

FMM CMSC 878R/AMSC 698R

Lecture 20

Nail Gumerov & Ramani Duraiswami

Outline

- Adaptive Multilevel FMM
 - Brief Historical Review
 - Statement of the Problem for Adaptive MLFMM
 - Data Structure
 - Determination of the Target Box Set
 - Building of D-tree
 - Building of C-forest
 - AMLFMM Algorithm
 - Upward Pass
 - Downward Pass
 - Final Summation

Summary of requirements for functions that can be used in FMM

- We have two sets of points:

$$\begin{aligned} X &= \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}, \quad \mathbf{x}_i \in \mathbb{R}^d, \quad i = 1, \dots, N, \\ Y &= \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_M\}, \quad \mathbf{y}_j \in \mathbb{R}^d, \quad j = 1, \dots, M. \end{aligned}$$

- We have functions (potentials):

$$\Phi(\mathbf{x}_i, \mathbf{y}) : \mathbb{R}^d \rightarrow \mathbb{R}, \quad \mathbf{y} \in \mathbb{R}^d, \quad i = 1, \dots, N.$$

- These functions can be factorized as (local expansion):

$$\Phi(\mathbf{x}_i, \mathbf{y}) = \mathbf{A}(\mathbf{x}_i, \mathbf{x}_*) \circ \mathbf{R}(\mathbf{y} - \mathbf{x}_*), \quad |\mathbf{y} - \mathbf{x}_*| < r < |\mathbf{x}_i - \mathbf{x}_*|, \quad i = 1, \dots, N$$

- These functions can be factorized as (far field expansion):

$$\Phi(\mathbf{x}_i, \mathbf{y}) = \mathbf{B}(\mathbf{x}_i, \mathbf{x}_*) \circ \mathbf{S}(\mathbf{x} - \mathbf{x}_*), \quad |\mathbf{y} - \mathbf{x}_*| > R > |\mathbf{x}_i - \mathbf{x}_*|, \quad i = 1, \dots, N$$

- The product is distributive operation with respect to addition

$$(u_1 \mathbf{A}_1 + u_2 \mathbf{A}_2) \circ \mathbf{F} = u_1 \mathbf{A}_1 \circ \mathbf{F} + u_2 \mathbf{A}_2 \circ \mathbf{F}, \quad \mathbf{F} = \mathbf{S}, \mathbf{R}$$

Summary of requirements for functions that can be used in FMM (2)

- R -expansion coefficients can be $R|R$ -translated:

$$|\mathbf{x} - \mathbf{x}_{*2}| < |\mathbf{x}_i - \mathbf{x}_{*1}| - |\mathbf{x}_{*1} - \mathbf{x}_{*2}| :$$

$$\mathbf{A}(\mathbf{x}_i, \mathbf{x}_{*2}) = (\mathbf{R}|\mathbf{R})(\mathbf{x}_{*2} - \mathbf{x}_{*1})\mathbf{A}(\mathbf{x}_i, \mathbf{x}_{*1})$$

- S -expansion coefficients can be $S|S$ -translated:

$$|\mathbf{x} - \mathbf{x}_{*2}| > |\mathbf{x}_{*1} - \mathbf{x}_{*2}| + |\mathbf{x}_i - \mathbf{x}_{*1}|,$$

$$\mathbf{B}(\mathbf{x}_i, \mathbf{x}_{*2}) = (\mathbf{S}|\mathbf{S})(\mathbf{x}_{*2} - \mathbf{x}_{*1})\mathbf{B}(\mathbf{x}_i, \mathbf{x}_{*1})$$

- S -expansion coefficients can be $S|R$ -translated (converted to R -expansion coefficients)

$$|\mathbf{x} - \mathbf{x}_{*2}| < |\mathbf{x}_{*1} - \mathbf{x}_{*2}| + |\mathbf{x}_i - \mathbf{x}_{*1}|,$$

$$\mathbf{A}(\mathbf{x}_i, \mathbf{x}_{*2}) = (\mathbf{S}|\mathbf{R})(\mathbf{x}_{*2} - \mathbf{x}_{*1})\mathbf{B}(\mathbf{x}_i, \mathbf{x}_{*1})$$

- And we are looking for sums:

$$v_j = \sum_{i=1}^N u_i \Phi(\mathbf{y}_j, \mathbf{x}_i), \quad j = 1, \dots, M.$$

- Some generalization are possible, say instead of $\Phi(\mathbf{y}_j, \mathbf{x}_i)$ we can consider $\Phi_i(\mathbf{y}_j)$, etc.

Both Types of Function Representations Considered satisfy these requirements

When using the diagonal forms of translation operators, we understand that representing vectors **A** and **B** are samples of the signature function at the quadrature nodes and **S**- and **R**- basis functions are samples of the singular and regular kernel at the same nodes. Operation \circ means weighted dot product (with the quadrature weights).

Adaptive Multilevel FMM

Some History:

- J. Carrier, L. Greengard & V. Rokhlin,
A fast adaptive multipole algorithm for particle simulations. SIAM J. Sci. Statist. Comput. 9, 669, 1988.
- L. Van Dommelen & E.A. Rundensteiner,
Fast, adaptive summation of point sources in the two-dimensional Poisson equation. J. Comp. Physics, 83, 126-147, 1989.
- H. Cheng, L. Greengard & V. Rokhlin, A fast adaptive multipole algorithm in three dimensions.

Here we present a new version and interpretation
of AMLFMM

What Does Mean “Adaptive”?

- The number of operations depends on data sets.
- Skipping boxes that do not contain sources and evaluation points in FMM is the first step of adaptation (one should not go through all boxes).
- Fully adaptive method should be able to design the computational process depending on 2 data sets: Source and Evaluation Points.
- The regular multilevel FMM uses pyramid data structure, while for the adaptive method we use tree (and/or forest) data structures.

General Idea

1). Potential at *any* level of hierarchical space subdivision can be computed as

$$\Phi(\mathbf{y}_j) = \sum_{\mathbf{x}_i \in E_2(n,l)} \Phi_i(\mathbf{y}_j) + \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{y}_j - \mathbf{x}_c^{(n,l)}), \quad \mathbf{y}_j \in E_1(n,l).$$

Contribution of neighborhood

Contribution of all other sources

2). Therefore for each point \mathbf{y}_j we can determine a level l_j of the space subdivision at which the number of sources in the neighborhood does not exceed some prescribed number q and the cost of direct computation of the field of sources in the neighborhood is $O(q)$.

To determine the second term in the right hand side we need to compute coefficients $\mathbf{D}^{(n,l)}$ only for level $l = l_j \leq L$ that is specific for each point \mathbf{y}_j and is determined by the number of points of the set \mathbb{X} in the neighborhood of that point. The set of box numbers and levels (n_j, l_j) , $j = 1, \dots, M$, for which we need to evaluate \mathbf{y}_j will provide a set of *target* boxes for which we need to obtain $\mathbf{D}^{(n_j, l_j)}$.

General Idea (2)

- 3). To compute coefficients $\mathbf{D}^{(n_j, l_j)}$ we will construct a tree which will show what other coefficients $\mathbf{D}^{(n, l)}$, $l < l_j$ are needed to obtain the value of $\mathbf{D}^{(n_j, l_j)}$. We will call such tree as *D-tree*. Since $\mathbf{D}^{(n_j, l_j)}$ depends not only on $\mathbf{D}^{(n, l)}$, but also on $\tilde{\mathbf{D}}^{(n, l)}$, we will also construct a *$\tilde{\mathbf{D}}$ -tree*, which shows the entire set of coefficients $\tilde{\mathbf{D}}^{(n, l)}$ that is needed for computation of $\mathbf{D}^{(n_j, l_j)}$.
- 4). More detailed look at the formulae for computation of coefficients show that the *$\tilde{\mathbf{D}}$ -tree* is the same as the *D-tree*.
- 5). Further, we will build a *C-forest* (which may be a single tree or a union of several trees), that will contain the numbers and levels of boxes, (n, l) , for which we need to have $\mathbf{C}^{(n, l)}$ coefficients to obtain $\tilde{\mathbf{D}}^{(n, l)}$ and further $\mathbf{D}^{(n, l)}$ belonging to the *D-tree* and *$\tilde{\mathbf{D}}$ -tree*.

General Idea (3)

The described scheme is:

- Adaptive, since it provides the level of the box specific for each point \mathbf{y}_j , which depends on the both sets \mathbb{X} and \mathbb{Y} .
- Presumably more economic than the regular FMM, since it is seeking for computation of the coefficients that are necessary to achieve the objective and does not need computation of unnecessary coefficients.
- Presumably more accurate than the regular FMM, since it avoids unnecessary translations.

Setting Data Structure. Step 0.

Generate the same data structure as for regular MLFMM.

Setting Data Structure. Step 1.

Determine the Set of Target Boxes.

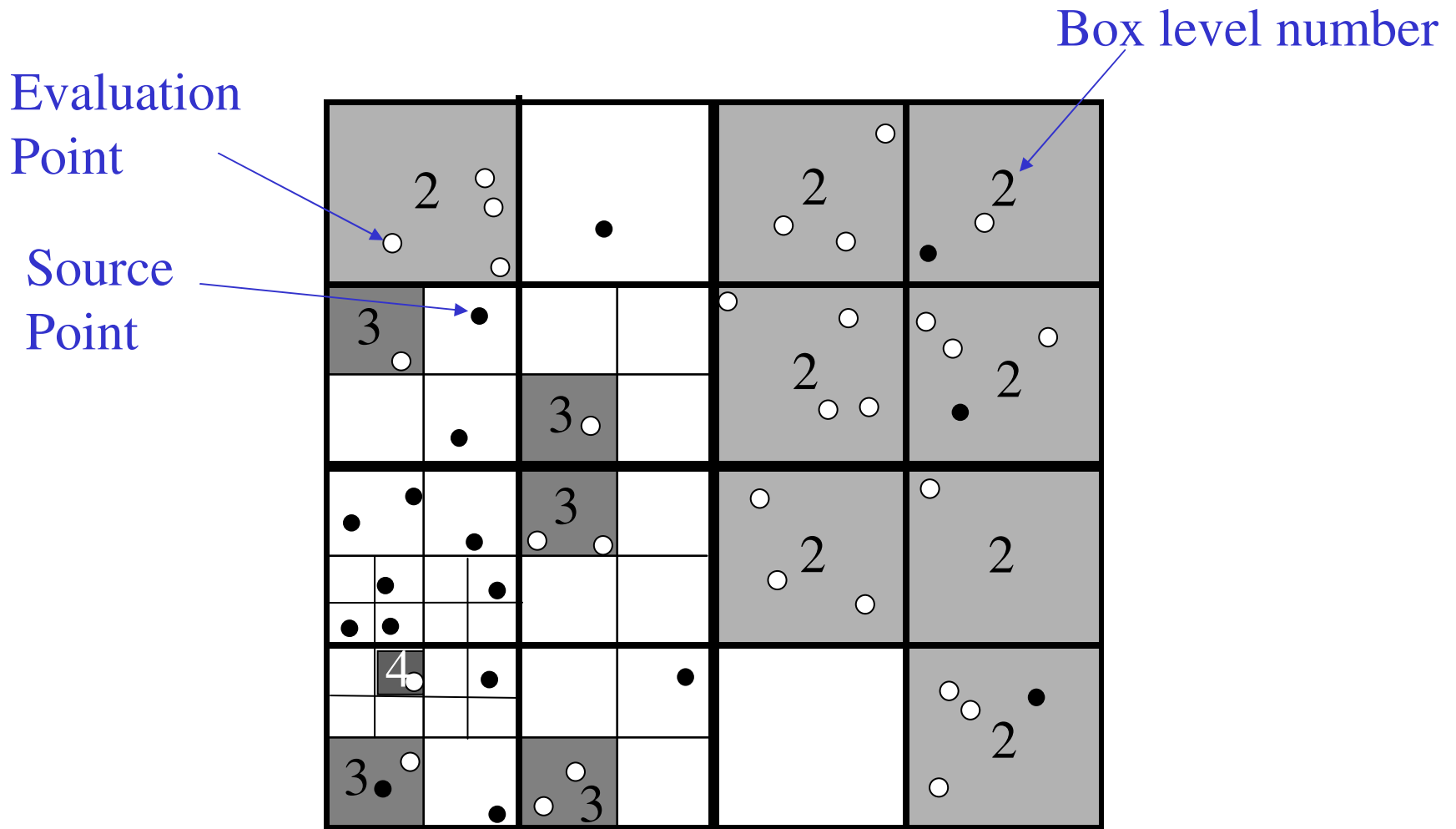
The first step of the present adaptive algorithm includes determination of all target boxes and an appropriate number of levels of the hierarchical space subdivision. This depends on the prescribed number q that is the maximum allowed number of sources which can be summed directly to evaluate the potential due to these sources. All these tasks can be performed within one step procedure that algorithmically can be described as follows. In our notation we denote the set of target boxes as \mathbb{T} and the number of levels in the hierarchical space subdivision as L . The set of target boxes at level l we will denote as \mathbb{T}_l . So

$$\mathbb{T} = \mathbb{T}_2 \cup \dots \cup \mathbb{T}_L.$$

- 1.** Set the level of consideration $l = 2$ and the remaining set of the evaluation points $\mathbb{Y}_{rem} = \mathbb{Y}$.
- 2.** For each box (n, l) , that contains a point $\mathbf{y}_j \in \mathbb{Y}_{rem}$ find the number N_n of points $\mathbf{x}_i \in E_2(n, l)$.
- 3.** If $N_n \leq q$ refer the box (n, l) to the set of target boxes \mathbb{T}_l and exclude all $\mathbf{y}_j \in E_1(n, l)$ from the set \mathbb{Y}_{rem} .
- 4.** If $\mathbb{Y}_{rem} = \emptyset$, set the maximum number of levels $L = l$ and stop the procedure, else increase the level, $l = l + 1$, and repeat steps 2-4 of this algorithm.

Example of Determination of Target Boxes

In this example target boxes are shaded. Each of them contains not more than 3 source points in the neighborhood.



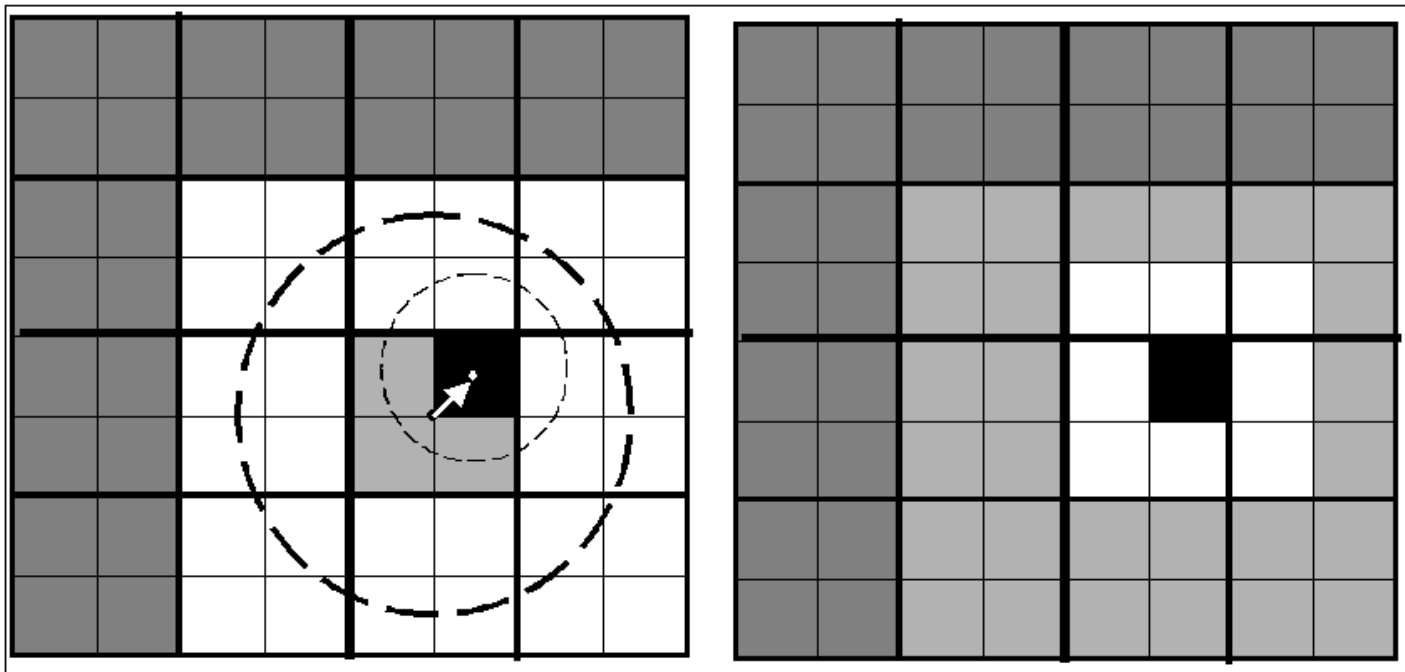
Setting Data Structure. Step 2.

Build D-tree.

Consider Step 2 of the MLFMM Downward Pass.

$$\Phi_{n,l}^{(3)}(\mathbf{x}) = \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{x} - \mathbf{x}_c^{(n,l)}),$$

$$\mathbf{D}^{(n,l)} = \tilde{\mathbf{D}}^{(n,l)} + (\mathbf{R}|\mathbf{R})\left(\mathbf{x}_c^{(n',l-1)} - \mathbf{x}_c^{(n,l)}\right)\mathbf{D}^{(n',l-1)}, \quad n' = \text{Parent}(n).$$



To compute D for a box we need D for the *Parent* of this box

Setting Data Structure. Step 2 (2).

Build D-tree.

Algorithm

Let us denote the set of the boxes in the D -tree as \mathbb{D} . We also denote through \mathbb{D}_l sets of boxes in the D -tree at level $l \geq 2$, so $\mathbb{D} = \mathbb{D}_2 \cup \dots \cup \mathbb{D}_L$. The tree can be generated from the target box set by applying function *Parent* until the level $l = 2$ reached. In other words if box $(n, l) \in \mathbb{D}$, $l > 2$, then box $(Parent(n), l - 1) \in \mathbb{D}$. The algorithm here is straightforward:

- 1.** Refer all target boxes at the finest level of space subdivision L to set \mathbb{D}_L and set $l = L$.
- 2.** If $l = 2$ stop the procedure, else set $l = l - 1$ and perform steps 2-4 in a loop.
- 3.** Determine $(Parent(n), l)$ for all boxes from \mathbb{D}_{l+1} and refer them to set \mathbb{D}_l .
- 4.** Make a union of \mathbb{D}_l and target boxes \mathbb{T}_l and return the result back to \mathbb{D}_l .

Note that in the D -tree all target boxes represent *leaves* of the tree (in other words end points for each branch of the tree). So \mathbb{T} is the set of all leaves in the D -tree and \mathbb{T}_l is the set of leaves of this tree at level $l = 2, \dots, L$.

Setting Data Structure. Step 2 (3).

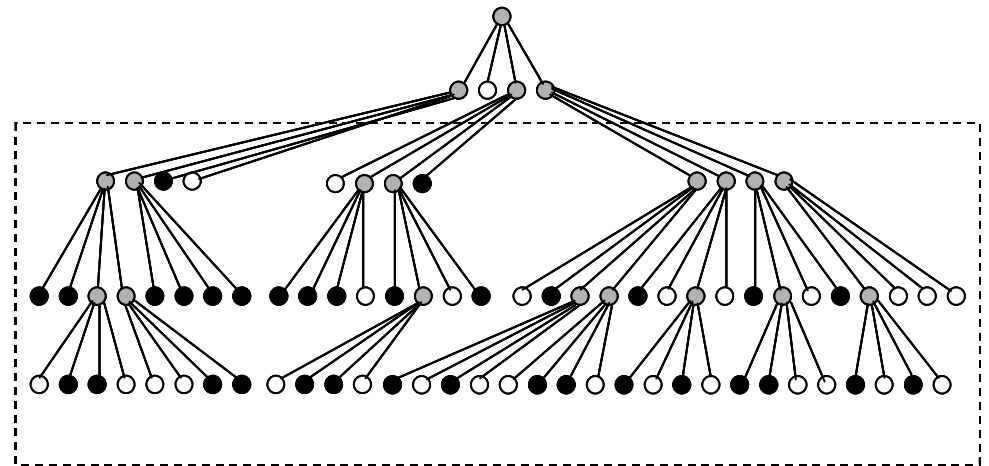
Build D-tree.

Example

Target Boxes



D-tree



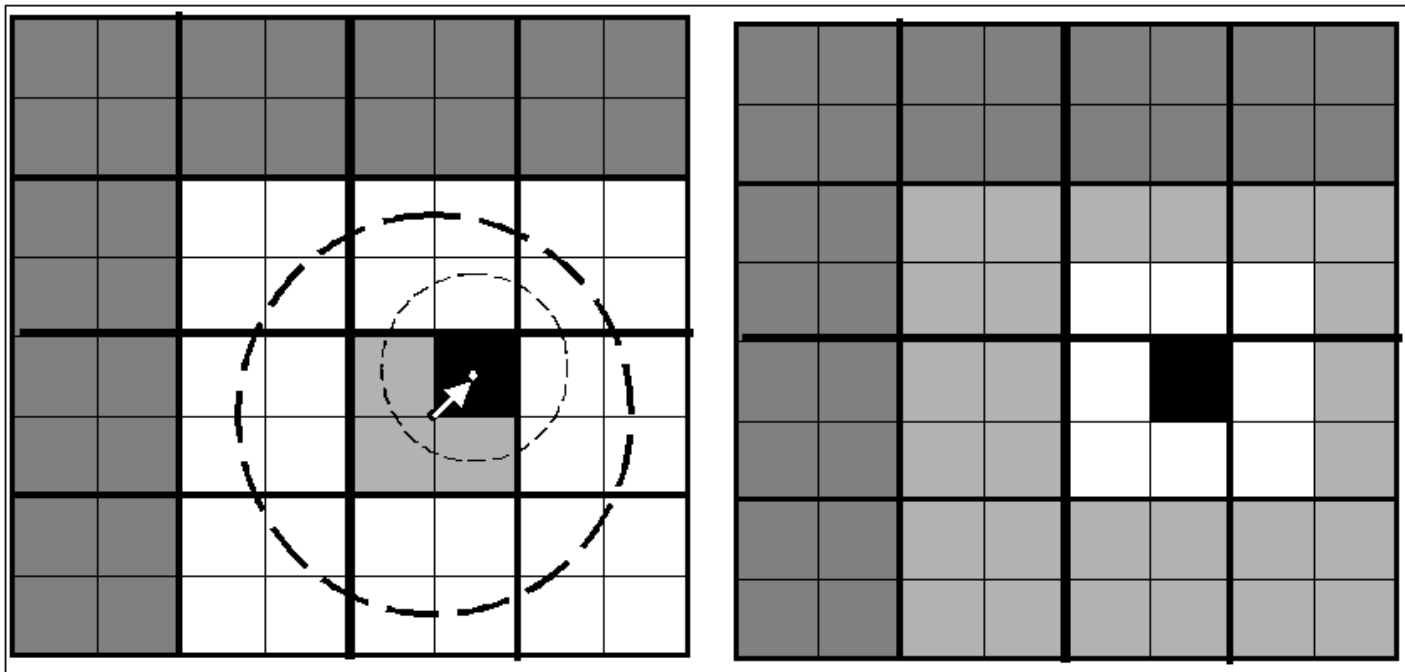
Setting Data Structure. Step 2 (4).

Build \tilde{D} -tree=D-tree.

Consider again Step 2 of the MLFMM Downward Pass.

$$\Phi_{n,l}^{(3)}(\mathbf{x}) = \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{x} - \mathbf{x}_c^{(n,l)}),$$

$$\mathbf{D}^{(n,l)} = \tilde{\mathbf{D}}^{(n,l)} + (\mathbf{R}|\mathbf{R})\left(\mathbf{x}_c^{(n',l-1)} - \mathbf{x}_c^{(n,l)}\right)\mathbf{D}^{(n',l-1)}, \quad n' = \text{Parent}(n).$$



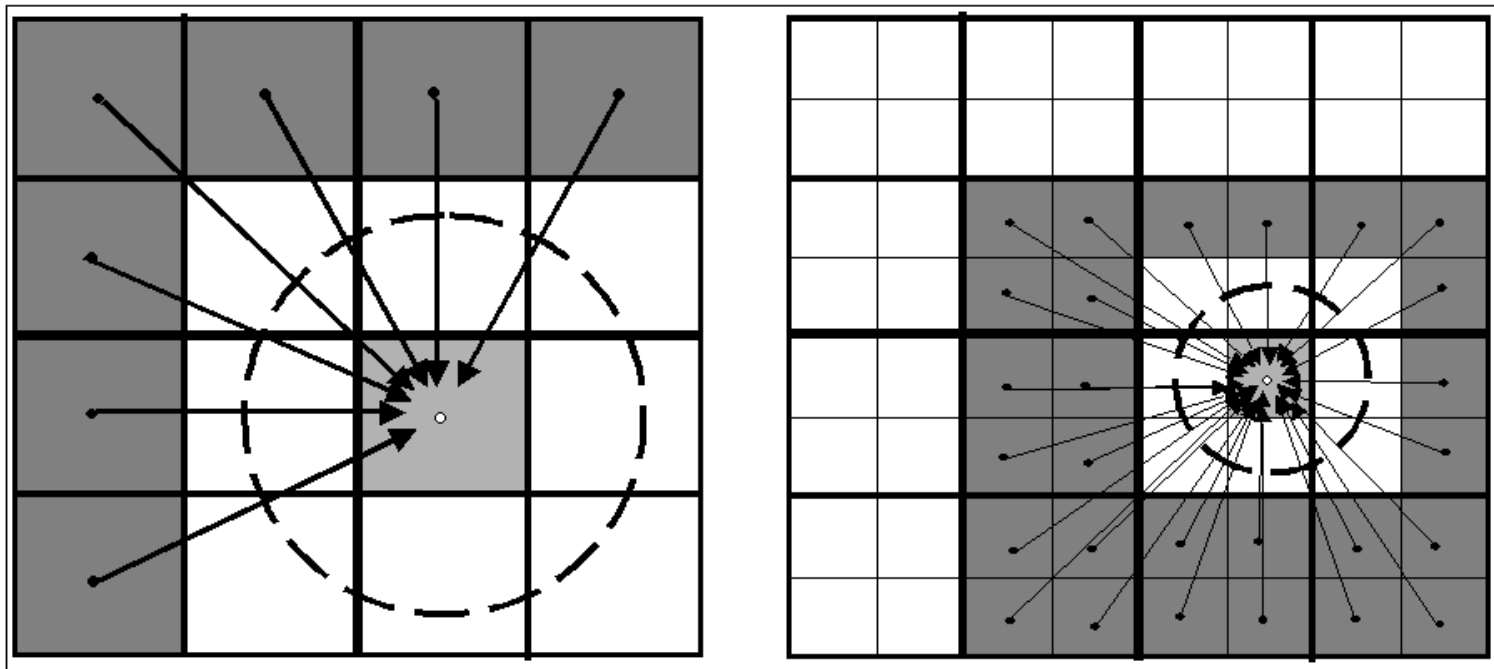
To compute D for a box we need \tilde{D} for the *Same* box

Setting Data Structure. Step 3 (1).

Build C-set.

Consider Step 1 of the MLFMM Downward Pass.

$$\Phi_{n,l}^{(4)}(\mathbf{x}) = \tilde{\mathbf{D}}^{(n,l)} \circ \mathbf{R}(\mathbf{x} - \mathbf{x}_c^{(n,l)}),$$
$$\tilde{\mathbf{D}}^{(n,l)} = \sum_{n' \in I_4(n,l)} (\mathbf{S}|\mathbf{R})\left(\mathbf{x}_c^{(n',l)} - \mathbf{x}_c^{(n,l)}\right) \mathbf{C}^{(n',l)}.$$



To compute $\tilde{\mathbf{D}}$ for a box we need \mathbf{C} for the I_4 neighborhood

Setting Data Structure. Step 3 (2).

Build C-set.

\mathbb{C} is the set of boxes for which we need to compute $\mathbf{C}^{(n,l)}$.

$\mathbb{C} = \mathbb{C}_2 \cup \dots \cup \mathbb{C}_{L-1}$ (for each level)

- 1.** For levels $l = L$ to $l = 2$ perform the following operation.
- 2.** For each $(n, l) \in \mathbb{D}_l$ refer boxes $\in I_4(n, l)$ and contain points of \mathbb{X} to \mathbb{C}_l .

Setting Data Structure. Step 4 (1).

Build C-forest.

The C -forest as it is clear from this name has a different structure than D -tree. We consider the structure called ‘forest’ as a union of K trees. The number K of these trees depends on the D -tree and the source data set \mathbb{X} , and we will determine it within the algorithm that constructs the C -forest. We denote these trees as $\mathbb{C}^1, \dots, \mathbb{C}^K$. These trees are independent, so

$$\mathbb{C}^k \cap \mathbb{C}^j = \emptyset, \quad k \neq j, \quad \mathbb{C} = \mathbb{C}^1 \cup \dots \cup \mathbb{C}^K.$$

Each tree will have its own coarsest and finest level, which we denote as l_k and L_k , respectively, $k = 1, \dots, K$. All these levels are inside the range $2 \leq l_k \leq L_k \leq L$, and will be determined within the algorithm provided below. We also denote as \mathbb{C}_l the sets of boxes in the C -forest at level $l \geq 2$, so $\mathbb{C} = \mathbb{C}_2 \cup \dots \cup \mathbb{C}_{L-1}$. The boxes of level l can belong to different trees. We denote the set of boxes at level l that belongs to the k -th tree as \mathbb{C}_l^k . It follows from the definition that $\mathbb{C}_l^k = \mathbb{C}^k \cap \mathbb{C}_l$.

Setting Data Structure. Step 4 (2).

Build C-forest.

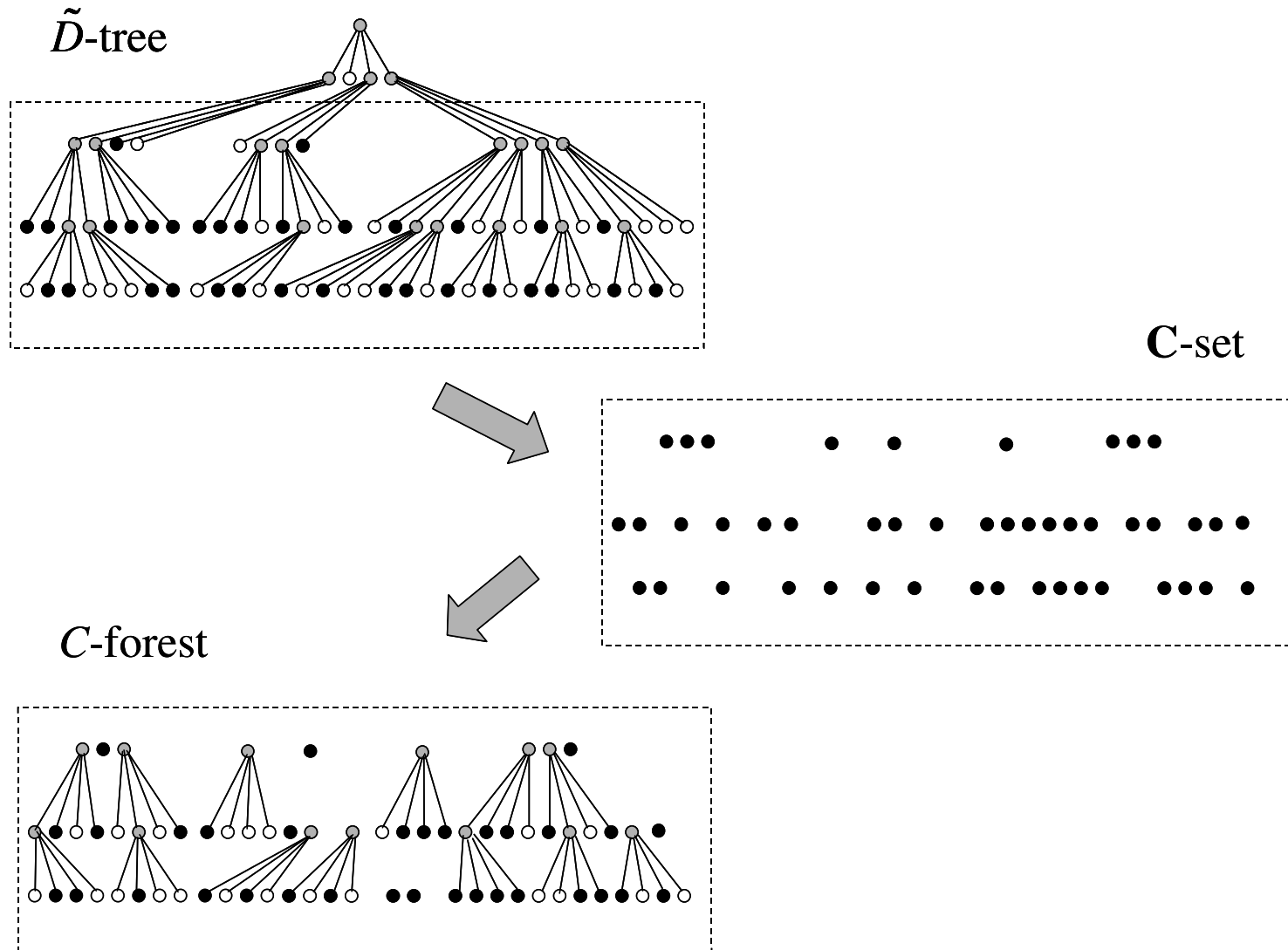
Algorithm

Determine K, l_k, L_k and sets \mathbb{C}_l^k , $l = 2, \dots, L$, $k = 1, \dots, K$.

- 1.** Find the number of elements P_2 in the set \mathbb{C}_2 . Set $K = P_2$, and $\mathbb{K} = [1, \dots, K]$. For $k = 1, \dots, K$ set $l_k = 2$, and refer each element to \mathbb{C}_2^k .
- 2.** For $l = 3, \dots, L$ perform the following loop.
- 3.** For $k \in \mathbb{K}$ find sets $\mathbb{G}_l^k = \text{Children}(\mathbb{C}_{l-1}^k) \cap \mathbb{C}_l$. If $\mathbb{G}_l^k \neq \emptyset$, refer all elements of \mathbb{G}_l^k to \mathbb{C}_l^k , else exclude k from the set \mathbb{K} and set $L_k = l - 1$.
- 4.** Find the number of elements P_l in the set $\mathbb{C}_l \setminus \left(\bigcup_{k \in \mathbb{K}} \mathbb{G}_l^k \right)$. Increase $K = K + P_l$,
 $\mathbb{K} = \mathbb{K} \cup [K + 1, \dots, K + P_l]$. For $k = K + 1, \dots, K + P_l$ set $l_k = l$, and refer each element of $\mathbb{C}_l \setminus \left(\bigcup_{k \in \mathbb{K}} \mathbb{G}_l^k \right)$ to \mathbb{C}_l^k .

Setting Data Structure. Step 4 (3).

Build C-forest.



Upward Pass (1)

- For each Tree in the forest can be computed independently;
- Skipping unnecessary translations, computing all leaves directly;
- Skipping unnecessary translations when going Upward.

Upward Pass (2)

1. For $k = 1, \dots, K$ (for each tree) perform the following steps.

2. Generate $\mathbb{C}^{(n,l)}$ for the leaf boxes, $l = L_k$, $\mathbb{C}^{(n,l)} \in \mathbb{C}_{L_k}^k$:

$$\forall (n, L_k) \in \mathbb{C}_{L_k}^k, \quad \Phi_{n, L_k}^{(1)}(\mathbf{y}) = \mathbb{C}^{(n, L_k)} \circ \mathbf{S}(\mathbf{y} - \mathbf{x}_c^{(n, L_k)}),$$

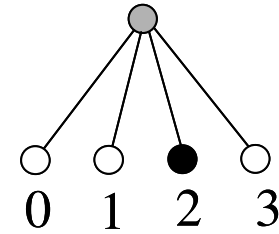
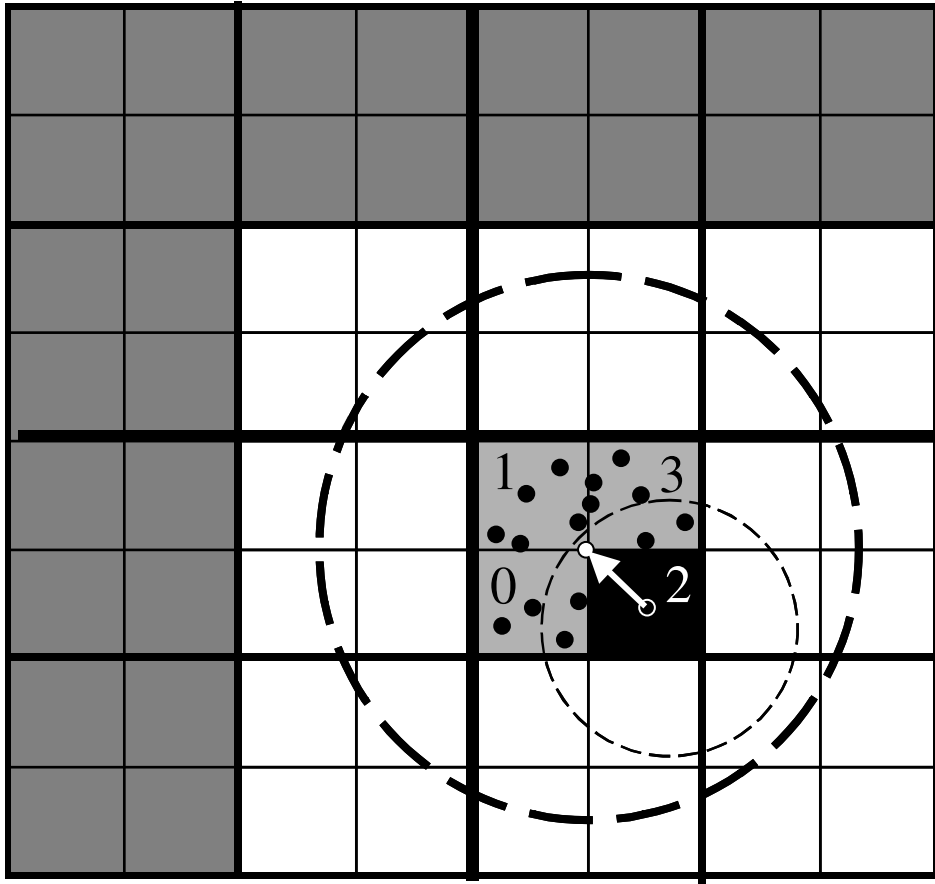
$$\mathbb{C}^{(n, L_k)} = \sum_{\mathbf{x}_i \in E_1(n, L_k)} u_i \mathbf{B}(\mathbf{x}_i, \mathbf{x}_c^{(n, L_k)}).$$

3. For $l = L_k - 1, \dots, l_k$ recursively generate $\mathbb{C}^{(n,l)}$ for all other boxes in tree \mathbb{C}^k :

$$\forall (n, l) \in \mathbb{C}_l^k, \quad \Phi_{n, l}^{(1)}(\mathbf{y}) = \mathbb{C}^{(n, l)} \circ \mathbf{S}(\mathbf{y} - \mathbf{x}_c^{(n, l)}),$$

$$\mathbb{C}^{(n, l)} = \sum_{\substack{n' \in \text{Children}(n) \\ \&(n', l+1) \in \mathbb{C}_{l+1}^k}} (\mathbf{S}|\mathbf{S})(\mathbf{x}_c^{(n, l)} - \mathbf{x}_c^{(n', l+1)}) \mathbb{C}^{(n', l+1)} + \sum_{\substack{\mathbf{x}_i \in E_1(n', l+1) \\ n' \in \text{Children}(n) \\ \&(n', l+1) \notin \mathbb{C}_{l+1}^k}} u_i \mathbf{B}(\mathbf{x}_i - \mathbf{x}_c^{(n, l)}).$$

Upward Pass (3)



Downward Pass

1. For $l = 2, \dots, L$ perform the following steps

2. For $(n, l) \in \mathbb{D}_l$ find $\tilde{\mathbf{D}}^{(n,l)}$:

$$\Phi_{n,l}^{(4)}(\mathbf{y}) = \tilde{\mathbf{D}}^{(n,l)} \circ \mathbf{R}(\mathbf{y} - \mathbf{x}_c^{(n,l)}),$$

$$\tilde{\mathbf{D}}^{(n,l)} = \sum_{n' \in I_4^1(n,l)} (\mathbf{S}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)} - \mathbf{x}_c^{(n',l)}\right) \mathbf{C}^{(n',l)} + \sum_{n' \in \text{Parent}(I_4^2(n,l))} (\mathbf{S}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)} - \mathbf{x}_c^{(n',l-1)}\right) \mathbf{C}^{(n',l-1)}.$$

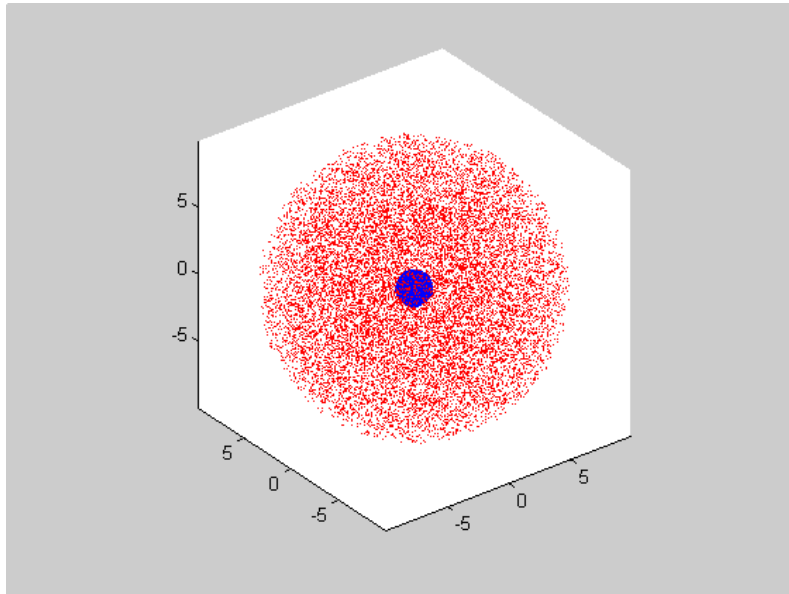
3. For $(n, l) \in \mathbb{D}_l$ find $\mathbf{D}^{(n,l)}$ (set $\mathbf{D}^{(n,2)} = \tilde{\mathbf{D}}^{(n,2)}$):

$$\Phi_{n,l}^{(3)}(\mathbf{y}) = \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{y} - \mathbf{x}_c^{(n,l)}),$$

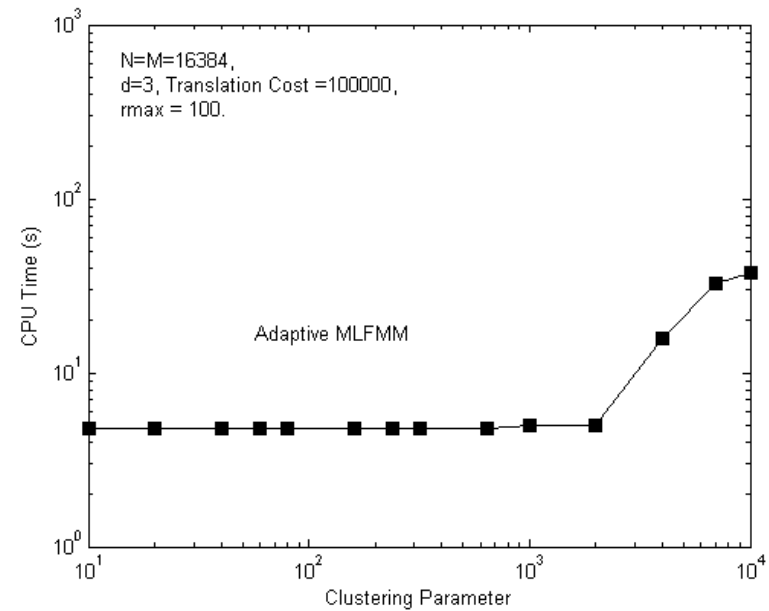
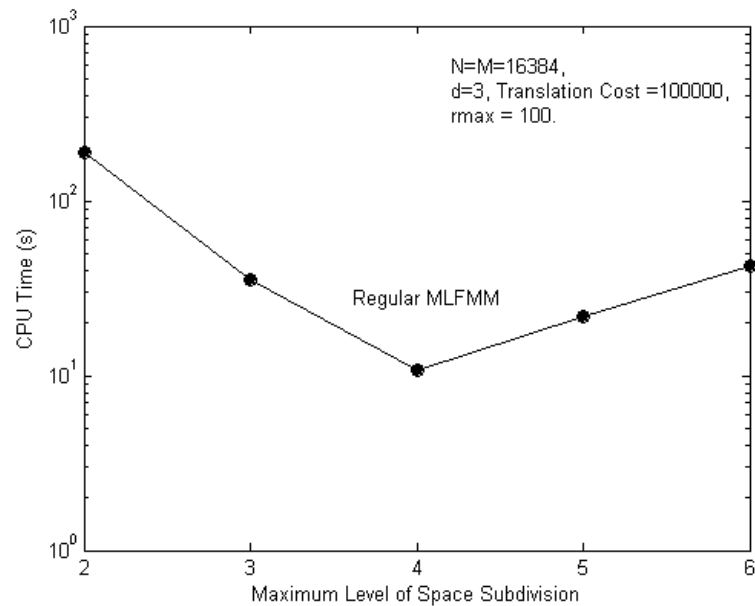
$$\mathbf{D}^{(n,l)} = \tilde{\mathbf{D}}^{(n,l)} + (\mathbf{R}|\mathbf{R})\left(\mathbf{x}_c^{(n,l)} - \mathbf{x}_c^{(n',l-1)}\right) \mathbf{D}^{(n',l-1)}, \quad n' = \text{Parent}(n).$$

Final Summation

$$\Phi(\mathbf{y}_j) = \sum_{\mathbf{x}_i \in E_2(n,l)} \Phi_i(\mathbf{y}_j) + \mathbf{D}^{(n,l)} \circ \mathbf{R}(\mathbf{y}_j - \mathbf{x}_c^{(n,l)}), \quad \mathbf{y}_j \in E_1(n,l).$$



Comparison of the Regular and Adaptive FMM. Sources (blue) distributed uniformly over the sphere. Targets (red) distributed uniformly inside a larger sphere.



Comparison of the Regular and Adaptive FMM. Gaussian clusters of sources (blue) distributed uniformly in side a cube. Targets (red) distributed uniformly inside the same cube.

