Hierarchical Interpolative Factorization for Elliptic Operators: Differential Equations

KENNETH L. HO
Stanford University

LEXING YING
Stanford University

Abstract

This paper introduces the hierarchical interpolative factorization for elliptic partial differential equations (HIF-DE) in two (2D) and three dimensions (3D). This factorization takes the form of an approximate generalized LU/LDL decomposition that facilitates the efficient inversion of the discretized operator. HIF-DE is based on the nested dissection multifrontal method but uses skeletonization on the separator fronts to sparsify the dense frontal matrices and thus reduce the cost. We conjecture that this strategy yields linear complexity in 2D and quasilinear complexity in 3D. Estimated linear complexity in 3D can be achieved by skeletonizing the compressed fronts themselves, which amounts geometrically to a recursive dimensional reduction scheme. Numerical experiments support our claims and further demonstrate the performance of our algorithm as a fast direct solver and preconditioner. MATLAB® codes are freely available.

1 Introduction

This paper considers elliptic partial differential equations (PDEs) of the form

$$-\nabla \cdot (a(x)\nabla u(x)) + b(x)u(x) = f(x), \quad x \in \Omega \subset \mathbb{R}^d, \tag{1.1}$$

with appropriate boundary conditions on $\partial\Omega$, where $a(x)$, $b(x)$, and $f(x)$ are given functions, and $d = 2$ or 3. Such equations are of fundamental importance in science and engineering and encompass (perhaps with minor modification) many of the PDEs of classical physics, including the Laplace, Helmholtz, Stokes, and time-harmonic Maxwell equations. We will further assume that (1.1) is not highly indefinite. Discretization using local schemes such as finite differences or finite elements then leads to a linear system

$$Au = f, \tag{1.2}$$

where $A \in \mathbb{R}^{N\times N}$ is sparse 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.

Communications on Pure and Applied Mathematics, 0001–0037 (PREPRINT)

1.1 Previous Work

A large part of modern numerical analysis and scientific computing has been devoted to the solution of (1.2). We classify existing approaches into several groups. The first consists of classical direct methods like Gaussian elimination or other standard matrix factorizations [19], which compute the solution exactly (in principle, to machine precision, up to conditioning) without iteration. Naive implementations generally have $O(N^3)$ complexity but can be heavily accelerated by exploiting sparsity [10]. A key example is the nested dissection multifrontal method (MF) [12, 15, 33], which performs elimination according to a special hierarchy of separator fronts in order to minimize fill-in. These fronts correspond geometrically to the cell interfaces in a domain partitioning and 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. This is a significant improvement and, indeed, MF has proven very effective in many environments. However, it remains unsuitable for truly large-scale problems, especially in 3D.

The second group is that of iterative methods [36], with conjugate gradient (CG) [29, 41] and multigrid [7, 24, 47] among the most popular techniques. These typically work well when $a(x)$ and $b(x)$ are smooth, in which case the number of iterations required is small and optimal $O(N)$ complexity can be achieved. However, the iteration count can grow rapidly in the presence of ill-conditioning, which can arise when the coefficient functions lack regularity or have high contrast. In such cases, convergence can be delicate and specialized preconditioners are often required. 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 and A^{-1} are numerically low-rank [4–6, 8] in order to dramatically lower the cost. The seminal work in this area is due to Hackbusch et al. [25–27], whose $\mathscr{H}$- and $\mathscr{H}^2$-matrices have been shown to achieve linear or quasilinear complexity. These methods were originally introduced for integral equations characterized by structured dense matrices but apply also to PDEs as a special case. Although their work has had significant theoretical impact, in practice the constants implicit in the asymptotic scalings tend to be quite large due to the recursive nature of the inversion algorithms, the use of expensive hierarchical matrix-matrix multiplication, and the lack of sparsity optimizations.

More recent developments aimed at improving practical performance have combined MF with structured matrix algebra on the dense frontal matrices only. This better exploits the inherent sparsity of A and has been carried out under both the $\mathscr{H}$- [20, 39, 40] and hierarchically semiseparable (HSS) [16, 17, 42–44] matrix frameworks, among other related schemes [1, 2, 34]. Those under the former retain

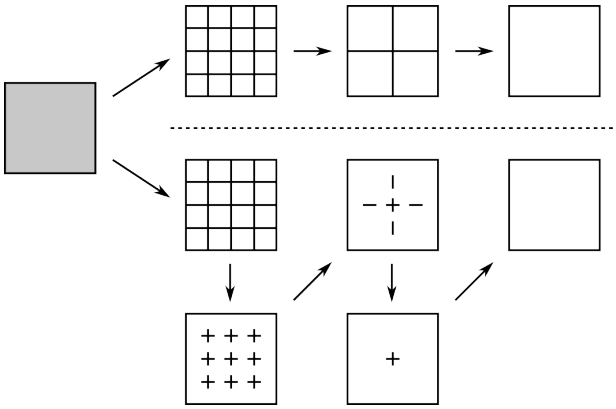


FIGURE 1.1. Schematic of MF (top) and HIF-DE (bottom) in 2D. The gray box (left) represents a uniformly discretized square; the lines in the interior of the boxes (right) denote the remaining DOFs after each level of elimination or skeletonization.

their quasilinear complexities and have improved constants but can still be somewhat expensive. On the other hand, those using HSS operations, which usually have much more favorable constants, are optimal in 2D but require $O(N^{4/3})$ work in 3D. In principle, it is possible to further reduce this to $O(N)$ work by using multilayer HSS representations, but this procedure is quite complicated and has yet to be achieved.

1.2 Contributions

In this paper, we introduce the hierarchical interpolative factorization for PDEs (HIF-DE), which produces an approximate generalized LU/LDL decomposition of A with linear or quasilinear complexity estimates. HIF-DE is based on MF but augments it with frontal compression using a matrix sparsification technique that we call skeletonization. The resulting algorithm is similar in structure to the accelerated MF solvers above and is sufficient for estimated scalings of $O(N)$ in 2D and $O(N \log N)$ in 3D. Unlike [16, 17, 20, 39, 40, 44], however, which keep the entire fronts but work with them implicitly using fast structured methods, our sparsification approach allows us to reduce the fronts explicitly (see also [42, 43]). This obviates the need for internal hierarchical matrix representations and substantially simplifies the algorithm. Importantly, it also makes any additional compression straightforward to accommodate, thereby providing a ready means to achieve estimated $O(N)$ complexity in 3D by skeletonizing the compressed fronts themselves. This corresponds geometrically to a recursive dimensional reduction, whose interpretation is directly enabled by the skeletonization formalism.

Figure 1.1 shows a schematic of HIF-DE as compared to MF in 2D. In MF (top), the domain is partitioned by a set of separators into "interior" square cells at each level of a tree hierarchy. Each cell is eliminated starting from the finest level

to the coarsest, leaving degrees of freedom (DOFs) only on the separators, which constitute the so-called fronts. This process can be understood as the compression of data from the cells to their interfaces, which evidently grow as we march up the tree, ultimately leading to the observed $O(N^{3/2})$ complexity.

In contrast, in HIF-DE (bottom), we start by eliminating interior cells as in MF but, before proceeding further, perform an additional level of compression by skeletonizing the separators. For this, we view the separator DOFs as living on the interfacial edges of the interior cells then skeletonize each cell edge. This respects the one-dimensional (1D) structure of the separator geometry and allows more DOFs to be eliminated, in effect reducing each edge to only those DOFs near its boundary. Significantly, this occurs *without any loss of existing sparsity*. The combination of interior cell elimination and edge skeletonization is then repeated up the tree, with the result that the frontal 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 eliminating interior cubic cells and then skeletonizing cubic faces at each level to execute a reduction from 3D to 2D to 1D at a total estimated cost of $O(N \log N)$. We can further reduce this to $O(N)$ (but at the price of introducing some fill-in) by adding subsequent cubic edge skeletonization at each level for full reduction to 0D. This tight control of the front size is critical for achieving near-optimal scaling.

Once the factorization has been constructed, it can be used to rapidly apply A^{-1} and therefore serves as a fast direct solver or preconditioner, depending on the accuracy. (It can also be used to apply A itself, but this is not particularly advantageous since A typically has only $O(N)$ nonzeros.) Other capabilities are possible, too, though they will not be pursued here.

HIF-DE can also be understood in relation to the somewhat more general *hierarchical interpolative factorization for integral equations* (HIF-IE) described in the companion paper [31], which, like other structured dense methods, can apply to PDEs as a special case. However, HIF-IE does not make use of sparsity and so is not very competitive in practice. HIF-DE remedies this by essentially embedding HIF-IE into the framework of MF in order to maximally exploit sparsity.

Extensive numerical experiments reveal strong evidence for quasilinear complexity and demonstrate that HIF-DE can accurately approximate elliptic partial differential operators in a variety of 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 our new skeletonization operation. In Section 3, we review MF, which will serve to establish the necessary algorithmic foundation as well as to highlight its fundamental difficulties. In Section 4, we present HIF-DE as an extension of MF with frontal skeletonization corresponding to recursive dimensional reduction. Although we cannot yet provide a rigorous complexity analysis, estimates based on well-supported rank assumptions

suggest that HIF-DE 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. The *neighbor set* of an index set c with respect to A is then $c^{\mathsf{N}} = \{i \notin c : A_{i,c} \text{ or } A_{c,i} \neq 0\}$.

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

For simplicity, we hereafter assume that the matrix A in (1.2) is symmetric, though this is not strictly necessary [31].

2.1 Sparse Elimination

Let

$$A = \begin{bmatrix} A_{pp} & A_{qp}^{\mathsf{T}} & \\ A_{qp} & A_{qq} & A_{rq}^{\mathsf{T}} \\ & A_{rq} & A_{rr} \end{bmatrix} \tag{2.1}$$

be a symmetric matrix defined over the indices (p, q, r). This matrix structure often appears in sparse PDE problems such as (1.2), 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 $\Omega \setminus \overline{\mathscr{D}}$, 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 $A_{pp} = L_p D_p L_p^{\mathsf{T}}$ in factored form, where L_p is a unit triangular matrix (up to permutation). If A_{pp} is nonsingular, then*

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

where

$$S_p = \begin{bmatrix} L_p^{-\mathsf{T}} & & \\ & I & \\ & & I \end{bmatrix} \begin{bmatrix} I & -D_p^{-1} L_p^{-1} A_{qp}^{\mathsf{T}} & \\ & I & \\ & & I \end{bmatrix}$$

and $B_{qq} = A_{qq} - A_{qp} A_{pp}^{-1} A_{qp}^{\mathsf{T}}$ *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 operator S_p carries out this elimination, which furthermore is particularly efficient since the interactions involving the large index set r are unchanged. However, some fill-in is generated through the Schur complement B_{qq}, which in general is completely dense. Clearly, the requirement that A_{pp} be invertible is satisfied if A is symmetric positive definite (SPD), as is the case for many such problems in practice.

In this paper, we often work with a collection C of disjoint index sets, where $A_{c,c'} = A_{c',c} = 0$ for any $c, c' \in C$ with $c \neq c'$. Applying Lemma 2.1 to each $p = c$, $q = c^{\mathsf{N}}$, and $r = (c \cup c^{\mathsf{N}})^{\mathsf{c}}$ gives $W^{\mathsf{T}} A W$ for $W = \prod_{c \in C} S_c$, where each set of DOFs $c \in C$ has been decoupled from the rest and the matrix product over C can be taken in any order. The resulting matrix has a block diagonal structure over the index groups

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

where the outer union is to be understood as acting on collections of index sets and $s = \{1, 2, \ldots, N\}$ is the set of all indices, but with dense fill-in covering $(W^{\mathsf{T}} A W)_{c^{\mathsf{N}}, c^{\mathsf{N}}}$ for each $c \in C$.

2.2 Interpolative Decomposition

Our next tool is the interpolative decomposition (ID) [9] for low-rank matrices, which we present in a somewhat nonstandard form below (see [31] for details).

LEMMA 2.2. *Let* $A = A_{:,q} \in \mathbb{R}^{m \times n}$ *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* $T_q \in \mathbb{R}^{k \times n}$ *such that* $A_{:,\check{q}} = A_{:,\hat{q}} T_q$.

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. If $\|T_q\|$ and $\|E\|$ are not too large, then the reconstruction of $A_{:,\check{q}}$ is stable and accurate. In this paper, we use the algorithm of [9] based on a simple pivoted QR decomposition to compute an ID that typically satisfies

$$\|T_q\| \le \sqrt{4k(n-k)}, \quad \|E\| \le \sqrt{1+4k(n-k)}\sigma_{k+1}(A),$$

where $\sigma_{k+1}(A)$ is the $(k+1)^{\text{st}}$ largest singular value of A, at a cost of $O(kmn)$ operations. Fast algorithms based on random sampling are also available [28], but these can incur some loss of accuracy (see also Section 4.5).

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 $\|E\| \lesssim \epsilon\|A\|$ for any specified relative precision $\epsilon > 0$.

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*

$$A = \begin{bmatrix} A_{pp} & A_{qp}^{\mathsf{T}} \\ A_{qp} & A_{qq} \end{bmatrix}$$

be symmetric with A_{qp} low-rank, and let $p = \hat{p} \cup \check{p}$ and T_p be such that $A_{q\check{p}} = A_{q\hat{p}}T_p$. Without loss of generality, write

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

and define

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

Then

$$Q_p^{\mathsf{T}} A Q_p = \begin{bmatrix} B_{\check{p}\check{p}} & B_{\hat{p}\check{p}}^{\mathsf{T}} & \\ B_{\hat{p}\check{p}} & A_{\hat{p}\hat{p}} & A_{q\hat{p}}^{\mathsf{T}} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix}, \tag{2.3}$$

where

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

so

$$S_{\check{p}}^{\mathsf{T}} Q_p^{\mathsf{T}} A Q_p S_{\check{p}} = \begin{bmatrix} D_{\check{p}} & & \\ & B_{\hat{p}\hat{p}} & A_{q\hat{p}}^{\mathsf{T}} \\ & A_{q\hat{p}} & A_{qq} \end{bmatrix} \equiv \mathscr{Z}_p(A), \tag{2.4}$$

where $S_{\check{p}}$ is the elimination operator of Lemma 2.1 *associated with $\check{p}$ and $B_{\hat{p}\hat{p}} = A_{\hat{p}\hat{p}} - B_{\hat{p}\check{p}} B_{\check{p}\check{p}}^{-1} B_{\hat{p}\check{p}}^{\mathsf{T}}$, 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 by 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 Cholesky decompositions [45] by combining the structure-preserving rank-revealing factorization [46] with reduced matrices [42].

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 (2.3) and consequently (2.4) 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 (2.4) by writing $\mathscr{Z}_p(A) \approx S_{\check{p}}^{\mathsf{T}} Q_p^{\mathsf{T}} A Q_p S_{\check{p}}$, 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

$$\mathscr{Z}_C(A) \approx U^{\mathsf{T}} A U, \quad U = \prod_{c \in C} Q_c S_{\check{c}},$$

where the redundant DOFs $\check{c}$ for each $c \in C$ have been decoupled from the rest and the matrix product 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

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

3 Multifrontal Factorization

In this section, we review MF, which constructs a multilevel LDL decomposition of A by using Lemma 2.1 to eliminate DOFs according to a hierarchical sequence of domain separators. Our presentation will tend to emphasize its geometric aspects [15]; more algebraic treatments can be found in [12, 33].

We begin with a detailed description of MF in 2D before extending to 3D in the natural way. The same presentation framework will also be used for HIF-DE in Section 4, which we hope will help make clear the specific innovations responsible for its improved complexity estimates.

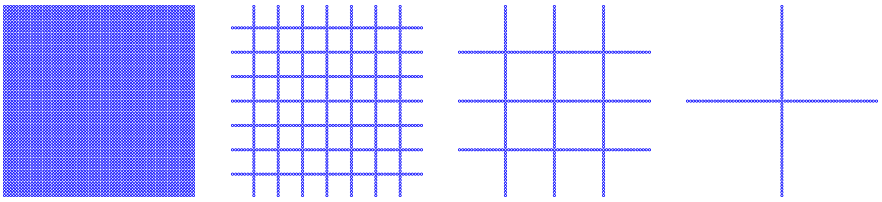


FIGURE 3.1. Active DOFs at each level ℓ of MF in 2D.

3.1 Two Dimensions

Consider the PDE (1.1) on $\Omega = (0,1)^2$ with zero Dirichlet boundary conditions, discretized using finite differences via the five-point stencil over a uniform $n \times n$ grid for simplicity. More general domains, boundary conditions, and discretizations can be handled without difficulty, but the current setting will serve to fix 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 grid points $x_j = hj = h(j_1, j_2)$ for $1 \le j_1, j_2 \le n-1$. The discrete system (1.2) then reads

$$-\frac{1}{h^2}(a_{j-e_1/2} + a_{j+e_1/2} + a_{j-e_2/2} + a_{j+e_2/2})u_j \\ -\frac{1}{h^2}\big(a_{j-e_1/2}u_{j-e_1} + a_{j+e_1/2}u_{j+e_1} \\ + a_{j-e_2/2}u_{j-e_2} + a_{j+e_2/2}u_{j+e_2}\big) + b_j u_j = f_j$$

at each x_j, where $a_j = a(hj)$ is sampled on the "staggered" dual grid for $e_1 = (1,0)$ and $e_2 = (0,1)$ the unit coordinate vectors, $b_j = b(x_j)$, $f_j = f(x_j)$, and u_j is the approximation to $u(x_j)$. The resulting matrix A is sparse and symmetric, consisting only of nearest-neighbor interactions. The total number of DOFs is $N = (n-1)^2$, 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 by 1D separators $mh(j_1, \cdot)$ and $mh(\cdot, j_2)$ for $1 \le j_1, j_2 \le 2^L - 1$ every $mh = n/2^L$ units in each direction into interior square cells $mh(j_1 - 1, j_1) \times mh(j_2 - 1, j_2)$ for $1 \le j_1, j_2 \le 2^L$. Observe that distinct cells do not interact with each other since they are buffered by the separators. Let C_0 be the collection of index sets corresponding to the active DOFs of each cell. Then elimination with respect to C_0 gives

$$A_1 = W_0^{\mathsf{T}} A_0 W_0, \quad W_0 = \prod_{c \in C_0} S_c,$$

where the DOFs $\bigcup_{c\in\boldsymbol{C}_0} c$ have been eliminated (and marked inactive). Let $s_1 = s_0 \setminus \bigcup_{c\in\boldsymbol{C}_0} c$ be the remaining active DOFs. The matrix A_1 is block diagonal with block partitioning

$$\theta_1 = \Big(\bigcup_{c\in\boldsymbol{C}_0} \{c\}\Big) \cup \{s_1\}.$$

Level ℓ

Defined at this stage are A_ℓ and s_ℓ. Partition Ω by 1D separators $2^\ell mh(j_1, \cdot)$ and $2^\ell mh(\cdot, j_2)$ for $1 \le j_1, j_2 \le 2^{L-\ell} - 1$ every $2^\ell mh = n/2^{L-\ell}$ units in each direction into interior square cells $2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2)$ for $1 \le j_1, j_2 \le 2^{L-\ell}$. Let $\boldsymbol{C}_\ell$ be the collection of index sets corresponding to the active DOFs of each cell. Elimination with respect to $\boldsymbol{C}_\ell$ then gives

$$A_{\ell+1} = W_\ell^{\mathsf{T}} A_\ell W_\ell, \quad W_\ell = \prod_{c\in\boldsymbol{C}_\ell} S_c,$$

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

$$\theta_{\ell+1} = \Big(\bigcup_{c\in\boldsymbol{C}_0} \{c\}\Big) \cup \cdots \cup \Big(\bigcup_{c\in\boldsymbol{C}_\ell} \{c\}\Big) \cup \{s_{\ell+1}\},$$

where $s_{\ell+1} = s_\ell \setminus \bigcup_{c\in\boldsymbol{C}_\ell} c$.

Level L

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

$$\theta_L = \Big(\bigcup_{c\in\boldsymbol{C}_0} \{c\}\Big) \cup \cdots \cup \Big(\bigcup_{c\in\boldsymbol{C}_{L-1}} \{c\}\Big) \cup \{s_L\}.$$

Combining over all levels gives

$$D = W_{L-1}^{\mathsf{T}} \cdots W_0^{\mathsf{T}} A W_0 \cdots W_{L-1},$$

where each W_ℓ is a product of unit upper triangular matrices, each of which can be inverted simply by negating its off-diagonal entries. Therefore,

$$A = W_0^{-\mathsf{T}} \cdots W_{L-1}^{-\mathsf{T}} D W_{L-1}^{-1} \cdots W_0^{-1} \equiv F, \tag{3.1a}$$

$$A^{-1} = W_0 \cdots W_{L-1} D^{-1} W_{L-1}^{\mathsf{T}} \cdots W_0^{\mathsf{T}} = F^{-1}. \tag{3.1b}$$

The factorization F is an LDL decomposition of A that is numerically exact (to machine precision, up to conditioning), whose inverse F^{-1} can be applied as a fast direct solver. Clearly, if A is SPD, then so are F and F^{-1}; in this case, F can, in fact, be written as a Cholesky decomposition by storing D in Cholesky form. We emphasize that F and F^{-1} are not assembled explicitly and are used only in their factored representations.

The entire procedure is summarized compactly as Algorithm 3.1. In general, we can construct the cell partitioning at each level using an adaptive quadtree [38],

which recursively subdivides the domain until each node contains only $O(1)$ DOFs, provided that some appropriate postprocessing is done to define "thin" separators in order to optimally exploit sparsity (see Section 4.5).

Algorithm 3.1 MF.

$A_0 = A$ ▷ initialize
for $\ell = 0, 1, \ldots, L-1$ **do** ▷ loop from finest to coarsest level
  $A_{\ell+1} = W_\ell^{\mathsf{T}} A_\ell W_\ell$ ▷ eliminate interior cells
end for
$A = W_0^{-\mathsf{T}} \cdots W_{L-1}^{-\mathsf{T}} A_L W_{L-1}^{-1} \cdots W_0^{-1}$ ▷ LDL decomposition

3.2 Three Dimensions

Consider now the analogous setting in 3D, where $\Omega = (0,1)^3$ is discretized using the seven-point stencil over a uniform $n \times n \times n$ mesh with grid points $x_j = hj = h(j_1, j_2, j_3)$ for $j = (j_1, j_2, j_3)$:

$$\frac{1}{h^2}(a_{j-e_1/2} + a_{j+e_1/2} + a_{j-e_2/2} + a_{j+e_2/2} + a_{j-e_3/2} + a_{j+e_3/2})u_j$$
$$- \frac{1}{h^2}\big(a_{j-e_1/2}u_{j-e_1} + a_{j+e_1/2}u_{j+e_1} + a_{j-e_2/2}u_{j-e_2} + a_{j+e_2/2}u_{j+e_2}$$
$$+ a_{j-e_3/2}u_{j-e_3} + a_{j+e_3/2}u_{j+e_3}\big) + b_j u_j = f_j,$$

where $e_1 = (1,0,0)$, $e_2 = (0,1,0)$, and $e_3 = (0,0,1)$. The total number of DOFs is $N = (n-1)^3$.

The algorithm extends in the natural way with 2D separators $2^\ell mh(j_1, \cdot, \cdot)$, $2^\ell mh(\cdot, j_2, \cdot)$, and $2^\ell mh(\cdot, \cdot, j_3)$ for $1 \le j_1, j_2, j_3 \le 2^{L-\ell} - 1$ every $2^\ell mh = n/2^{L-\ell}$ units in each direction now partitioning Ω into interior cubic cells

$$2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2) \times 2^\ell mh(j_3 - 1, j_3)$$

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 (3.1). General geometries can be treated using an adaptive octree.

3.3 Complexity Estimates

We next analyze the computational complexity of MF. This is determined by the size $|c|$ of a typical index set $c \in C_\ell$, which we write as $k_\ell = O(2^{(d-1)\ell})$ following the separator structure. Note furthermore that $|c^{\mathsf{N}}| = O(k_\ell)$ as well since c^{N} is restricted to the separators enclosing the DOFs c.

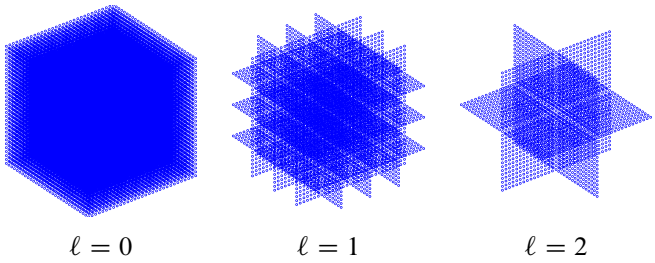


FIGURE 3.2. Active DOFs at each level ℓ of MF in 3D.

THEOREM 3.1 ([15]). *The cost of constructing the factorization F in* (3.1) *using MF is*

$$t_f = \sum_{\ell=0}^{L} 2^{d(L-\ell)} O(k_\ell^3) = \begin{cases} O(N), & d = 1, \\ O(N^{3(1-1/d)}), & d \geq 2, \end{cases} \tag{3.2}$$

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

$$t_{a/s} = \sum_{\ell=0}^{L} 2^{d(L-\ell)} O(k_\ell^2) = \begin{cases} O(N), & d = 1, \\ O(N \log N), & d = 2, \\ O(N^{2(1-1/d)}), & d \geq 3. \end{cases} \tag{3.3}$$

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 various local dense matrix operations (due to fill-in) at a total cost of $O((|c| + |c^{\mathrm{N}}|)^3) = O(k_\ell^3)$, following Lemma 2.1. Hence, we derive (3.2). A similar argument yields (3.3) by observing that each $c \in C_\ell$ requires local dense matrix-vector products with cost $O((|c| + |c^{\mathrm{N}}|)^2)$. □

Remark 3.2. 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 or F^{-1} is clearly $m_f = O(t_{a/s})$ and so is also given by (3.3). Theorem 3.1 is, in fact, valid for all d, including the 1D case where $k_\ell = O(1)$ and optimal linear complexity is achieved. It is immediate that the suboptimal complexities in 2D and 3D are due to the geometric growth of k_ℓ.

4 Hierarchical Interpolative Factorization

In this section, we present HIF-DE, which builds upon MF by introducing additional levels of compression based on skeletonizing the separator fronts. The key observation is that the Schur complements characterizing the dense frontal matrices accumulated throughout the algorithm possess significant rank structures. This can be understood by interpreting the matrix A_{pp}^{-1} in (2.2) as the discrete Green's function of a local elliptic PDE. By elliptic regularity, such Green's functions typically

have numerically low-rank off-diagional blocks. The same rank structure essentially carries over to the Schur complement B_{qq} itself, as indeed has previously been recognized and successfully exploited [1, 2, 16, 17, 20, 34, 39, 40, 42–44].

The interaction ranks of the Schur complement interactions (SCIs) constituting B_{qq} have been the subject of several analytic studies [4–6, 8], though none have considered the exact type with which we are concerned in this paper. Such an analysis, however, is not our primary goal, and we will be content simply with an empirical description. In particular, we have found through extensive numerical experimentation (Section 5) that standard multipole estimates [22, 23] appear to hold for SCIs. We hereafter take this as an assumption, from which we can expect that the skeletons of a given cluster of DOFs will tend to lie along its boundary [30, 31], thus exhibiting a dimensional reduction.

We are now in a position to motivate HIF-DE. Considering the 2D case for concreteness, the main idea is simply to employ an additional level $\ell + \frac{1}{2}$ of edge skeletonization after each level ℓ of interior cell elimination. This fully exploits the 1D geometry of the active DOFs and effectively reduces each front to 0D. An analogous strategy is adopted in 3D for reduction to either 1D by skeletonizing cubic faces or to 0D by skeletonizing faces then edges. In principle, the latter is more efficient but can generate fill-in and so must be used with care.

The overall approach of HIF-DE is closely related to that of [2, 16, 17, 20, 39, 40, 44], but our sparsification framework permits a much simpler implementation and analysis. As with MF, we begin first in 2D before extending to 3D.

4.1 Two Dimensions

Assume the same setup as Section 3.1. HIF-DE supplements interior cell elimination (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 by 1D separators $2^\ell mh(j_1, \cdot)$ and $2^\ell mh(\cdot, j_2)$ for $1 \le j_1, j_2 \le 2^{L-\ell} - 1$ into interior square cells $2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2)$ for $1 \le j_1, j_2 \le 2^{L-\ell}$. Let $\boldsymbol{C}_\ell$ be the collection of index sets corresponding to the active DOFs of each cell. Elimination with respect to $\boldsymbol{C}_\ell$ then gives

$$A_{\ell+1/2} = W_\ell^{\mathsf{T}} A_\ell W_\ell, \quad W_\ell = \prod_{c \in \boldsymbol{C}_\ell} S_c,$$

where the DOFs $\bigcup_{c \in \boldsymbol{C}_\ell} c$ have been eliminated. The matrix $A_{\ell+1/2}$ is block diagonal with block partitioning

$$\theta_{\ell+1/2} = \Big(\bigcup_{c \in \boldsymbol{C}_0} \{c\} \Big) \cup \Big(\bigcup_{c \in \boldsymbol{C}_{1/2}} \{\check{c}\} \Big) \cup \cdots \cup \Big(\bigcup_{c \in \boldsymbol{C}_\ell} \{c\} \Big) \cup \{s_{\ell+1/2}\},$$

where $s_{\ell+1/2} = s_\ell \setminus \bigcup_{c \in \boldsymbol{C}_\ell} c$.

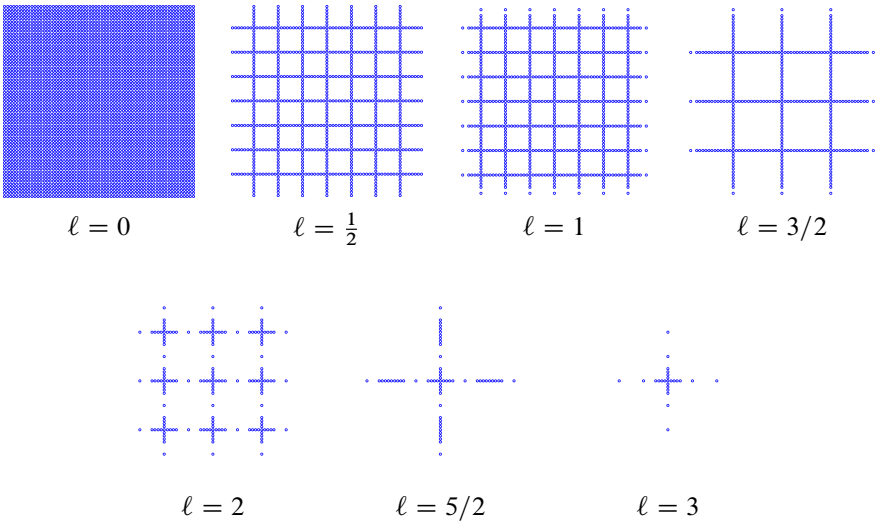


FIGURE 4.1. Active DOFs at each level ℓ of HIF-DE in 2D.

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

Partition Ω into Voronoi cells [3] 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

$$A_{\ell+1} = \mathscr{Z}_{C_{\ell+1/2}}(A_{\ell+1/2}) \approx U_{\ell+1/2}^{\mathsf{T}} A_{\ell+1/2} U_{\ell+1/2},$$
$$U_{\ell+1/2} = \prod_{c \in C_{\ell+1/2}} Q_c S_{\check{c}},$$

where the DOFs $\bigcup_{c \in C_{\ell+1/2}} \check{c}$ have been eliminated. Note that no fill-in is generated since the DOFs $\hat{c}$ for each $c \in C_{\ell+1/2}$ are already connected via SCIs from elimination at level ℓ. The matrix $A_{\ell+1}$ is block diagonal with block partitioning

$$\theta_{\ell+1} = \Big(\bigcup_{c \in C_0} \{c\}\Big) \cup \Big(\bigcup_{c \in C_{1/2}} \{\check{c}\}\Big) \cdots \cup \Big(\bigcup_{c \in C_\ell} \{c\}\Big) \cup \Big(\bigcup_{c \in C_{\ell+1/2}} \{\check{c}\}\Big) \cup \{s_{\ell+1/2}\},$$

where $s_{\ell+1} = s_{\ell+1/2} \setminus \bigcup_{c \in C_\ell} \check{c}$.

Level L

Combining over all levels gives

$$D \equiv A_L \approx U_{L-1/2}^{\mathsf{T}} W_{L-1}^{\mathsf{T}} \cdots U_{1/2}^{\mathsf{T}} W_0^{\mathsf{T}} A W_0 U_{1/2} \cdots W_{L-1} U_{L-1/2}$$

or, more simply,

$$D \approx V_{L-1/2}^{\mathsf{T}} \cdots V_{1/2}^{\mathsf{T}} V_0^{\mathsf{T}} A V_0 V_{1/2} \cdots V_{L-1/2},$$

where

$$V_\ell = \begin{cases} W_\ell, & \ell = 0, 1, \ldots, L-1, \\ U_\ell, & \text{otherwise}, \end{cases} \tag{4.1}$$

so

$$A \approx V_0^{-\mathsf{T}} V_{1/2}^{-\mathsf{T}} \cdots V_{L-1/2}^{-\mathsf{T}} D V_{L-1/2}^{-1} \cdots V_{1/2}^{-1} V_0^{-1} \equiv F, \tag{4.2a}$$

$$A^{-1} \approx V_0 V_{1/2} \cdots V_{L-1/2} D^{-1} V_{L-1/2}^{\mathsf{T}} \cdots V_{1/2}^{\mathsf{T}} V_0^{\mathsf{T}} = F^{-1}. \tag{4.2b}$$

This is a factorization very similar to that in (3.1) except

(1) it has twice as many factors,
(2) it is now an approximation, and
(3) the skeletonization matrices U_ℓ are composed of both upper and lower triangular factors and so are not themselves triangular (but are still easily invertible).

We call (4.2) an approximate generalized LDL decomposition, with F^{-1} serving as a direct solver at high accuracy or as a preconditioner otherwise.

Unlike MF, if A is SPD, then D and hence F now only approximate SPD matrices. The extent of this approximation is governed by Weyl's inequality.

THEOREM 4.1. *If $A, B \in \mathbb{R}^{N \times N}$ are symmetric, then*

$$|\lambda_i(A) - \lambda_i(B)| \le \|A - B\|, \quad i = 1, 2, \ldots, N,$$

where $\lambda_i(\cdot)$ returns the i^{th} largest eigenvalue of a symmetric matrix.

COROLLARY 4.2. *If A is SPD with $F = A + E$ symmetric such that $\|E\| \le \epsilon \|A\|$ for $\epsilon\kappa(A) < 1$, where $\kappa(A) = \|A\|\|A^{-1}\|$ is the condition number of A, then F is SPD.*

PROOF. By Theorem 4.1, $|\lambda_i(A) - \lambda_i(F)| \le \|E\| = \epsilon\|A\|$ for all $i = 1, 2, \ldots, N$, so

$$\left|\frac{\lambda_i(A) - \lambda_i(F)}{\lambda_i(A)}\right| \le \left|\frac{\lambda_i(A) - \lambda_i(F)}{\lambda_N(A)}\right| \le \epsilon\|A\|\|A^{-1}\| = \epsilon\kappa(A). \tag{4.3}$$

This implies that $\lambda_i(F) \ge (1 - \epsilon\kappa(A))\,\lambda_i(A)$, so $\lambda_i(F) > 0$ if $\epsilon\kappa(A) < 1$ since $\lambda_i(A) > 0$ by assumption. □

Remark 4.3. Equation (4.3) actually proves a much more general result, namely that all eigenvalues are approximated to relative precision $\epsilon\kappa(A)$.

The requirement that $\epsilon\kappa(A) < 1$ is necessary for F^{-1} to achieve any accuracy whatsoever and hence is quite weak. Therefore, F is SPD under very mild conditions, in which case (4.2) can be interpreted as a generalized Cholesky decomposition. Its inverse F^{-1} is then also SPD and can be used, e.g., as a preconditioner in CG.

The entire procedure is summarized as Algorithm 4.1.

Algorithm 4.1 HIF-DE.

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

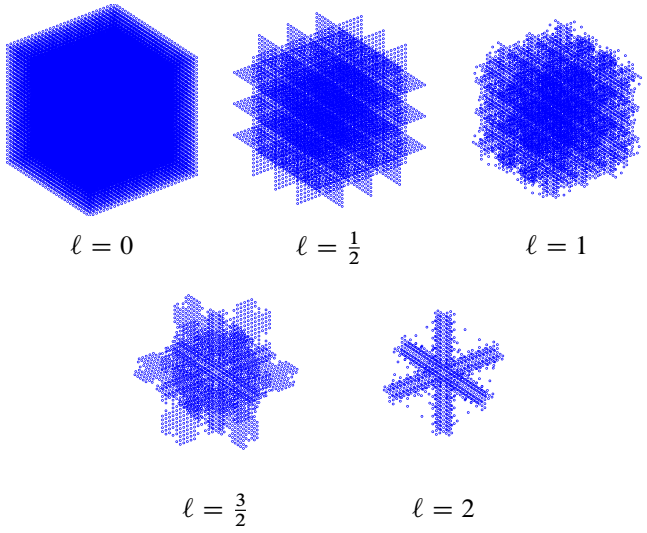


FIGURE 4.2. Active DOFs at each level ℓ of HIF-DE in 3D.

4.2 Three Dimensions

Assume the same setup as in Section 3.2. There are two variants of HIF-DE in 3D: a direct generalization of the 2D algorithm by combining interior cell elimination (3D to 2D) with face skeletonization (2D to 1D) and a more complicated version adding also edge skeletonization (1D to 0D) afterward. We will continue to refer to the former simply as HIF-DE and call the latter "HIF-DE in 3D with edge skeletonization." For unity of presentation, we will discuss only HIF-DE here, postponing the alternative formulation until Section 4.5. Figure 4.2 shows the active DOFs at each level for HIF-DE on a representative example.

Level ℓ

Partition Ω by 2D separators $2^\ell mh(j_1, \cdot, \cdot)$, $2^\ell mh(\cdot, j_2, \cdot)$, and $2^\ell mh(\cdot, \cdot, j_3)$ for $1 \le j_1, j_2, j_3 \le 2^{L-\ell} - 1$ into interior cubic cells

$$2^\ell mh(j_1 - 1, j_1) \times 2^\ell mh(j_2 - 1, j_2) \times 2^\ell mh(j_3 - 1, j_3)$$

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. Elimination with respect to C_ℓ then gives

$$A_{\ell+1/2} = W_\ell^{\mathsf{T}} A_\ell W_\ell, \quad W_\ell = \prod_{c \in \boldsymbol{C}_\ell} S_c,$$

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

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

Partition Ω into Voronoi cells about the face centers

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

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

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

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 a factorization of the form (4.2). The overall procedure is the same as that in Algorithm 4.1.

4.3 Accelerated Compression

A dominant contribution to the cost of HIF-DE is computing IDs for skeletonization. The basic operation required is the construction of an ID of $(A_{\ell+1/2})_{c^{\mathsf{c}},c}$, where $c \in C_{\ell+1/2}$ and $c^{\mathsf{c}} = s_{\ell+1/2} \setminus c$, following Lemma 2.4. We hereafter drop the dependence on ℓ for notational convenience. Note that $A_{c^{\mathsf{c}},c}$ is a tall-and-skinny matrix of size $O(N) \times |c|$, so forming its ID takes at least $O(N|c|)$ work. By construction, however, $A_{c^{\mathsf{c}},c}$ is very sparse and can be written without loss of generality as

$$A_{c^{\mathsf{c}},c} = \begin{bmatrix} A_{c^{\mathsf{N}},c} \\ 0 \end{bmatrix},$$

where the DOFs c^{N} are restricted to the immediately adjacent edges or faces, as appropriate. Thus, $|c^{\mathsf{N}}| = O(|c|)$ and an ID of the much smaller matrix $A_{c^{\mathsf{N}},c}$ of size $O(|c|) \times |c|$ suffices. In other words, the global compression of $A_{c^{\mathsf{c}},c}$ can

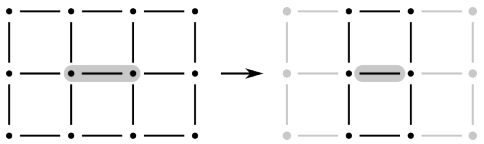


FIGURE 4.3. Accelerated compression by exploiting sparsity. In 2D, the number of neighboring edges (left) that must be included when skeletonizing a given edge (gray outline) can be substantially reduced by restricting to only the interior DOFs of that edge (right). An analogous setting applies for faces in 3D.

be performed via the local compression of $A_{c^{\mathsf{N}},c}$. This observation is critical for reducing the asymptotic complexity.

We can pursue further acceleration by optimizing $|c^{\mathsf{N}}|$ as follows. Consider the reference domain configuration depicted in Figure 4.3, which shows the active DOFs $s_{\ell+1/2}$ after interior cell elimination at level ℓ in 2D. The Voronoi partitioning scheme clearly groups together all interior DOFs of each edge, but those at the corner points $2^{\ell}mh(j_1, j_2)$ for $1 \le j_1, j_2 \le 2^{L-\ell} - 1$ are equidistant to multiple Voronoi centers and can be assigned arbitrarily (or, in fact, not at all). Let $c \in C_{\ell+1/2}$ be a given edge and suppose that it includes both of its endpoints. Then its neighbor set c^{N} includes all immediately adjacent edges as shown (left). But the only DOFs in c that interact with the edges to the left or right are precisely the corresponding corner points. Therefore, we can reduce c^{N} to only those edges belonging to the two cells on either side of the edge defining c by restricting to only its interior DOFs (right); i.e., we exclude from $C_{\ell+1/2}$ all corner points. This can also be interpreted as preselecting the corner points as skeletons (as must be the case because of the sparsity pattern of $A_{c^{\mathsf{c}},c}$) and suitably modifying the remaining computation. In 2D, this procedure lowers the cost of the ID by about a factor of $17/6 = 2.8333\ldots$. In 3D, an analogous situation holds for faces with respect to "corner" edges and the cost is reduced by a factor of $37/5 = 7.4$.

It is also possible to accelerate the ID using fast randomized methods [28] based on compressing $\Phi_c A_{c^{\mathsf{N}},c}$, where Φ_c is a small Gaussian random sampling matrix. However, we did not find a significant improvement in performance and so did not use this optimization in our tests for simplicity (see also Section 4.5).

4.4 Optimal Low-Rank Approximation

Although we have built our algorithms around the ID, it is actually not essential (at least with HIF-DE as presently formulated) and other low-rank approximations can just as well be used. Perhaps the most natural of these is the singular value decomposition (SVD), which is optimal in the sense that it achieves the minimal approximation error for a given rank [19]. Recall that the SVD of a matrix $A \in \mathbb{R}^{m\times n}$ is a factorization of the form $A = U\Sigma V^{\mathsf{T}}$, where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$

are orthogonal, and $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal with the singular values of A as its entries. The following is the analogue of Corollary 2.3 using the SVD.

LEMMA 4.4. *Let $A \in \mathbb{R}^{m\times n}$ with rank $k \le \min(m,n)$ and SVD*

$$A = U\Sigma V^{\mathsf{T}} = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} 0 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^{\mathsf{T}},$$

where $\Sigma_2 \in \mathbb{R}^{k\times k}$. Then

$$U^{\mathsf{T}}A = \Sigma V^{\mathsf{T}} = \begin{bmatrix} 0 \\ \Sigma_2 V_2^{\mathsf{T}} \end{bmatrix}, \quad AV = U\Sigma = \begin{bmatrix} 0 & U_2\Sigma_2 \end{bmatrix}.$$

The analogue of Lemma 2.4 is then the following:

LEMMA 4.5. *Let*

$$A = \begin{bmatrix} A_{pp} & A_{qp}^{\mathsf{T}} \\ A_{qp} & A_{qq} \end{bmatrix}$$

be symmetric for A_{qp} low-rank with SVD

$$A_{qp} = A_{p^c,p} = U_p \Sigma_p V_p^{\mathsf{T}} = \begin{bmatrix} U_{p,1} & U_{p,2} \end{bmatrix} \begin{bmatrix} 0 & \\ & \Sigma_{p,2} \end{bmatrix} \begin{bmatrix} V_{p,1} & V_{p,2} \end{bmatrix}^{\mathsf{T}}.$$

If $Q_p = \operatorname{diag}(V_p, I)$, then

$$\begin{aligned} Q_p^{\mathsf{T}} A Q_p &= \begin{bmatrix} V_p^{\mathsf{T}} A_{pp} V_p & \begin{bmatrix} 0 \\ \Sigma_{p,2} U_{p,2}^{\mathsf{T}} \end{bmatrix} \\ \begin{bmatrix} 0 & U_{p,2}\Sigma_{p,2} \end{bmatrix} & A_{qq} \end{bmatrix} \\ &\equiv \begin{bmatrix} B_{p_1,p_1} & B_{p_2,p_1}^{\mathsf{T}} & \\ B_{p_2,p_1} & B_{p_2,p_2} & \Sigma_{p,2} U_{p,2}^{\mathsf{T}} \\ & U_{p,2}\Sigma_{p,2} & A_{qq} \end{bmatrix} \end{aligned} \tag{4.4}$$

on conformably partitioning $p = p_1 \cup p_2$, so

$$S_{p_1}^{\mathsf{T}} Q_p^{\mathsf{T}} A Q_p S_{p_1} = \begin{bmatrix} D_{p_1} & & \\ & \widetilde{B}_{p_2,p_2} & \Sigma_{p,2} U_{p,2}^{\mathsf{T}} \\ & U_{p,2}\Sigma_{p,2} & A_{qq} \end{bmatrix},$$

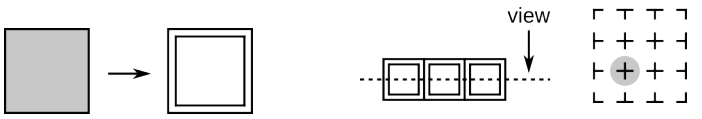
where S_{p_1} is the elimination operator of Lemma 2.1 *associated with p_1 and*

$$\widetilde{B}_{p_2,p_2} = B_{p_2,p_2} - B_{p_2,p_1} B_{p_1,p_1}^{-1} B_{p_2,p_1}^{\mathsf{T}},$$

where we assume that B_{p_1,p_1} is nonsingular.

The external interactions $U_{p,2}\Sigma_{p,2}$ with the SVD "skeletons" p_2 are a linear combination of the original external interactions A_{qp} involving all of p. Thus, the DOFs p_2 are, in a sense, delocalized across all points associated with p, though they can still be considered to reside on the separators.

The primary advantages of using the SVD over the ID are that (1) it can achieve better compression since a smaller rank may be required for any given precision



(A) Face skeletonization. (B) Edge configuration from top view at interior slice.

FIGURE 4.4. Loss of sparsity from edge skeletonization in 3D. Face skeletonization (left) typically leaves several layers of DOFs along the perimeter, which lead to thick edges (right) that connect DOFs across cubic cell boundaries upon skeletonization (3×3 grid of cells shown in example with a thick edge outlined in gray).

and (2) the sparsification matrix Q_p in (4.4) is orthogonal, which provides improved numerical stability, especially when used in a multilevel setting such as (4.2). However, there are several disadvantages as well, chief among them:

- the extra computational cost, which typically is about 2 to 3 times larger;
- the need to overwrite matrix entries involving the index set q in (4.4), which we remark is still sparse; and
- the loss of precise geometrical information associated with each DOF.

Of these, the last is arguably the most important since it destroys the dimensional reduction interpretation of HIF-DE, which is crucial for achieving estimated $O(N)$ complexity in 3D, as we shall see next.

4.5 Three-Dimensional Variant with Edge Skeletonization

In Section 4.2, we presented a "basic" version of HIF-DE in 3D based on interior cell elimination and face skeletonization, which from Figure 4.2 is seen to retain active DOFs only on the edges of cubic cells. All fronts are hence reduced to 1D, which yields estimated $O(N \log N)$ complexity for the algorithm (Section 4.6). Here, we seek to further accelerate this to $O(N)$ by skeletonizing each cell edge and reducing it completely to 0D, as guided by our assumptions on SCIs. However, a complication now arises in that fill-in can occur, which can be explained as follows.

Consider the 3D problem and suppose that both interior cell elimination and face skeletonization have been performed. Then as noted in Section 4.3, the remaining DOFs with respect to each face will be those on its boundary edges plus a few interior layers near the edges (Figure 4.4(A)) (the depth of these layers depends on the compression tolerance ϵ). Therefore, grouping the active DOFs by cell edge gives "thick" edges consisting not only of the DOFs on the edges themselves but also those in the interior layers in the four transverse directions surrounding each edge (Figure 4.4(B)). Skeletonizing these thick edges then generates SCIs acting on the skeletons of each edge group by Lemma 2.4, which generally causes DOFs to interact across cubic cell boundaries. The consequence of this is that the next level of interior cell elimination must take into account, in effect, thick separators

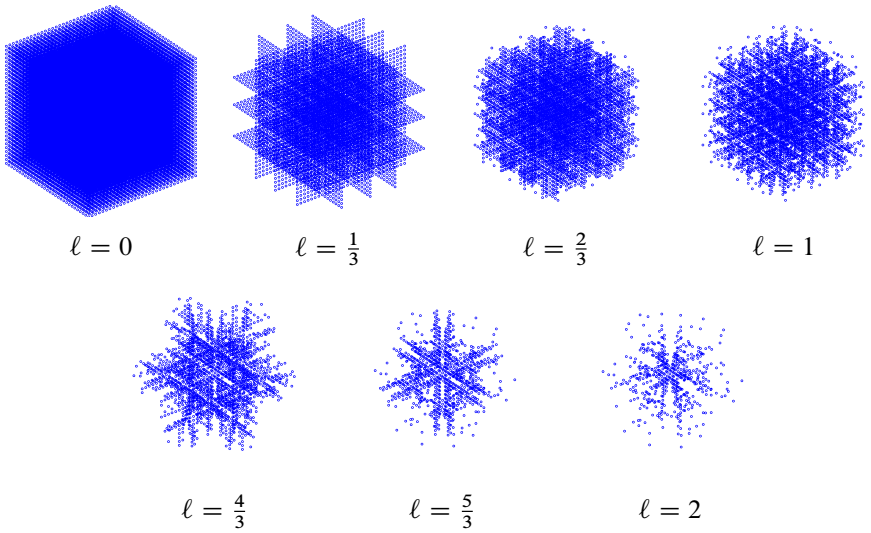


FIGURE 4.5. Active DOFs at each level ℓ of HIF-DE in 3D with edge skeletonization.

of width twice the layer depth, which can drastically reduce the number of DOFs eliminated and thus increase the cost. Of course, this penalty does not apply at any level ℓ before edge skeletonization has occurred. As a rule of thumb, edge skeletonization should initially be skipped until it reduces the number of active DOFs by a factor of at least the resulting separator width.

For completeness, we now describe HIF-DE in 3D with edge skeletonization following the structure of Section 4.2, where interior cell elimination (3D to 2D) at level ℓ is supplemented with face skeletonization (2D to 1D) at level $\ell + \frac{1}{3}$ and edge skeletonization (1D to 0D) at level $\ell + \frac{2}{3}$ for each $\ell = 0, 1, \ldots, L-1$. Figure 4.5 shows the active DOFs at each level for a representative example, from which we observe that further compression is clearly achieved on comparing with Figure 4.2.

Level ℓ

Partition Ω by separators into interior cells. If $\ell = 0$, then these are the same as those in the standard HIF-DE (Section 4.2), but if $\ell \geq 1$, then this must, in general, be done somewhat more algebraically according to the sparsity pattern of A_ℓ. We propose the following procedure. First, partition all active DOFs 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 $1 \leq j_1, j_2, j_3 \leq 2^{L-\ell}$. This creates an initial geometric partitioning C_ℓ, which we remark is unbuffered (no separators) and so does not satisfy the hypotheses of Section 2.1. Then for each $c \in C_\ell$ in some order:

(1) Let

$$c^{\mathsf{E}} = \{i \in c : (A_\ell)_{c^{\mathsf{c}},i} \neq 0\}, \quad c^{\mathsf{c}} = \Big(\bigcup_{c' \in \boldsymbol{C}_\ell} c' \Big) \setminus c$$

be the set of indices of the DOFs in c with external interactions.

(2) Replace c by $c \setminus c^{\mathsf{E}}$ in $\boldsymbol{C}_\ell$.

On termination, this process produces a collection $\boldsymbol{C}_\ell$ of interior cells with minimal separators adaptively constructed. Elimination with respect to $\boldsymbol{C}_\ell$ then gives

$$A_{\ell+1/3} = W_\ell^{\mathsf{T}} A_\ell W_\ell, \quad W_\ell = \prod_{c \in \boldsymbol{C}_\ell} S_c,$$

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

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

Partition Ω into Voronoi cells about the face centers

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

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

$$\begin{aligned}
A_{\ell+2/3} &= \mathscr{Z}_{\boldsymbol{C}_{\ell+1/3}}(A_{\ell+1/3}) \approx U_{\ell+1/3}^{\mathsf{T}} A_{\ell+1/3} U_{\ell+1/3},\\
U_{\ell+1/3} &= \prod_{c \in \boldsymbol{C}_{\ell+1/3}} Q_c S_{\check{c}},
\end{aligned}$$

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

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

Partition Ω into Voronoi cells about the edge centers

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

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

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

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

Level L

Combining the approximation over all levels gives

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

where V_ℓ is as defined in (4.1), so

$$A \approx V_0^{-\mathsf{T}} V_{1/3}^{-\mathsf{T}} V_{2/3}^{-\mathsf{T}} \cdots V_{L-1/3}^{-\mathsf{T}} D V_{L-1/3}^{-1} \cdots V_{2/3}^{-1} V_{1/3}^{-1} V_0^{-1} \equiv F, \tag{4.5a}$$

$$A^{-1} \approx V_0 V_{1/3} V_{2/3} \cdots V_{L-1/3} D^{-1} V_{L-1/3}^{\mathsf{T}} \cdots V_{2/3}^{\mathsf{T}} V_{1/3}^{\mathsf{T}} V_0^{\mathsf{T}} = F^{-1}. \tag{4.5b}$$

As in Section 4.1, if A is SPD, then so are F and F^{-1} provided that very mild conditions hold. We summarize the overall scheme as Algorithm 4.2.

Algorithm 4.2 HIF-DE in 3D with edge skeletonization.

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

Unlike for the standard HIF-DE, randomized methods (Section 4.3) now tend to be inaccurate when compressing SCIs. This could be remedied by considering instead $\Phi_c (A_{c^{\mathsf{N}},c} A_{c^{\mathsf{N}},c}^{\mathsf{T}})^\gamma A_{c^{\mathsf{N}},c}$ for some small integer $\gamma = 1, 2, \ldots$, but the expense of the extra multiplications usually outweighed any efficiency gains.

4.6 Complexity Estimates

We now investigate the computational complexity of HIF-DE. For this, we need to estimate the skeleton size $|\widehat{c}|$ for a typical index set $c \in C_\ell$ at fractional level ℓ. This is determined by the rank behavior of SCIs, which we assume satisfy standard multipole estimates [22, 23] as motivated by experimental observations. Then it can be shown [30, 31] that the typical skeleton size is

$$k_\ell = \begin{cases} O(\ell), & \delta = 1, \\ O(2^{(\delta-1)\ell}), & \delta \geq 2, \end{cases} \tag{4.6}$$

where δ is the intrinsic dimension of a typical DOF cluster at level ℓ, i.e., $\delta = 1$ for edges (2D and 3D) and $\delta = 2$ for faces (3D only). Note that we have suggestively used the same notation as for the index set size $|c|$ in Section 3.3, which can be justified by recognizing that the active DOFs $c \in C_\ell$ for any ℓ are obtained by merging skeletons from at most one integer level prior. We emphasize that (4.6) has yet to be proven, so all following results should formally be understood as conjectures, albeit ones with strong numerical support (Section 5).

THEOREM 4.6. *Assume that* (4.6) *holds. Then the costs of constructing the factorization* F *in* (4.2) *or* (4.5) *using HIF-DE with accelerated compression and of applying* F *or* F^{-1} *are, respectively,* $t_f, t_{a/s} = O(N)$ *in 2D;* $t_f = O(N \log N)$ *and* $t_{a/s} = O(N)$ *in 3D; and* $t_f, t_{a/s} = O(N)$ *in 3D with edge skeletonization.*

PROOF. The costs of constructing and applying the factorization are clearly

$$t_f = \sum_{\ell=0}^{L}{}' O\big(2^{d(L-\ell)} k_\ell^3\big), \quad t_{a/s} = \sum_{\ell=0}^{L}{}' O\big(2^{d(L-\ell)} k_\ell^2\big),$$

where prime notation denotes summation over all levels, both integer and fractional, and k_ℓ is as given by (4.6) for δ appropriately chosen. In 2D, all fronts are reduced to 1D edges, so $\delta = 1$; in 3D, compression on 2D faces has $\delta = 2$; and in 3D with edge skeletonization, we again have $\delta = 1$. The claim follows by direct computation. □

5 Numerical Results

In this section, we demonstrate the efficiency of HIF-DE 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 MF as mf2 in 2D and mf3 in 3D. Similarly, we call HIF-DE hifde2 and hifde3, respectively, and denote by hifde3x the 3D variant with edge skeletonization. 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 roughly half of the initial DOFs are eliminated. In select cases, the first few fractional levels of HIF-DE were skipped to optimize the running time. Diagonal blocks, i.e., A_{pp} in Lemma 2.1, were factored using the Cholesky decomposition if A is SPD and the (partially pivoted) LDL decomposition otherwise.

For each example, the following are given:

- ϵ: 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;

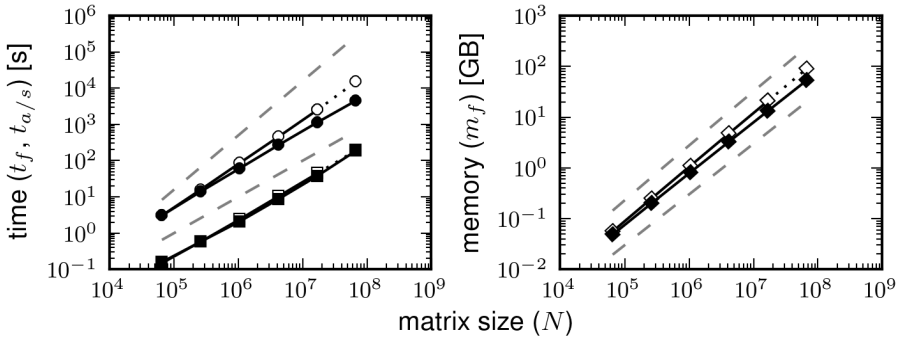


FIGURE 5.1. Scaling results for Example 1. Wall clock times t_f ($\circ$) and $t_{a/s}$ ($\square$) and storage requirements m_f ($\diamond$) are shown for mf2 (white) and hifde2 (black) at precision $\epsilon = 10^{-9}$. Dotted lines denote extrapolated values. Included also are reference scalings (gray 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.

- 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 (1.2) using CG [29, 41] (if SPD) or GMRES [37] with preconditioner F^{-1} to a tolerance of 10^{-12}, where f is a standard uniform random vector.

We also compare against MF, which is numerically exact.

The operator errors e_a and e_s were estimated using power iteration with a standard uniform random start vector [11, 32] and a convergence criterion of 10^{-2} relative precision in the matrix norm. This has a small probability of underestimating the error but seems to be quite robust in practice.

For simplicity, all PDEs were defined over $\Omega = (0, 1)^d$ with (arbitrary) Dirichlet boundary conditions as in Section 3, discretized on a uniform $n \times n$ or $n \times n \times n$ mesh using second-order central differences via the five-point stencil in 2D and the seven-point stencil in 3D.

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 $a(x) \equiv 1$, $b(x) \equiv 0$, and $\Omega = (0, 1)^2$, i.e., a simple Laplacian in the unit square. The resulting matrix A is SPD, which we factored using both mf2 and hifde2 at $\epsilon = 10^{-6}$, 10^{-9}, and 10^{-12} (the compression tolerances are for HIF-DE only). The data are summarized in Tables 5.1 and 5.2 with scaling results shown in Figure 5.1.

TABLE 5.1. Factorization results for Example 1.

		mf2			hifde2		
ϵ	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-06}	1023^2	—	—	—	56	5.5e+1	7.9e−1
	2047^2	—	—	—	57	2.4e+2	3.2e+0
	4095^2	—	—	—	57	1.0e+3	1.3e+1
	8191^2	—	—	—	52	4.0e+3	5.1e+1
10^{-09}	1023^2	—	—	—	85	6.1e+1	8.2e−1
	2047^2	—	—	—	93	2.7e+2	3.3e+0
	4095^2	—	—	—	99	1.1e+3	1.3e+1
	8191^2	—	—	—	102	4.5e+3	5.3e+1
10^{-12}	1023^2	—	—	—	114	6.7e+1	8.4e−1
	2047^2	—	—	—	125	2.9e+2	3.4e+0
	4095^2	—	—	—	134	1.3e+3	1.4e+1
	8191^2	—	—	—	144	5.1e+3	5.5e+1
—	1023^2	2045	8.6e+1	1.1e+0	—	—	—
	2047^2	4093	4.5e+2	4.8e+0	—	—	—
	4095^2	8189	2.5e+3	2.1e+1	—	—	—

TABLE 5.2. Matrix application results for Example 1.

		mf2	hifde2			
ϵ	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i
10^{-06}	1023^2	—	2.1e+0	8.3e−06	2.4e−03	6
	2047^2	—	9.1e+0	2.1e−05	1.5e−02	7
	4095^2	—	3.9e+1	8.4e−05	2.6e−01	13
	8191^2	—	1.8e+2	1.1e−04	6.3e−01	15
10^{-09}	1023^2	—	2.1e+0	5.5e−09	8.7e−07	4
	2047^2	—	8.6e+0	1.4e−08	6.5e−06	4
	4095^2	—	3.8e+1	2.9e−08	2.0e−05	4
	8191^2	—	1.9e+2	5.3e−08	1.0e−04	3
10^{-12}	1023^2	—	2.1e+0	5.5e−12	8.0e−10	3
	2047^2	—	8.9e+0	9.4e−12	3.0e−09	3
	4095^2	—	4.2e+1	3.2e−11	2.2e−08	3
	8191^2	—	1.8e+2	6.3e−11	6.5e−08	3
—	1023^2	2.4e+0	—	—	—	—
	2047^2	1.0e+1	—	—	—	—
	4095^2	4.4e+1	—	—	—	—

It is evident that $|s_L| \sim k_L$ behaves as predicted, with HIF-DE achieving significant compression over MF (but, of course, at the cost of introducing approximation error). Consequently, we find strong support for asymptotic complexities consistent with Theorems 3.1 and 4.6, though MF scales much better than predicted due to its favorable constants. We remark that obtaining a speedup in 2D is not our primary goal since MF is already so efficient in this regime. Still, we see a modest

FIGURE 5.2. Sample realization of a quantized high-contrast random field in 2D.

increase in performance and memory savings that allow us to run HIF-DE up to $N = 8191^2$, for which MF was not successful.

For all problem sizes tested, t_f and m_f are always smaller for HIF-DE, though $t_{a/s}$ is quite comparable. This is 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 $e_a = O(\epsilon)$ for all N and seems to increase only mildly 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 orthogonal. On the other hand, we expect the inverse approximation error to scale as $e_s = O(\kappa(A)e_a)$, where $\kappa(A) = O(N)$ for this example, and indeed we see that e_s is much larger due to ill-conditioning. When using F^{-1} to precondition CG, 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 $a(x)$ a quantized high-contrast random field defined as follows:

(1) Initialize by sampling each staggered grid point a_j from the standard uniform distribution.
(2) Impose some correlation structure by convolving $\{a_j\}$ with an isotropic Gaussian of width $4h$.
(3) Quantize by setting

$$a_j = \begin{cases} 10^{-2}, & a_j \leq \mu, \\ 10^{+2}, & a_j > \mu, \end{cases}$$

where μ is the median of $\{a_j\}$.

Figure 5.2 shows a sample realization of such a high-contrast random field in 2D. The matrix A now has condition number $\kappa(A) = O(\rho N)$, where $\rho = 10^4$ is the contrast ratio. Such high-contrast problems are typically extremely difficult to

TABLE 5.3. Factorization results for Example 2.

		mf2			hifde2		
ϵ	N	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-09}	1023^2	—	—	—	97	6.5e+1	8.3e−1
	2047^2	—	—	—	110	2.8e+2	3.3e+0
	4095^2	—	—	—	113	1.2e+3	1.3e+1
	8191^2	—	—	—	141	4.6e+3	5.4e+1
10^{-12}	1023^2	—	—	—	134	7.4e+1	8.7e−1
	2047^2	—	—	—	148	3.2e+2	3.5e+0
	4095^2	—	—	—	160	1.4e+3	1.4e+1
	8191^2	—	—	—	191	5.5e+3	5.7e+1
—	1023^2	2045	8.4e+1	1.1e+0	—	—	—
	2047^2	4093	4.6e+2	4.8e+0	—	—	—
	4095^2	8189	2.5e+3	2.1e+1	—	—	—

TABLE 5.4. Matrix application results for Example 2.

		mf2	hifde2			
ϵ	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i
10^{-09}	1023^2	—	2.3e+0	3.1e−09	2.0e−4	3
	2047^2	—	9.3e+0	2.5e−09	2.4e−4	3
	4095^2	—	3.9e+1	3.4e−08	3.1e−4	8
	8191^2	—	1.9e+2	4.5e−09	1.2e−3	4
10^{-12}	1023^2	—	2.3e+0	1.8e−12	1.7e−7	2
	2047^2	—	8.9e+0	2.3e−12	3.0e−7	2
	4095^2	—	4.0e+1	3.5e−12	5.8e−7	2
	8191^2	—	1.9e+2	4.5e−12	6.0e−7	2
—	1023^2	2.5e+0	—	—	—	—
	2047^2	1.0e+1	—	—	—	—
	4095^2	4.1e+1	—	—	—	—

solve by iteration. Data for mf2 and hifde2 at $\epsilon = 10^{-9}$ and 10^{-12} are given in Tables 5.3 and 5.4.

As expected, factorization results for MF are essentially the same as those in Example 1 since the elimination procedure is identical. Results are also very similar for HIF-DE, with only slightly increased skeleton sizes, presumably to resolve the more detailed structure of $a(x)$. Thus, high-contrast problems do not appear to pose any challenge. However, e_s naturally suffers due to the additional ill-conditioning, though F^{-1} remains a very good preconditioner for CG in all cases tested.

Example 3. We then turn to the Helmholtz equation (1.1) with $a(x) \equiv 1$ and $b(x) \equiv -k^2$, where $k = 2\pi\kappa$ is the wave frequency for κ the number of wavelengths in Ω. We kept a fixed number of 32 DOFs per wavelength by increasing k with $n = \sqrt{N}$. The resulting matrix is *indefinite* and was factored using both mf2 and hifde2 with $\kappa = 32$, 64, and 128 at $\epsilon = 10^{-6}$, 10^{-9}, and 10^{-12}. Since A

TABLE 5.5. Factorization results for Example 3.

			mf2			hifde2		
ϵ	N	κ	$\lvert s_L\rvert$	t_f	m_f	$\lvert s_L\rvert$	t_f	m_f
10^{-06}	1023^2	32	—	—	—	156	5.7e+1	1.2e+0
	2047^2	64	—	—	—	271	2.4e+2	4.8e+0
	4095^2	128	—	—	—	408	1.0e+3	1.9e+1
10^{-09}	1023^2	32	—	—	—	180	6.2e+1	1.2e+0
	2047^2	64	—	—	—	286	2.7e+2	4.9e+0
	4095^2	128	—	—	—	442	1.2e+3	2.0e+1
10^{-12}	1023^2	32	—	—	—	207	6.9e+1	1.3e+0
	2047^2	64	—	—	—	310	3.0e+2	5.1e+0
	4095^2	128	—	—	—	482	1.3e+3	2.0e+1
—	1023^2	32	2045	1.1e+2	1.6e+0	—	—	—
	2047^2	64	4093	7.2e+2	7.1e+0	—	—	—
	4095^2	128	8189	4.9e+3	3.0e+1	—	—	—

TABLE 5.6. Matrix application results for Example 3.

			mf2	hifde2			
ϵ	N	κ	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i
10^{-06}	1023^2	32	—	2.4e+0	5.5e−06	3.1e−3	4
	2047^2	64	—	9.7e+0	6.8e−06	7.4e−3	7
	4095^2	128	—	4.0e+1	3.7e−05	2.3e−2	10
10^{-09}	1023^2	32	—	2.3e+0	3.8e−09	2.7e−6	2
	2047^2	64	—	9.5e+0	5.9e−09	2.9e−5	6
	4095^2	128	—	3.7e+1	3.5e−08	8.5e−6	6
10^{-12}	1023^2	32	—	2.2e+0	4.8e−12	3.9e−9	2
	2047^2	64	—	9.6e+0	5.7e−12	1.1e−8	2
	4095^2	128	—	4.3e+1	6.8e−11	1.7e−8	2
—	1023^2	32	2.6e+0	—	—	—	—
	2047^2	64	1.1e+1	—	—	—	—
	4095^2	128	4.6e+1	—	—	—	—

is no longer SPD, F^{-1} now applies as a preconditioner for GMRES. The data are summarized in Tables 5.5 and 5.6 with scaling results in Figure 5.3.

Overall, the results are very similar to those in Example 1 but with larger skeleton sizes and some extra ill-conditioning of order $O(k)$. We remark, however, that HIF-DE is effective only at low to moderate frequency since the rank structures employed break down as $k \to \infty$. This can be understood by analogy with the Helmholtz Green's function, whose off-diagonal blocks are full-rank in the limit (though other rank structures are possible [13, 14]). Indeed, we can already see an increasing trend in $|s_L|$ beyond that observed in Examples 1 and 2. In the high-frequency regime, the only compression available is due to sparsity, with HIF-DE

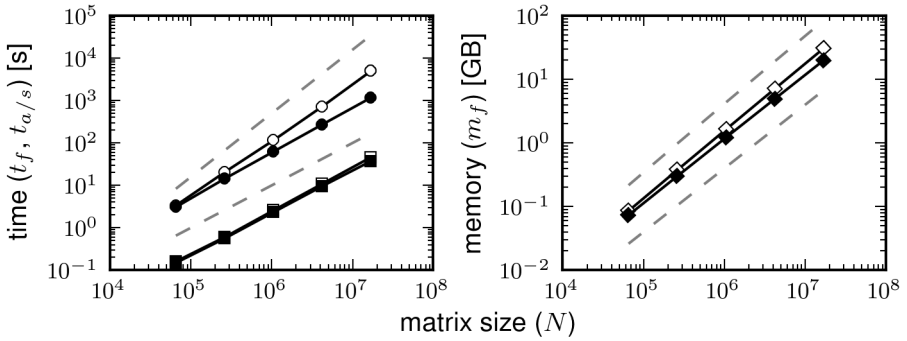


FIGURE 5.3. Scaling results for Example 3, comparing mf2 (white) with hifde2 (black) at precision $\epsilon = 10^{-9}$; all other notation as in Figure 5.1.

TABLE 5.7. Factorization results for Example 4.

		mf3			hifde3			hifde3x		
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}	31^3	—	—	—	950	1.0e+1	1.1e−1	331	1.0e+1	9.4e−2
	63^3	—	—	—	2019	1.9e+2	1.2e+0	578	1.7e+2	9.6e−1
	127^3	—	—	—	4153	2.8e+3	1.3e+1	890	2.2e+3	9.0e+0
10^{-6}	31^3	—	—	—	1568	1.1e+1	1.2e−1	931	1.1e+1	1.0e−1
	63^3	—	—	—	3607	3.0e+2	1.7e+0	2466	3.2e+2	1.3e+0
	127^3	—	—	—	7651	6.2e+3	2.0e+1	3562	6.2e+3	1.6e+1
10^{-9}	31^3	—	—	—	2030	1.3e+1	1.3e−1	1495	1.3e+1	1.1e−1
	63^3	—	—	—	5013	4.3e+2	2.0e+0	4295	4.7e+2	1.6e+0
	127^3	—	—	—	11037	1.1e+4	2.6e+1	7288	1.1e+4	2.1e+1
—	31^3	2791	1.6e+1	1.6e−1	—	—	—	—	—	—
	63^3	11719	8.2e+2	3.0e+0	—	—	—	—	—	—

essentially reducing to MF. Nonetheless, our results reveal no significant apparent failure and demonstrate that HIF-DE achieves linear complexity up to at least $\kappa \sim 10^2$.

5.2 Three Dimensions

We next present three examples in 3D generalizing each of the 2D cases above.

Example 4. Consider the 3D analogue of Example 1, i.e., (1.1) with $a(x) \equiv 1$, $b(x) \equiv 0$, and $\Omega = (0,1)^3$. Data for mf3, hifde3, and hifde3x at $\epsilon = 10^{-3}$, 10^{-6}, and 10^{-9} are given in Tables 5.7 and 5.8 with scaling results shown in Figure 5.4.

It is immediate that $t_f = O(N^2)$ and $t_{a/s} = O(N^{4/3})$ for MF, which considerably degrades its performance for large N. Indeed, we were unable to run mf3 for $N = 127^3$ because of the excessive memory cost. In contrast, HIF-DE scales much better, with $|s_L|$ growing consistently with (4.6) for both variants. This provides strong evidence for Theorem 4.6. However, the skeleton size is substantially larger than in 2D, and neither hifde3 nor hifde3x quite achieve quasilinear complexity as

TABLE 5.8. Matrix application results for Example 4.

		mf3	hifde3				hifde3x			
ϵ	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i	$t_{a/s}$	e_a	e_s	n_i
10^{-3}	31^3	—	1.8e−1	2.1e−03	5.6e−2	7	1.5e−1	3.6e−03	7.0e−2	8
	63^3	—	1.8e+0	5.0e−03	3.4e−1	11	1.3e+0	4.3e−03	3.3e−1	11
	127^3	—	1.9e+1	7.8e−03	7.5e−1	19	1.2e+1	4.8e−03	6.8e−1	17
10^{-6}	31^3	—	1.9e−1	8.5e−07	7.5e−6	3	1.4e−1	9.8e−07	9.8e−6	3
	63^3	—	2.1e+0	3.9e−06	5.8e−5	3	1.4e+0	2.5e−06	4.4e−5	3
	127^3	—	2.6e+1	2.3e−05	1.3e−3	4	1.9e+1	9.1e−06	2.6e−4	4
10^{-9}	31^3	—	1.5e−1	6.1e−10	3.4e−9	2	1.6e−1	7.4e−10	4.0e−9	2
	63^3	—	2.0e+0	4.0e−09	3.5e−8	2	1.7e+0	2.0e−09	2.1e−8	2
	127^3	—	3.3e+1	1.7e−08	4.6e−7	2	2.7e+1	6.4e−09	1.5e−7	2
—	31^3	2.0e−1	—	—	—	—	—	—	—	—
	63^3	3.3e+0	—	—	—	—	—	—	—	—

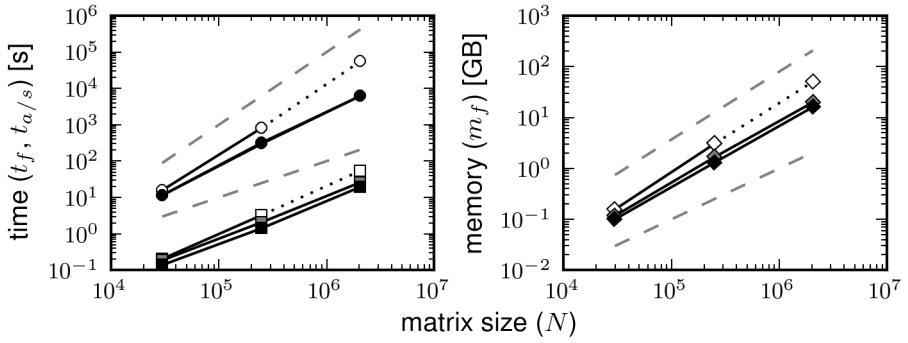


FIGURE 5.4. Scaling results for Example 4, comparing mf3 (white) with hifde3 (gray) and hifde3x (black) at precision $\epsilon = 10^{-6}$. Included also are reference scalings of $O(N)$ and $O(N^2)$ (left), and $O(N)$ and $O(N^{4/3})$ (right); all other notation as in Figure 5.1. The lines for hifde3 and hifde3x lie nearly on top of each other; for t_f (top left), they overlap almost exactly.

predicted: the empirical scaling for t_f for both algorithms at, e.g., $\epsilon = 10^{-6}$ is approximately $O(N^{1.4})$. We believe this to be a consequence of the large interaction ranks, which make the asymptotic regime rather difficult to reach. In parallel with Example 1, $e_a = O(\epsilon)$ but e_s is somewhat larger due to ill-conditioning. We found F^{-1} to be a very effective preconditioner throughout. There were no significant differences in either computation time or accuracy between hifde3 and hifde3x, though the latter does provide some appreciable memory savings.

Example 5. Now consider the 3D analogue of Example 2, i.e., Example 4 but with $a(x)$ a quantized high-contrast random field as previously defined, extended to 3D

TABLE 5.9. Factorization results for Example 5.

		mf3			hifde3			hifde3x		
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^{-6}	31^3	—	—	—	1441	1.1e+1	1.1e−1	948	1.1e+1	1.0e−1
	63^3	—	—	—	3271	2.5e+2	1.5e+0	2337	2.8e+2	1.2e+0
	127^3	—	—	—	6679	4.9e+3	1.7e+1	3294	4.9e+3	1.4e+1
10^{-9}	31^3	—	—	—	1893	1.2e+1	1.2e−1	1423	1.3e+1	1.1e−1
	63^3	—	—	—	4755	3.6e+2	1.8e+0	3924	4.0e+2	1.4e+0
	127^3	—	—	—	10913	9.4e+3	2.4e+1	7011	9.9e+3	1.9e+1
—	31^3	2791	1.5e+1	1.6e−1	—	—	—	—	—	—
	63^3	11719	8.4e+2	3.0e+0	—	—	—	—	—	—

TABLE 5.10. Matrix application results for Example 5.

		mf3	hifde3				hifde3x			
ϵ	N	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i	$t_{a/s}$	e_a	e_s	n_i
10^{-6}	31^3	—	1.8e−1	5.1e−07	6.1e−3	6	1.6e−1	6.4e−07	1.1e−2	5
	63^3	—	2.0e+0	2.1e−06	6.4e−2	7	1.6e+0	1.5e−06	5.8e−2	12
	127^3	—	2.2e+1	8.8e−06	3.4e−1	16	1.6e+1	6.0e−06	3.3e−1	16
10^{-9}	31^3	—	1.9e−1	3.3e−10	1.5e−5	4	1.4e−1	3.8e−10	1.3e−5	4
	63^3	—	2.2e+0	1.6e−09	1.7e−4	6	1.8e+0	1.9e−09	1.7e−4	4
	127^3	—	3.1e+1	1.8e−08	3.7e−3	8	2.3e+1	1.2e−08	3.5e−3	8
—	31^3	2.0e−1	—	—	—	—	—	—	—	—
	63^3	3.4e+0	—	—	—	—	—	—	—	—

in the natural way. Data for mf3, hifde3, and hifde3x at $\epsilon = 10^{-6}$ and 10^{-9} are given in Tables 5.9 and 5.10.

Again, the results are quite similar to those in Example 5 but with e_s necessarily larger by a factor of about ρ due to ill-conditioning. There are no evident difficulties arising from the high contrast ratio for either hifde3 or hifde3x.

Example 6. Finally, we consider the 3D analogue of Example 3, where now $k = 2\pi\kappa$ is increased in proportion to $n = N^{1/3}$ at a fixed resolution of 8 DOFs per wavelength. The matrix A is once again indefinite, which we factored using mf3, hifde3, and hifde3x with $\kappa = 4$, 8, and 16 at $\epsilon = 10^{-6}$ and 10^{-9}. The data are summarized in Tables 5.11 and 5.12 with scaling results in Figure 5.5.

All algorithms behave essentially as expected, but the skeleton size is substantially larger for hifde3 than in the Laplace case (Example 4). The same increase, however, was not observed for hifde3x. We take this to imply that the 1D nature of hifde3x is less sensitive to the oscillatory character of the Helmholtz problem, at least at low frequency, though any definitive conclusion is difficult to draw. The empirical complexity at $\epsilon = 10^{-6}$ is now $t_f \simeq O(N^{1.5})$ for hifde3 and $O(N^{1.3})$ for hifde3x. Both solvers remain quite favorable compared to mf3 and give very good preconditioners for GMRES.

TABLE 5.11. Factorization results for Example 6

			mf3			hifde3			hifde3x		
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
10^{-6}	31^3	4	—	—	—	1702	2.4e+1	1.8e−1	1215	1.8e+1	1.5e−1
	63^3	8	—	—	—	4275	8.2e+2	2.5e+0	2934	5.1e+2	1.9e+0
	127^3	16	—	—	—	10683	1.9e+4	3.0e+1	4071	8.1e+3	2.2e+1
10^{-9}	31^3	4	—	—	—	2144	3.7e+1	2.1e−1	1685	2.5e+1	1.7e−1
	63^3	8	—	—	—	5614	1.3e+3	3.1e+0	4684	9.3e+2	2.3e+0
	127^3	16	—	—	—	14088	3.4e+4	3.9e+1	7806	1.7e+4	2.9e+1
—	31^3	4	2791	6.4e+1	2.5e−1	—	—	—	—	—	—
	63^3	8	11719	5.5e+3	4.9e+0	—	—	—	—	—	—

TABLE 5.12. Matrix application results for Example 6.

			mf3	hifde3				hifde3x			
ϵ	N	κ	$t_{a/s}$	$t_{a/s}$	e_a	e_s	n_i	$t_{a/s}$	e_a	e_s	n_i
10^{-6}	31^3	4	—	1.6e−1	8.2e−07	3.4e−6	3	1.7e−1	6.5e−07	1.3e−5	3
	63^3	8	—	2.0e+0	2.4e−06	3.3e−5	3	1.8e+0	2.0e−06	4.5e−5	3
	127^3	16	—	3.0e+1	3.7e−06	1.3e−3	8	2.1e+1	9.7e−06	4.7e−4	4
10^{-9}	31^3	4	—	1.9e−1	5.0e−10	2.4e−9	2	1.7e−1	5.9e−10	1.1e−8	2
	63^3	8	—	2.4e+0	1.7e−09	2.1e−8	2	2.2e+0	2.0e−09	3.2e−8	2
	127^3	16	—	3.3e+1	3.3e−09	1.2e−6	6	2.6e+1	5.2e−09	1.4e−7	2
—	31^3	4	2.1e−1	—	—	—	—	—	—	—	—
	63^3	8	2.6e+0	—	—	—	—	—	—	—	—

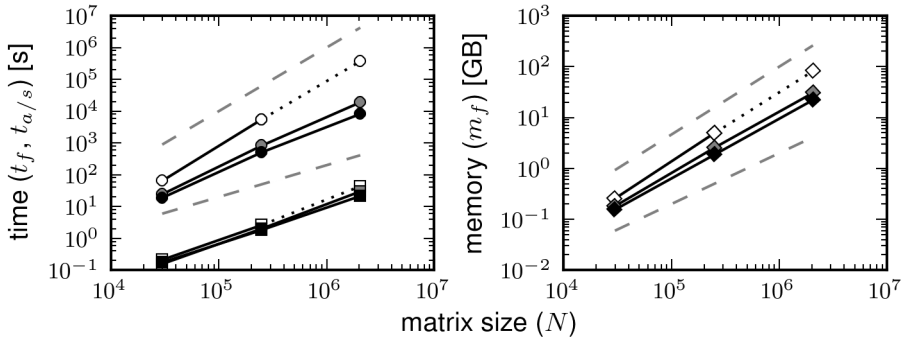


FIGURE 5.5. Scaling results for Example 6, comparing mf3 (white) with hifde3 (gray) and hifde3x (black) at precision $\epsilon = 10^{-6}$; all other notation as in Figure 5.4.

6 Generalizations and Conclusions

In this paper, we have introduced HIF-DE for the efficient factorization of discretized elliptic partial differential operators in 2D and 3D. HIF-DE combines MF [12, 15, 33] with recursive dimensional reduction via frontal skeletonization to

construct an approximate generalized LU/LDL decomposition at estimated quasilinear cost. The latter enables significant compression over MF and is critical for improving the asymptotic complexity, while the former is essential for optimally exploiting sparsity and hence for achieving good practical performance. The resulting factorization allows the rapid application of the matrix inverse, which provides a fast direct solver or preconditioner, depending on the accuracy. Furthermore, although we have focused here only on symmetric matrices, our techniques generalize also to the unsymmetric case by defining analogous two-sided elimination operators R_p and S_p in (2.2) as in [31] and by compressing

$$B_c = \begin{bmatrix} A_{c^{\mathsf{N}},c} \\ A_{c,c^{\mathsf{N}}} \end{bmatrix}$$

instead of just $A_{c^{\mathsf{N}},c}$.

While we have reported numerical data only for PDEs with Dirichlet boundary conditions, HIF-DE extends trivially to other types of boundary conditions as well. Preliminary tests with mixed Dirichlet-Neumann conditions reveal no discernible change in performance.

The skeletonization operator at the core of HIF-DE 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, 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 [7, 24, 47].

Although we have presently only considered sparse matrices arising from PDEs, the same basic approach can also be applied to structured dense matrices such as those derived from the integral equation formulations of elliptic PDEs. This is described in detail as algorithm HIF-IE in the companion paper [31], which uses skeletonization for all compression steps and likewise has quasilinear complexity estimates in both 2D and 3D. In particular, HIF-DE can be viewed as a heavily specialized version of HIF-IE by embedding it into the framework of MF in order to exploit sparsity. The elimination operations in MF can also be seen as a trivial form of skeletonization acting on overlapping subdomains. Indeed, [31] shows that recursive skeletonization [18, 21, 30, 35], a precursor of HIF-IE based on cell compression, is essentially equivalent to MF.

Some important directions for future research include:

- Obtaining analytical estimates of the interaction rank for SCIs, even for the simple case of the Laplacian. This would enable a much more precise understanding of the complexity of HIF-DE, which has yet to be rigorously established.
- Parallelizing HIF-DE, which, like MF, is organized according to a tree structure where each node at a given level can be processed independently of the rest. In particular, the frontal matrices are now much more compact,

which should support better parallelization, and we anticipate that the overall scheme will have significant impact on practical scientific computing. This is currently in active development.
- Investigating alternative strategies for reducing skeleton sizes in 3D, which can still be quite large, especially at high precision.
- Understanding the extent to which our current techniques can be adapted to highly indefinite problems, some of which have a Helmholtz character and possess rank structures of a different type than that exploited here [13, 14]. Such problems can be very challenging to solve iteratively and present a prime target area for future fast direct solvers.

Acknowledgment. We would like to thank Jack Poulson for 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] Amestroy, P. R.; Ashcraft, C.; Boiteau, O.; Buttari, A.; L'Excellent, J.-Y.; Weisbecker, C. Improving multifrontal methods by means of block low-rank representations. INPT-IRIT Technical Report RT/APO/12/6; also appeared as INRIA Technical Report RR-8199, December 2012. *SIAM J. Sci. Comput.*, submitted.

[2] Aminfar, A.; Ambikasaran, S.; Darve, E. A fast block low-rank dense solver with applications to finite-element matrices. Preprint, 2014. arXiv:1403.5337 [cs.NA]

[3] 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

[4] Bebendorf, M. Efficient inversion of the Galerkin matrix of general second-order elliptic operators with nonsmooth coefficients. *Math. Comp.* **74** (2005), no. 251, 1179–1199 (electronic). doi:10.1090/S0025-5718-04-01716-8

[5] 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

[6] Börm, S. Approximation of solution operators of elliptic partial differential equations by $\mathscr{H}$- and $\mathscr{H}^2$-matrices. *Numer. Math.* **115** (2010), no. 2, 165–193. doi:10.1007/s00211-009-0278-7

[7] Brandt, A. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.* **31** (1977), no. 138, 333–390. doi:10.2307/2006422

[8] Chandrasekaran, S.; Dewilde, P.; Gu, M.; Somasunderam, N. On the numerical rank of the off-diagonal blocks of Schur complements of discretized elliptic PDEs. *SIAM J. Matrix Anal. Appl.* **31** (2010), no. 5, 2261–2290. doi:10.1137/090775932

[9] 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

[10] Davis, T. A. *Direct methods for sparse linear systems.* Fundamentals of Algorithms, 2. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2006. doi:10.1137/1.9780898718881

[11] 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

[12] 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

[13] 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

[14] 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.

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

[16] Gillman, A.; Martinsson, P. G. A direct solver with $O(N)$ complexity for variable coefficient elliptic PDEs discretized via a high-order composite spectral collocation method. *SIAM J. Sci. Comput.* **36** (2014), no. 4, A2023–A2046. doi:10.1137/130918988

[17] 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

[18] 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

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

[20] Grasedyck, L.; Kriemann, R.; Le Borne, S. Domain decomposition based $\mathscr{H}$-LU preconditioning. *Numer. Math.* **112** (2009), no. 4, 565–600. doi:10.1007/s00211-009-0218-6

[21] 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

[22] 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

[23] 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

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

[25] 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

[26] 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

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

[28] 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

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

[30] 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

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

[32] 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

[33] Liu, J. W. H. The multifrontal method for sparse matrix solution: theory and practice. *SIAM Rev.* **34** (1992), no. 1, 82–109. doi:10.1137/1034004

[34] Martinsson, P.-G. A fast direct solver for a class of elliptic partial differential equations. *J. Sci. Comput.* **38** (2009), no. 3, 316–330. doi:10.1007/s10915-008-9240-6

[35] 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

[36] Saad, Y. *Iterative methods for sparse linear systems.* Second edition. Society for Industrial and Applied Mathematics, Philadelphia, 2003. doi:10.1137/1.9780898718003

[37] 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

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

[39] Schmitz, P. G.; Ying, L. A fast direct solver for elliptic problems on general meshes in 2D. *J. Comput. Phys.* **231** (2012), no. 4, 1314–1338. doi:10.1016/j.jcp.2011.10.013

[40] Schmitz, P. G.; Ying, L. A fast nested dissection solver for Cartesian 3D elliptic problems using hierarchical matrices. *J. Comput. Phys.* **258** (2014), 227–245. doi:10.1016/j.jcp.2013.10.030

[41] 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

[42] 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

[43] Xia, J. Randomized sparse direct solvers. *SIAM J. Matrix Anal. Appl.* **34** (2013), no. 1, 197–227. doi:10.1137/12087116X

[44] 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

[45] 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

[46] 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

[47] Xu, J. Iterative methods by space decomposition and subspace correction. *SIAM Rev.* **34** (1992), no. 4, 581–613. doi:10.1137/1034116

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

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

Received July 2014.