MULTI-LAYER HIERARCHICAL STRUCTURES AND FACTORIZATIONS

JIANLIN XIA*

Abstract. We propose multi-layer hierarchically semiseparable (MHS) structures for the fast factorizations of dense matrices arising from multi-dimensional discretized problems such as certain integral operators. The MHS framework extends hierarchically semiseparable (HSS) forms (which are essentially one dimensional) to higher dimensions via the integration of multiple layers of structures, i.e., structures within the dense generators of HSS forms. In the 2D case, we lay theoretical foundations for MHS structures and justify the feasibility of MHS approximations based on the fast multipole method (FMM) and algebraic techniques such as structure-preserving rank-revealing factorizations. Rigorous rank bounds and conditions for the structures are given. Representative subsets of mesh points and a multi-layer tree are used to intuitively illustrate the structure. The MHS framework makes it convenient to explore FMM structures and perform direct factorizations. We can naturally design and analyze MHS algorithms by taking advantage of existing methods and analysis for simple HSS methods. In particular, we can design fully stable and scalable multi-layer ULV (MULV) factorizations that can preserve the inner structures and have nearly linear complexity under certain conditions. An idea of reduced matrices is used to show the structured factorizations and the recursive sparsification of the mesh. We also establish intrinsic connections between dense MULV factorizations and sparse structured multifrontal factorizations, which bridges the gaps among different types of hierarchical solvers, and facilitates the sharing of ideas and the study and design of new algorithms. The new structures and algorithms can be used for the direct solution of some multi-dimensional discretized problems with nearly linear complexity and storage.

Key words. MHS structure, representative subset, MULV factorization, reduced matrix, fast direct solver, linear complexity

AMS subject classifications. 15A23, 65F05, 65F30

1. Introduction.

In recent years, rank structured matrices have been widely used for the fast direct solution of some integral and differential equations, especially elliptic problems. See [1, 4, 6, 7, 10, 11, 17, 19, 22, 23, 28, 29, 32] for a partial list of references. A basic idea of these methods is to approximate the dense matrices or fill-in by rank structured forms. Such dense matrices include discretized integral operators, inverses of discretized PDEs, and Schur complements in the direct factorizations of some sparse matrices. PDE/integral equation theories together with linear algebra techniques show that certain off-diagonal blocks of these dense matrices have small numerical ranks.

Several rank structured forms have been designed for the approximation of these dense matrices. Among the most widely used ones are hierarchical structured matrices such as $\mathcal{H}$ [3], $\mathcal{H}^2$ [4], and hierarchically semiseparable (HSS) matrices [6, 33]. $\mathcal{H}/\mathcal{H}^2$ matrices and matrices based on the fast multipole method (FMM) are applicable to 2D and 3D cases. However, these structures are often sophisticated, and the analysis of their numerical behaviors such as stability is not very convenient. The HSS structure aims at 1D cases, but is based on simple bisection and is much easier to use and analyze. It is thus suitable for practical implementations and is easily accessible to the general scientific computing community. In particular, efficient, stable, and scalable HSS operations (especially ULV-type factorizations [6, 33]) are available. Moreover, the hierarchical approximation accuracy and backward stability of HSS methods are well studied [24, 25]. For more general sparse problems, the applicability of HSS

*Department of Mathematics, Purdue University, West Lafayette, IN 47907 (xiaj@math.purdue.edu). The research of Jianlin Xia was supported in part by an NSF CAREER Award DMS-1255416.

matrices can be extended via the integration into sparse matrix techniques such as nested dissection [9] and the multifrontal method [8].

Existing HSS-based structured direct solvers work well for 1D discretized integral equations and 2D discretized elliptic PDEs. However, the efficiency is usually less satisfactory for higher dimensions. One strategy is to approximate the dense matrices corresponding to two dimensions still by HSS forms. Although this simplifies the implementation, the performance of the resulting methods is not optimal for large sizes due to the large off-diagonal ranks. For example, the sparse direct solvers in [28, 29] with HSS approximations of fill-in have up to $O(n^{4/3})$ complexity for 3D discretized elliptic PDEs, where n is the size of the sparse matrix. Special care needs to be taken to reach nearly $O(n)$ complexity.

In some recent studies, additional structures within some HSS approximations have been explored. In fact, it has long been noticed that, in some applications, the dense blocks (called generators) that define the HSS forms are also structured [7, 30, 34, 36]. By taking advantage of such structures, it is possible to design multi-dimensional structured algorithms for dense discretized matrices just based on simple HSS methods. For example, in a fast selected inversion algorithm [35], some diagonal and off-diagonal blocks of the inverse of a sparse matrix are approximated by HSS and low-rank forms, respectively. For some 2D discretized integral operators, the method in [7] exploits the inner structures with the aid of potential theories, and matrix compression is obtained directly based on thin layers of boundary mesh points. The method in [7] directly inverts the 2D discretized matrix, and intermediate operations such as HSS inversion and recompression can be quite expensive.

In this work, we propose a multi-layer hierarchically semiseparable (MHS) structure for multiple dimensions. We show the feasibility of using MHS forms to approximate some discretized kernel functions on 2D domains, and the idea can also be generalized to higher dimensions. We exploit additional structures under the general FMM framework. For some integral kernels on 2D domains, if HSS forms are used for the approximation, we show that the dense generators have inner HSS or low-rank structures. Unlike the method in [7], we consider general FMM interactions between all the subdomains resulting from nested bisection based on algebraic methods. A structure-preserving rank-revealing factorization [13, 36] is used to produce subsets of representative points in the mesh. Unlike in [7], the representative points are not limited to just boundary points. The points are organized into hierarchical levels, and help to justify the existence of additional structures within the HSS generators. Rigorous rank bounds for the low-rank structures are given, together with the conditions such as certain proper ordering. A systematic definition of the MHS form is then presented. The storage of the MHS form is nearly linear in the matrix size.

The MHS framework makes it very convenient to explore FMM structures and design efficient MHS algorithms. In particular, a fast multi-layer/MHS ULV (MULV) factorization is proposed. This generalizes HSS ULV factorizations in [6, 36] to multiple layers. In an outer HSS ULV factorization, the diagonal blocks are further factorized with an inner ULV factorization. The inner-layer structures are preserved during the elimination process, due to certain attractive properties of ULV factorizations. We can conveniently perform the elimination based on the representative subsets, so that the original MHS matrix is hierarchically reduced into smaller MHS forms which correspond to subsets of the mesh points. The original matrix and mesh are thus recursively sparsified. The final reduced MHS form is just an HSS form. The MULV factorization can thus be intuitively illustrated and performed. This is similar

to the idea of reduced HSS matrices in [36, 29] and the idea of skeletonization in [15]. Here, the multi-layer design enables us to recursively perform HSS ULV factorization so as to keep the algorithm simple to implement and analyze. No HSS inversion or recompression as in [7] is needed.

The MULV factorization has nearly linear complexity under certain conditions. Its numerical stability immediately follows from the HSS stability analysis in [25]. It is also fully scalable and has a great potential for large-scale parallel computations.

Additionally, we reveal some intrinsic connections between the *dense* HSS ULV factorization and the *sparse* multifrontal factorization, and between the MULV factorization and the structured multifrontal methods in [29, 32]. This helps the understanding of different types of hierarchical solvers, and enables us to exchange ideas among different hierarchical structured algorithms. This is very useful for the design of new algorithms and for the analysis of different solvers with similar techniques. Then naturally, we may also directly apply MHS methods to some sparse matrices.

We verify the feasibility of MHS approximations and the efficiency of MHS factorizations in some numerical tests. For some discretized matrices, an HSS approximation would have maximum off-diagonal rank (called HSS rank) growing quickly with the matrix size N, while in the MHS approximation, a rank bound for the structural measurement (called MHS rank) grows very slowly. For reasonable N, the MHS ranks are significantly smaller than the HSS ranks. The benefit of MHS structures is further demonstrated with the performance comparison of MHS and HSS factorizations.

The outline of the presentation is as follows. We first discuss structure-preserving low-rank factorizations and representative subsets in Section 2. The design of MHS structures is shown in detail in Section 3, followed by the development of MHS algorithms (especially MULV factorizations) in Section 4. Section 5 shows the numerical tests. The following is a list of some notation.

- For a set of points Ω, $|\Omega|$ denotes its cardinality.
- For a node i of a binary tree, $\mathrm{sib}(i)$ and $\mathrm{par}(i)$ denote the sibling and parent of i, respectively.
- For a binary tree $\mathcal{T}$, $\mathrm{root}(\mathcal{T})$ denotes its root.
- For a matrix A and index sets $\mathbf{I}$ and $\mathbf{J}$, $A|_{\mathbf{I}}$ denotes a submatrix of A consisting of its rows selected by $\mathbf{I}$, and $A|_{\mathbf{I}\times\mathbf{J}}$ corresponds to the selection of row and column entries based on $\mathbf{I}$ and $\mathbf{J}$, respectively.
- $\sigma_j(C)$ denotes the jth largest singular value of a matrix C.

2. Separated sets, structure-preserving low-rank approximations, and representative subsets. In this section, we generalize the concept of well-separated sets, introduce the notion of representative points in low-rank approximations, and discuss the proper ordering of representative points. These serve as preliminaries for our later design of the MHS structure.

2.1. Generalization of well-separated sets. Consider the discretization of a kernel function of the form $\phi(|y-z|)$, where y and z are points inside certain domains and $|y-z|$ is their distance. Some examples of ϕ are $\frac{1}{|y-z|}$, $\frac{1}{|y-z|^2}$, and $\log|y-z|$, where $y \neq z$. Suppose Ω_1 and Ω_2 are two sets of points. We look at the numerical rank of the following discretized matrix for a finite number of points y_i and z_j:

$$K = \left(\phi(|y_i - z_j|)\right)_{y_i\in\Omega_1, z_j\in\Omega_2}. \tag{2.1}$$

For convenience, we refer to K in (2.1) as the *interaction (matrix)* between Ω_1 and Ω_2.

In FMM, in general, two sets Ω_1 and Ω_2 are considered well separated if their distance is greater than or equal to their diameters so that the function $\phi(||y-z||)$ for $y \in \Omega_1$ and $z \in \Omega_2$ can be approximated by a series with a finite number ($r_0 = \log(\tau)$) of terms to reach any given accuracy τ [2, 12]. Correspondingly, the matrix K in (2.1) has a small numerical rank. For convenience, we say that ϕ has a *finite-term multipole expansion* (with respect to the tolerance τ), and denote this by

$$\phi(|y-z|) \approx \sum_{i=1}^{r_0} f_i(y) g_i(z). \tag{2.2}$$

Then consider more general cases where Ω_1 and Ω_2 are not well separated. For example, suppose Ω_1 and Ω_2 are 1D sets as in Figure 2.1(a). (Later, we do not strictly distinguish a domain and a set. In our figures, we usually only draw a domain to indicate a set.) The two sets are not well separated. In FMM, the sets are further partitioned. If we consider $\tilde{\Omega}_1$, the left subset of Ω_1 as in Figure 2.1(b) that is not adjacent to Ω_1, then $\tilde{\Omega}_1$ and Ω_2 are well separated. Thus, $(\phi(|y_i - z_j|))_{y_i \in \tilde{\Omega}_1, z_j \in \Omega_2}$ is numerically low rank. This process is recursively applied to $\Omega_1 \backslash \tilde{\Omega}_1$. It is repeated $O(\log |\Omega_1|)$ times, so that the numerical rank of K becomes $O(\log |\Omega_1|)$. In general, we partition the set with a smaller cardinality.

(a) Two adjacent sets in 1D (b) Well-separated sets in the usual sense

FIG. 2.1. *Separation of two sets in 1D.*

To consider more general cases in multiple dimensions, we relax the usual concept of well separated sets as follows.

DEFINITION 2.1. *(Separated sets) Let $\delta(y_0, \Omega_1)$ denote the distance between a point y_0 and a set Ω_1. Two sets Ω_1 and Ω_2 are separated (or α-separated) for a constant $\alpha > 0$ if there exist points y_0, z_0 and constants d_1, d_2, such that*

$$\delta(y_0, \Omega_1) \le d_1, \quad \delta(z_0, \Omega_2) \le d_2, \quad |y_0 - z_0| \ge d_1 + d_2 + \alpha \min\{d_1, d_2\}.$$

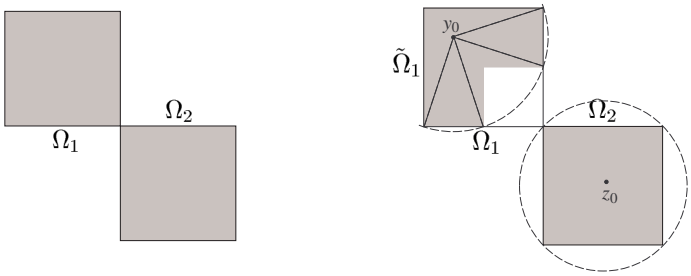
Ω_1 and Ω_2 are well separated if $\alpha > 1$.

For example, consider 2D sets Ω_1 and Ω_2 in Figure 2.2(a) that are not separated. On the other hand, we can get a subset $\tilde{\Omega}_1$ of Ω_1 as in Figure 2.2(b), so that we can choose $x_0 \in \tilde{\Omega}_1$ and $y_0 \in \Omega_2$ as shown and verify that $\tilde{\Omega}_1$ and Ω_2 are α-separated for a constant $\alpha > 0$. For example, if Ω_1 and Ω_2 are two unit squares, then

$$\delta(y_0, \Omega_1) \le d_1 \equiv \frac{\sqrt{10}}{4}, \quad \delta(z_0, \Omega_2) \le d_2 \equiv \frac{\sqrt{2}}{2},$$
$$|y_0 - z_0| = \frac{5}{4}\sqrt{2} \ge d_1 + d_2 + \alpha \min\{d_1, d_2\}, \quad \text{with } \alpha \approx 0.38.$$

After this, the set $\Omega_1 \backslash \tilde{\Omega}_1$ can be similarly processed via recursion. Thus, the numerical rank of K in (2.1) becomes $O(\log |\Omega_1|)$. Later for convenience, we say that Ω_1 and Ω_2 are weakly or logarithmically separated.

DEFINITION 2.2. *(Logarithmically-separated sets) Two sets of points Ω_1 and Ω_2 are logarithmically separated if one of the sets, say, Ω_1 can be partitioned into*

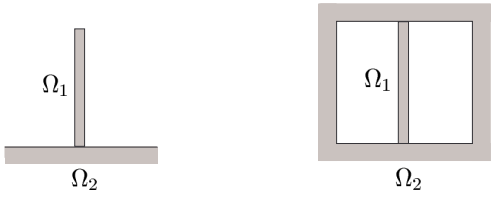


(a) Two adjacent sets in 2D (b) Well-separated sets in the usual sense

FIG. 2.2. *Separation of two sets in 2D.*

$O(\log|\Omega_1|)$ *subsets, all* α*-separated from* Ω_2 *for a constant* $\alpha > 0$*, except a constant number of subsets each containing* $O(1)$ *points.*

The sets Ω_1 and Ω_2 in Figures 2.1(a) and 2.2(a) are thus logarithmically separated. Similarly, it can be verified that Ω_1 and Ω_2 in Figure 2.3 are logarithmically separated.



(a) One example (b) Another example

FIG. 2.3. *Examples of logarithmically-separated sets in two dimensions, where the points in* Ω_1 *correspond to a narrow band.*

Based on these definitions, we have the following simple lemma.

LEMMA 2.3. *If two sets* Ω_1 *and* Ω_2 *are logarithmically separated and* ϕ *has a finite-term multipole expansion with respect to a tolerance* τ*, then the discretized matrix* K *in (2.1) has numerical rank* $O(\log(\min\{|\Omega_1|, |\Omega_2|\}))$ *with respect to* τ*.*

REMARK 2.1. (Kernel expansion assumption) In all the following discussions, we assume $\phi(|y-z|)$ has a finite-term multipole expansion with respect to the tolerance τ for y and z in two α-separated sets. Thus, when we say a matrix has a small numerical rank, we mean that the numerical rank is $O(1)$ with respect to the relative tolerance τ. We often do not explicitly mention the tolerance unless it is different from τ.

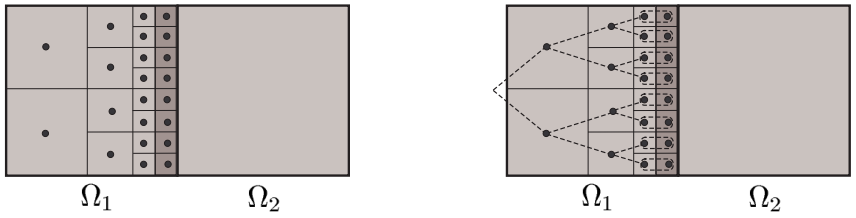
2.2. Structure-preserving rank-revealing factorization and representative points. For a matrix such as K in (2.1) with a small numerical rank r, a rank-revealing factorization may be used to compute a low-rank approximation to it. In particular, a reliable scheme is the strong rank-revealing QR (RRQR) factorization [13]. It can yield a factorization of the form (for convenience, an LQ factorization is written below)

$$K \approx \Pi \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} Q = \Pi \begin{pmatrix} I \\ E \end{pmatrix} (L_1 Q) \equiv UK|_{\mathbf{I}}, \quad \text{with} \tag{2.3}$$

$$U = \Pi \begin{pmatrix} I \\ E \end{pmatrix}, \quad E = L_2 L_1^{-1}, \quad K|_{\mathbf{I}} = L_1 Q.$$

where Π is a permutation matrix, L_1 is $r \times r$ and is made to have a determinant as large as possible, and the entries of E are bounded by a small constant. $K|_{\mathbf{I}}$ thus corresponds to selected rows of K. The factorization $UK|_{\mathbf{I}}$ is also called an interpolative decomposition [14] or structure-preserving rank-revealing (SPRR) factorization [36]. More intuitively, we say this is to select *representative rows* from K with $\mathbf{I}$. If K is from (2.1), then $\mathbf{I}$ corresponds to selected points in Ω_1. Therefore, this factorization can be understood as the selection of *representative points*.

If Ω_1 and Ω_2 are not well separated as in Figure 2.4(a), then partition Ω_1 into subsets at multiple levels, as done in FMM. Suppose the subsets are located within boxes at $l_{\max}$ levels, with the boundary level or level $l_{\max}$ right adjacent to Ω_2. For convenience, suppose each box at level $l_{\max}$ has a constant size h. Level $l_{\max}-1$ also consists of boxes of size h. Also, suppose the partition is fine enough so that the number of points within each box at these two levels is a constant r_0. The box sizes increase for boxes away from the boundary. The boxes at levels $l_{\max}-1, l_{\max}-2, \ldots$ are well separated from Ω_2, so that representative points can be selected from each box (marked as black dots), as also done in [7]. (In practice, all the points within a box at levels $l_{\max}$ and $l_{\max}-1$ can be considered as representative points.) The process yields a sequence of representative points that can be collected into a representative subset.



(a) Representative points at different levels (b) Tree organization of the points

FIG. 2.4. *Study of the interaction between Ω_1 and Ω_2 in FMM, and the selection of representative subsets in Ω_1. The locations of the representative subsets are for illustration purpose only.*

DEFINITION 2.4. *The collection $\mathbf{I}$ of all the representative points within Ω_1 due to the SPRR factorization of the interaction between Ω_1 and Ω_2 is called a* representative subset *of Ω_1 (with respect to Ω_2). For convenience, this process of selecting representative points is denoted by*

$$\mathbf{I} = \mathrm{SPRR}(\Omega_1, \Omega_2). \tag{2.4}$$

The complementary representative subset $\hat{\mathbf{I}}$ *of* $\mathbf{I}$ *in* Ω_1 *is* $\hat{\mathbf{I}} \equiv \Omega_1 \backslash \mathbf{I}$.

2.3. Proper ordering of representative points. The points in the representative subset $\mathbf{I}$ are located at $l_{\max} = O(\log|\mathbf{I}|)$ hierarchical levels. We can organize these points with the aid of a binary tree, called a *representative subset tree*, where the nodes correspond to the boxes and the representative points inside. A larger box at level $l-1$ and two smaller adjacent boxes at level l define a parent-children relationship. The boxes at levels $l_{\max}$ and $l_{\max}-1$ are associated with the leaves. See Figure 2.4(b). For convenience, we use $\mathbf{I}^{(l)}$ to denote all the representative points at level l of the tree, called the l-th *slice* of $\mathbf{I}$, so that

$$\mathbf{I} = \bigcup_{l=1}^{l_{\max}} \mathbf{I}^{(l)}. \tag{2.5}$$

In our design of MHS structures later, it is important to order the points within $\mathbf{I}$ in an appropriate way to obtain desired rank structures. Here, we order them following the postorder of the representative subset tree. This ensures that the representative points within each slice $\mathbf{I}^{(l)}$ are ordered consecutively in a uniform way.

DEFINITION 2.5. *(Proper order) If the points in* $\mathbf{I}$ *are ordered following the postorder of the corresponding representative subset tree, we say that* $\mathbf{I}$ *is* properly ordered.

3. MHS structures. We now show the detailed design of MHS structures and lay the theoretical foundation.

3.1. HSS structures and motivation for MHS structures. The HSS structure is a convenient tool to study the mutual interactions for points inside 1D domains [6, 33]. In an HSS form, an $N \times N$ matrix H is partitioned into a block 2×2 form, and the partition is then recursively done on the two diagonal blocks. This can be organized through a binary tree T called *HSS tree*. The resulting off-diagonal blocks at all levels are represented or approximated by low-rank forms.

In particular, assume i is a node of T with two children c_1 and c_2. In an HSS representation, i is associated some matrices $D_i, U_i, V_i, R_i, W_i, B_i$, called HSS generators. D_i is the diagonal block and U_i, V_i are off-diagonal basis matrices. More specifically, these generators are hierarchically defined as

(3.1)
$$D_i \equiv H|_{\mathcal{I}_i \times \mathcal{I}_i} = \begin{pmatrix} D_{c_1} & U_{c_1} B_{c_1} V_{c_2}^T \\ U_{c_2} B_{c_2} V_{c_1}^T & D_{c_2} \end{pmatrix}, \; U_i = \begin{pmatrix} U_{c_1} R_{c_1} \\ U_{c_2} R_{c_2} \end{pmatrix}, \; V_i = \begin{pmatrix} V_{c_1} W_{c_1} \\ V_{c_2} W_{c_2} \end{pmatrix},$$

whether $\mathcal{I}_i$ is the index set for D_i in H and satisfies the hierarchical relation $\mathcal{I}_i = \mathcal{I}_{c_1} \cup \mathcal{I}_{c_2}$. For the root node k, $\mathcal{I}_k = \{1 : N\}$. It can be seen that U and V are also basis matrices of the blocks $H|_{\mathcal{I}_i \times (\mathcal{I}_k \setminus \mathcal{I}_i)}$ and $H|_{(\mathcal{I}_k \setminus \mathcal{I}_i) \times \mathcal{I}_i}$, called *HSS blocks*. The maximum rank or numerical rank of all the HSS blocks is called the *HSS rank* of H.

HSS matrices can be constructed based on direct compression or randomized sampling [14]. Randomized sampling applied to a low-rank matrix Φ converts its direct compression into the compression of ΦX, where X is a random skinny matrix with column size slightly larger than the rank of Φ. An SPRR factorization similar to (2.3) can then be applied to Φ:

$$\Phi = U \Phi|_{\mathbf{I}}, \quad U = \Pi \begin{pmatrix} I \\ E \end{pmatrix}.$$

The idea of randomized HSS construction is initially given in [20]. For an order N dense matrix A with HSS rank r, an HSS representation or approximation can be computed based on the products of A with $O(r)$ random vectors, plus $O(rN)$ entries from A [20, 36]. The methods in [18, 26] use $O(r \log N)$ matrix-vector products, but without any explicit entries of H. The method in [26] further incorporates a strategy in [14] that can adaptively estimate the number of matrix-vector products needed. The HSS approximation error is given in [25].

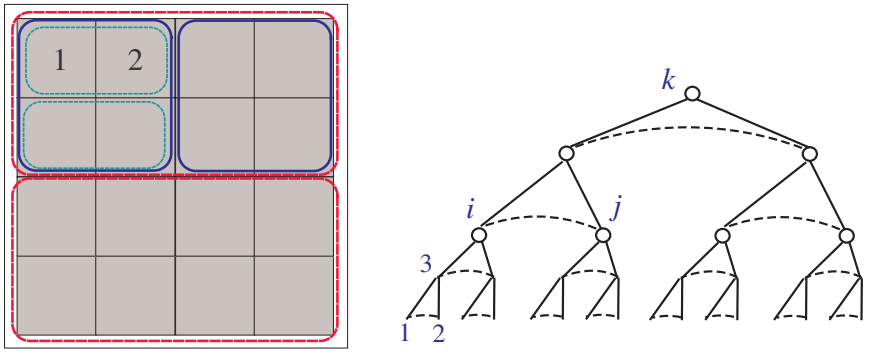
The HSS structure has some significant benefits, include its simplicity, well-established fast and stable operations, and the convenient error and stability analysis. However, the HSS structure focuses on 1D problems. For higher dimensions, it becomes less effective due to the high HSS ranks. On the other hand, many subblocks of the discretized matrix may still have small numerical ranks, following the idea of FMM (see Section 2.1).

As mentioned in Section 1, we seek to design multi-dimensional structures still based on HSS forms, so as to keep the structure simple and to take advantage of existing HSS algorithms and analysis. This involves the study of the interior structures within the HSS generators, so as to establish a new hierarchical structure consisting of multiple layers of simple HSS forms. To illustrate this, we consider the discretization of a kernel ϕ over a 2D set Ω with the assumption in Remark 2.1, and the discretized matrix is

$$A = (\phi(|y_i - y_j|))_{y_i, y_j \in \Omega}. \tag{3.2}$$

(The diagonal entries A_{ii} may be specified otherwise.) Suppose $N = |\Omega|$. The matrix A is $N \times N$ and symmetric.

3.2. MHS structures – outer layer structures. In the design of MHS structures, there are two layers of trees for the 2D case, an outer layer and an inner layer. To explore the outer layer tree structure, we use *nested bisection* to partition the domain/set Ω into a sequence of subdomains. That is, the domain is split into two subdomains, and each subdomain is recursively split. This is similar to the usual nested dissection partition [9], but does not involve a separator of mesh points. A postordered binary tree $\mathbf{T}$ called *nested bisection tree* is then set up, where each leaf corresponds to a bottom level subdomain, and each parent corresponds to an upper level domain or the union of the subdomains associated with its children. See Figure 3.1. Here, we suppose the root is at level 0 and the leaves are at the largest level.

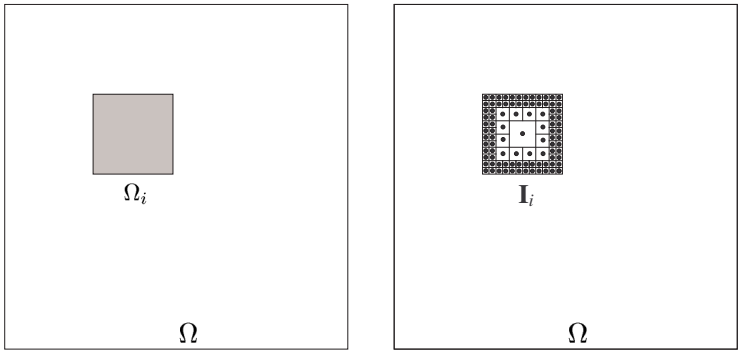


(a) Nested bisection of a domain (b) Nested bisection tree

FIG. 3.1. *Nested bisection of a domain and the corresponding tree* $\mathbf{T}$ *for the subdomains.*

After the application of nested bisection to the points in Ω, we reorder the matrix A in (3.2) following the ordering of the leaves of $\mathbf{T}$. Later, we suppose A is already reordered. We then construct an HSS approximation to A by compressing the HSS blocks $A|_{\Omega_i \times (\Omega \backslash \Omega_i)}$ for each subset $\Omega_i \subset \Omega$. This is done via the study of the interaction of Ω_i with its exterior or complement $\Omega \backslash \Omega_i$.

For example, consider Ω_i in Figure 3.2(a) and its interaction with $\Omega \backslash \Omega_i$. Similarly to Figure 2.4, Ω_i is partitioned into multiple levels of boxes of different sizes. The sizes of the boxes double when their locations are farther away from the boundary. These boxes not inside the boundary level are well separated from $\Omega \backslash \Omega_i$. Suppose there are m boxes along the boundary level, then the total number of boxes inside Ω_i that are well separated from $\Omega \backslash \Omega_i$ is $O(m)$. We can then select representative points from each box. The collection of these points is a representative subset $\mathbf{I}_i$ in Ω_i (Figure 3.2(b)).



(a) A subdomain Ω_i within Ω (b) Representative subset $\mathbf{I}_i$

FIG. 3.2. *Study of the interaction between Ω_i and $\Omega\backslash\Omega_i$ for a subdomain $\Omega_i \subset \Omega$.*

For convenience, we introduce the following notation.

- For a subset $\Omega_i \subset \Omega$, we use $\hat{\Omega}_i \equiv \Omega\backslash\Omega_i$ to denote the complement of Ω_i in Ω.
- As in Definition 2.4, $\mathbf{I}_i \subset \Omega_i$ denotes the representative subset within a set Ω_i with respect to $\hat{\Omega}_i$:

$$\mathbf{I}_i = \mathrm{SPRR}(\Omega_i, \hat{\Omega}_i).$$

- $\hat{\mathbf{I}}_i = \Omega_i\backslash\mathbf{I}_i$ denotes complementary representative subset of $\mathbf{I}_i$ in Ω_i.
- Π_i denotes an appropriate permutation matrix like in (2.3).

In the following, we study the off-diagonal blocks of A in (3.2) or the interactions among different subdomains of Ω. After the nested bisection ordering, we can write

$$A = \begin{pmatrix} A|_{\Omega_1\times\Omega_1} & A|_{\Omega_1\times\hat{\Omega}_1} \\ A|^T_{\Omega_1\times\hat{\Omega}_1} & A|_{\hat{\Omega}_1\times\hat{\Omega}_1} \end{pmatrix} \begin{matrix} \Omega_1 \\ \hat{\Omega}_1 \end{matrix}.$$

The submatrix $A|_{\Omega_1\times\hat{\Omega}_1}$ corresponds to the interaction between Ω_1 and $\hat{\Omega}_1$. According to the representative subset section, we have a factorization similar to (2.3):

$$A|_{\Omega_1\times\hat{\Omega}_1} \approx U_1 A|_{\mathbf{I}_1\times\hat{\Omega}_1}, \quad \text{with} \quad U_1 = \Pi_1 \begin{pmatrix} I \\ E_1 \end{pmatrix}. \tag{3.3}$$

The basis matrix U_1 is thus obtained.

Suppose $\Omega_2 \subset \Omega$ is the sibling set of Ω_1 in nested bisection. That is, Ω_1 and Ω_2 correspond to a pair of sibling nodes in the tree in Figure 3.1. Just like above, we can obtain a representative subset $\mathbf{I}_2 \subset \Omega_2$ (Figure 3.3(a)), so that

$$A|_{\Omega_2\times\hat{\Omega}_2} = \begin{pmatrix} A|_{\Omega_2\times\Omega_1} & A|_{\Omega_2\times\hat{\Omega}_3} \end{pmatrix} \approx U_2 A|_{\mathbf{I}_2\times\hat{\Omega}_2}, \quad \text{with} \quad U_2 = \Pi_2 \begin{pmatrix} I \\ E_2 \end{pmatrix}, \tag{3.4}$$

where $\hat{\Omega}_3$ is the complement of the parent set $\Omega_3 = \Omega_1 \cup \Omega_2$.

We can then write A as

$$A = \begin{pmatrix} A|_{\Omega_1\times\Omega_1} & A|_{\Omega_1\times\Omega_2} & A|_{\Omega_1\times\hat{\Omega}_3} \\ A|_{\Omega_2\times\Omega_1} & A|_{\Omega_2\times\Omega_2} & A|_{\Omega_2\times\hat{\Omega}_3} \\ A|_{\hat{\Omega}_3\times\Omega_1} & A|_{\hat{\Omega}_3\times\Omega_2} & A|_{\hat{\Omega}_3\times\hat{\Omega}_3} \end{pmatrix} \begin{matrix} \Omega_1 \\ \Omega_2 \\ \hat{\Omega}_3 \end{matrix},$$

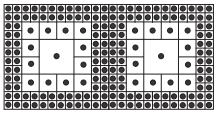
As in symmetric HSS constructions, it is natural to let

$$D_1 = A|_{\Omega_1\times\Omega_1}, \quad D_2 = A|_{\Omega_2\times\Omega_2}, \quad B_1 = A|_{\mathbf{I}_1\times\mathbf{I}_2}, \tag{3.5}$$

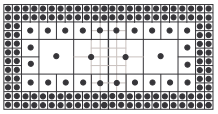
so that

$$D_3 \equiv \begin{pmatrix} A|_{\Omega_1\times\Omega_1} & A|_{\Omega_1\times\Omega_2} \\ A|_{\Omega_2\times\Omega_1} & A|_{\Omega_2\times\Omega_2} \end{pmatrix} \approx \begin{pmatrix} D_1 & U_1B_1U_2^T \\ U_2B_2U_1^T & D_2 \end{pmatrix}. \tag{3.6}$$

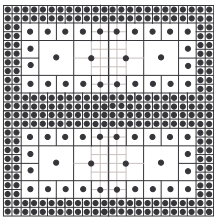
The choice of B_1 is due to the selection of the representative subsets [20, 36].



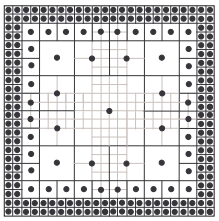
(a) Representative subsets in Ω_1 and Ω_2



(b) Representative subset in $\Omega_1 \cup \Omega_2$



(c) The upper level



(d) Representative subset at the upper level

FIG. 3.3. *Study of the interactions between some sets and their complements in* Ω.

In HSS construction, the next step is to conduct compression associated with the parent node 3 so as to find a nested basis matrix $U_3 = \begin{pmatrix} U_1 & \\ & U_2 \end{pmatrix}\begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$ for

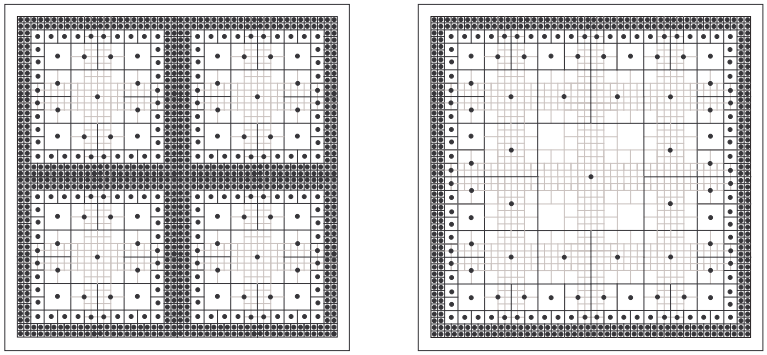
$$A|_{\Omega_3\times\hat{\Omega}_3} \equiv \begin{pmatrix} A|_{\Omega_1\times\hat{\Omega}_3} \\ A|_{\Omega_2\times\hat{\Omega}_3} \end{pmatrix}.$$

U_3 results from the interaction between Ω_3 and $\hat{\Omega}_3$. To find $\begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$, we need to study the interaction between $\mathbf{I}_1 \cup \mathbf{I}_2$ and $\hat{\Omega}_3$. That is, we select a representative subset $\mathbf{I}_3$ from $\mathbf{I}_1 \cup \mathbf{I}_2$. In Figure 3.3(a), we can see that some boundary representative points in $\mathbf{I}_1$ and $\mathbf{I}_2$ becomes interior points located within some boxes well separated from $\hat{\Omega}_3$. We just need to further select representative points from these points, and keep the other boundary representative points in $\mathbf{I}_1$ and $\mathbf{I}_2$ as in Figure 3.3(b). This representative subset selection yields

$$\begin{pmatrix} R_1 \\ R_2 \end{pmatrix} = \Pi_3 \begin{pmatrix} I \\ E_3 \end{pmatrix}.$$

When we move to upper levels, similar procedures apply. See Figures 3.3(c–d) and 3.4(a–b). This is repeated for all the nodes (except root($\mathbf{T}$)) of the nested bisection tree $\mathbf{T}$ (Figure 3.1), so as to produce an HSS approximation to A.

Clearly, the number of points within each representative subset $\mathbf{I}_i$ is the numerical rank of the HSS block $A|_{\Omega_i\times\hat{\Omega}_i}$. Thus, we have the following lemma, which is consistent with FMM and a rank pattern result in [27].



(a) Representative subsets within some subdomains at a certain level

(b) Representative subset at an upper level

FIG. 3.4. *Representative subsets at upper levels.*

LEMMA 3.1. *If Ω is an $M \times M$ uniform mesh, then the HSS block $A|_{\Omega_i \times \hat{\Omega}_i}$ associated with node i at level $\mathbf{l}$ of $\mathbf{T}$ has numerical rank*

$$\tilde{r}_{\mathbf{l}} = O(M_{\mathbf{l}}) \equiv O\left(M/2^{\lfloor \mathbf{l}/2 \rfloor}\right), \tag{3.7}$$

where the diagonal generator D_i of A has size $O(M_{\mathbf{l}}^2)$.

3.3. MHS structures – inner layer structures. Lemma 3.1 indicates that the HSS rank of A may be as large as $O(M) = O(\sqrt{N})$. An HSS approximation to A is generally not very effective. For example, it costs $O(N^{3/2})$ flops to factorize it. To reduce this cost to about $O(N)$, we study the inner layer structures or the structures within the HSS generators from the previous subsection.

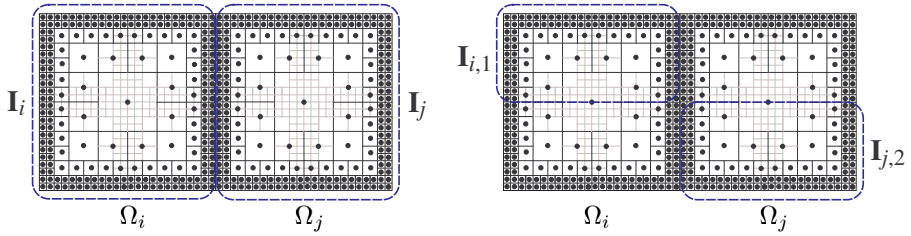
We set a switching level $\mathbf{l}_s$ for the nodes of the nested bisection tree $\mathbf{T}$, so that if a node is at a level above $\mathbf{l}_s$, we exploit the inner structures of the HSS generators. Thus, the generators below $\mathbf{l}_s$ are treated as in the regular HSS case. This avoids operating on blocks that are too small, and also ensures that the outer HSS generator sizes are large enough for the asymptotic inner-layer rank estimates to hold. We can establish a two-layer tree $\mathcal{T}$ from $\mathbf{T}$, which has outer-layer nodes from levels 0 to $\mathbf{l}_s$ of $\mathbf{T}$. A node i at level $\mathbf{l}_s$ of $\mathbf{T}$ is treated as a leaf of $\mathcal{T}$, and D_i is treated as an HSS form generator. The off-diagonal numerical ranks of D_i satisfy Lemma 3.1, and the HSS tree of D_i is simply the subtree of $\mathbf{T}$ associated with i, which is an inner-layer tree. The non-leaf nodes of $\mathcal{T}$ are also associated with inner-layer trees, as shown next.

3.3.1. Structures within the B generators. First, we show the structures within the B_i generators. According to the previous discussions, B_i has the form (see, e.g., (3.5))

$$B_i = A|_{\mathbf{I}_i \times \mathbf{I}_j}, \tag{3.8}$$

where $j = \mathrm{sib}(i)$. (Unlike in [7], B_i here may not be a square matrix). See Figure 3.5(a) for an illustration of the representative subsets $\mathbf{I}_i$ and $\mathbf{I}_j$. We show that B_i can be approximated by an HSS form when $\mathbf{I}_i$ and $\mathbf{I}_j$ are properly ordered.

Notice that the points within $\mathbf{I}_i$ are located at $O(\log |\mathbf{I}_i|)$ levels, similar to the discussions in Section 2.3 for Figure 2.4. We can similarly organize the points in $\mathbf{I}_i$ with the aid of a representative subset tree, where each node may have several



(a) Representative subsets $\mathbf{I}_i$ and $\mathbf{I}_j$ (b) Partition of $\mathbf{I}_i$ and $\mathbf{I}_j$

FIG. 3.5. *Study of the interaction between two representative subsets* $\mathbf{I}_i$ *and* $\mathbf{I}_j$ *for siblings* i *and* j *of* T.

children. The tree can then be used to obtain a proper order of $\mathbf{I}_i$ as in Definition 2.5. The representative subset trees for $\mathbf{I}_i$ and $\mathbf{I}_j$ can be used to recursively bipartition $\mathbf{I}_i$ and $\mathbf{I}_j$. For example, with one level of partition, we have

$$\mathbf{I}_i = \mathbf{I}_{i,1} \cup \mathbf{I}_{i,2}, \quad \mathbf{I}_j = \mathbf{I}_{j,1} \cup \mathbf{I}_{j,2}. \tag{3.9}$$

See Figure 3.5(b). Based on the partition, we have the following result.

THEOREM 3.2. *Suppose* Ω_i *and* Ω_j *are sibling subsets in the nested bisection of a mesh of uniformly distributed points, and* $\mathbf{I}_i$ *and* $\mathbf{I}_j$ *are representative subsets selected from* Ω_i *and* Ω_j*, respectively, and are properly ordered. Assume* $|\mathbf{I}_i| = O(m)$, $|\mathbf{I}_j| = O(m)$. *With the assumption in Remark 2.1,* B_i *in (3.8) and* $A|_{\mathbf{I}_i \times \mathbf{I}_i}$ *have HSS ranks* $O(\log^2 m)$ *and can thus be approximated by (rectangular) HSS forms. Therefore, if* Ω *is an* $M \times M$ *uniform mesh, then the* B_i *generators at level* $\mathbf{l}$ *of* $\mathcal{T}$ *have HSS ranks* $O(\log^2 M_{\mathbf{l}}) \equiv O\left(\log^2 \frac{M}{2^{\lfloor l/2 \rfloor}}\right)$.

Proof. We show that B_i can be approximated by a rectangular HSS form. The proof for $A|_{\mathbf{I}_i \times \mathbf{I}_i}$ is similar. Following the partition (3.9), we can write

$$B_i = \begin{pmatrix} A|_{\mathbf{I}_{i,1} \times \mathbf{I}_{j,1}} & A|_{\mathbf{I}_{i,1} \times \mathbf{I}_{j,2}} \\ A|_{\mathbf{I}_{i,2} \times \mathbf{I}_{j,1}} & A|_{\mathbf{I}_{i,2} \times \mathbf{I}_{j,2}} \end{pmatrix}.$$

Since $\mathbf{I}_i$ and $\mathbf{I}_j$ are properly ordered, it is sufficient to show that the numerical rank of $A|_{\mathbf{I}_{i,1} \times \mathbf{I}_{j,2}}$ is $O(\log^2 m)$, and the proof is similar for the off-diagonal blocks of $A|_{\mathbf{I}_{i,1} \times \mathbf{I}_{j,1}}$ and $A|_{\mathbf{I}_{i,2} \times \mathbf{I}_{j,2}}$ at lower levels.

The points within $\mathbf{I}_{i,1}$ are located at $l_{\max} \equiv O(\log m)$ levels. Following the notation in (2.5), we write $\mathbf{I}_{i,1} = \bigcup_{l=1}^{l_{\max}} \mathbf{I}_{i,1}^{(l)}$. Due to the partition, each slice $\mathbf{I}_{i,1}^{(l)}$ is logarithmically separated from $\mathbf{I}_{j,2}$ (see Figures 2.2 and 2.3). The numerical rank of $A|_{\mathbf{I}_{i,1}^{(l)} \times \mathbf{I}_{j,2}}$ is thus $O(\log m)$. Based on simple rank accumulation, the numerical rank of $A|_{\mathbf{I}_{i,1} \times \mathbf{I}_{j,2}}$ is $O(\log^2 m)$. If B_i is at level $\mathbf{l}$ of $\mathcal{T}$, $m = O(M_{\mathbf{l}})$ following Lemma 3.1. The HSS rank of B_i is then $O(\log^2 M_{\mathbf{l}})$. ∎

The HSS tree of B_i then serves as an inner-layer tree associated with node i of $\mathcal{T}$. If i is a leaf, the HSS trees of D_i and B_i are related, which will be explained in Remark 3.1 below.

3.3.2. Structures within the R generators. Next, we show the structures within R_i and R_j. Let $p = \mathrm{par}(i)$. The parent domain is $\Omega_p = \Omega_i \cup \Omega_j$. Due to the lower level compression, the compression of $A|_{\Omega_p \times \hat{\Omega}_p}$ reduces to the compression of

$A|_{(\mathbf{I}_i \cup \mathbf{I}_j)\times\hat{\Omega}_p}$. The SPRR factorization leads to

$$A|_{(\mathbf{I}_i \cup \mathbf{I}_j)\times\hat{\Omega}_p} \approx \begin{pmatrix} R_i \\ R_j \end{pmatrix} A|_{\mathbf{I}_p\times\hat{\Omega}_p} \quad \text{with}$$

$$\begin{pmatrix} R_i \\ R_j \end{pmatrix} = \Pi_p \begin{pmatrix} I \\ E_p \end{pmatrix} \begin{matrix} \mathbf{I}_p \\ \hat{\mathbf{I}}_p \end{matrix}, \tag{3.10}$$

where the identity matrix corresponds to the representative subset $\mathbf{I}_p$ from $\mathbf{I}_i \cup \mathbf{I}_j$, and E_p corresponds to $\hat{\mathbf{I}}_p = (\mathbf{I}_i \cup \mathbf{I}_j)\backslash\mathbf{I}_p$.

Similar to Figure 3.3, some boxes near the boundary of Ω_i and Ω_j become well separated from $\hat{\Omega}_p$. For convenience, we use $\tilde{\mathbf{I}}_p$ to denote the subset of points in $\mathbf{I}_i \cup \mathbf{I}_j$ that are inside those boxes, and call $\tilde{\mathbf{I}}_p$ the *compressible representative subset*. See Figure 3.6(a). Also, denote the representative subset of $\tilde{\mathbf{I}}_p$ with respect to $\hat{\Omega}_p$ by

$$\bar{\mathbf{I}}_p = \mathrm{SPRR}(\tilde{\mathbf{I}}_p, \hat{\Omega}_p), \tag{3.11}$$

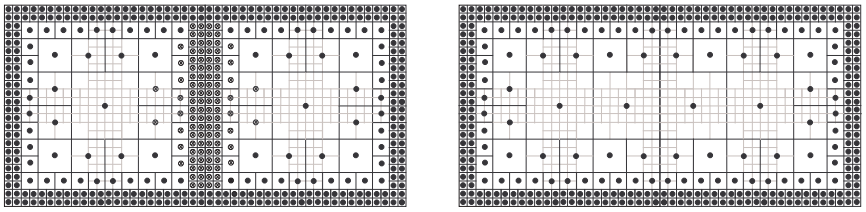
so that

$$\mathbf{I}_p = ((\mathbf{I}_i \cup \mathbf{I}_j)\backslash\tilde{\mathbf{I}}_p) \cup \bar{\mathbf{I}}_p. \tag{3.12}$$

Then we have

$$\hat{\mathbf{I}}_p = (\mathbf{I}_i \cup \mathbf{I}_j)\backslash\mathbf{I}_p = \tilde{\mathbf{I}}_p\backslash\bar{\mathbf{I}}_p. \tag{3.13}$$

That is, to obtain $\mathbf{I}_p$ from $\mathbf{I}_i \cup \mathbf{I}_j$, we replace $\tilde{\mathbf{I}}_p$ by its representative set $\bar{\mathbf{I}}_p$, and the points that we drop form the complementary representative subset $\hat{\mathbf{I}}_p$.



(a) Compressible representative subset $\tilde{\mathbf{I}}_p$ (marked as $\otimes$) in $\mathbf{I}_i \cup \mathbf{I}_j$

(b) $\mathbf{I}_p$ (with $\tilde{\mathbf{I}}_p$ replaced by $\bar{\mathbf{I}}_p$)

FIG. 3.6. *Forming the representative subset* $\mathbf{I_p}$ *from* $\mathbf{I_i} \cup \mathbf{I_j}$ *by replacing the compressible representative subset* $\tilde{\mathbf{I}}_\mathbf{p}$ *by its own representative subset* $\bar{\mathbf{I}}_\mathbf{p}$.

The following lemma shows the cardinality of $\bar{\mathbf{I}}_p$ and will be used later.

LEMMA 3.3. *Suppose* $|\mathbf{I}_i \cup \mathbf{I}_j| = O(m)$. *Then* $|\bar{\mathbf{I}}_p| = O(\log^2 m)$.

Proof. We prove this with the aid of the compressible representative subset $\tilde{\mathbf{I}}_p$, as shown in Figure 3.7. Just like in Figure 2.4, using a binary tree, we can organize the points in $\tilde{\mathbf{I}}_p$ into $O(\log|\tilde{\mathbf{I}}_p|)$ slices (see (2.5)):

$$\tilde{\mathbf{I}}_p = \bigcup_{l=1}^{l_{\max}} \tilde{\mathbf{I}}_p^{(l)}, \quad l_{\max} = O(\log|\tilde{\mathbf{I}}_p|). \tag{3.14}$$

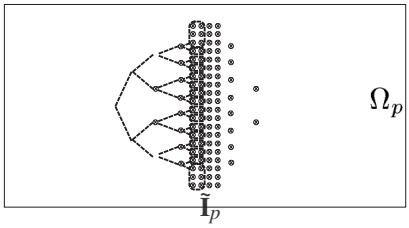


FIG. 3.7. *Tree organization of the compressible representative subset* $\tilde{\mathbf{I}}_p$ *in* $\mathbf{I_i} \cup \mathbf{I_j}$ *from Figure 3.6(a).*

Since $\bar{\mathbf{I}}_p \subset \tilde{\mathbf{I}}_p$ as in (3.13), $\bar{\mathbf{I}}_p$ can also be split into (at most) slices $\bar{\mathbf{I}}_p^{(l)}$. Within each slice $\bar{\mathbf{I}}_p^{(l)}$, there are at most $O(\log|\tilde{\mathbf{I}}_p^{(l)}|)$ points, since they are located within at most $O(\log|\tilde{\mathbf{I}}_p^{(l)}|)$ boxes that are well-separated from $\hat{\Omega}_p$. (In another word, $\tilde{\mathbf{I}}_p^{(l)}$ is logarithmically separated from $\hat{\Omega}_p$.) Thus, $\bar{\mathbf{I}}_p$ includes at most $O(\log^2|\tilde{\mathbf{I}}_p|) = O(\log^2 m)$ points because $|\tilde{\mathbf{I}}_p| \le |\mathbf{I}_i \cup \mathbf{I}_j| = O(m)$. ∎

We then study the structure of E_p in (3.10). Suppose $m = |\mathbf{I}_p|$. The strong RRQR factorization finds an $m \times m$ submatrix of $A|_{(\mathbf{I}_i \cup \mathbf{I}_j)\times\hat{\Omega}_p}$, denoted $A|_{\mathbf{I}_p\times\mathbf{J}_p}$ ($\mathbf{J}_p \subset \hat{\Omega}_p$), whose determinant is sufficiently large. That is, the selection of the representative subset $\mathbf{I}_p$ returns an approximate column basis matrix for $A|_{(\mathbf{I}_i\cup\mathbf{I}_j)\times\hat{\Omega}_p}$:

$$\Pi_p \begin{pmatrix} I \\ E_p \end{pmatrix} A|_{\mathbf{I}_p\times\mathbf{J}_p} = \Pi_p \begin{pmatrix} A|_{\mathbf{I}_p\times\mathbf{J}_p} \\ E_p A|_{\mathbf{I}_p\times\mathbf{J}_p} \end{pmatrix}, \tag{3.15}$$

which is also a submatrix of $A|_{(\mathbf{I}_i\cup\mathbf{I}_j)\times\hat{\Omega}_p}$. Thus,

$$E_p A|_{\mathbf{I}_p\times\mathbf{J}_p} = A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p}. \tag{3.16}$$

To study the rank structure of E_p, we use the following lemma.

LEMMA 3.4. *Suppose* $m \times m$ *matrices* C *and* D *satisfy* $\sigma_1(C) > 0$, $\sigma_m(D) > 0$, *and* $\sigma_1(CD) > 0$. *Then for* $1 \le r \le m$,

$$\frac{\sigma_r(CD)}{\sigma_1(CD)} \le \kappa_2(D)\frac{\sigma_r(C)}{\sigma_1(C)},$$

where $\kappa_2(D)$ *denotes the* 2*-norm condition number of* D.

Proof. It is known that (see, e.g., [16]), for $1 \le s \le m$,

$$\max_{1\le t\le m+s-1} \sigma_t(C)\sigma_{m+s-t}(D) \le \sigma_s(CD) \le \min_{1\le t\le s} \sigma_t(C)\sigma_{s+1-t}(D).$$

Setting $s = t = 1$ in the first inequality yields

$$\sigma_1(CD) \ge \sigma_1(C)\sigma_m(D).$$

Setting $s = t = r$ in the second inequality yields

$$\sigma_r(CD) \le \sigma_r(C)\sigma_1(D).$$

The result then follows. ∎

The rank structure of E_p is then given below.

THEOREM 3.5. *Assume* $|\mathbf{I}_p| = O(m)$, $|\mathbf{I}_q| = O(m)$. *With the assumption in Remark 2.1,* E_p *in (3.10) has numerical rank* $O(\log^2 m)$ *with respect to the tolerance* $\tau\kappa_2(A|_{\mathbf{I}_p\times\mathbf{J}_p})$. *Therefore, if* Ω *is an* $M \times M$ *uniform mesh, then for* p *at level* $\mathbf{l}$ *of* $\mathcal{T}$, E_p *has numerical rank* $O(\log^2 M_{\mathbf{l}}) \equiv O\left(\log^2 \frac{M}{2^{\lfloor l/2 \rfloor}}\right)$ *with respect to the tolerance* $\tau\kappa_2(A|_{\mathbf{I}_p\times\mathbf{J}_p})$.

Proof. Notice $\tilde{\mathbf{I}}_p \subset \Omega_p$ and $\mathbf{J}_p \subset \hat{\Omega}_p$. As in the proof of Lemma 3.3, each slice $\tilde{\mathbf{I}}_p^{(l)}$ in (3.14) is logarithmically separated from $\hat{\Omega}_p$ and also $\mathbf{J}_p$. Thus, $A|_{\tilde{\mathbf{I}}_p^{(l)}\times\mathbf{J}_p}$ has numerical rank $O(\log m)$ (with respect to the tolerance τ), and $A|_{\tilde{\mathbf{I}}_p\times\mathbf{J}_p}$ has numerical rank $O(\log^2 m)$. Accordingly, $A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p}$ has numerical rank $O(\log^2 m)$ since it is a submatrix of $A|_{\tilde{\mathbf{I}}_p\times\mathbf{J}_p}$. That is, for $r = O(\log^2 m)$,

$$\frac{\sigma_{r+1}(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})}{\sigma_1(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})} \le \tau.$$

The matrix E_p from the strong RRQR factorization has the form in (3.16) or

$$E_p = A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p}(A|_{\mathbf{I}_p\times\mathbf{J}_p})^{-1}.$$

According to Lemma 3.4,

$$\begin{aligned}\frac{\sigma_{r+1}(E_p)}{\sigma_1(E_p)} &\le \kappa_2\left((A|_{\mathbf{I}_p\times\mathbf{J}_p})^{-1}\right)\frac{\sigma_{r+1}(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})}{\sigma_1(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})} = \kappa_2(A|_{\mathbf{I}_p\times\mathbf{J}_p})\frac{\sigma_{r+1}(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})}{\sigma_1(A|_{\hat{\mathbf{I}}_p\times\mathbf{J}_p})}\\ &\le \tau\kappa_2(A|_{\mathbf{I}_p\times\mathbf{J}_p}).\end{aligned}$$

That is, E_p in (3.10) has numerical rank $r = O(\log^2 m)$ with respect to the tolerance $\tau\kappa_2(A|_{\mathbf{I}_p\times\mathbf{J}_p})$. ∎

This theorem indicates that, if the block $A|_{\mathbf{I}_p\times\mathbf{J}_p}$ resulting from the strong RRQR factorization is not too ill conditioned (which is the case if $\sigma_r(A|_{(\mathbf{I}_i\cup\mathbf{I}_j)\times\hat{\Omega}_p})$ is not significantly smaller than $\sigma_1(A|_{(\mathbf{I}_i\cup\mathbf{I}_j)\times\hat{\Omega}_p})$ [13]), then E_p in the representation (3.10) for the R generators has numerical rank $O(\log^2 m)$ with respect to a tolerance close to τ.

We can also show the rank structures within $A|_{\mathbf{I}_p\times\hat{\mathbf{I}}_p}$, which will be useful in the factorization stage.

THEOREM 3.6. *Suppose* $|\mathbf{I}_p| = O(m)$, $|\hat{\mathbf{I}}_p| = O(m)$. *With the assumption in Remark 2.1,* $A|_{\mathbf{I}_p\times\hat{\mathbf{I}}_p}$ *has numerical rank* $O(\log^2 m)$.

Proof. After row permutations, we can rewrite $A|_{\mathbf{I}_p\times\hat{\mathbf{I}}_p}$ as $\begin{pmatrix} A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\hat{\mathbf{I}}_p} \\ A|_{\bar{\mathbf{I}}_p\times\hat{\mathbf{I}}_p}\end{pmatrix}$. $A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\hat{\mathbf{I}}_p}$ is a submatrix of $A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\tilde{\mathbf{I}}_p}$. For each slice $\tilde{\mathbf{I}}_p^{(l)}$ in (3.14), it is easy to see that it is logarithmically separated from $\mathbf{I}_p\setminus\bar{\mathbf{I}}_p$, so that the numerical rank of $A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\tilde{\mathbf{I}}_p^{(l)}}$ is $O(\log m)$. The numerical ranks of $A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\tilde{\mathbf{I}}_p}$ and also $A|_{(\mathbf{I}_p\setminus\bar{\mathbf{I}}_p)\times\hat{\mathbf{I}}_p}$ are then $O(\log^2 m)$.

According to Lemma 3.3, $A|_{\bar{\mathbf{I}}_p\times\hat{\mathbf{I}}_p}$ has row size at most $O(\log^2 m)$. The result then follows. ∎

3.3.3. MHS representation. The discussions above indicate that A in (3.2) can be approximated by an HSS form with structured generators. To systematically take advantage of all these structures, we define the MHS representation as follows.

DEFINITION 3.7. *A multi-layer* hierarchically semiseparable (MHS) matrix *is an HSS matrix whose generators are further HSS, MHS, or low-rank matrices. In particular, a two-layer MHS matrix $\mathcal{A}$ with a corresponding* MHS tree $\mathcal{T}$ *is recursively defined as follows. $\mathcal{T}$ includes two layers of postordered binary trees. The outer-layer tree has nodes $i = 1, 2, \ldots, k$, where k is the root. Each node i is associated with HSS generators $D_i, U_i, V_i, R_i, W_i, B_i$. Furthermore, the generators are structured as described below:*

1. *all the generators D_i associated with leaves i of $\mathcal{T}$ are in HSS forms;*
2. *all the B_i generators associated with node $i \neq k$ of $\mathcal{T}$ are in HSS forms;*
3. *all the R, W generators used to construct the generators U_i, V_i associated with nonleaf nodes $i \neq k$ of $\mathcal{T}$ as in (3.1) are in the forms of $\begin{pmatrix} R_{c_1} \\ R_{c_2} \end{pmatrix} = \Pi_i \begin{pmatrix} I \\ E_i \end{pmatrix}$ and $\begin{pmatrix} W_{c_1} \\ W_{c_2} \end{pmatrix} = \Theta_i \begin{pmatrix} I \\ F_i \end{pmatrix}$, respectively, where c_1 and c_2 are the children of i, Π_i and Θ_i are permutation matrices, and E_i and F_i are low-rank matrices.*

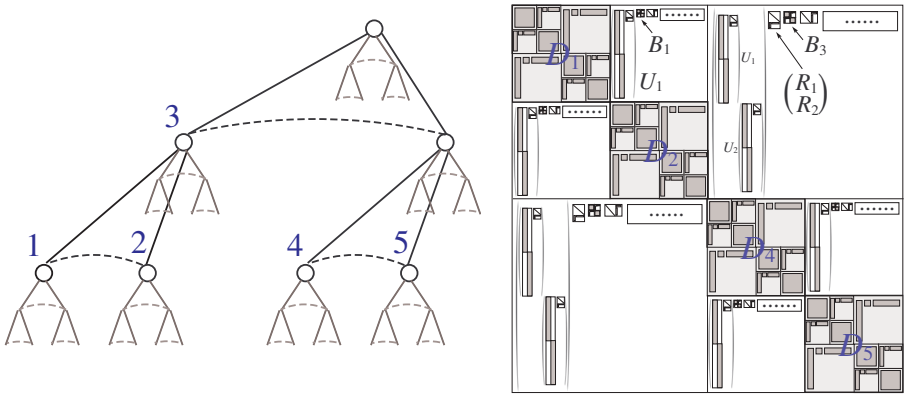
Each node i of $\mathcal{T}$ is associated with an inner-layer HSS tree for the D_i or B_i generator. All the inner-layer HSS generators and the low-rank forms of E_i and F_i are called the MHS generators. The outer-layer HSS rank is the HSS rank of $\mathcal{A}$ when $\mathcal{A}$ is considered as an HSS matrix, and the MHS rank *of $\mathcal{A}$ is the maximum of the HSS ranks of all the leaf level D_i generators, of the HSS ranks of all the B_i generators, and of the ranks of all E_i, F_i.*

REMARK 3.1. We make some remarks on certain practical issues about the generators in the definition.

1. Permutations may also be involved in the leaf level D_i generators and the B_i generators in order for them to have HSS forms. This is to accommodate possible reordering of the corresponding representative points. The permutations do not interfere with the off-diagonal rank structures, since the U, V basis matrices also involve permutations.
2. The B_i generators may be non-square matrices. (In [7], when HSS approximations are used for integral equation solution, B is restricted to be square.) Following the factorization process and Remark 4.1 in Section 4 below, it makes sense to store the following matrix as a square HSS matrix instead:
$$\Pi_p^T \begin{pmatrix} D_i & B_i \\ B_j & D_j \end{pmatrix} \Theta_p,$$
where $j = \text{sib}(i)$, $p = \text{par}(i)$.
3. For a leaf i, there is no restriction on the structure of the U_i, V_i generators, which are formed based on the generators associated with the inner HSS generators of D_i.

Thus, a two-layer MHS structure is an outer-layer HSS structure with an extra inner layer of HSS or low-rank structures. This is also called an HSS2D structure in the report [30]. See Figure 3.8 for an illustration. Here by an MHS structure, we usual mean the two-layer one. For notational consistency, suppose the MHS tree has $\mathbf{l}_s$ outer levels, with the root at level 0. When we say a node of $\mathcal{T}$ is at level $\mathbf{l}$, we mean the outer level $\mathbf{l}$.

The storage of $\mathcal{A}$ can be counted as follows. Let N be its size, $\tilde{r}$ be its outer HSS rank, and r be its MHS rank. Each HSS form D_i or B_i generator has size $O(\tilde{r})$ and needs storage $O(r\tilde{r})$. The storage is similar for each structured R_i or W_i generator.



(a) MHS tree (b) MHS structure

FIG. 3.8. *An MHS matrix corresponding to a two-layer MHS tree, where the inner layer trees in (a) are for the structured D, B generators.*

The outer HSS tree has $O(\frac{N}{\tilde{r}})$ nodes. Thus, the total storage is

$$\sigma = O(r\tilde{r} \cdot \frac{N}{\tilde{r}}) = O(rN).$$

When $\mathcal{A}$ is used to approximate A in (3.2) with Ω an $M \times M$ uniform mesh, we have $r = O(\log^2 N)$ according to Theorems 3.2 and 3.5 above. Thus, $\sigma = O(N \log^2 N)$.

However, this overestimates since the rank structures associated with nodes at different levels $\mathbf{l}$ are different. Similar to the rank patterns in [27], we can take into consideration the different levels and improve the storage estimate. According to Lemma 3.1, the outer off-diagonal ranks depending on $\mathbf{l}$ as $\tilde{r}_{\mathbf{l}}$ in (3.7). Thus, the sizes of D_i, R_i, W_i, B_i, etc. depend on $\mathbf{l}$. Each leaf level D_i generator has size $O(M_{\mathbf{l}_s}^2)$, and Lemma 3.1 further means that the storage of D_i is $O(M_{\mathbf{l}_s}^2 \log M_{\mathbf{l}_s})$ [27]. The leaf level U_i, V_i generators needs no extra storage due to the third item in Remark 3.1. The R_i, W_i, B_i generators at level $\mathbf{l}$ have sizes $O(M_{\mathbf{l}})$ and need storage $O(M_{\mathbf{l}} \log^2 M_{\mathbf{l}})$ based on Theorems 3.2 and 3.5. Thus, the total storage for the MHS matrix is

$$\begin{aligned}
\sigma &= 2^{\mathbf{l}_s} O(M_{\mathbf{l}_s}^2 \log M_{\mathbf{l}_s}) + \sum_{\mathbf{l}=1}^{\mathbf{l}_s} 2^{\mathbf{l}} O\left(M_{\mathbf{l}} \log^2 M_{\mathbf{l}}\right) \\
&= O(M^2 \log M_{\mathbf{l}_s}) + \sum_{\mathbf{l}=1}^{\mathbf{l}_s} 2^{\mathbf{l}} O\left(\frac{M}{2^{\lfloor \mathbf{l}/2 \rfloor}} \log^2 \frac{M}{2^{\lfloor \mathbf{l}/2 \rfloor}}\right) \\
&= O(M^2 \log M_{\mathbf{l}_s}) + \sum_{\hat{\mathbf{l}}=0}^{\lfloor \mathbf{l}_s/2 \rfloor} 2^{\hat{\mathbf{l}}} O(M(\log M - \hat{\mathbf{l}})^2).
\end{aligned}$$

Based on the formula $\sum_{\hat{\mathbf{l}}=0}^{k} 2^{\hat{\mathbf{l}}}(n - \hat{\mathbf{l}})^2 \approx 2^{k+1}(n-k)^2$ (where some low-order terms are dropped for sufficiently large n and k), we have

$$\begin{aligned}
\sigma &= O(M^2 \log M_{\mathbf{l}_s}) + O(M 2^{\lfloor \mathbf{l}_s/2 \rfloor} (\log M - \lfloor \mathbf{l}_s/2 \rfloor)^2 \\
&= O(M^2 \log M_{\mathbf{l}_s}) + O\left(\frac{M^2}{M_{\mathbf{l}_s}} (\log M - \lfloor \mathbf{l}_s/2 \rfloor)^2\right).
\end{aligned}$$

In the ideal situation, when $\lfloor \mathbf{l}_s/2 \rfloor \approx \log M$ and $M_{\mathbf{l}_s} = O(1)$, we get $\sigma = O(M^2)$. However, since the rank estimates and the flop counts are asymptotic, this may not actually hold. A more realistic estimate is for $M_{\mathbf{l}_s}$ to depend on M as

$$M_{\mathbf{l}_s} (\approx M/2^{\lfloor \mathbf{l}_s/2 \rfloor}) = O(\log M). \tag{3.17}$$

In this case, $\sigma = O(M^2 \log\log M)$. This also means

$$\mathbf{l}_s \approx 2\log M - O(\log\log M), \tag{3.18}$$

and the leaf level D_i generators have $O(\log\log M)$ levels in their inner HSS trees. If $M_{\mathbf{l}_s} = O(\sqrt{M})$, then

$$\sigma = O(M^2 \log M) = O(N \log N).$$

In practical implementations, this is a more realistic estimate.

4. MHS algorithms and MULV factorizations. Due to the multi-layer structure, it is convenient to reuse some basic ideas and algorithms in HSS methods to design MHS algorithms such as construction, factorization, solution, and multiplication. We mainly focus on the factorization of an MHS approximation $\mathcal{A}$ to the matrix A in (3.2), and briefly mention other algorithms.

4.1. MHS constructions. One MHS construction strategy is based on the kernel expansion in (2.2). We first construct the outer-layer representation following the derivation procedure in Section 3 and then explore the inner-layer structures. Initially, evaluate the functions f and g in (2.2) at all the points in Ω. We may also integrate the stabilization strategies in [5] for the purpose of stability.

Then for a leaf i of $\mathcal{T}$, find a numerical column basis $\tilde{U}_i$ for $A|_{\Omega_i \times \hat{\Omega}_i}$ based on the kernel expansion. $\tilde{U}_i$ is then converted into the generator U_i in a form like in (2.3). This enables us to identify the representative subset $\mathbf{I}_i$.

For a nonleaf node p with children i and j, we find a numerical column basis $\tilde{U}_p$ for $\begin{pmatrix} A|_{\mathbf{I}_i \times \hat{\Omega}_p} \\ A|_{\mathbf{I}_j \times \hat{\Omega}_p} \end{pmatrix}$. As mentioned in Section 3.3.2, this only needs to be done on $A|_{\tilde{\mathbf{I}}_p \times \hat{\Omega}_p}$. The representative subset selection (3.11) yields $\bar{\mathbf{I}}_p$ and then $\mathbf{I}_p$ in (3.12). The representative subset selection also gives E_p in (3.15).

Next, we find the inner structures. For a leaf i, the kernel expansion or direct off-diagonal compression can be used to find an HSS approximation to D_i. Since $B_i \equiv A|_{\mathbf{I}_i \times \mathbf{I}_j}$, the kernel expansion can be used to find an HSS approximation to B_i [5]. We then compute a low-rank approximation to E_p based on (3.16). That is, we can write an HSS approximation to $A|_{\mathbf{I}_p \times \mathbf{J}_p}$ and a low-rank approximation to $A|_{\hat{\mathbf{I}}_p \times \mathbf{J}_p}$, and then obtain E_p via HSS solutions:

$$E_p = (A|_{\mathbf{I}_p \times \mathbf{J}_p})^{-1} A|_{\hat{\mathbf{I}}_p \times \mathbf{J}_p}.$$

Clearly, writing the outer-layer HSS approximation only costs $O(M^2 \log M) = O(N \log N)$ flops, following the rank pattern in Lemma 3.1. While it costs $O(M^3) = O(N^{3/2})$ flops to find the inner-layer structures. On the other hand, for certain situations [7, 21], it is be possible to analytically identify the representative subsets, so as to reduce the total cost to $O(N \log N)$ or even less. In such cases, potential theories can greatly benefit the compression.

Algebraic MHS construction strategies can also be proposed, and are our primary interest. A straightforward HSS construction from the dense matrix A would cost $O(N^2)$. A faster way is to use a randomized HSS construction in [20, 36] together with FMM for matrix-vector multiplications. Due to the rank pattern in Lemma 3.1, the complexity is $O(N^{3/2})$ [27]. Although this complexity is not optimal, the construction is simpler than the analytical ones and the performance is still competitive in practice. In addition, if MHS approximation is used to handle intermediate dense fill-in in the factorization of sparse discretized elliptic problems on 3D domains, the $O(N^{3/2})$ complexity is sufficient to ensure a roughly linear complexity for the overall sparse factorization [28].

It is still an open problem to construct MHS approximations to problems with small MHS ranks in nearly $O(N)$ complexity using only algebraic techniques. On the other hand, for problems with small HSS ranks and fast matrix-vector multiplications, randomized HSS construction can reach nearly linear complexity [18, 20, 36].

4.2. Multi-layer ULV (MULV) factorization. A major significance of the MHS structure is the fast factorization. The factorization of an MHS form is very convenient based on HSS ULV factorizations [6, 33]. Multiple layers of ULV factorizations will be involved. Thus, we call the MHS factorization a multi-layer ULV or MULV factorization.

Recall that the basic idea of HSS ULV factorizations is the hierarchical reduction of an HSS matrix into smaller *reduced HSS matrices* [33, 28]. In particular, for one version of ULV factorization [36], due to the structured basis matrices as in (2.3), the reduced matrices eventually correspond to the representative subsets selected in SPRR factorizations. Thus, the factorization can be intuitively illustrated in terms of the representative subsets. The basic idea is introduced in [36, 29] and also appears in [15]. Here, we improve the idea by incorporating an additional layer of structures. Furthermore, we can even relate the MULV factorization to sparse multifrontal type factorizations that also involve hierarchical tree operations. The nice stability and scalability of MHS factorizations can thus be easily seen. These are explained in detail below. Suppose $\mathcal{A}$ is an MHS approximation to A with generators as in Definition 3.7. Since A here is symmetric, the V, W generators are same as the U, R generators, respectively, and the MULV factorization below is a symmetric one. It can be easily modified to get a nonsymmetric version.

4.2.1. Outer factorization framework. In order to facilitate the inner-layer structured operations, the MHS factorization uses an outer factorization framework based on a ULV factorization modified from [36]. We first explain the fundamental operations in terms of representative subsets, and then discuss the inner-layer computations in the next subsection. One fundamental operation is to introduce zeros into the off-diagonal blocks via the introduction of zeros into the column basis matrices. Due to the form of a basis matrix like in (3.3), (3.4), and (3.10), a strategy is proposed in [36] to introduce zeros without any cost. Here, we use a modified form. For example, consider a basis matrix $\Pi_i \begin{pmatrix} I \\ E_i \end{pmatrix} \begin{matrix} \mathbf{I}_i \\ \hat{\mathbf{I}}_i \end{matrix}$ associated with a leaf node i, where the index sets are marked and $\mathbf{I}_i$ is the representative subset within Ω_i. Let

$$P_i = \begin{matrix} \hat{\mathbf{I}}_i \\ \mathbf{I}_i \end{matrix} \begin{pmatrix} I & \\ -E_i & I \end{pmatrix} \Pi_i^T. \tag{4.1}$$

Then

$$P_i\Pi_i \begin{pmatrix} I \\ E_i \end{pmatrix} = \begin{matrix} \hat{\mathbf{I}}_i \\ \mathbf{I}_i \end{matrix} \begin{pmatrix} I \\ 0 \end{pmatrix}. \tag{4.2}$$

This process is done analytically and no actual operation is performed. P_i here is different from the one [36], and the reason is that we can write it as the following form that will be convenient for our inner structured operations later:

$$P_i = (I + \tilde{E}_i)\Pi_i^T.$$

Another way to see the benefit is that $I + \tilde{E}_i$ is an HSS form when E_i in (4.1) has an inner low-rank structure. P_i is thus a permuted HSS matrix.

Accordingly, the diagonal generator D_i needs to be updated. For later notation consistency, let $\hat{D}_i \equiv D_i$. Multiply P_i to $\hat{D}_i$ on the left and P^T on the right:

$$\tilde{D}_i = P_i\hat{D}_iP_i^T = (I + \tilde{E}_i)(\Pi_i^T\hat{D}_i\Pi_i)(I + \tilde{E}_i^T). \tag{4.3}$$

Due to the permutation, $\Pi_i^T\hat{D}_i\Pi_i$ may be partitioned as

$$\Pi_i^T\hat{D}_i\Pi_i \equiv \begin{pmatrix} A|_{\mathbf{I}_i\times\mathbf{I}_i} & A|_{\mathbf{I}_i\times\hat{\mathbf{I}}_i} \\ A|^T_{\mathbf{I}_i\times\hat{\mathbf{I}}_i} & A|_{\hat{\mathbf{I}}_i\times\hat{\mathbf{I}}_i} \end{pmatrix}. \tag{4.4}$$

Then we get

$$\tilde{D}_i = \Pi_i^T\hat{D}_i\Pi_i + \tilde{E}_i(\Pi_i^T\hat{D}_i\Pi_i) + (\Pi_i^T\hat{D}_i\Pi_i)\tilde{E}_i^T + \tilde{E}_i(\Pi_i^T\hat{D}_i\Pi_i)\tilde{E}_i^T \equiv \begin{pmatrix} \tilde{D}_{i;1,1} & \tilde{D}_{i;1,2} \\ \tilde{D}_{i;1,2}^T & \tilde{D}_{i;2,2} \end{pmatrix}, \tag{4.5}$$

where

$$\begin{aligned} \tilde{D}_{i;1,1} &= A|_{\mathbf{I}_i\times\mathbf{I}_i}, \\ \tilde{D}_{i;1,2} &= A|_{\mathbf{I}_i\times\hat{\mathbf{I}}_i} - A|_{\mathbf{I}_i\times\mathbf{I}_i}E_i^T, \\ \tilde{D}_{i;2,2} &= A|_{\hat{\mathbf{I}}_i\times\hat{\mathbf{I}}_i} - E_iA|_{\mathbf{I}_i\times\hat{\mathbf{I}}_i} - A|^T_{\mathbf{I}_i\times\hat{\mathbf{I}}_i}E_i^T + E_iA|_{\mathbf{I}_i\times\mathbf{I}_i}E_i^T. \end{aligned} \tag{4.6}$$

The second fundamental operation is to partially eliminate the diagonal block. Treat the $(2,2)$ block $\tilde{D}_{i;2,2}$ as the pivot block and compute a partial UDU factorization (which is a variation of the partial LDL factorization)

$$\tilde{D}_i = \begin{pmatrix} I & \tilde{L}_{i;1,2} \\ & \tilde{L}_{i;2,2} \end{pmatrix} \begin{pmatrix} S_i & \\ & \Lambda_i \end{pmatrix} \begin{pmatrix} I & \\ \tilde{L}_{i;1,2}^T & \tilde{L}_{i;2,2}^T \end{pmatrix}, \tag{4.7}$$

where S_i is the Schur complement of the $(2,2)$ block:

$$S_i = \tilde{D}_{i;1,1} - \tilde{D}_{i;1,2}\tilde{D}_{i;2,2}^{-1}\tilde{D}_{i;1,2}^T = A|_{\mathbf{I}_i\times\mathbf{I}_i} - \tilde{D}_{i;1,2}\tilde{D}_{i;2,2}^{-1}\tilde{D}_{i;1,2}^T. \tag{4.8}$$

The pivot block $\tilde{D}_{i;2,2}$ can then be eliminated, which clearly corresponds to the removal of the complementary representative subset $\hat{\mathbf{I}}_i$ from the domain, so that only the representative subset $\mathbf{I}_i$ remains.

The next operation is to merge blocks and form a reduced matrix. Let $p = \mathrm{par}(i)$ and $j = \mathrm{sib}(i)$, and

$$\hat{D}_p \equiv \begin{matrix} \mathbf{I}_i \\ \mathbf{I}_j \end{matrix} \begin{pmatrix} S_i & B_i \\ B_i^T & S_j \end{pmatrix}, \quad \hat{U}_p = \begin{matrix} \mathbf{I}_i \\ \mathbf{I}_j \end{matrix} \begin{pmatrix} R_i \\ R_j \end{pmatrix}, \tag{4.9}$$

where the index sets marked for the blocks of $\hat{D}_p$ are due to (4.6), (4.8), and (3.8). Then we can remove i and j from the tree, so that p becomes a leaf with new generators $\hat{D}_p, \hat{U}_p$. Then the above process can be repeated on p. Every time a node is eliminated, we get a reduced HSS matrix.

4.2.2. Inner structured computations. In the MULV factorization, the computations in the outer ULV factorization above are performed in structured forms.

1. Since E_i in (4.1) is low rank (see Section 3.3.2), so does $\tilde{E}_i$ in (4.5). According to the discussions below, $\Pi_i^T \hat{D}_i \Pi_i$ in (4.4) has an HSS approximation, so that $\tilde{D}_i$ in (4.5) also has an HSS approximation and can be computed via a low-rank update of the HSS form of $\Pi_i^T \hat{D}_i \Pi_i$.
2. Then the partial diagonal factorization (4.7) is replaced by a partial ULV factorization. Different versions in [29, 33] may be used. The Schur complement S_i in (4.8) is also an HSS form, and its generators can be obtained from the update of those of $\tilde{D}_{i;1,1}$. Notice that an essential property of the ULV Schur complement computation is that S_i and $\tilde{D}_{i;1,1}$ share the same off-diagonal basis generators, so that the HSS rank of S_i is bounded by the HSS rank of $\tilde{D}_i$ (see [33, 29]). This is important to preserve the inner-layer structures. Thus, the partial factorization looks like

$$\tilde{D}_{i;2,2} = \tilde{L}_{i;2,2} \Lambda_i \tilde{L}_{i;2,2}^T \qquad \text{— HSS ULV factorization,} \tag{4.10}$$

$$S_i = \tilde{D}_{i;1,1} - \tilde{D}_{i;1,2} \tilde{D}_{i;2,2}^{-1} \tilde{D}_{i;1,2}^T \qquad \text{— partial generator update.} \tag{4.11}$$

   The update $\tilde{D}_{i;1,2} \tilde{D}_{i;2,2}^{-1} \tilde{D}_{i;1,2}^T$ is not explicitly formed. Instead, a reduced HSS matrix [28, 33] resulting from the ULV factorization of $\tilde{D}_{i;2,2}$ is used for the quick update. This needs no HSS inversion or recompression.
3. In the merge step (4.9), the subblocks of $\hat{D}_p$ are then all approximated by HSS forms. We can show that $\Pi_p^T \hat{D}_p \Pi_p$ itself also has an HSS approximation. According to (4.8), S_i is $A|_{\mathbf{I}_i \times \mathbf{I}_i}$ plus a low-rank update. A similar situation holds for S_j. Also, notice (3.8). Thus,

$$\begin{aligned} \tilde{D}_p &\equiv \Pi_p^T \hat{D}_p \Pi_p \\ &= A|_{(\mathbf{I}_p \cup \hat{\mathbf{I}}_p) \times (\mathbf{I}_p \cup \hat{\mathbf{I}}_p)} - \Pi_p^T \begin{pmatrix} \tilde{D}_{i;1,2} \tilde{D}_{i;2,2}^{-1} \tilde{D}_{i;1,2}^T & \\ & \tilde{D}_{j;1,2} \tilde{D}_{j;2,2}^{-1} \tilde{D}_{j;1,2}^T \end{pmatrix} \Pi_p. \end{aligned} \tag{4.12}$$

   Based on Theorem 3.6 and similarly to the study of the structures of the B generators in Section 3.3, $A|_{(\mathbf{I}_p \cup \hat{\mathbf{I}}_p) \times (\mathbf{I}_p \cup \hat{\mathbf{I}}_p)}$ can be approximated by an HSS form. Thus, $\Pi_p^T \hat{D}_p \Pi_p$ can also be approximated by an HSS form after a low-rank update. This means that we can proceed with the structured factorization of $\tilde{D}_p$ with i replaced by p in (4.5). Note that (4.12) is only used to show the HSS structure of $\Pi_p^T \hat{D}_p \Pi_p$, and is not used in actual computations due to the second statement above.

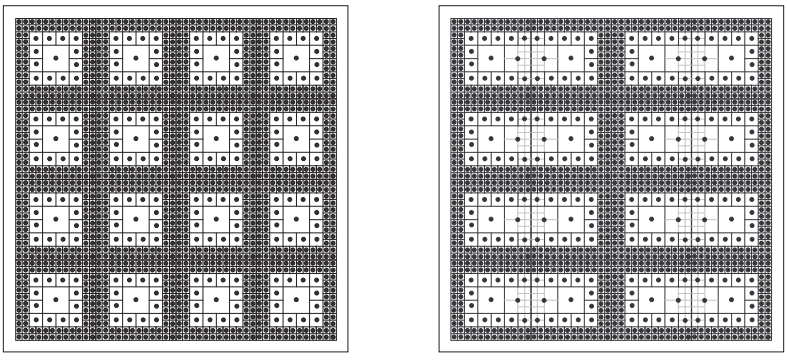
When the nodes at a level are eliminated, we get a *reduced MHS matrix*. Let $\mathcal{A}^{(\mathbf{l}_s)} \equiv \mathcal{A}$, and let $\mathcal{A}^{(\mathbf{l}-1)}$ be the reduced matrix resulting from the elimination of the nodes at level $\mathbf{l}$ of $\mathcal{A}^{(\mathbf{l})}$. Within a reduced MHS matrix, a leaf level diagonal generator $\hat{D}_i$ has an HSS form when it is permuted as $\Pi_i^T \hat{D}_i \Pi_i$. All the other generators of the reduced MHS matrix come from the original generators of $\mathcal{A}$. In the elimination process, due to (4.3), the HSS rank of S_i may be higher than that of $A|_{\mathbf{I}_i \times \mathbf{I}_i}$. When the factorization proceeds to p at upper levels, algebraically, it is possible for the HSS rank of $\Pi_p^T \hat{D}_p \Pi_p$ to slightly grow. This growth is expected to be proportional to a low-degree power of $\mathbf{l}_s - \mathbf{l}$ for p at level $\mathbf{l}$. This is similar to the off-diagonal rank behavior of $A|_{(\mathbf{I}_p \cup \hat{\mathbf{I}}_p) \times (\mathbf{I}_p \cup \hat{\mathbf{I}}_p)}$. Numerical tests confirm the slow growth, though a rigorous derivation of the actual growth pattern is not available yet.

REMARK 4.1. There are some practical issues to pay attention to. Since Π_p results from an RRQR factorization, it only separates $\mathbf{I}_i \cup \mathbf{I}_j$ into sets $\mathbf{I}_p$ and $\hat{\mathbf{I}}_p$, but does not guarantee that $\mathbf{I}_p$ and $\hat{\mathbf{I}}_p$ are properly ordered as needed for the HSS approximation of $A|_{(\mathbf{I}_p \cup \hat{\mathbf{I}}_p)\times(\mathbf{I}_p \cup \hat{\mathbf{I}}_p)}$. Thus, in practice, we also reorder the sets $\mathbf{I}_p$ and $\hat{\mathbf{I}}_p$ following the geometric connectivity in the mesh so as to ensure the HSS structure. This does not affect the matrix representation. For convenience, we assume Π_p also includes such internal reordering. Thus, to be more specific, the structure within the B generators appear as part of the HSS form of

$$\tilde{D}_p = \Pi_p^T \hat{D}_p \Pi_p = \Pi_p^T \begin{pmatrix} S_i & B_i \\ B_i^T & S_j \end{pmatrix} \Pi_p. \tag{4.13}$$

If i and j are leaf nodes, the storage of D_i, D_j, B_i can be similarly based on the matrix $\Pi_p^T \begin{pmatrix} D_i & B_i \\ B_i^T & D_j \end{pmatrix} \Pi_p$ as an HSS form. In the actual implementation, for simplicity, the internal ordering for $\mathbf{I}_p$ and $\hat{\mathbf{I}}_p$ may be obtained with the reverse Cuthill-McKee (RCM) method.

4.2.3. Overall MULV factorization algorithm and properties. The entire MULV factorization recursively produces the reduced MHS matrices $\mathcal{A}^{(\mathbf{l})}, \mathbf{l} = \mathbf{l}_s - 1, \ldots, 1, 0$. The elimination of the nodes i at one level $\mathbf{l}$ corresponds to the removal of all the complementary representative subsets $\hat{\mathbf{I}}_i$ at level $\mathbf{l}$. Thus, Figures 3.3 and 3.4 can naturally also be used to illustrate this factorization process. For example, we start from $\mathcal{A}$ and the domain Ω. After eliminating the complementary representative subsets at level $\mathbf{l}_s$, we obtain the reduced matrix $\mathcal{A}^{(\mathbf{l}_s-1)}$ corresponding to the mesh like in Figure 4.1(a). Another level of elimination yields Figure 4.1(b). By continuing this, we get the meshes in Figures 3.4(a–b). Notice that the final reduced matrix $\mathcal{A}^{(0)}$ is a standard HSS matrix, and a regular HSS ULV factorization can be applied. Thus, the factorization process also corresponds to the recursive sparsification of the mesh and dimension reduction for $\mathcal{A}$.



(a) Eliminating complementary representative subsets at the bottom levels

(b) Eliminating another level

FIG. 4.1. *Illustration of the MHS ULV (MULV) factorization process with the aid of representative subsets.*

See Algorithm 1 for a sketch of the factorization scheme. The design of the MHS framework makes it very convenient to organize and understand the scheme despite the large amount of technical and implementation details.

Algorithm 1 MULV factorization of an MHS matrix

procedure MULV
  for node $i = 1, 2, \ldots,$ root$(\mathcal{T})$ **do**
    if i is a leaf **then**
      $\hat{D}_i \leftarrow D_i$ ▷ *Leaf level inner HSS form*
    else
      $\hat{D}_i \leftarrow \begin{pmatrix} S_{c_1} & B_{c_1} \\ B_{c_1}^T & S_{c_2} \end{pmatrix}$ ▷ *Merging child level HSS contributions*
    end if
    if $i <$ root$(\mathcal{T})$ **then**
      $\tilde{D}_i \leftarrow (I + \tilde{E}_i)(\Pi_i^T \hat{D}_i \Pi_i)(I + \tilde{E}_i^T)$ ▷ *HSS construction (e.g., randomized construction via fast multiplication of this matrix and vectors)*
      Compute partial ULV factorization of $\tilde{D}_i$ as in (4.10)–(4.11)
      $S_i \leftarrow$ HSS form Schur complement
    else ▷ *Root node*
      $\Pi_i \leftarrow$ permutation of $\mathbf{I}_i$ based on connectivity
      $\tilde{D}_i \leftarrow \Pi_i^T \hat{D}_i \Pi_i$ ▷ *HSS construction*
      Compute ULV factorization of $\tilde{D}_i$
    end if
  end for
end procedure

Remark 4.2. It can be observed that this MULV factorization is very similar to the structured multifrontal methods in [28, 29, 32] for sparse factorizations. The representative subsets can be regarded as (generalized) separators in nested dissection. The outer ULV factorization is similar to the multifrontal framework, and the inner ULV factorization corresponds to the intermediate ULV factorizations in the structured multifrontal methods. The matrices S_i in (4.8) and $\tilde{D}_p$ in (4.13) play roles similar to the update matrix and frontal matrix in the multifrontal method, respectively. The merging step (4.9) and the permutation $\Pi_p^T \hat{D}_p \Pi_p$ are similar to the extend-add operation in the multifrontal method. All such intrinsic connections are valuable in the sense that they enable us to share ideas among different types of hierarchical methods. For example, instead of keeping explicit S_i, we may use randomized method as in the randomized multifrontal method in [29] so as to pass skinny matrix-vector products instead of S_i itself in the communication with the parent node. In addition, we may then also directly apply MHS methods to some sparse problems. More details on exploiting such connections will appear in future work [31].

The design of the MHS structure also makes it very convenient to analyze the MHS methods. For example, we can show the complexity of the factorization similarly to HSS complexity studies.

Theorem 4.1. *Suppose $\mathcal{A}$ is an $N \times N$ MHS matrix with MHS rank r, where the R_i, B_i generators at level $\mathbf{l}$ have sizes $O(M_{\mathbf{l}}) = O(\frac{M}{2^{\lfloor \mathbf{l}/2 \rfloor}})$ with $M = N^{1/2}$, and the leaf level D_i generators have sizes $O(M_{\mathbf{l}_s}^2)$ for $M_{\mathbf{l}_s}$ in (3.17) and $\mathbf{l}_s$ in (3.18). If all the matrices $\tilde{D}_i$ in Algorithm 1 have HSS ranks $O(r)$, then the complexity of Algorithm 1 is $O(r^2 N)$, and the storage for the factors is $O(rN)$.*

Proof. The structured operations associated with a node at level $\mathbf{l}$ cost at most

$O(r^2 M_{\mathbf{l}})$ flops. Thus from (3.17), the total MULV factorization cost is

$$\xi_{\text{fact}} = 2^{\mathbf{l}_s} O(r^2 M_{\mathbf{l}_s}^2) + \sum_{\mathbf{l}=1}^{\mathbf{l}_s} 2^{\mathbf{l}} O\left(r^2 M_{\mathbf{l}}\right) = 2^{\mathbf{l}_s} O(r^2 \log^2 M) + \sum_{\mathbf{l}=1}^{\mathbf{l}_s} 2^{\mathbf{l}} O\left(\frac{r^2 M}{2^{\lfloor \mathbf{l}/2 \rfloor}}\right)$$
$$= O(r^2 M^2) + \sum_{\hat{\mathbf{l}}=0}^{\lfloor \mathbf{l}_s/2 \rfloor} 2^{\hat{\mathbf{l}}} O(r^2 M) = O(r^2 M^2) = O(r^2 N).$$

The storage can be similarly estimated. ∎

REMARK 4.3. In the ideal situation, when $r = O(1)$, the complexity is then linear in N. In practice, r is likely also proportional to a low-degree power of $\log M$, so that the total factorization cost is proportional to N times a polylogarithmic term of N. To simplify the implementation, we may also keep S_i in a dense form. The total factorization cost would then be increased by a factor that is a polylogarithmic term of N.

Similarly, we can show the stability of the MHS factorization by recursively using the HSS stability analysis in [24, 25]. The details are expected to appear in [28]. The factorization is also fully scalable in the sense that all the operations at the same hierarchical level in each layer can be performed simultaneously.

4.3. Other algorithms. We can similarly design an MHS solution algorithm based on the MULV factors. Again, this follows an outer layer HSS ULV solution scheme [36, 33], with inner operations performed in structured forms. The solution cost is $O(rN)$ following the assumptions in Theorem 4.1. MHS matrix-vector and matrix-matrix multiplication schemes can also be conveniently designed based on the corresponding HSS algorithms in [6, 19].

We may also develop an MHS selected inversion method to computed the diagonal blocks of the inverse. Following Remark 4.2, this is very similar to the selected inversion method in [35] that is based on an outer multifrontal inversion with inner HSS operations.

5. Numerical experiments. To verify the effectiveness of MHS structures and the efficiency of MHS factorizations, we solver the linear system

$$Ax = b, \tag{5.1}$$

where A is a discretized matrix as in (3.2). We report some rank bounds to show the feasibility of MHS approximations, and then demonstrate the performance of MHS factorizations by reporting the factorization and solution costs, storage, and solution accuracies. The following measurements are used.

- $\tilde{r}$: (outer) HSS rank when A is approximated by an HSS form;
- $\tilde{r}_{\tilde{D}}$: maximum size of all the matrices $\tilde{D}_p$ as in (4.13) in the ULV factorization when A is approximated by an HSS form;
- r: MHS rank;
- $r_{\tilde{D}}$: maximum of the inner HSS ranks of all the matrices $\tilde{D}_p$ as in (4.13), which measures the intermediate rank structures in the MULV factorization;
- ξ_{fact}: factorization flop count;
- ξ_{sol}: solution flop count;
- σ: storage for the structured matrix approximation in terms of the number of nonzeros;

- $\frac{\|A\tilde{x}-b\|_2}{\||A|\cdot|x|+|b|\|_2}$: relative residual, where $\tilde{x}$ is the numerical solution and b is generated with the exact solution x set to be random;
- $\frac{\|x-\tilde{x}\|_2}{\|x\|_2}$: relative accuracy.

In practice, we use appropriate sizes of the resulting compressed forms for the rank measurements. For example, if E is approximated by a compressed form GH^T, the column size of G is used as the rank measurement for E. Also, simplifications as in Remarks 4.1 and 4.3 are used in the implementation. Two types of discretized functions ϕ in (3.2) are considered in two examples.

EXAMPLE 1. First, consider ϕ to be the 2D Laplace free-space Green's function and

$$A_{ij} = \begin{cases} 1, & i = j, \\ \frac{h^2}{2\pi} \log|y_i - y_j|, & i \neq j, \end{cases}$$

where y_j's are points on a uniform grid in the domain $[-1,1]\times[-1,1]$ with M points in each direction.

We inspect whether A can be approximated by compact MHS forms for a given tolerance. More specifically, we look at the inner structures within an outer HSS approximation to A, as discussed in Section 3.3. Accordingly, to show the advantage of MHS structures over HSS structures, we also show the performance when A is directly approximated by an HSS form (which is the outer layer HSS form of the MHS approximation). A relative tolerance 10^{-6} is used for all the compression steps. The matrix size $N = M^2$ ranges from 128^2 to 2048^2. The number of outer HSS levels $\mathbf{l}_s$ varies accordingly.

As shown in Table 5.1, when the mesh dimension M doubles, the HSS rank $\tilde{r}$ roughly doubles, while the MHS rank r remains about the same. For $N = 2048^2$, r is almost 40 times as small as $\tilde{r}$. Similarly, $\tilde{r}_{\tilde{D}}$ roughly doubles, while $r_{\tilde{D}}$ only slowly increases. This confirms the feasibility of MHS approximations for the problem.

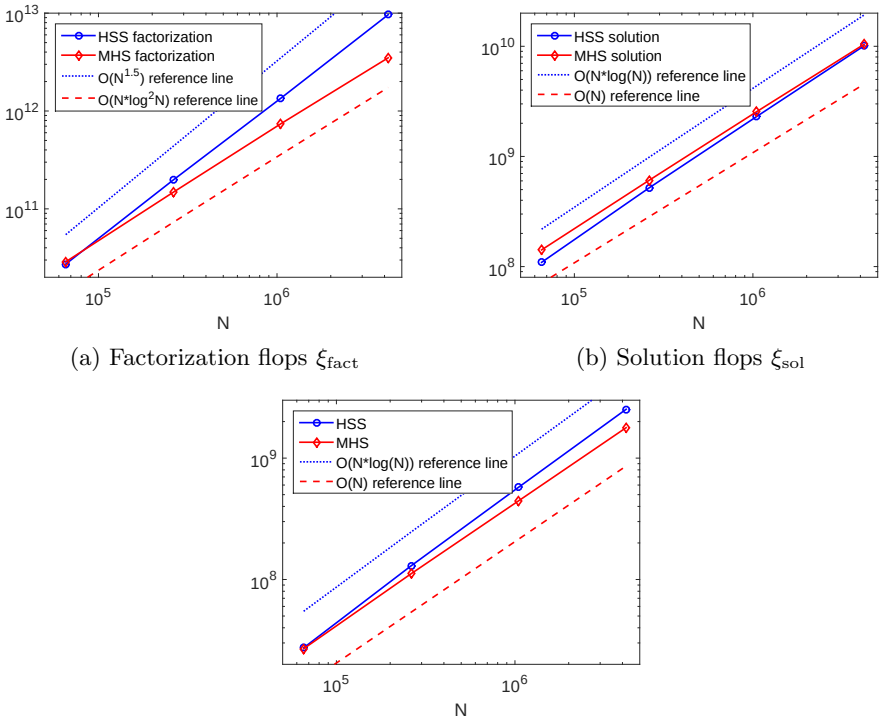
TABLE 5.1
Example 1. Rank structures of the MHS approximation and MULV factorization as compared with those of the HSS approximation and ULV factorization, respectively.

N		128^2	256^2	512^2	1024^2	2048^2
$\mathbf{l}_s$		6	8	10	12	14
HSS	$\tilde{r}$	247	472	898	1710	3302
	$\tilde{r}_{\tilde{D}}$	496	945	1804	3443	6626
MHS	r	80	80	80	83	84
	$r_{\tilde{D}}$	80	109	139	169	189

Table 5.2 compares the performance of HSS ULV factorization and MULV factorization. To show the asymptotic complexity of the MULV factorization, we also plot ξ_{fact} in Figure 5.1(a). It indicates that the MULV factorization roughly follows the $O(N\log^2 N)$ complexity. On the other hand, the HSS ULV factorization costs $O(N^{1.5})$ [27] based on Lemma 3.1. As discussed in [27], the HSS method is still very competitive in terms of the solution cost. Both MHS and HSS factorizations have nearly $O(N)$ solution costs. Still, the MHS solution cost grows slower with N. See Figure 5.1(b). The storage for the MHS and HSS matrices is given in Figure 5.1(c). The accuracies of the MHS solution is given in Table 5.3.

TABLE 5.2
Example 1. Costs and storage of the MHS and HSS methods.

N		128^2	256^2	512^2	1024^2	2048^2
HSS	ξ_{fact}	3.81e09	2.72e10	1.97e11	1.35e12	9.64e12
	ξ_{sol}	2.24e07	1.09e08	5.18e08	2.30e09	1.01e10
	σ	5.60e06	2.74e07	1.30e08	5.76e08	2.53e09
MHS	ξ_{fact}	5.37e09	2.87e10	1.48e11	7.39e11	3.50e12
	ξ_{sol}	3.20e07	1.42e08	6.07e08	2.55e09	1.04e10
	σ	6.35e06	2.69e07	1.12e08	4.45e08	1.78e09



(a) Factorization flops ξ_{fact} (b) Solution flops ξ_{sol}

(c) Matrix storage σ

FIG. 5.1. *Example 1. Costs and storage of the MHS method as compared with the HSS method.*

EXAMPLE 2. Then we replace ϕ in the previous example by the 3D Laplace free-space Green's function, and the matrix A is given by

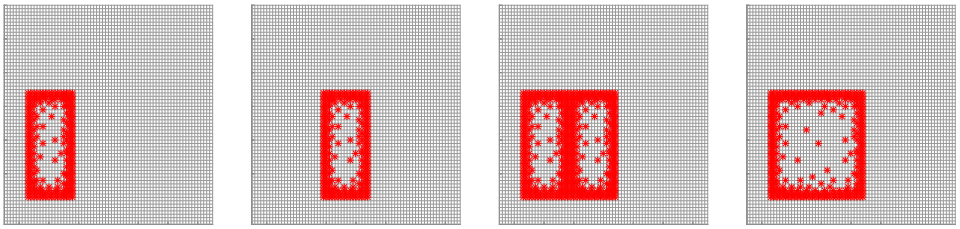
$$A_{ij} = \begin{cases} 1, & i = j, \\ -\frac{h^2}{4\pi} \frac{1}{|y_i - y_j|}, & i \neq j. \end{cases}$$

In this case, unlike the previous example where representative points cluster near the boundaries of a domain [7], the SPRR factorization yields representative points that may also be away from the boundaries. This can be observed from Figure 5.2. See Figure 5.3 for the collection of representative subsets for all the nodes at each level. They correspond to a reduced MHS matrix. This is consistent with Figure

TABLE 5.3
Example 1. Accuracies of the MHS solution.

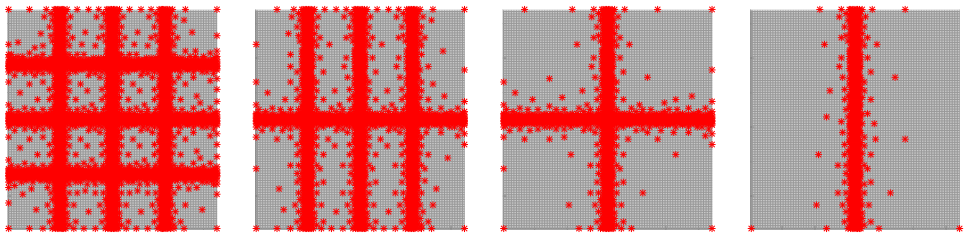
N	128^2	256^2	512^2	1024^2	2048^2
$\frac{\|x-\tilde{x}\|_2}{\|x\|_2}$	$4.13e-6$	$6.98e-6$	$5.76e-6$	$1.79e-5$	$1.42e-5$
$\frac{\|A\tilde{x}-b\|_2}{\|\|A\|\cdot\|x\|+\|b\|\|_2}$	$1.87e-6$	$2.94e-6$	$2.51e-6$	$7.92e-6$	$6.25e-6$

4.1 and clearly shows how the complimentary representative subsets $\hat{\mathbf{I}}_i$ at each level are eliminated and the mesh is sparsified. Note that in the MULV factorization, the elimination of $\hat{\mathbf{I}}_i$ is also done through a ULV factorization, so a similar inner sparsification process is involved.



(a) $\mathbf{I}_i$ (b) $\mathbf{I}_j$ (c) $\mathbf{I}_i \cup \mathbf{I}_j$ (d) $\mathbf{I}_p$

FIG. 5.2. *Example 2. Some examples of representative subsets in a mesh (zoomed in), and the selection of* $\mathbf{I}_p$ *from lower level representative subsets* $\mathbf{I}_i$ *and* $\mathbf{I}_j$*, where* $j = \mathrm{sib}(i)$*,* $p = \mathrm{par}(i)$*.*



(a) $\mathbf{l} = 3$ (b) $\mathbf{l} = 2$ (c) $\mathbf{l} = 1$ (d) $\mathbf{l} = 0$

FIG. 5.3. *Example 2. Collection of representative subsets at each level* $\mathbf{l}$ *corresponding to a reduced MHS matrix, where the mesh is* 128×128*.*

The rank structures are reported in Table 5.4. Although the MHS ranks in this case are higher than in the previous case, they are still much smaller than the outer HSS ranks. Note that the ordering of the representative points impacts the measured MHS ranks, as mentioned in Remark 4.1. On the other hand, the rank structure of the E_p matrices in Theorem 3.5 do not depend on any ordering. Thus, we also report the maximum of the numerical ranks of the E_p matrices for all the nodes p of $\mathcal{T}$, denoted r_E. This bound is a more precise measurement of the intrinsic structures within the off-diagonal basis generators. r_E is quite smaller than r in Table 5.4.

The costs and storage of the MULV method are given in Table 5.5. Due to the higher MHS ranks than in the previous example, it needs N to be much larger to see a significant advantage over the HSS method. However, the asymptotic complexities

TABLE 5.4
Example 2. Rank structures of the MHS approximation and MULV factorization as compared with those of the HSS approximation and ULV factorization, respectively.

N		128^2	256^2	512^2	1024^2	2048^2
$\mathbf{l}_s$		6	8	10	12	14
HSS	$\tilde{r}$	377	743	1471	2932	5893
	$\tilde{r}_{\tilde{D}}$	755	1491	2948	5882	11836
MHS	r_E	99	114	131	139	152
	r	99	136	201	384	510
	$r_{\tilde{D}}$	159	274	354	483	647

of the MHS method can already be observed from Figure 5.4. According to Theorems 3.2, 3.5, and 4.1, the MULV factorization cost is expected to be around $O(N \log^4 N)$, as verified by Figure 5.4(a). Similar asymptotic behaviors can be observed for the solution cost and storage.

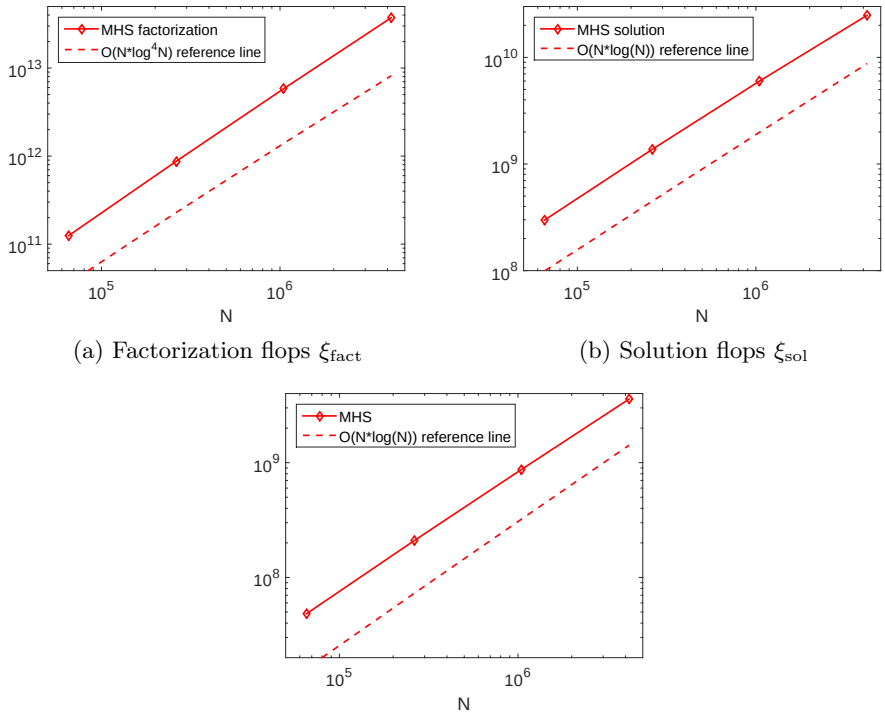
TABLE 5.5
Example 2. Costs and storage of the MHS and HSS methods.

N		128^2	256^2	512^2	1024^2	2048^2
HSS	ξ_{fact}	9.06e09	8.36e10	7.30e11	6.21e12	5.21e13
	ξ_{sol}	4.40e07	1.88e08	9.79e08	4.89e09	2.38e10
	σ	8.50e06	4.70e07	2.45e08	1.22e09	5.95e09
MHS	ξ_{fact}	1.63e10	1.25e11	8.73e11	5.83e12	3.73e13
	ξ_{sol}	5.97e07	2.98e08	1.37e09	6.00e09	2.48e10
	σ	1.06e07	4.84e07	2.09e08	8.74e08	3.58e09

TABLE 5.6
Example 2. Accuracies of the MHS solution.

N	128^2	256^2	512^2	1024^2	2048^2
$\frac{\|x-\tilde{x}\|_2}{\|x\|_2}$	$5.69e-5$	$7.34e-5$	$2.23e-4$	$3.33e-4$	$9.98e-4$
$\frac{\|A\tilde{x}-b\|_2}{\|\|A\|\cdot\|x\|+\|b\|\|_2}$	$2.30e-5$	$2.74e-5$	$8.90e-5$	$9.60e-5$	$4.20e-4$

6. Conclusions. In this work, we design MHS structures and propose MHS factorizations. The MHS structure extends the HSS structure to multiple dimensions by recursively incorporating HSS and low-rank structures into the generators of outer-layer HSS forms. Based on FMM and algebraic methods for selecting representative points, we have shown the existence of MHS structures within the approximation of some multi-dimensional discretized dense matrices. The multi-layer design makes it convenient to explore multi-dimensional FMM structures by taking advantage of existing HSS algorithms and analysis. In particular, the MULV factorization integrates multiple hierarchical ULV factorizations, which also has a strong connection to hierarchical sparse factorizations. This provides a great potential to unify dense and sparse structured factorization methods. The factorization is fully stable and scalable, and has nearly linear complexity. The work provides a proof-of-concept study



(a) Factorization flops ξ_{fact}

(b) Solution flops ξ_{sol}

(c) Matrix storage σ

FIG. 5.4. *Example 2. Costs and storage of the MHS method.*

for multi-layer hierarchical structures. The MHS structure in two dimensions can be used approximate some 2D discretized integral equations or dense Schur complements in some 3D discretized PDEs, which can lead to direct solvers with nearly linear complexity. Further optimization of the practical ordering strategies for the representative points is expected to be done. Efficient implementations will be developed for the purpose of large-scale dense and sparse direct solutions for multi-dimensional problems.

REFERENCES

[1] A. AMINFAR, S. AMBIKASARAN, E. DARVE, *A fast block low-rank dense solver with applications to finite-element matrices*, J. Comp. Phys., 304 (2016), pp. 170–188.

[2] R. BEATSON AND L. GREENGARD, *A Short Course on Fast Multipole Methods*, In M. Ainsworth, & al (Eds.), Wavelets, multilevel methods, and elliptic PDEs, pp. 1–37, (Numerical Mathematics and Scientific Computation.) Oxford University Press.

[3] S. BÖRM, L. GRASEDYCK, AND W. HACKBUSCH, *Introduction to hierarchical matrices with applications*, Eng. Anal. Bound. Elem, 27 (2003), pp. 405–422.

[4] S. BÖRM AND W. HACKBUSCH, *Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices*, Computing, 69 (2002), pp. 1–35.

[5] D. CAI AND J. XIA, *Bridging the gap between the fast multipole method and HSS structures*, Purdue GMIG Report 15-15, April 2015.

[6] S. CHANDRASEKARAN, P. DEWILDE, M. GU, AND T. PALS, *A fast ULV decomposition solver for hierarchically semiseparable representations*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 603–622.

[7] E. CORONA, P. G. MARTINSSON, AND D. ZORIN, *An $O(N)$ direct solver for integral equations*

in the plane, Appl. Comput. Harmon. Anal. 38 (2015), pp. 284–317.
[8] I. S. Duff and J. K. Reid, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Bans. Math. Software, 9 (1983), pp. 302–325.
[9] J. A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal., 10 (1973), pp. 345–363.
[10] A. Gillman, P. Young, and P. G. Martinsson, *A direct solver with $O(N)$ complexity for integral equations on one-dimensional domains*, Front. Math. China, 7 (2012), pp. 217–247.
[11] L. Grasedyck, R. Kriemann, and S. Le Borne, *Domain decomposition based $\mathcal{H}$-LU preconditioning*, Numer. Math., 112 (2009), pp. 565–600.
[12] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comp. Phys., 73 (1987), pp. 325–348.
[13] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong-rank revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.
[14] N. Halko, P. G. Martinsson, and J. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288.
[15] K. L. Ho and L. Ying, *Hierarchical interpolative factorization for elliptic operators: integral equations*, Comm. Pure Appl. Math., 69 (2016), pp. 1314–1353.
[16] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, 1994.
[17] R. Kriemann and S. Le Borne, *H-FAINV: Hierarchically factored approximate inverse preconditioners*, Comput. Vis. Sci. 17 (2015), pp. 135–150.
[18] L. Lin, J. Lu, and L. Ying, *Fast construction of hierarchical matrix representation from matrix-vector multiplication*, J. Comput. Phys., 230 (2011), pp. 4071–4087.
[19] W. Lyons, *Fast Algorithms with Applications to PDEs*, PhD thesis, University of California Santa Barbara, USA, 2005.
[20] P. G. Martinsson, *A fast randomized algorithm for computing a hierarchically semiseparable representation of a matrix*, SIAM. J. Matrix Anal. Appl., 32 (2011), pp. 1251–1274.
[21] P. G. Martinsson and V. Rokhlin, *A fast direct solver for boundary integral equations in two dimensions*, J. Comput. Phys., 205 (2005), pp. 1–23.
[22] P. G. Schmitz and L. Ying, *A fast direct solver for elliptic problems on general meshes in 2D*, J. Comput. Phys., 231 (2012), pp. 1314–1338.
[23] S. Wang, M. V. de Hoop, and J. Xia, *Acoustic inverse scattering via Helmholtz operator factorization and optimization*, J. Comput. Phys., 229 (2010), pp. 8445–8462.
[24] Y. Xi and J. Xia, *On the stability of some hierarchical rank structured matrix algorithms*, SIAM J. Matrix Anal. Appl., to appear, (2016).
[25] Y. Xi, J. Xia, S. Cauley, and V. Balakrishnan, *Superfast and stable structured solvers for Toeplitz least squares via randomized sampling*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 44–72.
[26] Y. Xi, J. Xia, and R. Chan, *A fast randomized eigensolver with structured LDL factorization update*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 974–996.
[27] J. Xia, *On the complexity of some hierarchical structured matrix algorithms*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 388–410.
[28] J. Xia, *Efficient structured multifrontal factorization for general large sparse matrices*, SIAM J. Sci. Comput., 35 (2013), pp. A832–A860.
[29] J. Xia, *Randomized sparse direct solvers*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 197–227.
[30] J. Xia, *O(n) complexity randomized 3D direct solver with HSS2D structure*, Purdue GMIG Report 14-18, April 2014.
[31] J. Xia, *Unifying hierarchically structured sparse and dense matrix factorizations*, under preparation, 2015.
[32] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Superfast multifrontal method for large structured linear systems of equations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1382–1411.
[33] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li, *Fast algorithms for hierarchically semiseparable matrices*, Numer. Linear Algebra Appl., 17 (2010), pp. 953–976.
[34] J. Xia and M. Gu, *A multi-structured stable and superfast Toeplitz solver*, preprint, 2009.
[35] J. Xia, Y. Xi, S. Cauley, and V. Balakrishnan, *Fast sparse selected inversion*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 1283–1314.
[36] J. Xia, Y. Xi, and M. Gu, *A superfast structured solver for Toeplitz linear systems via randomized sampling*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 837–858.