A STABLE AND EFFICIENT MATRIX VERSION OF THE FAST 
MULTIPOLE METHOD 
DIFENG CAI* AND JIANLIN XIA*

Abstract. It is well known that the fast multipole method (FMM) works nicely for evaluating matrix-vector products involving discretized kernel functions. In particular, for 1D problems, hierarchically semiseparable (HSS) provides a fast and stable way for relevant matrix operations, especially direct factorizations. However, in structured matrix computations with HSS matrices, HSS forms have always been constructed based on algebraic compression of the off-diagonal blocks via SVDs or randomized sampling. In this work, we show how to directly obtain an HSS representation to the matrix derived in FMM based on simple Taylor expansion of the kernel function. Moreover, a simple scaling strategy is introduced to ensure the stability of FMM based on truncated Taylor expansion of arbitrary degree. A sequence of numerical aspects are studied, including 2D problems.

Key words. fast multipole method, HSS matrix, stability 

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

1. Introduction. Given a matrix

\[
A = \begin{bmatrix} \kappa(x_i, y_j) \end{bmatrix}_{m \times n},
\]

where \( \mathbf{x} = \{x_1, \ldots, x_m\}, \mathbf{y} = \{y_1, \ldots, y_n\} \) are two sets of points on the complex plane and \( \kappa(x, y) \) is a kernel function, in the form such as \( 1/|x - y|, \log |x - y|, \) \((x, y \in \mathbb{C}, x \neq y), \) etc., we know that the task of multiplying \( A \) by a vector can be handled efficiently in linear complexity by fast multipole method (FMM) first introduced in [14]. It is shown in [15] that FMM is essentially a hierarchical factorization of matrix \( A \) to a given accuracy and the cost of forming such a factorization is linear. In fact, this kind of hierarchical matrix is a typical example of \( \mathcal{H}^2 \)-matrices ([9], [11]), which admits fast matrix-vector multiplication. For convenience, we may call the hierarchical matrix derived in fast multipole method the FMM matrix.

Despite the reputation of FMM as a fast summation technique, little is known about whether the structure of FMM matrices can be used to derive a factorization of \( A \) that facilitates solving the \( n \)-by-\( n \) linear system

\[
Aq = z
\]

in \( O(n) \) complexity and is suitable for parallel computations.

On the other hand, we know that a special class of \( \mathcal{H}^2 \) matrices, called hierarchical semi-separable (HSS) matrices ([4], [5]), admit a wide range of matrix operations, such as matrix-vector multiplication([10], [3]), ULV factorization([5]), etc., all of which can be carried out in linear complexity and are well-suited for parallel computations. However, HSS representations have always been constructed based on algebraic compression of the off-diagonal blocks via SVDs(cf.[5]) or randomized sampling(cf.[13]), which is usually expensive.

With the strength and weakness of both matrices (FMM matrices and HSS matrices) being stated, and by noting the fact that HSS matrices can be regarded as a subclass of FMM matrices, we ask the following two questions:

*Department of Mathematics, Purdue University, West Lafayette, IN 47907, USA (cai92@purdue.edu, xiaj@math.purdue.edu). The research of Jianlin Xia was supported in part by an NSF CAREER Award DMS-1255416 and an NSF grant DMS-1115572.
1. Is there a way to construct an HSS representation directly as is the case for FMM matrices, so that algebraic compressions are no longer needed?

2. Is it possible to rapidly transform an FMM matrix into an HSS matrix, so that a direct factorization (for example, ULV factorization) can be employed and hence (1.2) can be solved in linear complexity?

It has been noticed in [5] that it is possible to form an HSS matrix using FMM techniques under certain assumptions, but no detailed results or examples were given. In this paper, we consider the fast multipole method for essentially one-dimensional (1D) problems (where points are distributed on a planar curve) and show that in this case, the answers to the two questions above are affirmative. To be more precise, the following results are presented in this paper:

1. The 1D FMM matrix naturally bears an HSS representation and no algebraic compression is needed to derive the HSS representation. In fact, by using Taylor expansion to approximate the kernel function, we can write down the HSS representation of $A$ by hand when points are uniformly distributed;

2. With the HSS representation, the complexity for computing the ULV factorization ([5]) as well as solving (1.2) is $O(n)$;

3. HSS representation for adaptive FMM can also be constructed with no difficulty;

4. Our method is stable in the sense that the magnitude of each entry in the hierarchical factorization is either bounded by 1 or proportional to the magnitude of the kernel function evaluated at centers of expansion.

The following notation will be used:

- A bold symbol such as $v$ denotes a vector or a set with entries $v_i$;
- $i = 1 : m$ denotes an enumeration of index $i$ from 1 to $m$;
- For a matrix $A$, $A(i, j)$ denotes the entry of $A$ located at $i$th row and $j$th column;
- $\odot$ denotes the elementwise multiplication (Hadamard product) of two matrices, as is used in [15];
- $\text{diag}(\ldots)$ denotes the block diagonal matrix with blocks in $(\ldots)$.

The rest of this paper is organized as follows. In Section 2, we present by using Taylor expansion and by introducing scaling parameters a stable low-rank factorization for matrix $A$ when two sets of points are separated from each other. In Section 3, we review the one-dimensional fast multipole method, and derive the corresponding matrix representation, similar to [15]. In Section 4, we introduce the notion of HSS matrices and derive the HSS representation for the FMM matrix constructed in Section 3. In Section 5, we show how to derive HSS representation for adaptive FMM. In Section 6, numerical aspects are investigated, including stability of FMM after suitable scaling introduced in Section 2, complexity analysis, structures of HSS generators derived in Section 5 etc.

**Remark 1.1.** Since the hierarchical structure of FMM matrix only depends on the distribution of points, so does the corresponding HSS representation. That is to say, once the nested partition is done, the HSS representation can be readily derived.

2. Stable kernel expansion and low-rank approximation.

2.1. Kernel expansion and low-rank approximation. It is well-known (cf. [2, Chapter 2]) that a degenerate approximation of the kernel function

$$\kappa(x, y) \approx \sum_{k=0}^{r-1} \sum_{l=0}^{k} c_{k,l} f_l(x) g_{k-l}(y)$$
yields a low-rank approximation of the kernel matrix $A$ if $r$ is small compared to size of $A$. For simplicity, we only consider the kernel $\kappa(x, y) = 1/(x - y)$ with $x, y \in \mathbb{C}$ and $x \neq y$, though our approach in later sections is applicable to other kernels as long as a degenerate approximation exists.

For two points $a, b \in \mathbb{C}$ and a real number $\tau \in (0, 1)$, if $x, y \in \mathbb{C}$ are such that $\frac{|(x - a) - (y - b)|}{|b - a|} \leq \tau$, then it can be easily deduced from Taylor expansion (cf. [2, Chapter 2]) that

$$\frac{1}{x - y} = \sum_{k=0}^{r-1} c_k \sum_{l=0}^{k} f_l(x - a) f_{k-l}(b - y) + \epsilon_r,$$

where

$$f_k(x) = \frac{x^k}{k!}, \quad c_k = -\frac{k!}{(b - a)^{k+1}}, \quad |\epsilon_r| \leq \frac{\tau^r}{|b - a|(1 - \tau)}.$$

Note that

$$\frac{|x - y|}{|b - a|} \leq \frac{|(x - a) - (y - b)| + |a - b|}{|b - a|} \leq 1 + \tau.$$ 

Hence the truncation error $\epsilon_r$ can be estimated by

$$|\epsilon_r| \leq \frac{\tau^r}{|b - a|(1 - \tau)} \leq \frac{(1 + \tau)\tau^r}{(1 - \tau)} \frac{1}{|x - y|},$$

which indicates that the relative error of approximation (2.1) is bounded by

$$\frac{(1 + \tau)\tau^r}{(1 - \tau)}.$$

If $x$ and $y$ are well-separated in the sense below, a low-rank approximation of $A$ can be easily derived.

**Definition 2.1 ([15]).** Let $x = \{x_1, \ldots, x_m\}$ and $y = \{y_1, \ldots, y_n\}$ be two sets of points in $\mathbb{C}$. Let $a \in \mathbb{C}$ and $\delta_a > 0$ be the center and radius of $x$ in the sense that

$$|x - a| \leq \delta_a, \quad \forall x \in x.$$

Let $b \in \mathbb{C}$ and $\delta_b > 0$ be the center and radius of $y$. Given $\tau \in (0, 1)$, $x$ and $y$ are said to be well-separated with separation ratio $\tau$ if the following admissibility condition holds

$$\delta_a + \delta_b \leq \tau|a - b|.$$

In [14], the separation ratio $\tau$ is set to $\frac{2}{3} \approx 0.67$.

According to (2.1), the low-rank approximation of $A$ follows

$$A = U_0 B_0 V_0^T + A \odot E.$$
where
\[(2.6)\]
\[
U_0 = \begin{bmatrix}
f_0(x_1 - a) & \cdots & f_{r-1}(x_1 - a) \\
\vdots & \ddots & \vdots \\
f_0(x_m - a) & \cdots & f_{r-1}(x_m - a)
\end{bmatrix}, \quad V_0 = \begin{bmatrix}
f_0(y_1 - b) & \cdots & f_{r-1}(y_1 - b) \\
\vdots & \ddots & \vdots \\
f_0(y_n - b) & \cdots & f_{r-1}(y_n - b)
\end{bmatrix},
\]
\[
B_0 = \begin{bmatrix}
c_0 & \cdots & c_{r-2} & c_{r-1} \\
\vdots & \ddots & \vdots & \vdots \\
c_{r-2} & \cdots & & \\
c_{r-1} & & & 
\end{bmatrix}
\]
\[
\left[ (-1)^0 \right. \quad \cdots \quad \left. (-1)^{r-1} \right]
\]
with \(c_k = \frac{k!}{(b - a)^{k+1}}\), and \(|E(i,j)| \leq (1 + \tau)\tau^r/(1 - \tau)\).

We see that \(U_0\) and \(V_0\) are determined by the sets \(x\) and \(y\), respectively. The matrix \(B_0\) is an \(r \times r\) square matrix that depends only on \((b - a)\).

### 2.2. Scaling parameters.

It should be pointed out that the factorization in (2.5) is not numerically stable due to the factorial terms in \(c_k\) and \(f_k\). To resolve the issue, we introduce some scaling parameters.

We wish to bound the factorial term in \(c_k\) and \(f_k(x - a)\). To find appropriate scaling parameters, we employ Stirling’s approximation, which reads:
\[(2.7)\]
\[
\lim_{r \to \infty} \frac{r!}{\sqrt{2\pi r} \left( \frac{r}{e} \right)^r} = 1, \quad \text{or} \quad r! \sim \sqrt{2\pi r} \left( \frac{r}{e} \right)^r \text{ for large } r.
\]

We introduce for each cluster a set of scaling parameters. That is, for cluster \(x\) with center \(a\) and radius \(\delta_a\), we define
\[(2.8)\]
\[
\eta_{a,k} = \begin{cases} 
1, & k = 0, \\
\left( \frac{k}{2\pi r} \right) \left( \frac{1}{\delta_a} \right)^k, & k = 1, \ldots, r - 1. 
\end{cases}
\]
Replacing \(a\) with \(b\) in the above equation yields the scaling parameter associated with cluster \(y\).

With scaling parameters, the expansion in (2.1) can be written as
\[(2.9)\]
\[
\kappa(x, y) = \sum_{k=0}^{r-1} c_k \sum_{l=0}^{k} f_l(x - a) f_{k-l}(b - y) + \epsilon_r \\
= \sum_{k=0}^{r-1} c_k \sum_{l=0}^{k} \eta_{a,l}^{-1} \eta_{b,k-l}^{-1} (\eta_{a,l} f_l(x - a)) (\eta_{b,k-l} f_{k-l}(b - y)) + \epsilon_r.
\]

Compared with (2.5), (2.6) the factorization of \(A\) now becomes:
\[(2.10)\]
\[
A = M_{a,b} + A \odot E, \quad \text{with } M_{a,b} = \hat{U}_a \hat{B}_{a,b} \hat{V}_b^T
\]
and
\[(2.11)\]
\[
\hat{U}_a = [\eta_{a,k} f_k(x_l - a)]_{l=1:m_k} = U_a S_a, \quad \hat{V}_b = [\eta_{b,k} f_k(y_l - b)]_{l=1:n_k} = V_b S_b, \\
\hat{B}_{a,b} = S_a^{-1} B_{a,b} S_b^{-1}, \quad S_i = \text{diag}(\eta_{i,k})_{k=0:r-1}.
\]

The matrices \(U_0, V_0, B_0\) are defined in (2.6).

The choice of scaling parameters in (2.8) will be explained in Section 6.2, where we investigate the stability issues.
3. Matrix representation for 1D FMM. Before we construct the HSS form for 1D FMM, it would be beneficial for us to review the matrix representation for 1D FMM. For pedagogical reasons, we focus on FMM on the unit interval \( I = [0, 1] \) in \( \mathbb{R} \) and we assume the points are sampled from a uniform distribution in \( I \). The matrix version for 3D FMM is presented nicely in [15].

3.1. Notation.
1. Let \( T \) denote a perfect binary tree of \( L \) levels \((L \geq 3)\). The root node of \( T \) is simply written as root. For node \( i \in T \), we use \( S(i) \) to denote the sibling of node \( i \), \( P(i) \) the parent of node \( i \), and \( lv(i) \) the level in \( T \) where node \( i \) is located. For example, \( lv(\text{root}) = 1 \). A node \( i \) is called a leaf node if \( lv(i) = L \) and a nonleaf node otherwise. Tree \( T \) will be used throughout this paper.
2. Let \( \kappa(x, y) \) be a function in \( \mathbb{R}^2 \) and \( X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}, Y = \{y_1, \ldots, y_n\} \subseteq \mathbb{R} \). Consider the matrix \( A = (\kappa(x_i, y_j))_{i=1}^m_{j=1}^n \). Consider the matrix

\[ A_{ij} = \kappa(x_i, y_j) \]

for \( 1 \leq i \leq m, 1 \leq j \leq n \), \( x_i \in X, y_j \in Y \). For two intervals \( I_i, I_j \) in \( \mathbb{R} \), we use \( A_{ij} \) to denote the submatrix of \( A \) with entries \( \kappa(x, y) \) where \( x \in I_i \cap X \) and \( y \in I_j \cap Y \).
3. \( M_{i,j} \) denotes the rank-\( r \) approximation of \( A_{ij} \) as in (2.10).
4. For a vector \( v \), we use \( e \) to denote the length of \( v \). Therefore, \( v_{e-k}, v_e \) denote the \( (e-k) \)th component, the last component of \( v \), respectively. (Here \( e \) stands for end.)
5. Suppose \( p \) is an index and \( \xi = [\xi_1, \ldots, \xi_s], \mu = [\mu_1, \ldots, \mu_t] \) are index sets, we use \( U_{\xi}, T_{\xi,p}, B_{\xi,\mu} \) to denote the following matrices

\[
\begin{bmatrix}
\text{diag}(U_{\xi_1}, \ldots, U_{\xi_s}) & T_{\xi_1,p}
\end{bmatrix},
\begin{bmatrix}
B_{\xi_1,\mu_1} & \ldots & B_{\xi_1,\mu_t}
\end{bmatrix},
\begin{bmatrix}
T_{\xi,p} & B_{\xi_2,\mu_1} & \ldots & B_{\xi_2,\mu_t}
\end{bmatrix}
\]

respectively, where \( T_{\xi,p}(i = 1, \ldots, s) \) are matrices of same size. So are \( B_{\xi,\mu_j}(i = 1, \ldots, s, j = 1, \ldots, t) \).

3.2. Approximation. Consider two sets of points \( X = \{x_1, \ldots, x_m\} \) and \( Y = \{y_1, \ldots, y_n\} \), where \( x_1 < x_2 < \cdots < x_m \) and \( y_1 < y_2 < \cdots < y_n \) are sampled from the uniform distribution in \( I = [0, 1] \). We assume here for notational convenience that \( m = n \). Consider kernel matrix \( A \) in (1.1) associated with the kernel function

\[ \kappa(x, y) = \frac{1}{x - y}, \quad x \neq y, \]

where the value \( \kappa(x, x) \) can be arbitrary.

Given an error \( \epsilon \), we show in this section that the 1D FMM procedure produces a hierarchical matrix \( \tilde{A} \) such that

\[ A = \tilde{A} + A \odot E, \quad \text{with } |E(i, j)| \leq \epsilon, \]

and that the complexity of computing matrix-vector product for matrix \( \tilde{A} \) is \( O(rn) \), where \( r \) is a small constant proportional to \( |\log \epsilon| \).

3.3. Nested partition. For a given accuracy \( \epsilon \), according to (2.3) with \( \tau = 1/2 \), we choose approximation order \( r \) such that \( 3(1/2)^r \leq \epsilon \). Let \( T \) be a perfect binary tree with \( L \) levels such that \( n/2^{L-1} = O(r) \).

In the case of uniform distribution, we partition \( I \) hierarchically according to \( T \). Namely, at partition level \( l(1 \leq l \leq L) \), \( I \) is partitioned into \( 2^{l-1} \) equal parts. We denote by \( I_l \) the subinterval corresponding to node \( i \in T \).
Remark 3.1. For convenience, we identify the cluster $X \cap I_i$ (or $Y \cap I_j$) with $I_i$. That is, the center of $X \cap I_i$ is regarded as $a_i$ (center of $I_i$) and the radius of $X \cap I_i$ is set to $\delta_i = |I_i|/2$.

3.4. Near neighbors and interaction list. Given $i, j \in \mathcal{T}$, by interaction between $I_i$ and $I_j$, or simply interaction between $i$ and $j$, we mean the matrix $A|_{I_i \times I_j}$.

Given two nodes $i, j$ at level $l$ in $\mathcal{T}$, we say $I_i$ and $I_j$ (or simply $i$ and $j$) are near neighbors if they’re not well-separated in the sense of (2.4). Correspondingly, we have two types of interactions. The interaction between near neighbors are called near-field interaction. The interaction between well-separated sets are called far-field interaction.

The notion of interaction list introduced below clarifies which interaction we should consider at each level of partition.

Definition 3.1 ([17]). The interaction list for node $i$ at level $l$ is a list of nodes at level $l$ which are well-separated from node $i$ and whose parent nodes are near neighbors of parent of $i$. We use $\mathcal{L}(i)$ to denote the interaction list for node $i$.

3.5. Matrix representation for far-field interactions. For a node $i \in \mathcal{T}$ such that $\mathcal{L}(i)$ is not empty, since $i$ and $j$ are well-separated for each $j \in \mathcal{L}(i)$, according to (2.10), (2.11), we have

$$A|_{I_i \times I_j} = M_{i,j} + A|_{I_i \times I_j} \odot E, \quad \text{with} \quad M_{i,j} = \hat{U}_i \hat{B}_{i,j} \hat{V}_j^T$$

and $\hat{U}_i, \hat{V}_j, \hat{B}_{i,j}$ are derived from (2.11), (2.8) by replacing $a$ with $a_i$, $b$ with $a_j$, $x$ with points in $X \cap I_i$, and $y$ with points in $Y \cap I_j$. For example,

$$\eta_{i,k} = \begin{cases} 1, & k = 0, \\ \left(\tfrac{k}{(2\pi r)^{\frac{d}{2}} r^{\frac{1}{2}}}\right)^k, & k = 1, \ldots, r - 1, \end{cases}$$

where $\delta_i$ denotes the radius of $I_i$ as in Remark 3.1, and

$$\hat{B}_{i,j} = S_{i,j}^{-1} B_0 S_{i,j}^{-1}, \quad \text{with} \quad S_i = \text{diag}(\eta_{i,k})_{k=0,r-1}.$$

Since $\hat{U}_i$ is only dependent on node $i$, similar to the concept of local expansion vector introduced in [15], we call $\hat{U}_i$ (or $\hat{V}_i$) a local expansion matrix associated with node $i$ and cluster $X \cap I_i$ (or $Y \cap I_i$). Obviously, if $X = Y$, then $\hat{U}_i = \hat{V}_i$.

3.6. Telescoping expansion.

3.6.1. Levelwise factorization. For $1 \leq l \leq L$, let $A^{(l)}$ be the matrix derived from $A$ by retaining only submatrices $A|_{I_i \times I_j}$ where $i$ is at level $l$ and $j \in \mathcal{L}(i)$ and zeroing out other blocks in $A$. The nonzero patterns of $A^{(l)}(l = 3, 4, 5, 6)$ are illustrated in Fig.3.1. Based on the analysis above, we see that the matrix $A$ can be decomposed levelwisely into matrices corresponding to far-field interactions and near-field interactions:

$$A = A^{(1)} + \cdots + A^{(L)} + N,$$

where $N$ corresponds to near-field interactions in leaf level (level $L$) of partition. It can be seen that $N$ is a banded matrix with bandwidth $O(r)$.

Since there is no far-field interactions in the first and second level of partition, the interaction list for each node in those levels is empty. Therefore,

$$A^{(1)} = A^{(2)} = 0.$$
At level \( l \geq 3 \), since each nonzero block \( A |_{l \times l} \) represents a far-field interaction with \( j \in \mathcal{L}(i) \), we know from Section 3.5 that the block admits a low-rank approximation (3.3). Define at this moment

\[
\hat{B}_{i,j} = \begin{cases} 
0, & \text{if } j \notin \mathcal{L}(i), \\
S_j^{-1}B_0S_j^{-1}, & \text{if } j \in \mathcal{L}(i) \text{ (as in (3.5))}.
\end{cases}
\]

Suppose \( \mathbf{d} = [d_1, \ldots, d_{2^l-1}] \) denotes the list of nodes at level \( l \), ordered from left to right. Then together with the notation introduced in Section 3.1, \( A^{(l)} \) can be approximated by the following factorization

\[
A^{(l)} = K^{(l)} + A^{(l)} \odot E^{(l)}, \quad \text{with } K^{(l)} = U^{(l)}B^{(l)} \left( V^{(l)} \right)^T,
\]

and

\[
U^{(l)} = \text{diag}(\hat{U}_{d_1}, \ldots, \hat{U}_{d_{l-1}}) = \hat{U}_{\mathbf{d}}, \quad V^{(l)} = \text{diag}(\hat{V}_{d_1}, \ldots, \hat{V}_{d_{l-1}}) = \hat{V}_{\mathbf{d}}, \quad B^{(l)} = \hat{B}_{\mathbf{d}, \mathbf{d}}.
\]

It can be seen that, for each fixed \( l \), the nonzero patterns of \( A^{(l)}, K^{(l)}, E^{(l)} \) are same, and the nonzero pattern of \( B^{(l)} \) is same to \( A^{(l)} \) except that the size of each block in \( B^{(l)} \) is \( r \)-by-\( r \).

Combining (3.6), (3.8), we have the following decomposition of \( A \):

\[
A = K^{(3)} + \cdots + K^{(L)} + N + A \odot E
\]

where \( E = E^{(3)} + \cdots + E^{(L)} \). Since the nonzero blocks for \( E^{(i)} \) and \( E^{(j)} \) do not overlap if \( i \neq j \), it follows that \( E \) satisfies (3.2).

Due to the factorization of \( K^{(l)} \) \( (3 \leq l \leq L) \) in (3.8) and the sparsity of \( N \), it can be seen that the decomposition in (3.10) gives us an \( \mathcal{O}(r n L) = \mathcal{O}(\log \epsilon n \log n) \) algorithm for computing the matrix-vector product of \( A \) within a prescribed accuracy \( \epsilon \).

3.6.2. Translation relation and hierarchical basis. A linear complexity FMM exploits the relation(called translation relation) of local expansions between a parent node and its children. Specifically, the translation relation reads

\[
f_k(x-a_p) = \frac{(x-a_p)^k}{k!} = \sum_{l=0}^{k} \frac{(x-a_c)^l}{l!} \frac{(a_c-a_p)^{k-l}}{(k-l)!} = \sum_{l=0}^{k} \eta_{c,l} f_l(x-a_c) \eta_{c,r-l} f_{k-l}(a_c-a_p).
\]

Therefore,

\[
[\eta_{p,0} f_0(x-a_p), \ldots, \eta_{p,r-1} f_{r-1}(x-a_p)] = [\eta_{c,0} f_0(x-a_c), \ldots, \eta_{c,r-1} f_{r-1}(x-a_c)] T_{c,p},
\]

Fig. 3.1: nonzero patterns of \( K^{(l)}(or A^{(l)}, E^{(l)}) \) for \( l = 3, \ldots, L \) and \( K \) (with \( L = 6 \)).
where the translation matrix is given by

$$T_{c,p} = S_c^{-1} \begin{bmatrix} f_0(a_c - a_p) & \cdots & f_{r-1}(a_c - a_p) \\ \vdots & & \vdots \\ f_0(a_c - a_p) \end{bmatrix} S_p$$  (3.12)

with $S_i$ defined in (2.11).

Using the translation relation in (3.11), we derive below the relation of local expansions between a parent node $p$ and its children $c_1, c_2$.

$$\hat{U}_p = \begin{bmatrix} \hat{U}_{c_1} \\ \hat{U}_{c_2} \end{bmatrix}, \quad \hat{V}_p = \begin{bmatrix} \hat{V}_{c_1} \\ \hat{V}_{c_2} \end{bmatrix}$$  (3.13)

Therefore, the hierarchy of basis at two levels is given by

$$U^{(l-1)} = \text{diag}(\hat{U}_p) = \text{diag} \left( \begin{bmatrix} \hat{U}_{c_1} \\ \hat{U}_{c_2} \end{bmatrix} \right) \text{diag} \left( \begin{bmatrix} T_{c_1,p} \\ T_{c_2,p} \end{bmatrix} \right) = U^{(1)} T^{(l-1)},$$  (3.14)

$$V^{(l-1)} = \text{diag}(\hat{V}_p) = \text{diag} \left( \begin{bmatrix} \hat{V}_{c_1} \\ \hat{V}_{c_2} \end{bmatrix} \right) \text{diag} \left( \begin{bmatrix} T_{c_1,p} \\ T_{c_2,p} \end{bmatrix} \right) = V^{(1)} T^{(l-1)},$$

where node $p$ is at level $l - 1$ with children $c_1$ and $c_2$, and

$$T^{(l-1)} = \text{diag} \left( \begin{bmatrix} T_{c_1,p} \\ T_{c_2,p} \end{bmatrix} \right).$$

Therefore, by applying (3.14) recursively to (3.8), we have the following factorization of $K^{(l)}$ ($l \geq 3$):

$$K^{(l)} = U^{(L)} T^{(L-1)} \ldots T^{(l)} B^{(l)} \left( T^{(l)} \right)^T \ldots \left( T^{(3)} B^{(3)} \left( T^{(3)} \right)^T + B^{(4)} \right) \ldots \left( T^{(L-1)} \right)^T + B^{(L)} \left( V^{(L)} \right)^T.$$  (3.15)

Inserting (3.15) into (3.10), we obtain the telescoping expansion of $A$:

$$A = U^{(L)} \left( T^{(L-1)} \left( \ldots \left( T^{(3)} B^{(3)} \left( T^{(3)} \right)^T + B^{(4)} \right) \ldots \right) \left( T^{(L-1)} \right)^T + B^{(L)} \right) \left( V^{(L)} \right)^T + N + A \odot E = \hat{A} + A \odot E.$$  (3.16)

Here $\hat{A} = A - A \odot E$ is the hierarchical matrix produced by FMM. We assign each node $i$ ($i \neq \text{root}$) with the submatrices related to $i$ in the telescoping expansion in (3.16), namely, matrices $T_{i,j}(i), B_{i,j}(j \in L(i)), \hat{U}_i, \hat{V}_i$ (if $i$ is a leaf node), all of which are of size $O(r)$. Therefore, the complexity of matrix vector product for each node is $O(r^2)$, resulting in a complexity of $O(rn)$ for computing the matrix-vector product of $\hat{A}$, as $n = O(r2^L)$ and $N$ is a banded matrix with bandwidth being $O(r)$.

**Remark 3.2.** Note that the translation relation (3.11) holds for any pair of nodes $p,c$ in $\mathcal{T}$, because (3.11) is merely a consequence of binomial theorem. Therefore, if $I_p$ is partitioned into $s (s \geq 1)$ subintervals (not necessarily of equal length) $I_{c_1}, \ldots, I_{c_s}$, then a general form of (3.13) follows:

$$\hat{U}_p = \text{diag} \left( \hat{U}_{c_1}, \ldots, \hat{U}_{c_s} \right) T_{c,p},$$  (3.17)

where $c = [c_1, \ldots, c_s]$ and the notation introduced in Section 3.1 is used in (3.17).
4. From FMM matrix to HSS matrix.

4.1. Overview of HSS matrix. We follow the postordering tree notation as in [19]. An \( m \times n \) matrix \( A \) is said to be a Hierarchically Semi-Separable (HSS) matrix if there exist a binary tree \( T \) of \( L \) level (\( L \geq 3 \)) and matrices \( D_i, U_i, V_i, R_i, W_i, B_i \) (called HSS generators) associated with node \( i \) in \( T \) s.t.

\[
\begin{align*}
D_{\text{root}} &= A, & D_i &= \begin{bmatrix} D_c1 & U_c1B_c1V_cT_c1 & U_c1B_c1V_cT_c2 \\ U_c2B_c2V_cT_c1 & D_c2 \end{bmatrix}, \\
U_i &= \begin{bmatrix} U_c1 \\ U_c2 \end{bmatrix}, & V_i &= \begin{bmatrix} V_c1 \\ V_c2 \end{bmatrix}, \\
R_i &= \begin{bmatrix} R_c1 \\ R_c2 \end{bmatrix}, & W_i &= \begin{bmatrix} W_c1 \\ W_c2 \end{bmatrix},
\end{align*}
\]

where root denotes the root node of \( T \), \( c_1, c_2 \) are left child and right child of node \( i \), respectively, and \( i \neq \text{root} \) in (4.2).

From the recursive definitions in (4.2), we see that all \( U_i, V_i \) can be obtained if we only know \( U_i, V_i \) for each \( i \) at leaf level and \( R_i, W_i \) for all \( i \in T \) such that \( lv(i) \geq 3 \). Thus the original matrix \( A \) can be derived if we are given \( D_i, U_i, V_i \) for each leaf node \( i \), \( R_i, W_i \) for all \( i \) with \( lv(i) \geq 3 \) and \( B_i \) for all \( i \) with \( lv(i) \geq 2 \). Consequently, those are the HSS generators that are explicitly stored to represent \( A \).

Let \( \gamma_i, \phi_i \) denote the index sets such that \( D_i = A_{\gamma_i, \phi_i} \). Then we see from (4.1), (4.2) that the row block \( A_{\gamma_i, \phi_i} \) can be factorized into the form \( A_{\gamma_i, \phi_i} = U_iG_i \) for some matrix \( G_i \), which implies that the column vectors in \( U_i \) can span a vector space determined by vectors in \( A_{\gamma_i, \phi_i} \). Hence we call \( U_i \) the row basis for the row block \( A_{\gamma_i, \phi_i} \). Similarly, \( V_i^T \) is called column basis for the column block \( A_{\gamma_i, \phi_i}^T \). \( U_i, V_i \) are called basis matrices.

From the definition of an HSS matrix \( A \), we see that \( A \) has a telescoping expansion([13]):

\[
A = U(L) \left( U(L-1) \left( \ldots \left( U(2)B(1) \left( V(2)^T + B(2) \right) \ldots \right) \left( V(L-1)^T + B(L-1) \right) \right) \left( V(L)^T + B(L) \right) \right),
\]

Fig. 4.1: HSS representation corresponding to a perfect binary tree of 3 levels with postordering
where
\[
U^{(l)} = \begin{cases} 
\text{diag}(U_i)_{l=1}, & \text{if } l = L, \\
\text{diag} \left( \begin{bmatrix} R_{c_1} \\
R_{c_2} \end{bmatrix} \right)_{l=1} & \text{if } l < L,
\end{cases}
\]
and
\[
V^{(l)} = \begin{cases} 
\text{diag}(V_i)_{l=1}, & \text{if } l = L, \\
\text{diag} \left( \begin{bmatrix} W_{c_1} \\
W_{c_2} \end{bmatrix} \right)_{l=1} & \text{if } l < L,
\end{cases}
\]
and \(c_1, c_2\) denote the left child and right child of node \(i\), respectively. Obviously, the expansion in (4.3) is in the same form as the telescoping expansion for FMM in (3.16) if we ignore the term \(N + A \odot E\) in (3.16). However, the two telescoping expansions are actually different. In (3.16), we notice that \(B^{(l)}\), whose nonzero pattern is similar to \(A^{(l)}\) illustrated in Fig.3.1, is not block-diagonal, while in (4.3), \(B^{(l)}\) is a block-diagonal matrix. It can be easily seen from (4.3) that the complexity for multiplying an \(n\times n\) matrix \(A\) by a vector is \(O(rn)\) assuming the size of each \(B_i\) \((i \neq \text{root})\), \(D_i\) \((i\text{ a leaf node})\) is \(r \times r\).

4.2. General idea of transforming FMM to HSS. We clarify the notations used hereafter.

- The matrices with an overhead hat, e.g., \(\hat{U}_i\), are from FMM procedure in Section 3.
- The matrices with no overhead decorations, e.g., \(U_i\) are HSS generators.
- The matrices with an overhead bar, e.g., \(\bar{U}_i\), are intermediate matrices. The intermediate matrices \(\bar{U}_i, \bar{V}_i\) will not be stored or computed, but \(\bar{B}_i\) will be used in computing the HSS generator \(B_i\).

It should be noted that the HSS representation we are constructing is for the FMM matrix \(\hat{A}\) in (3.16), not exactly \(A\). Nonetheless, for convenience, we still say HSS representation for \(A\), assuming no confusion is made.

If we define matrix \(K\) by

\[
K = K^{(3)} + \cdots + K^{(L)},
\]
(3.10) then reads:

\[
A = K + N + A \odot E,
\]
(4.5)

where \(N\) represents near-field interactions between all pairs of adjacent nodes at leaf level and \(K\) is an approximation to \(A^{(3)} + \cdots + A^{(L)}\) whose nonzero blocks, derived from nested partition, represent the far-field interactions from level 3 to level \(L\).

The general idea of constructing the HSS representation of \(A\) is to find the HSS representation for matrix \(K\) and matrix \(N\) respectively, then merge those two sets of HSS generators using a technique in [18] to form an HSS representation for \(A\). The nonzero pattern of matrix \(K\) is shown as the rightmost figure in Fig.3.1, where the zero blocks are in white.

The difficulty here is to find the HSS of \(K\). We do this in three steps:

1. First we factorize each off-diagonal block \(K|_{l \times I_j}\) with \(j = S(i)\) into a form \(K|_{l \times I_j} = \hat{U}_i \bar{B}_i \bar{V}_j^T\), where \(\hat{U}_i (\bar{V}_j)\) only depends on node \(i(j)\) and some of its descendants.
2. Next, based on $\bar{U}_i, \bar{V}_j$, we construct the column and row basis $U_i, V_j^T$, such that $K|_{I_i \times I_j} = U_i B_j V_j^T$, where $U_i, B_j, V_j$ are HSS generators of $K$.

3. Finally, we show the hierarchy of basis matrices. That is, for certain matrices $R, W$, relations in (4.2) hold.

The three steps above will be elaborated in Section 4.3, Section 4.4 and Section 4.5, respectively. The HSS representations for $K$ and $N$ will be summarized in Section 4.6 and Section 4.7, respectively. In Section 4.8, we merge two sets of HSS generators (for $K$ and $N$, resp.) to build up an HSS representation for $A$.

Remark 4.1. In HSS construction for $K$, the interaction between two adjacent intervals at leaf level is set to 0 since this interaction is considered in $N$ instead of $K$.

4.3. Factorization of off-diagonal block $K|_{I_i \times I_j}$. For $i, j \in \mathcal{T}$, by block $(i, j)$, we mean the submatrix $K|_{I_i \times I_j}$. In the context of the HSS construction, we’re only interested in block $(i, S(i))$, where $i \neq \text{root}$. For notational convenience, we set $j = S(i)$ and this notation will be used throughout this section. In addition, unless otherwise stated, we assume $i$ is a left child.

It should be noticed that the FMM procedure yields a partition of $I_i \cup I_j$, leading to a blockwise agglomeration (cf.[12]) of block $(i, j)$. Based on the blockwise agglomeration, we show in this subsection how to construct a factorization for block $(i, j)$ in the form $K|_{I_i \times I_j} = \bar{U}_i \bar{B}_i \bar{V}_j^T$.

4.3.1. Factorization at level $L$ (leaf level). For each leaf node $i$, according to Remark 4.1, we know that block $(i, j)$ is a zero matrix, which yields a trivial factorization

\begin{equation}
K|_{I_i \times I_j} = \bar{U}_i \bar{B}_i \bar{V}_j^T, \quad \text{with} \quad \bar{U}_i = \hat{U}_i, \quad \bar{B}_i = 0, \quad \bar{V}_j = \hat{V}_j.
\end{equation}

Fig. 4.2: interaction between $i$ and $j$ at level $L - 2$, giving rise to a siblinged partition (Def. 4.1) of $I_i$, where the nodes in dotted box are not used in the partition of $I_i \cup I_j$
**4.3.2. Factorization at level** \( l (l \leq L - 1) \). For each node \( i \) at level \( l (l \leq L - 1) \), the FMM procedure gives us a partition of block \((i, j)\) corresponding to the partition of intervals \( I_i, I_j \). See Fig.4.2 and Fig.4.3 for instance. We term this kind of partition for \( I_i \) as **siblinged partition** \( \alpha_i \), whose definition is given below, since this partition for \( I_i \) is determined by considering only the interaction between \( i \) and its sibling \( j \).

**Definition 4.1 (siblinged partition).** For node \( i \in T \), associated with interval \( I_i \), and its sibling \( j \in T \) associated with \( I_j \) in the nested partition described in Section 3.3, the siblinged partition of \( I_i \), denoted by \( \alpha_i \), is a partition produced by the FMM procedure corresponding to the nested partition when we consider only the interaction between \( I_i \) and \( I_j \).

For node \( i \) in Fig.4.2, \( I_i \) is partitioned into 3 subintervals, i.e., \( I_{d_1}, I_{c_3}, I_{c_4} \). We found it convenient to express \( \alpha_i \) as an array by setting \( \alpha_i = [d_1, c_3, c_4] \) corresponding to the partition of \( I_i \) in this case. Analogously, we set \( \alpha_j = [c_5, c_6, d_4] \).

Following the notation introduced in Section 3.1, that is, using \( v_e \) to denote the last component of vector \( v \), we can write for general \( \alpha_i (i \in T) \) that

\[
\alpha_i = [\alpha_1, \alpha_2, \ldots, \alpha_{e-1}, \alpha_e].
\]

We first look at the case \( l = L - 2 \) and then derive a factorization for a general \( l (l \leq L - 1) \).

From (3.3), we know that each submatrix \( M_{s,t} \) in the partition of block \((i, j)\) illustrated in Fig.4.3 has a low-rank factorization. By employing translation relations, we obtain

\[
M_{d_1,d_3} = \hat{U}_{d_1} \hat{B}_{d_1,d_3} \hat{V}_{d_3}^T = \hat{U}_{d_1} \hat{B}_{d_1,d_3} \left[ \begin{array}{c} \hat{V}_{c_5} T_{c_5,d_3} \\ \hat{V}_{c_4} T_{c_4,d_3} \end{array} \right]^T = \hat{U}_{d_1} \hat{B}_{d_1,d_3} \left[ \begin{array}{cc} T_{c_5,d_3} & \hat{V}_{c_5} \end{array} \right] \left[ \begin{array}{c} \hat{V}_{c_5} T_{c_5,d_3} \\ \hat{V}_{c_4} T_{c_4,d_3} \end{array} \right] = \hat{U}_{d_1} \hat{B}_{d_1,d_3} \left[ T_{c_5,d_3} \hat{V}_{c_5} \right] \left[ \begin{array}{c} T_{c_5,d_3} \hat{V}_{c_5} \\ \hat{V}_{c_4} T_{c_4,d_3} \end{array} \right],
\]

\[
M_{d_2,d_4} = \hat{U}_{d_2} \hat{B}_{d_2,d_4} \hat{V}_{d_4}^T = \left[ \begin{array}{c} \hat{U}_{c_3} T_{c_3,d_2} \\ \hat{U}_{c_4} T_{c_4,d_2} \end{array} \right] \hat{B}_{d_2,d_4} \hat{V}_{d_4}^T.
\]
A low-rank factorization for block \((i, j)\) in Fig. 4.3 now reads

\[
K|_{I_i \times I_j} = \begin{bmatrix} \hat{U}_{d_1} & \hat{U}_{c_3} \\ \hat{U}_{c_4} \end{bmatrix} \begin{bmatrix} \hat{B}_{d_1, d_2} & T_{c_5, d_3}^T \\ 0 & \hat{B}_{c_3, c_6} \end{bmatrix} \begin{bmatrix} \hat{B}_{d_1, d_4} & \hat{B}_{c_3, c_6} \\ T_{c_3, d_2} & \hat{B}_{c_4, c_6} \end{bmatrix} \begin{bmatrix} \hat{V}_{c_5} \\ \hat{V}_{c_6} \end{bmatrix}^T,
\]

where

\[
\hat{U}_i = \hat{U}_{\alpha(i)} = \text{diag} \left( \hat{U}_{d_1}, \hat{U}_{c_3}, \hat{U}_{c_4} \right), \quad \hat{V}_j = \hat{V}_{\alpha(i)} = \text{diag} \left( \hat{V}_{c_5}, \hat{V}_{c_6}, \hat{V}_{d_4} \right)
\]
correspond to partitions of \(I_i, I_j\) in Fig. 4.2, respectively.

Generally, for node \(i\) at level \(l (l \leq L - 1)\), the factorization of block \((i, j)\) can be carried out in a similar fashion. That is, according to siblinged partitions \(\alpha(i), \alpha(j)\) of \(I_i, I_j\), respectively, we first derive \(\hat{U}_i, \hat{V}_j\):

\[
\hat{U}_i = \hat{U}_{\alpha(i)} = \text{diag} \left( \hat{U}_{\alpha_1(i)}, \ldots, \hat{U}_{\alpha_x(i)} \right), \quad \hat{V}_j = \hat{V}_{\alpha(i)} = \text{diag} \left( \hat{V}_{\alpha_1(i)}, \ldots, \hat{V}_{\alpha_y(i)} \right).
\]

By applying low-rank factorization (3.3) to each block in the agglomeration of \(K|_{I_i \times I_j}\) and using translation relations in (3.17), \(\hat{B}_i\) can be computed as

\[
\hat{B}_i = G_1^{(i)} + \cdots + G_{L-\nu(i)}^{(i)} \quad \text{with} \quad G_k^{(i)} = \begin{bmatrix} 0 \\ H_k^{(i)} \\ 0 \end{bmatrix},
\]

where \(H_k^{(i)}\) is an \(r(k + 1)\)-by-\(r(k + 1)\) matrix defined by

\[
H_k^{(i)} = \begin{bmatrix} \hat{B}_{\rho, \xi_{(i)}}^T T_{\nu}, \xi & \hat{B}_{\rho, \xi}^T \\ 0 & T_{\sigma}, \hat{B}_{\rho, \xi} \end{bmatrix}.
\]

Here the indices are given by

\[
\rho = c_{e-k}, \quad \rho' = S(\rho), \quad \xi = \sigma_{k+1}, \quad \xi' = S(\xi), \quad \nu = [\nu_1, \ldots, \nu_k] = [\sigma_{k+1}, \ldots, \sigma_k], \quad \sigma = [\sigma_1, \ldots, \sigma_k] = [\sigma_{e-k+1}, \ldots, \sigma_{e-k}] = [\alpha_{e-k+1}, \ldots, \alpha_{e-k}].
\]

If \(i\) is a right child, then

\[
\hat{B}_i = \begin{cases} \hat{B}_j^T, & \text{if } \kappa(x, y) = \kappa(y, x) \\ -\hat{B}_j^T, & \text{if } \kappa(x, y) = -\kappa(y, x) \end{cases}.
\]

In conclusion, for each node \(i\) (\(i \neq \text{root}\)), the factorization for block \((i, j)\) now reads:

\[
K|_{I_i \times I_j} = \hat{U}_i \hat{B}_i \hat{V}_j^T,
\]

where the definition of each factor can be found in (4.6), (4.10), (4.11), (4.12).

In addition to the factorization above, there is an alternative choice for computing the interaction matrix \(\hat{B}_i\), whose value is not exactly same to the one given in (4.11). For example, consider factorizing block \((d_2, d_4)\) in Fig. 4.2. Instead of using local expansion \(\hat{U}_{d_2}\) at \(d_2\) and then expressing \(\hat{U}_{d_2}\) in terms of \(c_3, c_4\) via translation matrices.
(see (4.8)), we factorize block \((d_2, d_4)\) by computing the interaction between \(d_4\) and \(c_3, c_4\) directly, namely,

\[
\tilde{M}_{d_2, d_4} = \begin{bmatrix}
\tilde{U}_{c_2} \tilde{B}_{c_3, d_4} \tilde{V}_d^T \\
\tilde{U}_{c_4} \tilde{B}_{c_3, d_4} \tilde{V}_d^T
\end{bmatrix}.
\]

Note that \(\tilde{M}_{d_2, d_4}\) may not be equal to \(M_{d_2, d_4}\) in (4.8) because the center of expansion is different, but both of them are good approximations to \(A|_{I_{d_2} \times I_{d_4}}\) with relative error satisfying the bound in (2.3). With this kind of expansion technique, let \(\mu = \alpha^{(i)}, \sigma = \alpha^{(j)}\), the factorization of block \((i, j)\) for a nonleaf node \(i\) now reads

\[
(4.14) \quad \tilde{U}_i \tilde{B}_i \tilde{V}_j^T = \hat{U}_i \hat{B}_{\mu, \sigma} \hat{V}_j^T,
\]

where \(\tilde{U}_i, \tilde{V}_j\) are defined in (4.10), and

\[
(4.15) \quad \hat{B}_i = \hat{B}_{\mu, \sigma} = \begin{bmatrix}
\hat{B}_{\mu_1, \sigma_2} & \cdots & \hat{B}_{\mu_1, \sigma_c} \\
\vdots & \ddots & \vdots \\
\hat{B}_{\mu_c, \sigma_2} & \cdots & \hat{B}_{\mu_c, \sigma_c}
\end{bmatrix},
\]

in which for notational convenience, we set (note that this definition is different from the one in (3.7))

\[
\hat{B}_{p, q} = \begin{cases}
0, & \text{if } p, q \text{ are not well-separated,} \\
S_i^{-1} B_0 S_j^{-1}, & \text{if } p, q \text{ are well-separated.}
\end{cases}
\]

In comparison to (4.9), it can be seen that \(\hat{B}_i\) is derived by replacing \(\hat{B}_{p, q} T_{q, u}^T\) (or \(T_{p, v} \hat{B}_{v, q}\)) in \(B_i\) with \(\hat{B}_{p, q}\). That is to say, we do not need to compute translation matrices in determining \(\hat{B}_i\). Due to the convenience that (4.15) brings in notation and implementation, we will use the corresponding factorization in (4.14) when constructing the HSS form of adaptive FMM.

### 4.4. Finding column basis for \(K|_{I_i \times I_j}\)

For each node \(i\), we define the block row \(i\) to be the submatrix \(K|_{I_i \times I_j}\). Similarly, the block column \(i\) is defined to be the submatrix \(K|_{I_j \times I_i}\).

In order to derive HSS generators, we need to find column/row basis for each block row/column \(i\). We show in this subsection how to find these basis matrices \(U_i, V_i\) by considering the interaction between \(I_i\) and \(I_j\).

#### 4.4.1. Basis at level \(L\) (leaf level)

For a leaf node \(i\), we observe that

\[
U_i = \tilde{U}_i = \hat{U}_i, \quad V_i = \tilde{V}_i = \hat{V}_i,
\]

because the interaction between \(I_i\) and \(I_j\) is a far-field interaction according to Remark 4.1.

#### 4.4.2. Basis at level \(l (l \leq L - 1)\)

To find a column basis \(U_i\) for \(K|_{I_i \times I_j}\), the partition of \(I_i\) should be fine enough to represent the interaction between \(I_i\) and \(I_j\). Obviously, \(\tilde{U}_i\) is not a column basis for \(K|_{I_i \times I_j}\) in general because the partition of \(I_i\) only accounts for the interaction between two siblings \(i\) and \(j\).

Note that the interaction between \(I_i\) and \(I_j\), where \(i'\) at level \(lv(i)\) is not a neighbor of \(i\), is a far-field interaction, which can be computed without partitioning \(I_i\). This leads to the following crucial observation when points are uniformly distributed.
Observation 4.1. In finding the column basis for block row \( i \), it suffices to consider the interaction between \( I_i \) and its neighbors.

For this reason, we define the neighbored partition \( \Omega^{(i)} \) for \( I_i \), where the interaction between \( i \) and its two neighbors is taken into account.

Definition 4.2 (neighbored partition). Suppose \( i \in \mathcal{T} \) is a node with two neighbors \( j, q \in \mathcal{T} \), and let \( I_i, I_j, I_q \) be the associated intervals, respectively. The neighbored partition of \( I_i \), denoted by \( \Omega^{(i)} \), is a partition produced by the FMM procedure corresponding to the nested partition when we consider only the interaction between \( I_i \) and \( I_j \cup I_q \).

Same as what we did in (4.7), we can express \( \Omega^{(i)} \) as an array and use \(|\Omega^{(i)}|\) to denote its length.

As an example, the neighbored partition \( \Omega^{(i)} \) of \( I_i \) in Fig.4.2 where \( i, j \) are at level \( L - 2 \) is illustrated in Fig.4.5, which is a superposition of siblinged partition \( \alpha^{(i)} \) in Fig.4.2 and the partition of \( I_i \) in Fig.4.4.

Now the basis matrices \( U_i, V_i \) can be obtained from neighbored partitions, i.e.,

\[
U_i = \hat{U}_{\Omega^{(i)}} = \text{diag} \left( \hat{U}_{\Omega_i^{(1)}}, \ldots, \hat{U}_{\Omega_i^{(e)}} \right), \quad V_i = \hat{V}_{\Omega^{(i)}} = \text{diag} \left( \hat{V}_{\Omega_i^{(1)}}, \ldots, \hat{V}_{\Omega_i^{(e)}} \right).
\]

4.4.3 Factorization of \( K_{|I_i \times I_j|} \) using basis matrices. In this subsection, we deduce a factorization of \( K_{|I_i \times I_j|} \) using basis matrices \( U_i, V_j \) obtained before. In order to characterize the inclusion relations of different partitions for same interval, we introduce the following notations and concepts.
DEFINITION 4.3. For a partition $\omega$ for an interval $I$, if $|\omega| > 1$, we define $\omega|_1$ to be the subpartition of $\omega$ restricted to the left half subinterval of $I$, and $\omega|_2$ the subpartition of $\omega$ restricted to the right half subinterval of $I$. Notationally, we have $\omega = [\omega|_1, \omega|_2]$.

DEFINITION 4.4. For two partitions $\omega, \tilde{\omega}$ of an interval $I$, we define $\omega \preceq \tilde{\omega}$ if one of the following is satisfied:
1. $|\omega| = 1 \leq |\tilde{\omega}|$;
2. $|\omega| > 2, |\tilde{\omega}| > 2$ and $\omega|_j \preceq \tilde{\omega}|_j$ for $j = 1, 2$.

It can be easily checked that the binary relation $\preceq$ is a partial order. $\omega \preceq \tilde{\omega}$ means that partition $\tilde{\omega}$ is more (or equally) refined than partition $\omega$. Obviously, from the definitions of $\alpha^{(i)}$ and $\Omega^{(i)}$, we have $\alpha^{(i)} \preceq \Omega^{(i)}$.

Before moving on, we define generalized translation matrix below.

DEFINITION 4.5 (generalized translation matrix). Let $\omega, \tilde{\omega}$ be two partitions of interval $I$ satisfying $\omega \preceq \tilde{\omega}$. The generalized translation matrix representing the translation from $\tilde{\omega}$ to $\omega$ is defined as

$$T_{\tilde{\omega}, \omega} = \text{diag}(T_{\zeta_k, \omega_k})_{k=1:e}$$

with $e$ denoting the length of $\omega$ and vector $\zeta_k$ given by

$$\zeta_k = \tilde{\omega} \cap \{ \omega_k, \text{ descendants of } \omega_k \}.$$

Since $\alpha^{(i)} \preceq \Omega^{(i)}$, the relation between $\tilde{U}_i$ and $U_i$ can be easily derived:

$$U_i = \hat{U}_{\alpha^{(i)}} = \hat{U}_{\Omega^{(i)}} T_{\Omega^{(i)}, \alpha^{(i)}} = U_i T_{\Omega^{(i)}, \alpha^{(i)}}.$$

Similar relation for $\tilde{V}_i$ and $V_i$ can be obtained by replacing each $U$ with $V$ in (4.19).

With column and row basis being computed, from (4.13), (4.19) we arrive at a final factorization for block $(i, j)$:

$$K|_{i_1 \times j_1} = \hat{U}_i \hat{B}_i \hat{V}_j^T = \hat{U}_{\alpha^{(i)}} \hat{B}_i \hat{V}_{\alpha^{(j)}}^T = \hat{U}_{\Omega^{(i)}} T_{\Omega^{(i)}, \alpha^{(i)}} \hat{B}_i \hat{V}_{\Omega^{(j)}, \alpha^{(j)}}^T \hat{V}_{\Omega^{(j)}} = U_i B_i V_j^T,$$

where

$$B_i = T_{\Omega^{(i)}, \alpha^{(i)}} \hat{B}_i \hat{V}_{\Omega^{(j)}, \alpha^{(j)}}^T.$$

So far, $U, V, B$ generators for the HSS representation of $K$ are given in (4.17), (4.20). Recall that for $U, V$ generators, only those corresponding to leaf nodes are stored explicitly, so $U_i, V_i$ need not be computed if $i$ is not a leaf node.

4.5. Hierarchical basis for $K$. In this subsection, we show that for a parent node $p(p \neq \text{root})$ with left child $c_1$ and right child $c_2$, the corresponding column basis $U_p$ can be derived from $U_{c_1}$ and $U_{c_2}$. More precisely, we construct $R$ and $W$ so that the relations in (4.2) are satisfied (with $i$ replaced by $p$).

From the definition of $\Omega^{(c_1)}(j = 1, 2)$ and Observation 4.1, it is easy to see that $\Omega^{(p)}|_j \preceq \Omega^{(c_j)}$. Thus we have the translation relation

$$\hat{U}_{\Omega^{(p)}|_j} = \hat{U}_{\Omega^{(c_j)}} T_{\Omega^{(c_j)}, \Omega^{(p)}|_j}, \quad j = 1, 2,$$
which implies

\[ U_p = \hat{U}_{\Omega(p)} = \text{diag} \left( \hat{U}_{\Omega(p)[1]}, \hat{U}_{\Omega(p)[2]} \right) \]
\[ = \text{diag} \left( \hat{U}_{\Omega(c_1)}, \hat{U}_{\Omega(c_2)} T_{\Omega^{(c_1)}}, \hat{U}_{\Omega(c_2)} T_{\Omega^{(c_2)}}, \hat{U}_{\Omega(p)} \right) \]
\[ = \begin{bmatrix} \hat{U}_{\Omega(c_1)} \\ \hat{U}_{\Omega(c_2)} \end{bmatrix} \begin{bmatrix} T_{\Omega^{(c_1)}}, \Omega^{(p)} |_1 \\ 0 \\ T_{\Omega^{(c_2)}}, \Omega^{(p)} |_2 \end{bmatrix} \]
\[ = \begin{bmatrix} U_{c_1} \\ U_{c_2} \end{bmatrix} \begin{bmatrix} T_{\Omega^{(c_2)}}, \Omega^{(p)} |_1 \\ 0 \\ T_{\Omega^{(c_2)}}, \Omega^{(p)} |_2 \end{bmatrix}. \]

We therefore derive that

\[ R_i = \begin{cases} 
T_{\Omega^{(i)}, \Omega^{(p)} |_1}, & \text{if } i \text{ is a left child,} \\
0, & \text{if } i \text{ is a right child,} 
\end{cases} \]

(4.21)

where the definition of generalized translation matrices \( T_{\Omega^{(i)}, \Omega^{(p)} |_1}, T_{\Omega^{(i)}, \Omega^{(p)} |_2} \) can be found in (4.18).

Since \( R_i \) and \( W_i \) are only determined by the partition of \( I_i \) and are independent of the points in \( I_i \), we see that \( W_i = R_i \).

### 4.6. HSS representation for \( K \)

So far, all the HSS generators for \( K \) have been derived. We collect them together below.

\[ U_i = \hat{U}_i, \quad V_i = \hat{V}_i, \quad i \text{ is a leaf node,} \]

(4.22)

\[ R_i = \begin{cases} 
T_{\Omega^{(i)}, \Omega^{(p)} |_1}, & \text{if } i \text{ is a left child,} \\
0, & \text{if } i \text{ is a right child,} 
\end{cases} \quad W_i = R_i, \]

\[ B_i = T_{\Omega^{(i)}, \alpha^{(i)}}, \bar{B}_j T_{\Omega^{(j)}, \alpha^{(j)}}. \]

where \( j \) is the sibling of node \( i \), the formula for \( \hat{U}_i, \hat{V}_i \) can be found in (2.11), and the formula for \( \bar{B}_j \) can be found in (4.11), (4.12).

For a nonleaf node \( i \) at level \( l \), the sizes of \( R, W, B \) generators are now in order:

\[ \text{size}(R_i) = \text{size}(W_i) = [2(L - l)r, 2(L - l + 1)r], \]
\[ \text{size}(B_i) = [2(L - l)r, 2(L - l)r]. \]

(4.23)
4.7. HSS representation for $N$ - a banded matrix. The HSS representation for a banded matrix can be easily derived (cf. [17]).

\[ U_i = I, \quad V_i = I, \quad i \text{ is a leaf node}, \]

\[ R_i = \begin{cases} 
[I, 0] & \text{if } i \text{ is a left child at leaf level}, \\
[0, I] & \text{if } i \text{ is a right child at leaf level}, \\
\text{diag}(I, 0), & \text{if } i \text{ is a nonleaf left child}, \\
\text{diag}(0, I), & \text{if } i \text{ is a nonleaf right child}, 
\end{cases} \]

(4.24) \[ W_i = \begin{cases} 
[I, 0] & \text{if } i \text{ is a left child at leaf level}, \\
[0, I] & \text{if } i \text{ is a right child at leaf level}, \\
\text{diag}(I, 0), & \text{if } i \text{ is a nonleaf left child}, \\
\text{diag}(0, I), & \text{if } i \text{ is a nonleaf right child}, 
\end{cases} \]

\[ B_i = \begin{cases} 
M & \text{if } i \text{ is a leaf node}, \\
M & \text{if } i \text{ is a nonleaf left child}, \\
0 & \text{if } i \text{ is a nonleaf right child}, 
\end{cases} \]

where $M$ is a submatrix of $N$ corresponding to near-field interaction between $i$ and its certain neighbor.

4.8. HSS representation for $A$. With two sets of HSS generators for $K$ and $N$ at hand, denoted as $\{D_i(K), U_i(K), \ldots\}$ and $\{D_i(N), U_i(N), \ldots\}$, respectively, it is easy to verify (cf. [18]) that the HSS generators for $A$ are given by:

\[ D_i(A) = D_i(K) + D_i(N), \quad B_i(A) = \text{diag}(B_i(K), B_i(N)), \]

(4.25) \[ U_i(A) = \begin{bmatrix} U_i(K) & U_i(N) \end{bmatrix}, \quad V_i(A) = \begin{bmatrix} V_i(K) & V_i(N) \end{bmatrix}, \\
R_i(A) = \text{diag}(R_i(K), R_i(N)), \quad W_i(A) = \text{diag}(W_i(K), W_i(N)). \]

5. HSS form of adaptative FMM. One of the prominent features of FMM is that the algorithm can be easily made adaptive, which makes FMM extremely suitable for handling non-uniform particle distributions and complex geometries. For non-uniform distribution, we always assume that a complete binary tree is produced after nested partition procedure, namely, each non-root node has a sibling.

We’re glad to see that despite the shape of partition tree in adaptive FMM, the methodology we have employed in previous section to find the HSS form is still valid.

Once the adaptive partition is done and the binary tree is generated, we determine for each leaf node the near neighbors, from which the near-field interaction matrix $N$ can be identified. Since $N$ is mostly banded (i.e., only a small number of blocks away from diagonal are nonzero), the main task is to find the HSS approximation of $A - N$. This can be done by following the three steps described in Section 4.2. We show next how to carry out each step in detail.

To proceed, we first generalize the notion of neighbored partition introduced in Section 4.4.2.

**Definition 5.1** (generalized neighbored partition). For a node $i, i \neq \text{root}$, if $i$ is a child of root, the neighbored partition $\Omega^{(i)} = \alpha^{(i)}$; otherwise, the neighbored
partition $\Omega^{(i)}$ is a superposition of the siblinged partition $\alpha^{(i)}$ and the neighbored partition $\Omega^{(p)}$ restricted to $I_i$ where $p = \mathcal{P}(i)$.

Note that since $\alpha^{(i)}$ is never empty, $\Omega^{(i)}$ will not be empty ($i \neq$ root). For uniform distribution, $\Omega^{(i)}$ can be determined due to Observation 4.1 without recursion and it can be seen that the resulting $\Omega^{(i)}$ is indeed consistent with the definition above. For non-uniform distribution, $\Omega^{(i)}$ must be determined in a top-down fashion as described in the definition.

Now we can complete the task of finding the HSS form for $A - N$ in three steps. In the first step, since we only need to consider the interaction between $i$ and $j$, which is totally local, the siblinged partitions $\alpha^{(i)}, \alpha^{(j)}$ are easy to derive. Then we employ the factorization in (4.14) with $\tilde{B}_i = \hat{B}_{\alpha^{(i)},\alpha^{(j)}}$. In the second step, as siblinged partitions are already computed in first step, we are able to derive neighbored partitions from Definition 5.1. The basis matrices $U_i, V_i$ are then determined by neighbored partition of $I_i$ via (4.17) and $B_i$ can be derived similar to (4.20). In the third step, $R_i, W_i$ can be easily computed via translation matrices as in Section 4.5.

As a result, the HSS approximation to $A - N$ is same to (4.22) by replacing $\bar{B}_i$ with $\tilde{B}_i = \hat{B}_{\alpha^{(i)},\alpha^{(j)}}$.


6.1.1. Construction of HSS. Since all HSS generators can be derived, it follows immediately that the cost for HSS construction only consists of computing each entry for HSS generators in (4.25) or (4.22), which is proportional to the total size of those generators. Assume $n = r2^L$ and there are $r2^{L+1-l}$ points in $\Gamma_i$ for each node $i$ at level $l$ ($l = 1, \ldots, L$). Therefore, according to (4.23), the total cost for HSS construction is

\[
C_{HSS\text{construction}} \leq 2r2^{L-1} + \sum_{l=1}^{2} 2(L - l + 1)^22^{(l-1)} = O(rn).
\]

6.1.2. Matrix vector product. Under the same assumption as above, similar to the complexity analysis at the end of Section 3.6.2, we deduce from (4.23) that, the complexity for computing matrix-vector product is

\[
C_{\text{matvec}} = O \left( \sum_{l=1}^{2} (L - l + 1)^22^{(l-1)} \right) = O(rn).
\]

6.1.3. ULV factorization. It can be seen from (4.23) that the numerical rank of each HSS off-diagonal block at level $l$ is bounded by $2(L - l + 1)r$. With this rank pattern, according to Theorem 6.1 in [18], the complexity for computing the $ULV$ factorization is $O(n)$, and the cost for solution procedure is also $O(n)$.

6.2. Stability. The issue of numerical stability of FMM based on Taylor expansion was only briefly mentioned in [8] and no detailed analysis or complete solution was given. In this section, by estimating the magnitude of each element in the telescoping expansion of $A$ in (3.16), we conclude that, with scaling parameters introduced in Section 2.2, the FMM based on Taylor expansion is numerically stable. Our calculation shows that the maximal elementwise magnitudes in $U, V, T$ matrices in (3.16) are all bounded by $1$ and the relative elementwise magnitude in $B$ is bounded by $e \approx 2.71828$. As a result, the same bounds hold for HSS generators, which would lead
to a backward stable ULV factorization due to results in [16]. The estimates in this section are valid for two-dimensional adaptive FMM as well.

According to HSS generators given in (4.22), we only need to focus on these matrices: $\hat{U}_i, \hat{B}_{i,j}, \hat{T}_{i,p},$ where $j$ and $i$ are well-separated and $p$ is an ancestor of $i$.

**Lemma 6.1.** For any positive integer $r$,

$$\max_{k=1, \ldots, r} \frac{1}{k!} \left( \frac{k}{e} \left( 2\pi r \right)^{\frac{1}{r}} \right)^k \leq 1.$$  

*Proof.* We define $g(k) = \frac{1}{k!} \left( \frac{k}{e} \left( 2\pi r \right)^{\frac{1}{r}} \right)^k$. Let $s = \frac{1}{r} \left( 2\pi r \right)^{\frac{1}{r}}$. Since $g(k+1)/g(k) = s \left( 1 + \frac{1}{k} \right)^k$ and $0 < s < 1$, we find that the monotonicity of $g(k)$ as $k$ increases has at most three possibilities:

1. $g(k)$ decreases monotonically;
2. $g(k)$ increases monotonically;
3. $g(k)$ first decreases then increases.

Therefore, we conclude that

$$g(k) \leq \max \{g(1), g(r)\} = \max \{s, \frac{(r/e)^r \sqrt{2\pi r}}{r!} \} \leq 1,$$

where $g(1) = s < 1$ and $(r/e)^r \sqrt{2\pi r} \leq r!$ for all $r \geq 1$. $\Box$

**Lemma 6.2.** Let $0 < \tau < 0.8$ and $r$ be any positive integer. Then

$$\max_{k=1, \ldots, r} \tau^k k! \left( \frac{k}{e} \left( 2\pi r \right)^{\frac{1}{r}} \right)^{-k} < 3\tau.$$  

*Proof.* This is proved in the same way as Lemma 6.1. Note that with $g(k)$ defined in the proof of Lemma 6.1, the quantity in this lemma can be written as $h(k) := \tau^k/g(k)$. We have

$$h(k+1)/h(k) = \tau g(k)/g(k+1) = \tau s^{-1} \left( 1 + \frac{1}{k} \right)^{-k} < 0.8e \left( 1 + \frac{1}{k} \right)^{-k} < 1, \quad \forall k \geq 2.$$  

This implies that $h(k)$ is decreasing if $k \geq 2$, thus

$$\max_{1 \leq k \leq r} h(k) \leq \max \{h(1), h(2)\} < 3\tau,$$

and the proof is complete. $\Box$

**Lemma 6.3.** Let $\tau > 0$ and $k$ be any positive integer, then

$$\max_{s \in (0, \tau]} s^l (\tau - s)^{k-l} = \tau^k \left( \frac{l}{k} \right)^l \left( \frac{k-l}{k} \right)^{k-l}, \quad l = 1, \ldots, k-1.$$  

*Proof.* Let $\phi(s) = s^l (\tau - s)^{k-l}$. Consider $\psi(s) = \log \phi(s)$. Setting $\psi'(s) = 0$, we obtain the only critical point $s_0 = \tau l/k$. It is easy to check that $\phi(s_0)$ is the maximum, thus $\phi(s) \leq \phi(s_0)$, where $\phi(s_0)$ equals the right-hand side in (6.2). $\Box$
6.2.1. Estimate for $\hat{U}_i$. According to (2.11), we need to estimate $|\eta_{i,k}f_k(x-a_i)|$ for $k = 0, 1, \ldots, r - 1$, where $x$ is a point in cluster $i$. Obviously, if $k = 0$, the quantity equals 1. For $1 \leq k \leq r - 1$, we deduce from Lemma 6.1 that

$$ (6.3) \quad |\eta_{i,k}f_k(x-a_i)| = \left(\frac{k}{e^{(2\pi r)\frac{k}{|a_i-a_i|} - 1}}\right)^k \frac{|x-a_i|^k}{k!} \leq 1, $$

where we have used the fact that $|x-a_i| \leq \delta_i$. Thus we conclude that the magnitude of each element in $\hat{U}_i$ is bounded by 1. As a result, the maximal elementwise magnitude for $U, V$ generators of $A$ is 1.

6.2.2. Estimate for $\hat{B}_{i,j}$. According to (2.11) or (2.9), we need to estimate $|c_\kappa \eta_{i,l}^{-1} \eta_{j,k-1}^{-1}|$ for $k = 0, 1, \ldots, r - 1$, $l = 0, 1, \ldots, k$, where cluster $i$ and cluster $j$ are well-separated.

For such clusters $i$ and $j$, an immediate yet important estimate from the definition of being well-separated in (2.4) can be easily obtained below

$$ (6.4) \quad \frac{\delta_i}{|a_i-a_j|} \leq \tau \frac{\delta_i}{\delta_i + \delta_j}, \quad \frac{\delta_j}{|a_i-a_j|} \leq \tau \frac{\delta_j}{\delta_i + \delta_j}. $$

According to the definition of $c_\kappa$ in (2.2) and the formula for scaling parameters in (3.4), we see that if $k = 0$, the quantity we need to estimate is equal to $1/|a_i-a_j|$; if $k \geq 1$, we do a simple case analysis on $l$.

1. $l = 0$.

$$ |c_\kappa \eta_{i,l}^{-1} \eta_{j,k-1}^{-1}| = \frac{1}{|a_i-a_j|} k! \left(\frac{k}{e^{(2\pi r)\frac{k}{|a_i-a_i|}} - 1}\right)^k \left(\frac{\delta_i}{|a_i-a_j|}\right)^k \left(\frac{\delta_j}{\delta_i + \delta_j}\right)^k \leq \frac{1}{|a_i-a_j|} k! \left(\frac{k}{e^{(2\pi r)\frac{k}{|a_i-a_i|}} - 1}\right)^k \left(\frac{\tau \delta_i}{\delta_i + \delta_j}\right)^k \leq 3 \tau |\kappa(a_i,a_j)|. $$

2. $1 \leq l \leq k - 1$.

$$ |c_\kappa \eta_{i,l}^{-1} \eta_{j,k-1}^{-1}| = \frac{1}{|a_i-a_j|} k! \left(\frac{k}{e^{(2\pi r)\frac{k}{|a_i-a_i|}} - 1}\right)^k l^{-l}(k-l)^{-(k-l)} \left(\frac{\delta_i}{|a_i-a_j|}\right)^l \left(\frac{\delta_j}{\delta_i + \delta_j}\right)^{k-l} \leq \frac{1}{|a_i-a_j|} k! \left(\frac{k}{e^{(2\pi r)\frac{k}{|a_i-a_i|}} - 1}\right)^k l^{-l}(k-l)^{-(k-l)} \frac{\tau \delta_i}{\delta_i + \delta_j} \left(\frac{\delta_j}{\delta_i + \delta_j}\right)^{k-l} \leq 3 \tau |\kappa(a_i,a_j)|. $$

3. $l = k$. Same as case 1.

Hence we conclude that the magnitude for each element in $\hat{B}_{i,j}$ is bounded by $e|\kappa(a_i,a_j)|$ with $\kappa$ given in (3.1).

6.2.3. Estimate for translation matrix $T_{i,p}$. According to (3.12), we need to estimate $|\eta_{p,k} \eta_{i,l}^{-1} f_{k-l}(a_i-a_p)|$, where $p$ is an arbitrary ancestor of $i$. 

If \( k = 0 \), the quantity is equal to 1. For \( k \geq 1 \), we proceed to do case analysis on \( l \) as in the estimate for \( \hat{B}_{i,j} \). Before we do the analysis, an observation below regarding the partition would be helpful.

\[
|a_i - a_p| \leq \delta_p - \delta_i.
\]

This inequality holds true because the upper bound in (6.5) measures the distance from the center of box \( p \) (i.e., \( a_p \)) to the center of its farthest descendant box at the same level (hence with same size) as box \( i \).

1. \( l = 0 \). \( |\eta_{p,k} f_k(a_i - a_p)| \leq 1 \) from (6.3).
2. \( 1 \leq l \leq k - 1 \).

\[
|\eta_{p,k}\eta_{i,l}\eta_{k-l}(a_i - a_p)| = k^k l^{-l} \left( \frac{1}{e} (2\pi r)^{\frac{3}{2}} \right)^{k-l} \left( \frac{\delta_i}{\delta_p} \right)^l \left( \frac{1}{k} \right)^{k-l} \frac{|a_i - a_p|^{k-l}}{(k-l)!}
\]

\[
\leq k^k l^{-l} \frac{1}{(k-l)!} \left( \frac{1}{e} (2\pi r)^{\frac{3}{2}} \right)^{k-l} \left( \frac{\delta_i}{\delta_p} \right)^l \left( 1 - \frac{\delta_i}{\delta_p} \right)^{k-l}
\]

Lemma 6.3

\[
= \frac{1}{(k-l)!} \left( \frac{k-l}{e} (2\pi r)^{\frac{3}{2}} \right)^{k-l} \frac{1}{e} \left( \frac{k-l}{k} \right) \frac{1}{1} \leq 1.
\]

3. \( l = k \). \( |\eta_{p,k}\eta_{i,k}^{-1}| = \frac{\delta_i}{\delta_p} \leq 1 \).

Due to the above elementwise estimate on \( T_{i,p} \), we conclude that the magnitude for each element in \( R, W \) generators are bounded by 1.

**Remark 6.1.** The same set of scaling parameters is also valid for kernels like \( 1/(x - y)^2 \), \( \log |x - y| \), etc., and the estimates above for \( \hat{U}, T_{i,p} \) do not change, while the estimate for \( \hat{B}_{i,j} \) only changes by a small constant depending on the kernel function. It can be seen that the stability analysis in this section holds true for both 1D and 2D FMM.

**6.3. Structures in HSS generators.** With an HSS representation for \( A \), we can perform \( ULV \) factorization ([5]). We notice that the structure of the HSS generators can be used to speed up the computation in \( ULV \) process. Firstly it should be noted that \( U_i, V_i, R_i, W_i \) are mostly sparse, thus instead of treating those matrices as dense matrices when performing multiplication, we only multiply nonzero blocks in those matrices. Secondly, since \( U_i \) is composed of two block diagonal matrices, it suffices to compute \( QR \) factorization for certain nonzero blocks in \( U_i \) instead of the whole matrix itself. For example, in view of \( U_7 \) (see Fig.6.1) constructed in Section 4, we notice that its first and last block rows are of full rank due to the identity matrix and other block rows can be compressed. As a result, it suffices to compute \( QR \) factorizations for the nonzero blocks in the second block row and the third block row. Since each \( R_i \) is a block diagonal matrix, it is easily seen that the structure of each \( U \) generator is preserved after each level of compression and merging.

**7. Numerical results.** We will assume that matrix \( A \) in (1.1) is a square matrix and \( \tilde{A} \) is the HSS representation of \( A \) constructed using our method. The following notation is used in this section.
Fig. 6.1: Nonzero pattern of $U_7$

Table 7.1: HSS approximation error to kernel matrix

<table>
<thead>
<tr>
<th>$r$</th>
<th>$L$</th>
<th>rel error</th>
<th>abs error</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>2.55E-02</td>
<td>1.25E+01</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>6.49E-04</td>
<td>3.20E-01</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>1.65E-05</td>
<td>8.14E-03</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>4.22E-07</td>
<td>2.08E-04</td>
</tr>
<tr>
<td>25</td>
<td>10</td>
<td>1.07E-08</td>
<td>5.29E-06</td>
</tr>
</tbody>
</table>

7.1. Approximation accuracy. Since the $\hat{A}$ is the same matrix produced by FMM in (3.16), the approximation error satisfies the estimate given in (2.3).

$$\kappa(x,y) = \frac{1}{x-y}, \quad \text{if } x \neq y, \text{ and } \kappa(x,x) = 1.$$ 

where $n = 12800$ and points $x = y$ are uniformly distributed in $[0,1]$.

$$A = (\kappa(x_i,y_j))_{n \times n}, \quad \hat{A} = \text{HSS approximation for } A$$

See Table 7.1, where $\text{rel error} = \max |a_{i,j} - \hat{a}_{i,j}|/|a_{i,j}|$, $\text{abs error} = \max |a_{i,j} - \hat{a}_{i,j}|$.

7.2. Stability. We conduct numerical experiment for (unnecessarily) large $r$ to show the stability of our algorithm.
Table 7.2: Stability test for uniform distribution
\[
\begin{array}{|c|c|c|c|}
\hline
r & L & \frac{\|x - \hat{x}\|}{\|x\|} & \frac{\|b - A\hat{x}\|}{\|b\|} \\
\hline
20 & 10 & 1.07E-07 & 2.99E-09 \\
50 & 10 & 3.11E-13 & 2.27E-14 \\
100 & 10 & 8.07E-13 & 4.49E-14 \\
150 & 10 & 1.02E-12 & 7.92E-14 \\
\hline
\end{array}
\]

Table 7.3: Stability test for nonuniform distribution
\[
\begin{array}{|c|c|c|c|}
\hline
r & L & \frac{\|x - \hat{x}\|}{\|x\|} & \frac{\|b - A\hat{x}\|}{\|b\|} \\
\hline
20 & 14 & 5.24E-07 & 8.42E-11 \\
50 & 14 & 1.55E-11 & 3.60E-15 \\
100 & 14 & 2.54E-11 & 3.46E-15 \\
150 & 14 & 2.71E-11 & 3.78E-15 \\
\hline
\end{array}
\]

7.2.1. Uniform distribution. Same point set as before.
\[
Ax = b, \\
\hat{A}\hat{x} = b.
\]

The point set is given by Fig.7.1. See Table 7.2 for test results.

Fig. 7.1: a nonuniform distribution in $[0, 1]$, $n = 12800$ points

7.2.2. Nonuniform distribution. See Table 7.3

7.3. Complexity and accuracy for ULV factorization.

7.3.1. ULV - Uniform distribution. Points are uniformly distributed on the unit circle. Original system: $Ax = b$, Approximation: $\hat{A}\hat{x} = b$.

See Table 7.4 for flop counts of ULV factorization and Table 7.5 for corresponding accuracy.

7.3.2. ULV - Nonuniform distribution.
\[
x(t) = \sin(2\pi t) + \cos^4(4\pi t), \quad y(t) = \cos^3(2\pi t)
\]

See Table 7.6 for flop counts of ULV factorization and Table 7.7 for corresponding accuracy.
Table 7.4: ULV flop counts - uniform distribution

<table>
<thead>
<tr>
<th># points</th>
<th>QR2HSS</th>
<th>FMM2HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>ULV sol</td>
</tr>
<tr>
<td>1600</td>
<td>2.68E+07</td>
<td>7.65E+05</td>
</tr>
<tr>
<td>3200</td>
<td>5.48E+07</td>
<td>1.48E+06</td>
</tr>
<tr>
<td>6400</td>
<td>1.12E+08</td>
<td>2.98E+06</td>
</tr>
<tr>
<td>12800</td>
<td>2.27E+08</td>
<td>5.91E+06</td>
</tr>
<tr>
<td>25600</td>
<td>4.58E+08</td>
<td>1.17E+07</td>
</tr>
</tbody>
</table>

Table 7.5: ULV error - uniform distribution

<table>
<thead>
<tr>
<th># points</th>
<th>QR2HSS</th>
<th>FMM2HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1600</td>
<td>7.68E-05</td>
<td>1.38E-06</td>
</tr>
<tr>
<td>3200</td>
<td>2.84E-04</td>
<td>2.23E-06</td>
</tr>
<tr>
<td>6400</td>
<td>2.87E-04</td>
<td>2.21E-06</td>
</tr>
<tr>
<td>12800</td>
<td>1.00E-03</td>
<td>3.20E-06</td>
</tr>
<tr>
<td>25600</td>
<td>1.94E-03</td>
<td>4.51E-06</td>
</tr>
</tbody>
</table>

7.4. Accuracy and Stability in 2D.

\[
\kappa(x, y) = \frac{1}{|x - y|} \quad \text{if } x \neq y, \text{ and } \kappa(x, x) = 1.
\]

\(x = y\) are nonuniformly distributed in \([0, 1] \times [0, 1]\).

See Table 7.8 for approximation error.

8. Conclusions. This paper presents an approach to rapidly construct the HSS representation from an FMM matrix for 1D problem. No algebraic compression (like SVD) is used and all HSS generators can be derived once the information about the partition of the curve is given. It is shown that the complexity for both HSS construction and solving the linear system in (1.2) is \(O(n)\). The method is stable based on stability analysis given in previous section. This method provides a way to solve linear systems associated with certain kernel functions evaluated on planar curves in optimal complexity and also a way to derive an HSS representation directly without algebraic compressions. Obviously, the approximation based on Taylor expansion can be substituted by other approximations, such as, polynomial interpolation([6]), SVD([20]), numerical integration([1]), etc., which implies that this method can also be applied even if the analytic property of the kernel function is not available.

The limitation of this method is obviously seen: since the off-diagonal blocks in a FMM matrix for 2D or 3D problems no longer have the low-rank property, the method is not efficient, i.e., linear complexity can not be achieved, if applied directly to problems in higher dimensions. It will be our future work to resolve this issue.

REFERENCES

Fig. 7.2: a nonuniform distribution on boundary of a stingray

Table 7.6: ULV flop counts - nonuniform distribution

<table>
<thead>
<tr>
<th># points</th>
<th>QR2HSS</th>
<th>FMM2HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ULV</td>
<td>sol</td>
</tr>
<tr>
<td>1600</td>
<td>3.88E+08</td>
<td>4.43E+06</td>
</tr>
<tr>
<td>3200</td>
<td>1.10E+09</td>
<td>7.82E+06</td>
</tr>
<tr>
<td>6400</td>
<td>3.32E+09</td>
<td>1.61E+07</td>
</tr>
<tr>
<td>12800</td>
<td>9.60E+09</td>
<td>3.31E+07</td>
</tr>
<tr>
<td>25600</td>
<td>2.74E+10</td>
<td>6.87E+07</td>
</tr>
</tbody>
</table>

Table 7.7: ULV error - nonuniform distribution

<table>
<thead>
<tr>
<th># points</th>
<th>QR2HSS</th>
<th>FMM2HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1600</td>
<td>6.93E-05</td>
<td>1.82E-10</td>
</tr>
<tr>
<td>3200</td>
<td>8.62E-04</td>
<td>2.43E-10</td>
</tr>
<tr>
<td>6400</td>
<td>1.11E-02</td>
<td>7.30E-10</td>
</tr>
<tr>
<td>12800</td>
<td>5.81E-02</td>
<td>1.36E-09</td>
</tr>
<tr>
<td>25600</td>
<td>1.47E-01</td>
<td>1.53E-09</td>
</tr>
</tbody>
</table>

Fig. 7.3: a nonuniform distribution in unit square, $n = 2500$
Table 7.8: HSS approximation error to 2D kernel matrix

<table>
<thead>
<tr>
<th>r</th>
<th>L</th>
<th>rel error</th>
<th>abs error</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>10</td>
<td>6.76E-05</td>
<td>9.79E-04</td>
</tr>
<tr>
<td>30</td>
<td>10</td>
<td>5.56E-07</td>
<td>7.80E-06</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>4.57E-09</td>
<td>6.21E-08</td>
</tr>
<tr>
<td>50</td>
<td>10</td>
<td>3.76E-11</td>
<td>4.94E-10</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>2.13E-15</td>
<td>1.60E-14</td>
</tr>
<tr>
<td>150</td>
<td>10</td>
<td>2.21E-15</td>
<td>1.61E-14</td>
</tr>
</tbody>
</table>


