.e., (1.1) with $a(x) \equiv 0$, $b(x) \equiv c(x) = 1$, $K(r) = 1/(4\pi r)$, and $\Omega = (0, 1)^3$, discretized over a uniform grid with adaptive quadratures for the diagonal entries. Data for rskelf3 and hifie3 at $\varepsilon = 10^{-3}$ and 10^{-6} are given in Tables 5.10 and 5.11 with scaling results in Figure 5.4.

It is immediate that $t_f = O(N^2)$ and $t_{a/s} = O(N^{4/3})$ for RSF, which considerably degrades its performance for large N. Indeed, we were unable to run rskelf3 for $N = 128^3$ because of the excessive memory cost. In contrast, HIF-IE scales much better but does not quite achieve $O(N)$ complexity as stated in Theorem 4.3: the empirical scaling for t_f at $\varepsilon = 10^{-3}$, for instance, is approximately $O(N^{1.3})$. We believe this to be a consequence of the large interaction ranks in 3D, which

TABLE 5.10. Factorization results for Example 5.

		rskelf3			hifie3		
ε	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-3}	32^3	5900	5.4e+2	1.0e+0	969	1.6e+2	2.7e−1
	64^3	24005	3.9e+4	1.9e+1	1970	3.4e+3	2.6e+0
	128^3	—	—	—	3981	5.5e+4	2.5e+1
10^{-6}	32^3	11132	2.4e+3	2.8e+0	6108	2.1e+3	1.4e+0
	64^3	—	—	—	16401	1.0e+5	2.0e+1

TABLE 5.11. Matrix application results for Example 5.

		rskelf3	hifie3			
ε	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i
10^{-3}	32^3	4.0e−1	1.6e−1	3.1e−4	2.7e−2	6
	64^3	6.2e+0	1.5e+0	3.6e−4	4.4e−2	7
	128^3	—	1.4e+1	1.2e−3	7.2e−2	8
10^{-6}	32^3	1.1e+0	5.2e−1	1.2e−7	2.8e−5	3
	64^3	—	6.1e+0	2.4e−7	9.5e−5	3

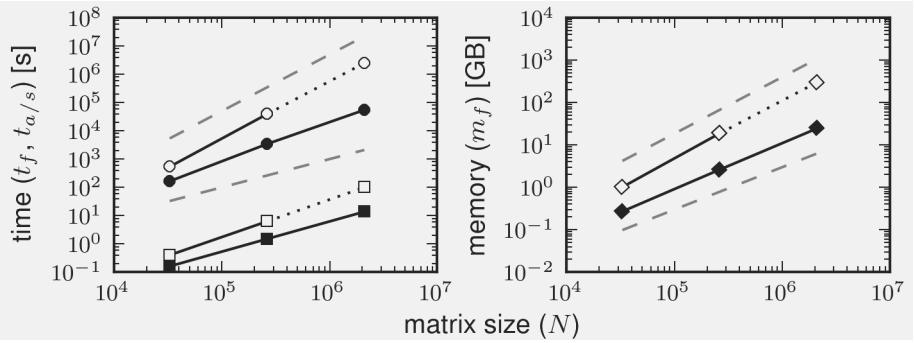

FIGURE 5.4. Scaling results for Example 5, comparing rskelf3 (white) and hifie3 (black) at precision $\varepsilon = 10^{-3}$. Dotted lines denote extrapolated values. Included also are reference scalings of $O(N)$ and $O(N^2)$ (left), and $O(N)$ and $O(N^{4/3})$ (right); all other notation as in Figure 5.1.

make the asymptotic regime rather difficult to reach. Still, even the experimental growth rate of $k_\ell \simeq O(2^\ell)$ would be sufficient for theoretical $O(N \log N)$ complexity. In parallel with Example 1, $e_a = O(\varepsilon)$ but e_s is somewhat larger due to ill-conditioning. We found F^{-1} to be a very effective preconditioner throughout.

Example 6. Finally, we consider the 3D analogue of Example 2, i.e., Example 5 but with $a(x) \equiv 1$. This is a well-conditioned second-kind IE, which we factored using rskelf3, hifie3, and hifie3x. The data are summarized in Tables 5.12 and 5.13 with scaling results shown in Figure 5.5.

TABLE 5.12. Factorization results for Example 6.

		rskelf3			hifie3			hifie3x		
ε	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-3}	32^3	5900	5.4e+2	1.0e+0	1271	2.1e+2	3.9e−1	3127	5.0e+2	6.6e−1
	64^3	24005	4.0e+4	1.9e+1	2023	3.3e+3	3.7e+0	7141	1.3e+4	8.5e+0
	128^3	—	—	—	5105	5.2e+4	3.6e+1	17491	3.5e+5	1.1e+2
10^{-6}	32^3	11132	2.4e+3	2.8e+0	5611	1.6e+3	1.4e+0	8620	2.4e+3	2.2e+0
	64^3	—	—	—	12558	5.4e+4	1.6e+1	25797	8.6e+4	3.4e+1

TABLE 5.13. Matrix application results for Example 6.

		rskelf3	hifie3			hifie3x		
ε	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	$t_{a/s}$	e_a	e_s
10^{-3}	32^3	4.0e−1	2.0e−1	4.6e−3	5.0e−3	2.2e−1	1.1e−4	1.3e−4
	64^3	6.6e+0	1.8e+0	4.4e−2	4.7e−2	3.1e+0	6.2e−4	6.8e−4
	128^3	—	1.7e+1	6.7e−2	7.3e−2	5.1e+1	1.7e−3	1.9e−3
10^{-6}	32^3	1.0e+0	5.7e−1	8.5e−6	9.7e−6	7.4e−1	2.9e−7	3.4e−7
	64^3	—	6.4e+0	5.9e−5	6.8e−5	1.2e+1	1.5e−6	1.8e−6

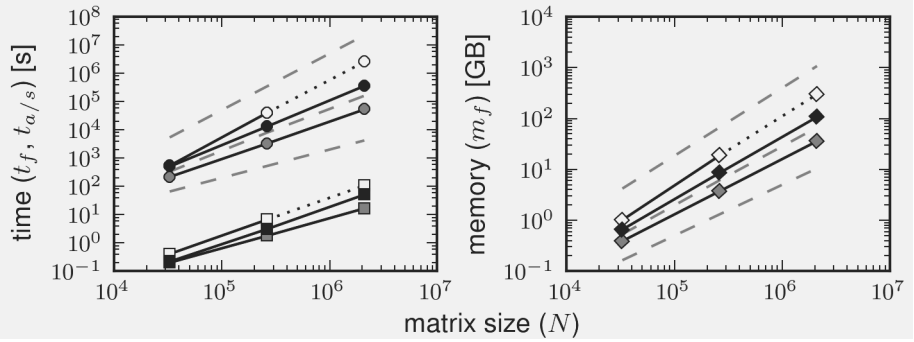

FIGURE 5.5. Scaling results for Example 6, comparing rskelf3 (white), hifie3 (gray), and hifie3x (black) at precision $\varepsilon = 10^{-3}$. Included also are reference scalings of $O(N)$, $O(N \log^6 N)$, and $O(N^2)$ (left); and $O(N)$, $O(N \log^2 N)$, and $O(N^{4/3})$ (right). All other notation is as in Figure 5.4.

Algorithms rskelf3 and hifie3 behave very similarly as in Example 5 but with some error propagation for hifie3 as discussed in Section 4.4. Full accuracy is restored using hifie3x but at the cost of significantly larger skeleton sizes. The empirical complexity of hifie3x hence suffers but remains quite favorable compared to that of rskelf3. We also find a good fit with the complexity estimates of Corollary 4.4, though the presumed penalty for not yet reaching the asymptotic regime may imply that the proposed bounds are overly pessimistic.

6 Generalizations and Conclusions

In this paper, we have introduced HIF-IE for the efficient factorization of discretized integral operators associated with elliptic PDEs in 2D and 3D. HIF-IE combines a novel matrix sparsification framework with recursive dimensional reduction to construct an approximate generalized LU decomposition at estimated quasilinear cost. The latter enables significant compression over RS and is critical for improving the asymptotic complexity, while the former substantially simplifies the algorithm and permits its formulation as a factorization. This representation allows the rapid application of both the matrix and its inverse, and therefore provides a generalized FMM, direct solver, or preconditioner, depending on the accuracy. We have also presented RSF, a factorization formulation of RS [25, 27, 39, 43] that is closely related to MF [19, 23] for sparse matrices. Indeed, a key observation underlying both RSF and HIF-IE is that structured dense matrices can be sparsified very efficiently via the ID. This suggests that well-developed sparse techniques can be applied, and we anticipate that fully exploring this implication will lead to new fast algorithms for dense linear algebra.

The skeletonization operator at the core of RSF and HIF-IE can be interpreted in several ways. For example, we can view it as an approximate local change of basis in order to gain sparsity. Unlike traditional approaches [1, 7, 17], however, this basis is determined optimally on the fly using the ID. Skeletonization can also be regarded as adaptive numerical upscaling or as implementing specialized restriction and prolongation operators in the context of multigrid methods [32].

Although we have presently only considered matrices arising from IEs, the same methods can also be applied (with minor modification) to various general structured matrices such as those encountered in Gaussian process modeling [3, 12] or sparse differential formulations of PDEs [6, 24, 51]. In particular, HIF-IE can be heavily specialized to the latter setting by explicitly taking advantage of existing sparsity. The resulting *hierarchical interpolative factorization for differential equations* (HIF-DE) is described in the companion paper [41] and likewise achieves estimated linear or quasilinear complexity in 2D and 3D.

Some important directions for future research include:

- Obtaining analytical estimates of the interaction rank for SCIs, even for the simple case of the Laplace kernel (1.4). This would enable a much more precise understanding of the complexity of HIF-IE, which has yet to be rigorously established.
- Parallelizing RSF and HIF-IE, both of which are organized according to a tree structure where each node at a given level can be processed independently of the rest. The parallelization of HIF-IE holds particular promise and should have significant impact on practical scientific computing.
- Investigating alternative strategies for reducing skeleton sizes in 3D, which can still be quite large, especially at high precision. New ideas may be required to build truly large-scale direct solvers.

- Understanding the extent to which our current techniques can be adapted to highly oscillatory kernels, which possess rank structures of a different type than that exploited here [20, 21]. Such high-frequency problems can be extremely difficult to solve by iteration and present a prime target area for future fast direct methods.

Acknowledgment. We would like to thank Leslie Greengard for many helpful discussions, Lenya Ryzhik for providing computing resources, and the anonymous referees for their careful reading of the manuscript, which have improved the paper tremendously. K.L.H. was partially supported by the National Science Foundation under award DMS-1203554. L.Y. was partially supported by the National Science Foundation under award DMS-1328230 and the U.S. Department of Energy's Advanced Scientific Computing Research program under award DE-FC02-13ER26134/DE-SC0009409.

Bibliography

[1] Alpert, B.; Beylkin, G.; Coifman, R.; Rokhlin, V. Wavelet-like bases for the fast solution of second-kind integral equations. *SIAM J. Sci. Comput.* **14** (1993), no. 1, 159–184. doi:10.1137/0914010

[2] Ambikasaran, S.; Darve, E. An $O(N \log N)$ fast direct solver for partial hierarchically semi-separable matrices: with application to radial basis function interpolation. *J. Sci. Comput.* **57** (2013), no. 3, 477–501. doi:10.1007/s10915-013-9714-z

[3] Ambikasaran, S.; Foreman-Mackey, D.; Greengard, L.; Hogg, D. W.; O'Neil, M. Fast direct methods for Gaussian processes and analysis of NASA Kepler mission data. Preprint, 2015. arXiv:1403.6015 [math.NA]

[4] Aurenhammer, F. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.* **23** (1991), no. 3, 345–405. doi:10.1145/116873.116880

[5] Barnes, J.; Hut, P. A hierarchical $O(N \log N)$ force-calculation algorithm. *Nature* **324** (1986), no. 4, 446–449. doi:10.1038/324446a0

[6] Bebendorf, M.; Hackbusch, W. Existence of $\mathscr{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients. *Numer. Math.* **95** (2003), no. 1, 1–28. doi:10.1007/s00211-002-0445-6

[7] Beylkin, G.; Coifman, R.; Rokhlin, V. Fast wavelet transforms and numerical algorithms. I. *Comm. Pure Appl. Math.* **44** (1991), no. 2, 141–183. doi:10.1002/cpa.3160440202

[8] Bremer, J. A fast direct solver for the integral equations of scattering theory on planar curves with corners. *J. Comput. Phys.* **231** (2012), no. 4, 1879–1899. doi:10.1016/j.jcp.2011.11.015

[9] Brigham, E. O. *The fast Fourier transform and its applications*. Prentice Hall, Englewood Cliffs, N.J., 1988.

[10] Chandrasekaran, S.; Dewilde, P.; Gu, M.; Lyons, W.; Pals, T. A fast solver for HSS representations via sparse matrices. *SIAM J. Matrix Anal. Appl.* **29** (2006/07), no. 1, 67–81. doi:10.1137/050639028

[11] Chandrasekaran, S.; Gu, M.; Pals, T. A fast ULV decomposition solver for hierarchically semiseparable representations. *SIAM J. Matrix Anal. Appl.* **28** (2006), no. 3, 603–622. doi:10.1137/S0895479803436652

[12] Chen, J.; Wang, L.; Anitescu, M. A fast summation tree code for Matérn kernel. *SIAM J. Sci. Comput.* **36** (2014), no. 1, A289–A309. doi:10.1137/120903002

[13] Chen, Y. A fast, direct algorithm for the Lippmann-Schwinger integral equation in two dimensions. Modeling and computation in optics and electromagnetics. *Adv. Comput. Math.* **16** (2002), no. 2-3, 175–190. doi:10.1023/A:1014450116300

[14] Cheng, H.; Gimbutas, Z.; Martinsson, P. G.; Rokhlin, V. On the compression of low rank matrices. *SIAM J. Sci. Comput.* **26** (2005), no. 4, 1389–1404. doi:10.1137/030602678

[15] Colton, D.; Kress, R. *Inverse acoustic and electromagnetic scattering theory.* Applied Mathematical Sciences, 93. Springer, Berlin, 1992. doi:10.1007/978-3-662-02835-3

[16] Corona, E.; Martinsson, P.-G.; Zorin, D. An $O(N)$ direct solver for integral equations on the plane. *Appl. Comput. Harmon. Anal.* **38** (2015), no. 2, 284–317. doi:10.1016/j.acha.2014.04.002

[17] Dahmen, W. Wavelet and multiscale methods for operator equations. *Acta numerica, 1997*, 55–228. Acta Numerica, 6. Cambridge University Press, Cambridge, 1997. doi:10.1017/S0962492900002713

[18] Dixon, J. D. Estimating extremal eigenvalues and condition numbers of matrices. *SIAM J. Numer. Anal.* **20** (1983), no. 4, 812–814. doi:10.1137/0720053

[19] Duff, I. S.; Reid, J. K. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Trans. Math. Software* **9** (1983), no. 3, 302–325. doi:10.1145/356044.356047

[20] Engquist, B.; Ying, L. Fast directional multilevel algorithms for oscillatory kernels. *SIAM J. Sci. Comput.* **29** (2007), no. 4, 1710–1737 (electronic). doi:10.1137/07068583X

[21] Engquist, B.; Ying, L. A fast directional algorithm for high frequency acoustic scattering in two dimensions. *Commun. Math. Sci.* **7** (2009), no. 2, 327–345.

[22] Fong, W.; Darve, E. The black-box fast multipole method. *J. Comput. Phys.* **228** (2009), no. 23, 8712–8725. doi:10.1016/j.jcp.2009.08.031

[23] George, A. Nested dissection of a regular finite element mesh. *SIAM J. Numer. Anal.* **10** (1973), 345–363. doi:10.1137/0710032

[24] Gillman, A.; Martinsson, P.-G. An $O(N)$ algorithm for constructing the solution operator to 2D elliptic boundary value problems in the absence of body loads. *Adv. Comput. Math.* **40** (2014), no. 4, 773–796. doi:10.1007/s10444-013-9326-z

[25] Gillman, A.; Young, P. M.; Martinsson, P.-G. A direct solver with $O(N)$ complexity for integral equations on one-dimensional domains. *Front. Math. China* **7** (2012), no. 2, 217–247. doi:10.1007/s11464-012-0188-3

[26] Golub, G. H.; Van Loan, C. F. *Matrix computations.* Third edition. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, 1996.

[27] Greengard, L.; Gueyffier, D.; Martinsson, P.-G.; Rokhlin, V. Fast direct solvers for integral equations in complex three-dimensional domains. *Acta Numer.* **18** (2009), 243–275. doi:10.1017/S0962492906410011

[28] Greengard, L.; Rokhlin, V. A fast algorithm for particle simulations. *J. Comput. Phys.* **73** (1987), no. 2, 325–348. doi:10.1016/0021-9991(87)90140-9

[29] Greengard, L.; Rokhlin, V. A new version of the Fast Multipole Method for the Laplace equation in three dimensions. *Acta Numerica* **6** (1997), 229–269. doi:10.1017/S0962492900002725

[30] Gu, M.; Eisenstat, S. C. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM J. Sci. Comput.* **17** (1996), no. 4, 848–869. doi:10.1137/0917055

[31] Guenther, R. B.; Lee, J. W. *Partial differential equations of mathematical physics and integral equations*. Prentice Hall, Englewood Cliffs, N.J., 1988.

[32] Hackbusch, W. *Multigrid methods and applications.* Springer Series in Computational Mathematics, 4. Springer, Berlin, 1985. doi:10.1007/978-3-662-02427-0

[33] Hackbusch, W. A sparse matrix arithmetic based on $\mathscr{H}$-matrices. I. Introduction to $\mathscr{H}$-matrices. *Computing* **62** (1999), no. 2, 89–108. doi:10.1007/s006070050015

[34] Hackbusch, W.; Börm, S. Data-sparse approximation by adaptive $\mathscr{H}^2$-matrices. *Computing* **69** (2002), no. 1, 1–35. doi:10.1007/s00607-002-1450-4

[35] Hackbusch, W.; Khoromskij, B. N. A sparse $\mathscr{H}$-matrix arithmetic. II. Application to multi-dimensional problems. *Computing* **64** (2000), no. 1, 21–47.

[36] Hackbusch, W.; Nowak, Z. P. On the fast matrix multiplication in the boundary element method by panel clustering. *Numer. Math.* **54** (1989), no. 4, 463–491. doi:10.1007/BF01396324

[37] Halko, N.; Martinsson, P. G.; Tropp, J. A. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.* **53** (2011), no. 2, 217–288. doi:10.1137/090771806

[38] Hestenes, M. R.; Stiefel, E. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards* **49** (1952), 409–436 (1953).

[39] Ho, K. L.; Greengard, L. A fast direct solver for structured linear systems by recursive skeletonization. *SIAM J. Sci. Comput.* **34** (2012), no. 5, A2507–A2532. doi:10.1137/120866683

[40] Ho, K. L.; Greengard, L. A fast semidirect least squares algorithm for hierarchically block separable matrices. *SIAM J. Matrix Anal. Appl.* **35** (2014), no. 2, 725–748. doi:10.1137/120902677

[41] Ho, K. L.; Ying, L. Hierarchical interpolative factorization for elliptic operators: differential equations. *Comm. Pure Appl. Math.*, forthcoming.

[42] Kuczyński, J.; Woźniakowski, H. Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM J. Matrix Anal. Appl.* **13** (1992), no. 4, 1094–1122. doi:10.1137/0613066

[43] Martinsson, P. G.; Rokhlin, V. A fast direct solver for boundary integral equations in two dimensions. *J. Comput. Phys.* **205** (2005), no. 1, 1–23. doi:10.1016/j.jcp.2004.10.033

[44] Martinsson, P. G.; Rokhlin, V. An accelerated kernel-independent fast multipole method in one dimension. *SIAM J. Sci. Comput.* **29** (2007), no. 3, 1160–1178. doi:10.1137/060662253

[45] Martinsson, P.-G.; Rokhlin, V.; Tygert, M. On interpolation and integration in finite-dimensional spaces of bounded functions. *Commun. Appl. Math. Comput. Sci.* **1** (2006), 133–142 (electronic). doi:10.2140/camcos.2006.1.133

[46] Pan, X.-M.; Wei, J.-G.; Peng, Z.; Sheng, X.-Q. A fast algorithm for multiscale electromagnetic problems using interpolative decomposition and multilevel fast multipole algorithm. *Radio Sci.* **47** (2012), RS1011. doi:10.1029/2011RS004891

[47] Saad, Y.; Schultz, M. H. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.* **7** (1986), no. 3, 856–869. doi:10.1137/0907058

[48] Samet, H. The quadtree and related hierarchical data structures. *Comput. Surveys* **16** (1984), no. 2, 187–260.

[49] van der Vorst, H. A. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.* **13** (1992), no. 2, 631–644. doi:10.1137/0913035

[50] Xia, J. Efficient structured multifrontal factorization for general large sparse matrices. *SIAM J. Sci. Comput.* **35** (2013), no. 2, A832–A860. doi:10.1137/120867032

[51] Xia, J.; Chandrasekaran, S.; Gu, M.; Li, X. S. Superfast multifrontal method for large structured linear systems of equations. *SIAM J. Matrix Anal. Appl.* **31** (2009), no. 3, 1382–1411. doi:10.1137/09074543X

[52] Xia, J.; Chandrasekaran, S.; Gu, M.; Li, X. S. Fast algorithms for hierarchically semiseparable matrices. *Numer. Linear Algebra Appl.* **17** (2010), no. 6, 953–976. doi:10.1002/nla.691

[53] Xia, J.; Xi, Y.; Gu, M. A superfast structured solver for Toeplitz linear systems via randomized sampling. *SIAM J. Matrix Anal. Appl.* **33** (2012), no. 3, 837–858. doi:10.1137/110831982

[54] Ying, L.; Biros, G.; Zorin, D. A kernel-independent adaptive fast multipole algorithm in two and three dimensions. *J. Comput. Phys.* **196** (2004), no. 2, 591–626. doi:10.1016/j.jcp.2003.11.021

KENNETH L. HO
Stanford University
Department of Mathematics
450 Serra Mall
Building 380
Stanford, CA 94305
USA
E-mail: klho@stanford.edu

LEXING YING
Stanford University
Department of Mathematics
450 Serra Mall
Building 380
Stanford, CA 94305
USA
E-mail: lexing@stanford.edu

Received July 2014.