

Wave scattering with the UV multilevel partitioning method: 1. Two-dimensional problem of perfect electric conductor surface scattering

Leung Tsang, Dong Chen, and Peng Xu

Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China

Qin Li and Vikram Jandhyala

Department of Electrical Engineering, University of Washington, Seattle, Washington, USA

Received 18 November 2003; revised 21 April 2004; accepted 16 July 2004; published 9 October 2004.

[1] A UV method with multilevel partitioning (UVMLP) is developed to solve electromagnetic problem. In this paper we demonstrate the technique to treat electromagnetic problem for large surface in a two-dimensional (2-D) problem. Using the multilevel partitioning (MLP), the decomposition preprocessing, and the matrix vector multiplication require CPU of $O(N \log N)$ per iteration with a smaller constant factor for matrix column multiplication than decomposition. The memory requirement is of $O(N \log N)$. We demonstrate the technique for a rough surface scattering problem with surface length up to 13,000 wavelengths and RMS height up to ten wavelengths. Computations are based on using PC with a single Pentium 4–2.4 GHz processor and 1 G RAM. For the case of 65,536 unknowns, it requires about CPU from 3.3 to 5.33 s per iteration and a total CPU of 3.9–14.3 min with 49–146 conjugate gradient iterations. **INDEX TERMS:** 0644 Electromagnetics: Numerical methods; 0659 Electromagnetics: Random media and rough surfaces; 0669 Electromagnetics: Scattering and diffraction; **KEYWORDS:** fast solver, rough surface, integral equation

Citation: Tsang, L., D. Chen, P. Xu, Q. Li, and V. Jandhyala (2004), Wave scattering with the UV multilevel partitioning method: 1. Two-dimensional problem of perfect electric conductor surface scattering, *Radio Sci.*, 39, RS5010, doi:10.1029/2003RS003009.

1. Introduction

[2] The solutions of wave equations for rough surface scattering can be calculated by using integral equations with Green's function and the method of moments (MOM). The Green's function integral equation approach has the advantages that it has taken into account the propagation of waves from one point to another. However, using Greens function will lead to matrix equations with a full impedance matrix. The full matrix relates the source locations to the observation points through propagation. For such problems, the common methodology is to use iterative method of solutions of the matrix equation. Thus the bottleneck is the calculation of the product of the impedance matrix and a column vector. The column vector is the trial

solution in the iterative procedure. If the matrix is of order $N \times N$, then for each matrix-column vector product, it requires $O(N^2)$ operations. Furthermore, the memory requirement is of $O(N^2)$. Thus the solution becomes prohibitively expensive when N is large. In recent years, two techniques have been used to accelerate such impedance matrix-column vector computation. The techniques also save on computer memory. The methods are the sparse matrix canonical grid method (SMCG) [Tsang *et al.*, 1995; Chan and Tsang, 1995; Tsang *et al.*, 2001], and the steepest descent multilevel fast multipole method (SDFMM) [Rohklin, 1990; Michielssen and Chew, 1996; Jandhyala *et al.*, 1998]. Both methods have been used for large-scale 3-D simulations. The SMCG exhibits computational complexity of $O(N \log N)$ and memory requirement of $O(N)$. The SDFMM method exhibits computational complexity of $O(N)$ for rough surface problems and memory requirement of $O(N)$. An advantage of the SMCG method is that it only requires a translational invariant Green's function and can be read-

ily applied with the multilayered medium Green's function. The SMCG method is FFT based, so that parallel implementation can be accomplished by using a parallel version of FFT [Li *et al.*, 2000]. There is also the forward and backward method with an acceleration scheme [Chou and Johnson, 1998]. However, this method emphasizes on the reduction of iterations for rough surface scattering.

[3] Recently, Kapur and Long [1997] proposed a QR decomposition method by using the property of the smoothness of the impedance matrix elements. In their proposed QR approach, a static rank map is first determined. Using the map, the QR with modified Gram-Schmidt procedure is applied to the matrix blocks with recursive partitioning and merging. Results indicate $O(N \log N)$ computational complexity. The QR method has so far been demonstrated only for object of moderate size in terms of wavelengths.

[4] In this paper, a UV decomposition method with multilevel partitioning (UVMLP) is developed to solve large-scale electromagnetic 2-D rough surface scattering problem. In the accompanying paper [Tsang *et al.*, 2004] the technique is extended to 3-D problem. Although 3-D problems are encountered in the real world, 2-D rough surface problem remains important for benchmark comparisons. In the method, we apply the multilevel matrix partitioning (MLP) to partition the impedance matrix \bar{Z} . An upper bound of the rank of the matrix blocks is derived. On the basis of this upper bound, sampling techniques are next used to determine the rank of each block. Using this multilevel partitioning, the impedance matrix is written as a product of U and V matrix. The decomposition preprocessing and the matrix-vector multiplication both have CPU computational complexity of $O(N \log N)$ with a smaller constant factor for matrix-column multiplication than decomposition. The memory requirement is of order $O(N \log N)$. We demonstrate the technique for a rough surface scattering problem with surface length up to 13000 wavelengths and RMS height up to 10 wavelengths. For the case of 65536 unknowns, it requires about CPU 3.3 s per iteration and a total CPU of 3.9 min with 49 conjugate gradient iterations. For the case of 131072 unknowns it requires about CPU 7.4 s per iteration and a total CPU of 14.5 min with 96 conjugate gradient iterations. The increase in total CPU from 65536 unknowns to 131072 unknowns is due to the increase in the number of iterations. We also demonstrate the technique for TE and TM polarizations at large incident angles. There are only slight difference of CPU and number of iterations from realization to realization. Because the UV decomposition is applied independently to each level and each block, the procedure facilitates parallel implementation. However, parallel implementation has not been implemented in this paper. All

CPU quoted are that of a single processor with a single Pentium 4–2.4 GHz processor.

[5] In section 2, we formulate the electromagnetic problem of scattering and describe the matrix multilevel partitioning. In section 3, we describe the upper bound of matrix rank for each block. Then we describe the sampling procedure for the blocks in the partitioned matrix, the UV decomposition algorithm and the matrix-column vector multiplication. In section 4, the computational complexity and memory requirement are derived. In section 5, numerical results are illustrated. The CPU complexity, memory requirement and matrix ranks are also shown numerically.

2. Two-Dimensional Wave Scattering by Rough Surface With Multilevel Partitioning

[6] Consider a 2-D problem of an incident wave $\psi_{inc}(\bar{r})$ impinging upon a PEC rough surface with height profile $z = f(x)$. For the TE case of electromagnetic scattering with $\bar{E} = \hat{y}E$, using the Dirichlet boundary condition $\psi(\bar{r}) = 0$ for \bar{r} on S . The integral equation can be written as [Tsang *et al.*, 2001]

$$\psi_{inc}^{TE}(\bar{r}') = \int_S ds g(\bar{r}, \bar{r}') \hat{n} \cdot \nabla \psi^{TE}(\bar{r}), \quad (1a)$$

for \bar{r} and \bar{r}' on S .

For the TM case, $\bar{H} = \hat{y}H$, the integral equation is [Tsang *et al.*, 2001]

$$\psi_{inc}^{TM}(\bar{r}') = - \int_P ds \psi^{TM}(\bar{r}) \hat{n} \cdot \nabla g(\bar{r}, \bar{r}') + \frac{1}{2} \psi^{TM}(\bar{r}'), \quad (1b)$$

for \bar{r} and \bar{r}' on S ,

where $g(\bar{r}, \bar{r}') = \frac{i}{4} H_0^{(1)}(k|\bar{r} - \bar{r}'|)$ is the 2-D Green's function. \int_P denotes a principle value of integral. After discretization, equation (1) can be converted to matrix equations:

$$\sum_{n=1}^N Z_{mn} W_n = b_m, \quad m = 1, 2, \dots, N. \quad (2)$$

In matrix notation, equation (2) can be written as

$$\bar{Z} \bar{W} = \bar{b}, \quad (3)$$

where

$$\begin{cases} W_n^{TE} = W^{TE}(x_n) = \sqrt{1 + \left(\frac{df}{dx}\right)^2} (\hat{n} \cdot \nabla \psi^{TE}(\bar{r}))_{z=f(x_n)} \\ W_n^{TM} = W^{TM}(x_n) = \psi^{TM}(\bar{r})_{z=f(x_n)} \end{cases} \quad (4a)$$

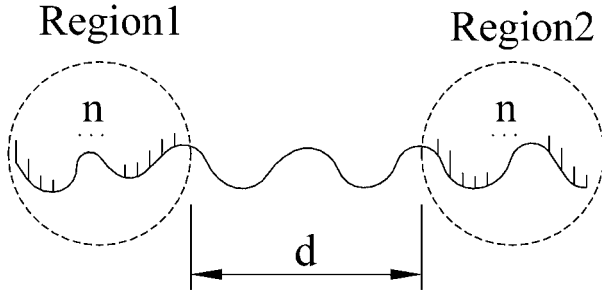


Figure 1. Two well-separated regions of rough surface with sampling points.

$$b_m = b(x_m) = \psi_{inc}(x_m, f(x_m)) \quad (4b)$$

$$Z_{mn}^{TE} = \begin{cases} \Delta x \cdot g(x_n, f(x_n); x_m, f(x_m)) & \text{for } m \neq n \\ \frac{i\Delta x}{4} \left[1 + i \frac{2}{\pi} \ln \left(\frac{\gamma k}{4e} \Delta x \sqrt{1 + (f'(x_m))^2} \right) \right] & \text{for } m = n \end{cases}$$

$$Z_{mn}^{TM} = \begin{cases} -\Delta x \sqrt{1 + (f'(x_m))^2} \hat{n} \cdot \nabla g(x_n, f(x_n); x_m, f(x_m)) & \text{for } m \neq n \\ \frac{1}{2} & \text{for } m = n \end{cases} \quad (4c)$$

$$\gamma = 1.78107.$$

Consider the interactions between two well-separated regions of the rough surface, as shown in Figure 1. Region 1 and region 2 contain n points and n points, respectively. The minimum distance of the two regions is d . According to equation (3), the interactions can be written as a $n \times n$ matrix $\underline{\underline{A}}$. Suppose the rank of matrix $\underline{\underline{A}}$ is r . When d is large enough and generally $d > 1.0\lambda$, $r \ll n$. $\underline{\underline{A}}$ can be written as

$$\underline{\underline{A}} = \overline{\underline{\underline{U}}} \overline{\underline{\underline{V}}}, \quad (5)$$

where $\overline{\underline{\underline{U}}}$ is dimension $n \times r$, and $\overline{\underline{\underline{V}}}$ is dimension $r \times n$.

[7] Using matrix $\overline{\underline{\underline{U}}}$ and $\overline{\underline{\underline{V}}}$ instead of matrix $\underline{\underline{A}}$ to multiply with a $n \times 1$ vector $\underline{\underline{X}}$, the computational time can be decreased from $O(n^2)$ to $O(2rn)$.

$$\overline{\underline{\underline{A}}} \underline{\underline{X}} = \overline{\underline{\underline{U}}} (\overline{\underline{\underline{V}}} \underline{\underline{X}}) \quad (6)$$

[8] The impedance elements are partitioned into multilevel $i = 1, 2, 3, \dots, p$ as shown in Figure 2 (top), with $\overline{\underline{\underline{Z}}} = \overline{\underline{\underline{Z}}}^{(0)} + \overline{\underline{\underline{Z}}}^{(1)U} + \overline{\underline{\underline{Z}}}^{(2)U} + \dots + \overline{\underline{\underline{Z}}}^{(p)U} + \overline{\underline{\underline{Z}}}^{(1)L} + \overline{\underline{\underline{Z}}}^{(2)L} + \dots + \overline{\underline{\underline{Z}}}^{(p)L}$, within each level, the ratio of maximum data separation $x_{\max}^{(i)}$ and minimum separation $x_{\min}^{(i)}$ is kept on a constant. Where the number in parentheses stands for the level of the group, superscript U and L denote the upper matrix and lower matrix respectively. We

illustrate the partitioning in the following example. The approach can be generalized to any order of multilevel partitioning. The multilevel partitioning has been used in the steepest descent fast multipole method (SDFMM; *Michielssen and Chew [1996]*). In the following simple example, we use the notations as in the work of *Tsang et al. [2001]*.

[9] The impedance elements are decomposed into various levels based on multilevel grouping. To illustrate, let the number of elements in the first-level group be M and the number of groups of the first-level be L . Then the total number of elements is $N = LM$ (e.g., $M = 20$, $L = 64$, then $N = 1280$). We illustrate it for the case $p = 5$. Generalization can readily be made to other values of M and L . For numerical implementation in this paper, we select p to be more than 10 for cases of large surfaces. Let $L = 2^{p+1}$, where p corresponds to the level of decomposition of the impedance matrix. The first level groups have M elements each. Beyond the first level, the number of elements increases by a factor of 2 for each higher level (Table 1). The impedance matrix is decomposed ($p = 5$ for this case)

$$\overline{\underline{\underline{Z}}} = \overline{\underline{\underline{Z}}}^{(0)} + \overline{\underline{\underline{Z}}}^{(1)U} + \overline{\underline{\underline{Z}}}^{(2)U} + \dots + \overline{\underline{\underline{Z}}}^{(5)U} + \overline{\underline{\underline{Z}}}^{(1)L} + \overline{\underline{\underline{Z}}}^{(2)L} + \dots + \overline{\underline{\underline{Z}}}^{(5)L}. \quad (7)$$

The upper matrix has column index larger than row index for nonzero elements. It is the reverse for the lower matrix.

[10] Let $\overline{\underline{\underline{Z}}}_{m_i n_i}^{(i)}$ denote the interaction of elements between group m_i and group n_i in level i . They are all full matrices.

$$\overline{\underline{\underline{Z}}}_{m_i n_i}^{(1)} = \text{dimension } M \times M \quad (8)$$

$$\overline{\underline{\underline{Z}}}_{m_i n_i}^{(2)} = \text{dimension } 2M \times 2M \quad (9)$$

Table 1. Group Levels, Number of Elements, and Number of Groups^a

Level of Group	Number of Elements	Number of groups
1	M	64
2	2M	32
3	4M	16
4	8M	8
5	16M	4
6	32M	2

^aFor $L = 64$, $p = 5$. Here the grouping is as shown in the bottom panel of Figure 2. Group m at level n is denoted as m_n .

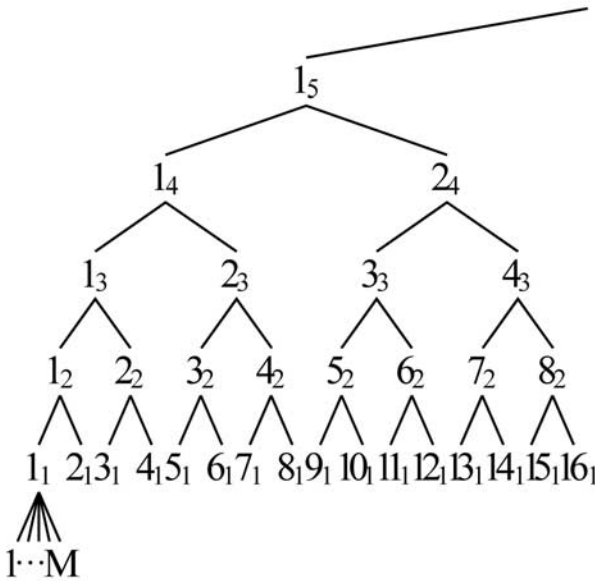
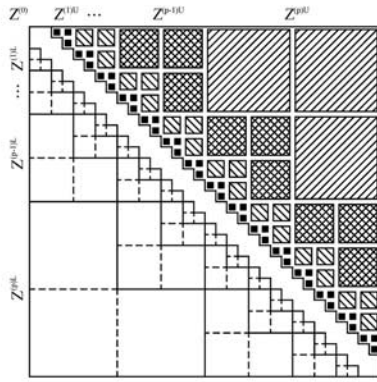


Figure 2. Multilevel structure. (top) Block structure. (bottom) Grouping structure.

Note that $\bar{\bar{Z}}_{m_i n_i}^{(i)}$ are defined differently from $\bar{Z}^{(i)}$. For example, $\bar{Z}^{(2)}$ is of dimension $N \times N$ while $\bar{\bar{Z}}_{m_i n_i}^{(2)}$ is of dimension $2M \times 2M$. The various definitions of impedance matrix elements should be distinguished.

$$\bar{\bar{0}}^{(1)} = \text{zero matrix of dimension } M \times M \quad (10)$$

$$\bar{\bar{0}}^{(2)} = \text{zero matrix of dimension } 2M \times 2M \quad (11)$$

$$\bar{\bar{0}}^{(i)} = \text{zero matrix of dimension } (2^{i-1}M) \times (2^{i-1}M) \quad (12)$$

Suppose we use $M = 20$. Then examples are

$$\bar{\bar{Z}}_{3_1 9_1}^{(1)} = \begin{bmatrix} Z_{41,161} & \cdots & Z_{41,180} \\ \vdots & \ddots & \vdots \\ Z_{60,161} & \cdots & Z_{60,180} \end{bmatrix} = 20 \times 20 \quad (13)$$

$$\bar{\bar{Z}}_{3_2 9_2}^{(2)} = \begin{bmatrix} \bar{\bar{Z}}_{5_1 17_1}^{(1)} & \bar{\bar{Z}}_{5_1 18_1}^{(1)} \\ \bar{\bar{Z}}_{6_1 17_1}^{(1)} & \bar{\bar{Z}}_{6_1 18_1}^{(1)} \end{bmatrix} = 40 \times 40. \quad (14)$$

Thus $\bar{\bar{Z}}_{m_1 n_1}^{(1)}$, $\bar{\bar{Z}}_{m_2 n_2}^{(1)}$, $\bar{\bar{Z}}_{m_3 n_3}^{(1)}$, etc., keep track of all the individual impedance matrix elements.

[11] The 0th-level impedance matrix $\bar{\bar{Z}}^{(0)}$ represents interaction at level 1 between itself and its neighbors on the two sides.

$$\bar{\bar{Z}}^{(0)} = \begin{bmatrix} \bar{\bar{Z}}_{1_1 1_1}^{(1)} & \bar{\bar{Z}}_{1_1 2_1}^{(1)} & \bar{\bar{0}}^{(1)} & \bar{\bar{0}}^{(1)} & \cdots \\ \bar{\bar{Z}}_{2_1 1_1}^{(1)} & \bar{\bar{Z}}_{2_1 2_1}^{(1)} & \bar{\bar{Z}}_{2_1 3_1}^{(1)} & \bar{\bar{0}}^{(1)} & \cdots \\ \bar{\bar{0}}^{(1)} & \bar{\bar{Z}}_{3_1 2_1}^{(1)} & \bar{\bar{Z}}_{3_1 3_1}^{(1)} & \bar{\bar{Z}}_{3_1 4_1}^{(1)} & \cdots \\ \bar{\bar{0}}^{(1)} & \bar{\bar{0}}^{(1)} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} = N \times N. \quad (15)$$

For $\bar{\bar{Z}}^{(0)}$, the impedance matrix elements can be calculated directly.

[12] The impedance matrix of each level is generated by following rules:

[13] 1. In each level, the impedance matrices only interact with itself or its nearest neighbors.

[14] 2. Each impedance matrix elements in $\bar{\bar{Z}}$ can occur only once in the matrix decomposition. If a pair had interacted previously in lower level groups, it cannot interact again in the current level. That entry has to be set to zero at the current level.

[15] To generate the next level, i.e., first-level impedance matrix $\bar{\bar{Z}}^{(1)}$, we first apply rule (1) to get the $N \times N$ matrix

$$\begin{bmatrix} \bar{\bar{Z}}_{1_2 1_2}^{(2)} & \bar{\bar{Z}}_{1_2 2_2}^{(2)} & \bar{\bar{0}}^{(2)} & \bar{\bar{0}}^{(2)} & \cdots \\ \bar{\bar{Z}}_{2_2 1_2}^{(2)} & \bar{\bar{Z}}_{2_2 2_2}^{(2)} & \bar{\bar{Z}}_{2_2 3_2}^{(2)} & \bar{\bar{0}}^{(2)} & \cdots \\ \bar{\bar{0}}^{(2)} & \bar{\bar{Z}}_{3_2 2_2}^{(2)} & \bar{\bar{Z}}_{3_2 3_2}^{(2)} & \bar{\bar{Z}}_{3_2 4_2}^{(2)} & \cdots \\ \bar{\bar{0}}^{(2)} & \bar{\bar{0}}^{(2)} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}. \quad (16)$$

However, according to rule (2), the diagonal matrix elements should be set to zero because they have already interacted in $\bar{Z}^{(0)}$. Setting them to zero then gives

$$\begin{bmatrix} \bar{0}^{(2)} & \bar{Z}_{1_2 2_2}^{(2)} & \bar{0}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{Z}_{2_2 1_2}^{(2)} & \bar{0}^{(2)} & \bar{Z}_{2_2 3_2}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{Z}_{3_2 2_2}^{(2)} & \bar{0}^{(2)} & \bar{Z}_{3_2 4_2}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{0}^{(2)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}. \quad (17)$$

However, in

$$\bar{Z}_{1_2 2_2}^{(2)} = \begin{bmatrix} \bar{Z}_{1_1 3_1}^{(1)} & \bar{Z}_{1_1 4_1}^{(1)} \\ \bar{Z}_{2_1 3_1}^{(1)} & \bar{Z}_{2_1 4_1}^{(1)} \end{bmatrix}, \quad (18)$$

$\bar{Z}_{2_1 3_1}^{(1)}$ has already been included in $\bar{Z}^{(0)}$. Thus we set that entry to zero also and define, with a prime superscript,

$$\bar{Z}_{1_2 2_2}^{(2)'} = \begin{bmatrix} \bar{Z}_{1_1 3_1}^{(1)} & \bar{Z}_{1_1 4_1}^{(1)} \\ \bar{0}^{(1)} & \bar{Z}_{2_1 4_1}^{(1)} \end{bmatrix} = \text{dimension } 2M \times 2M. \quad (19)$$

Similarly, we define

$$\bar{Z}_{2_2 1_2}^{(2)'} = \begin{bmatrix} \bar{Z}_{3_1 1_1}^{(1)} & \bar{0}^{(1)} \\ \bar{Z}_{4_1 1_1}^{(1)} & \bar{Z}_{4_1 2_1}^{(1)} \end{bmatrix}. \quad (20)$$

Thus the first level impedance matrix assumes the final form:

$$\begin{aligned} \bar{Z}^{(1)} &= \bar{Z}^{(1)U} + \bar{Z}^{(1)L} \\ \bar{Z}^{(1)U} &= \begin{bmatrix} \bar{0}^{(2)} & \bar{Z}_{1_2 2_2}^{(2)'} & \bar{0}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{Z}_{2_2 3_2}^{(2)'} & \bar{0}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{Z}_{3_2 4_2}^{(2)'} & \dots \\ \bar{0}^{(2)} & \bar{0}^{(2)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix} \\ &= \begin{bmatrix} \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{1_1 3_1}^{(1)} & \bar{Z}_{1_1 4_1}^{(1)} & \bar{0}^{(1)} & \dots & \dots & \dots & \dots \\ \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{2_1 4_1}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \dots & \dots & \dots \\ \dots & \dots & \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{3_1 5_1}^{(1)} & \bar{Z}_{3_1 6_1}^{(1)} & \bar{0}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{4_1 6_1}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \dots \\ \dots & \dots & \dots & \dots & \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{5_1 7_1}^{(1)} & \bar{Z}_{5_1 8_1}^{(1)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{6_1 8_1}^{(1)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \end{aligned} \quad (22)$$

$$\begin{aligned} \bar{Z}^{(1)L} &= \begin{bmatrix} \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{Z}_{2_2 1_2}^{(2)'} & \bar{0}^{(2)} & \bar{0}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{Z}_{3_2 2_2}^{(2)'} & \bar{0}^{(2)} & \bar{0}^{(2)} & \dots \\ \bar{0}^{(2)} & \bar{0}^{(2)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix} \\ &= \begin{bmatrix} \bar{0}^{(1)} & \dots & \dots & \dots & \dots & \dots & \dots \\ \bar{0}^{(1)} & \bar{0}^{(1)} & \dots & \dots & \dots & \dots & \dots \\ \bar{Z}_{3_1 1_1}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \dots & \dots & \dots & \dots \\ \bar{Z}_{4_1 1_1}^{(1)} & \bar{Z}_{4_1 2_1}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \dots & \dots & \dots \\ \bar{0}^{(1)} & \bar{0}^{(1)} & \bar{Z}_{5_1 3_1}^{(1)} & \bar{0}^{(1)} & \bar{0}^{(1)} & \dots & \dots \\ \dots & \bar{0}^{(1)} & \bar{Z}_{6_1 3_1}^{(1)} & \bar{Z}_{6_1 4_1}^{(1)} & \bar{0}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix}. \end{aligned} \quad (23)$$

For the second-level impedance matrix, we have

$$\bar{Z}^{(2)} = \bar{Z}^{(2)U} + \bar{Z}^{(2)L}. \quad (24)$$

Note that

$$\bar{Z}_{1_3 2_3}^{(3)} = \begin{bmatrix} \bar{Z}_{1_2 3_2}^{(2)} & \bar{Z}_{1_2 4_2}^{(2)} \\ \bar{Z}_{2_2 3_2}^{(2)} & \bar{Z}_{2_2 4_2}^{(2)} \end{bmatrix}, \quad (25)$$

but $\bar{Z}_{2_2 3_2}^{(2)}$ has already been included in previous level impedance matrix $\bar{Z}^{(0)}$ or $\bar{Z}^{(1)U} + \bar{Z}^{(1)L}$. Thus we set that part to zero and define, with a prime superscript,

$$\begin{aligned} \bar{Z}_{1_3 2_3}^{(3)'} &= \begin{bmatrix} \bar{Z}_{1_2 3_2}^{(2)'} & \bar{Z}_{1_2 4_2}^{(2)} \\ \bar{0}^{(2)} & \bar{Z}_{2_2 4_2}^{(2)} \end{bmatrix} = \text{dimension } 2^2 M \times 2^2 M \\ &= \begin{bmatrix} \bar{Z}_{1_1 5_1}^{(1)} & \bar{Z}_{1_1 6_1}^{(1)} & \bar{Z}_{1_1 7_1}^{(1)} & \bar{Z}_{1_1 8_1}^{(1)} \\ \bar{Z}_{2_1 5_1}^{(1)} & \bar{Z}_{2_1 6_1}^{(1)} & \bar{Z}_{2_1 7_1}^{(1)} & \bar{Z}_{2_1 8_1}^{(1)} \\ \bar{0} & \bar{0} & \bar{Z}_{3_1 7_1}^{(1)} & \bar{Z}_{3_1 8_1}^{(1)} \\ \bar{0} & \bar{0} & \bar{Z}_{4_1 7_1}^{(1)} & \bar{Z}_{4_1 8_1}^{(1)} \end{bmatrix}. \end{aligned} \quad (26)$$

Similarly, let

$$\bar{Z}_{2_3 1_3}^{(3)'} = \begin{bmatrix} \bar{Z}_{3_2 1_2}^{(2)} & \bar{0}^{(2)} \\ \bar{Z}_{4_2 1_2}^{(2)} & \bar{Z}_{4_2 2_2}^{(2)} \end{bmatrix}. \quad (27)$$

Thus the second-level upper matrix $\overline{\overline{Z}}^{(2)U}$ is

$$\overline{\overline{Z}}^{(2)U} = \begin{bmatrix} \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{1_3 2_3}^{(3)'} & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \dots \\ \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{2_3 3_3}^{(3)'} & \overline{\overline{0}}^{(3)} & \dots \\ \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{3_3 4_3}^{(3)'} & \dots \\ \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

$$= \begin{bmatrix} \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{1_2 3_2}^{(2)} & \overline{\overline{Z}}_{1_2 4_2}^{(2)} & \overline{\overline{0}}^{(2)} & \dots & \dots & \dots & \dots \\ \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{2_2 4_2}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \dots & \dots & \dots \\ \dots & \dots & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{3_2 5_2}^{(2)} & \overline{\overline{Z}}_{3_2 6_2}^{(2)} & \overline{\overline{0}}^{(2)} & \dots & \dots \\ \dots & \dots & \dots & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{4_2 6_2}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \dots \\ \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{5_2 7_2}^{(2)} & \overline{\overline{Z}}_{5_2 8_2}^{(2)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(2)} & \overline{\overline{0}}^{(2)} & \overline{\overline{Z}}_{6_2 8_2}^{(2)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

$$= \begin{bmatrix} \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{1_1 5_1}^{(1)} & \overline{\overline{Z}}_{1_1 6_1}^{(1)} & \overline{\overline{Z}}_{1_1 7_1}^{(1)} & \overline{\overline{Z}}_{1_1 8_1}^{(1)} & \overline{\overline{0}}^{(1)} & \dots \\ \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{2_1 5_1}^{(1)} & \overline{\overline{Z}}_{2_1 6_1}^{(1)} & \overline{\overline{Z}}_{2_1 7_1}^{(1)} & \overline{\overline{Z}}_{2_1 8_1}^{(1)} & \overline{\overline{0}}^{(1)} & \dots \\ \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{3_1 7_1}^{(1)} & \overline{\overline{Z}}_{3_1 8_1}^{(1)} & \overline{\overline{0}}^{(1)} & \dots \\ \dots & \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{4_1 7_1}^{(1)} & \overline{\overline{Z}}_{4_1 8_1}^{(1)} & \overline{\overline{0}}^{(1)} & \dots \\ \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{5_1 9_1}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \overline{\overline{Z}}_{6_1 9_1}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(1)} & \overline{\overline{0}}^{(1)} & \dots \end{bmatrix} \quad (28)$$

A similar expression can be found for $\overline{\overline{Z}}^{(2)L}$.

[16] Similar derivation can be applied to higher levels.

For level 3, $\overline{\overline{Z}}^{(3)} = \overline{\overline{Z}}^{(3)U} + \overline{\overline{Z}}^{(3)L}$ with

$$\overline{\overline{Z}}^{(3)U} = \begin{bmatrix} \overline{\overline{0}}^{(4)} & \overline{\overline{Z}}_{1_4 2_4}^{(4)'} & \overline{\overline{0}}^{(4)} & \overline{\overline{0}}^{(4)} & \dots \\ \overline{\overline{0}}^{(4)} & \overline{\overline{0}}^{(4)} & \overline{\overline{Z}}_{2_4 3_4}^{(4)'} & \overline{\overline{0}}^{(4)} & \dots \\ \overline{\overline{0}}^{(4)} & \overline{\overline{0}}^{(4)} & \overline{\overline{0}}^{(4)} & \overline{\overline{Z}}_{3_4 4_4}^{(4)'} & \dots \\ \overline{\overline{0}}^{(4)} & \overline{\overline{0}}^{(4)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

$$= \begin{bmatrix} \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{1_3 3_3}^{(3)} & \overline{\overline{Z}}_{1_3 4_3}^{(3)} & \overline{\overline{0}}^{(3)} & \dots & \dots & \dots & \dots \\ \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{2_3 4_3}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \dots & \dots & \dots \\ \dots & \dots & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{3_3 5_3}^{(3)} & \overline{\overline{Z}}_{3_3 6_3}^{(3)} & \overline{\overline{0}}^{(3)} & \dots & \dots \\ \dots & \dots & \dots & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{4_3 6_3}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \dots \\ \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{5_3 7_3}^{(3)} & \overline{\overline{Z}}_{5_3 8_3}^{(3)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \overline{\overline{0}}^{(3)} & \overline{\overline{0}}^{(3)} & \overline{\overline{Z}}_{6_3 8_3}^{(3)} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \quad (29)$$

A similar expression can be found for $\overline{\overline{Z}}^{(3)L}$.

[17] Continuing in this manner, the original impedance matrix is decomposed up to $\overline{\overline{Z}}^{(5)} = \overline{\overline{Z}}^{(5)U} + \overline{\overline{Z}}^{(5)L}$, all level impedance matrices are shown in the top of Figure 2.

[18] We note that for each level the matrices are in block structure. The distance separation between receiving elements and transmitting elements of different levels are illustrated as follows. For the submatrix $\overline{\overline{Z}}_{m_i n_i}^{(i)}$ in level i , we define n_i is the transmitting elements group; m_i is the receiving elements group.

[19] Level 1: $\overline{\overline{Z}}_{1_1 3_1}^{(1)}$ is dimension $M \times M$; transmitting elements group, 3₁; receiving elements group, 1₁; maximum separation $x_{\max}^{(1)} \cong 4M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong M\Delta x$. $\overline{\overline{Z}}_{1_1 4_1}^{(1)}$ is dimension $M \times M$; transmitting elements group, 4₁; receiving elements group, 1₁; maximum separation $x_{\max}^{(1)} \cong 4M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong M\Delta x$. $\overline{\overline{Z}}_{2_1 4_1}^{(1)}$ is dimension $M \times M$; transmitting elements group, 4₁; receiving elements group, 2₁; maximum separation $x_{\max}^{(1)} \cong 4M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong M\Delta x$.

[20] Level 2: $\overline{\overline{Z}}_{1_2 3_2}^{(2)}$ is dimension $2M \times 2M$; transmitting elements group, 3₂; receiving elements group, 1₂; maximum separation $x_{\max}^{(1)} \cong 8M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 2M\Delta x$. $\overline{\overline{Z}}_{1_2 4_2}^{(2)}$ is dimension $2M \times 2M$; transmitting elements group, 4₂; receiving elements group, 1₂; maximum separation $x_{\max}^{(1)} \cong 8M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 2M\Delta x$. $\overline{\overline{Z}}_{2_2 4_2}^{(2)}$ is dimension $2M \times 2M$; transmitting elements group, 4₂; receiving elements group, 2₂; maximum separation $x_{\max}^{(1)} \cong 8M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 2M\Delta x$.

[21] Level 3: $\overline{\overline{Z}}_{1_3 3_3}^{(3)}$ is dimension $4M \times 4M$; transmitting elements group, 3₃; receiving elements group, 1₃; maximum separation $x_{\max}^{(1)} \cong 16M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 4M\Delta x$. $\overline{\overline{Z}}_{1_3 4_3}^{(3)}$ is dimension $4M \times 4M$; transmitting elements group, 4₃; receiving elements group, 1₃; maximum separation $x_{\max}^{(1)} \cong 16M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 4M\Delta x$. $\overline{\overline{Z}}_{2_3 4_3}^{(3)}$ dimension $4M \times 4M$; transmitting elements group, 4₃; receiving elements group, 2₃; maximum separation $x_{\max}^{(1)} \cong 16M\Delta x$; and minimum separation $x_{\min}^{(1)} \cong 4M\Delta x$.

[22] The ratio of $x_{\max}^{(i)}$ and $x_{\min}^{(i)}$ is kept in constant of each level. On the basis of steepest descent representation of Green's function, the matrix rank of this block has an upper bound of Q (Appendix A). We apply UV to this block as follows.

3. UV Decomposition With Sampling Procedure to the Blocks

[23] Consider a matrix $\overline{\overline{A}}$ of dimension $n_{\perp} \times n$. UV decomposition tries to find the rank of matrix $\overline{\overline{A}}$ and form

the matrices \overline{U} and \overline{V} . The procedure is described as follows:

[24] 1. Pick r_0 columns uniformly of \overline{A} to construct a new matrix \overline{U}_0 of dimension $n \times r_0$, $r_0 \ll n$:

$$\overline{U}_0 = [\overline{A}_{i_1} \overline{A}_{i_2} \cdots \overline{A}_{i_{r_0}}], \quad (30)$$

where \overline{A}_{i_j} is the i_j th column vector of matrix \overline{A} , the series number $i_j \in 1, 2, \dots, n, j = 1, 2, \dots, r_0$.

[25] 2. Pick i_1 th, i_2 th, \dots, i_{r_0} th row vector of \overline{U}_0 to form matrix \overline{W}_0 of dimension $r_0 \times r_0$:

$$\overline{W}_0 = \begin{bmatrix} \overline{U}_{0i_1}^T \\ \overline{U}_{0i_2}^T \\ \vdots \\ \overline{U}_{0i_{r_0}}^T \end{bmatrix}, \quad (31)$$

where $\overline{U}_{0i_j}^T$ is the i_j th row vector of matrix \overline{U}_0 .

[26] 3. Compute the ‘‘condition number’’ C_{w_0} of matrix \overline{W}_0 . Unlike the traditional condition number, we define the condition number of matrix \overline{W}_0 as follows.

[27] Apply modified Gram-Schmidt algorithm [Golub and Van Loan, 1989] for factoring

$$\overline{W}_0 = \overline{Q}_0 \overline{R}_0, \quad (32)$$

where \overline{Q}_0 is the unitary matrix of dimension $r_0 \times r_0$; \overline{R}_0 is the upper triangular matrix of dimension $r_0 \times r_0$.

[28] The ‘‘condition number,’’ C_{w_0} of matrix \overline{W}_0 is the absolute value of the ratio of the first diagonal element to the last diagonal element in matrix \overline{R}_0 :

$$C_{w_0} = |R_{11}/R_{r_0 r_0}|, \quad (33)$$

where R_{11} is the first diagonal element of matrix \overline{R}_0 ; $R_{r_0 r_0}$ is the last diagonal element of matrix \overline{R}_0 .

[29] 4. If C_{w_0} is too large, such as greater than 1000, it indicates that matrix \overline{W}_0 is nearly singular. In other words, the column vectors of \overline{W}_0 are not totally independent. Let $r_1 = r_0 - 1$, skip back to step 1. Construct matrices \overline{U}_1 and \overline{W}_1 , and compute the condition number C_{w_1} of \overline{W}_1 . If C_{w_1} is moderate, such as dozens or hundreds, then r_1 is the rank of matrix \overline{A} ; otherwise let $r_2 = r_1 - 1$, skip to step 1 and do this procedure again, until find r_i . Build \overline{U}_i and \overline{W}_i . The condition number C_{w_i} of \overline{W}_i is moderate. Thus r_i is the rank of matrix \overline{A} .

[30] 5. If C_{w_0} is very small, such as less than 10, it means the column vectors of \overline{W}_0 are entirely independent. Let $r_1 = r_0 + 1$, skip back to step 1. Construct matrices \overline{U}_1 and \overline{W}_1 , and compute the condition number C_{w_1} of \overline{W}_1 . If C_{w_1} is moderate, then r_0 is the rank of matrix \overline{A} ; otherwise let $r_2 = r_1 + 1$, skip to step 1 again,

until find condition number C_{w_i} of matrix \overline{W}_i is moderate. Thus $r_i - 1$ is the rank of matrix \overline{A} .

[31] 6. Let the rank of matrix \overline{A} is r , matrix \overline{U} and \overline{W} is of dimension $n \times r$, and $r \times r$, respectively,

$$\overline{U} = [\overline{A}_{i_1} \overline{A}_{i_2} \cdots \overline{A}_{i_r}] \quad (34)$$

$$\overline{W} = \begin{bmatrix} \overline{U}_{i_1}^T \\ \overline{U}_{i_2}^T \\ \vdots \\ \overline{U}_{i_r}^T \end{bmatrix}, \quad (35)$$

where $i_j \in 1, 2, \dots, n, j = 1, 2, \dots, r$.

[32] An alternative method is discussed in the work of Tsang *et al.* [2004], where the rank is determined by coarse-coarse sampling and singular value decomposition (SVD).

[33] 7. Pick i_1 th, i_2 th, \dots, i_r th row vector of matrix \overline{A} to construct matrix \overline{V} of dimension $r \times n$:

$$\overline{V} = \begin{bmatrix} \overline{A}_{i_1}^T \\ \overline{A}_{i_2}^T \\ \vdots \\ \overline{A}_{i_r}^T \end{bmatrix}, \quad (36)$$

where $\overline{A}_{i_j}^T$ is the i_j th row vector of matrix \overline{A} .

[34] 8. Build matrix \overline{V} of dimension $r \times n$ in terms of matrix \overline{W} and \overline{V} .

$$\overline{V} = \overline{W}^{-1} \overline{V} \quad (37)$$

Then $\overline{A} = \overline{U} \overline{V}$.

[35] From the first-level, we build UV decomposition with sampling procedure to $\overline{Z}_{1,3,1}^{(1)}, \overline{Z}_{1,4,1}^{(1)}, \overline{Z}_{2,4,1}^{(1)}, \overline{Z}_{3,5,1}^{(1)}, \overline{Z}_{3,6,1}^{(1)}, \overline{Z}_{4,6,1}^{(1)}, \overline{Z}_{5,7,1}^{(1)}, \overline{Z}_{5,8,1}^{(1)}, \overline{Z}_{6,8,1}^{(1)}, \dots$; next the second level, do that of to $\overline{Z}_{1,3,2}^{(2)}, \overline{Z}_{1,4,2}^{(2)}, \overline{Z}_{2,4,2}^{(2)}, \overline{Z}_{3,5,2}^{(2)}, \overline{Z}_{3,6,2}^{(2)}, \overline{Z}_{4,6,2}^{(2)}, \overline{Z}_{5,7,2}^{(2)}, \overline{Z}_{5,8,2}^{(2)}, \overline{Z}_{6,8,2}^{(2)}, \dots$; until the p th level. Thus we can do matrix-vector product with it.

4. CPU Complexity and Memory Requirement for Decomposition of Matrix Column Multiplication

[36] Next, we illustrate the computational steps of the product of impedance matrix \overline{Z} and a column vector \overline{b} .

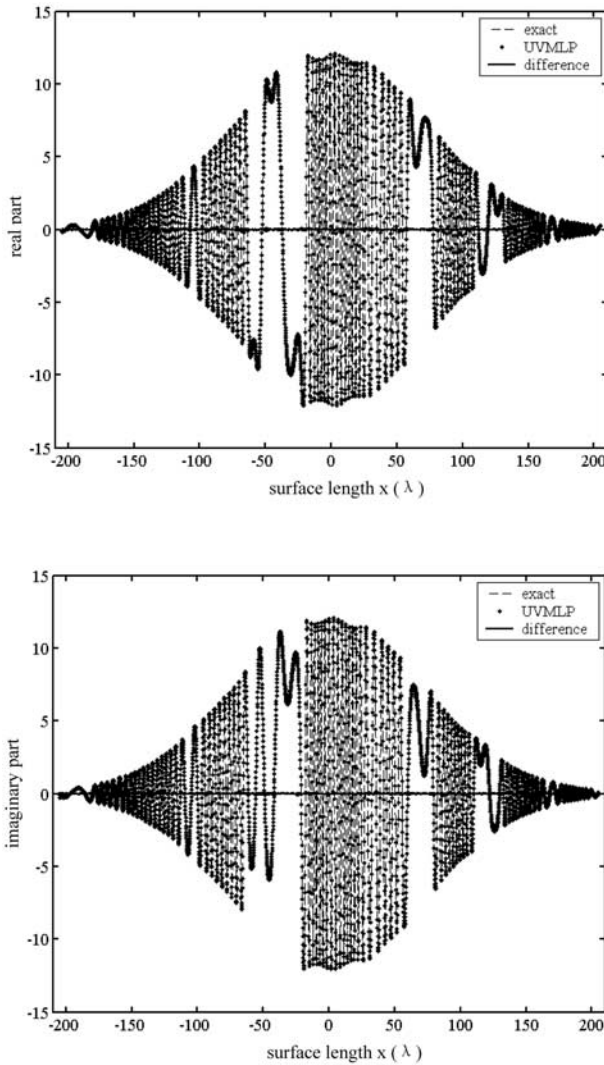


Figure 3. Comparison between UVMLP method and exact solution. Here $h = 2.0\lambda$, $l = 20.0\lambda$, $\Delta x = 0.1\lambda$, and $\theta_i = 11.46^\circ$. (top) Real part. (bottom) Imaginary part.

Generally, the rank is dependent on distance d , surface slope [Tsang *et al.*, 2001, equation 4.1.113] and block's size; for convenience, suppose each submatrix's rank of $r = \text{constant}$.

[37] 1. Compute $\bar{\bar{Z}}^{(0)}\bar{b}$ by direct multiplication. The number of computational steps is $3MN - 2M^2$. The memory for storing $\bar{\bar{Z}}^{(0)}$ is $3MN - 2M^2$.

[38] 2. Compute $\bar{\bar{Z}}_{1,3,1}^{(1)}\bar{b} = \bar{U}_{1,3,1}^{(1)}(\bar{V}_{1,3,1}^{(1)}\bar{b})$, where $\bar{U}_{1,3,1}^{(1)}$, $\bar{V}_{1,3,1}^{(1)}$ are the UV decomposition of $\bar{\bar{Z}}_{1,3,1}^{(1)}$, \bar{b} is the subvector of \bar{b} corresponds to the multiplication with $\bar{\bar{Z}}_{1,3,1}^{(1)}$. The number of computational steps is $rM + rM =$

$2rM$. The memory for storing $\bar{U}_{1,3,1}^{(1)}$ and $\bar{V}_{1,3,1}^{(1)}$ is $rM + rM = 2rM$.

[39] 3. Compute $\bar{\bar{Z}}_{1,4,1}^{(1)}\bar{b} = \bar{U}_{1,4,1}^{(1)}(\bar{V}_{1,4,1}^{(1)}\bar{b})$, where $\bar{U}_{1,4,1}^{(1)}$, $\bar{V}_{1,4,1}^{(1)}$ are the UV decomposition of $\bar{\bar{Z}}_{1,4,1}^{(1)}$, \bar{b} is the subvector of \bar{b} corresponds to the multiplication with $\bar{\bar{Z}}_{1,4,1}^{(1)}$. The number of computational steps is $rM + rM = 2rM$. The memory for storing $\bar{U}_{1,4,1}^{(1)}$ and $\bar{V}_{1,4,1}^{(1)}$ is $rM + rM = 2rM$.

[40] 4. Compute $\bar{\bar{Z}}_{2,4,1}^{(1)}\bar{b} = \bar{U}_{2,4,1}^{(1)}(\bar{V}_{2,4,1}^{(1)}\bar{b})$, where $\bar{U}_{2,4,1}^{(1)}$, $\bar{V}_{2,4,1}^{(1)}$ are the UV decomposition of $\bar{\bar{Z}}_{2,4,1}^{(1)}$, \bar{b} is the subvector of \bar{b} corresponds to the multiplication with $\bar{\bar{Z}}_{2,4,1}^{(1)}$. The number of computational steps is $rM + rM = 2rM$. The memory for storing $\bar{U}_{2,4,1}^{(1)}$ and $\bar{V}_{2,4,1}^{(1)}$ is $rM + rM = 2rM$.

[41] 5. For the first-level upper matrix $\bar{\bar{Z}}^{(1)U}$, there are $(L/2 - 1)$ submatrices similar to $\bar{\bar{Z}}_{1,3,1}^{(1)}$, $\bar{\bar{Z}}_{1,4,1}^{(1)}$ and $\bar{\bar{Z}}_{2,4,1}^{(1)}$, respectively. Thus the number of computational steps for $\bar{\bar{Z}}^{(1)U}\bar{b}$ is $(L/2 - 1)(2rM + 2rM + 2rM) = 3r(N - 2M)$. The memory requirement is $(L/2 - 1)(2rM + 2rM + 2rM) = 3r(N - 2M)$.

[42] 6. Compute the first-level lower matrix $\bar{\bar{Z}}^{(1)L}\bar{b}$. The number of computational steps is $(L/2 - 1)(2rM + 2rM + 2rM) = 3r(N - 2M)$. The memory requirement is $(L/2 - 1)(2rM + 2rM + 2rM) = 3r(N - 2M)$.

[43] 7. Compute $\bar{\bar{Z}}_{1,3,2}^{(2)}\bar{b} = \bar{U}_{1,3,2}^{(2)}(\bar{V}_{1,3,2}^{(2)}\bar{b})$. The number of computational steps is $2rM + 2rM = 4rM$. The memory for storing $\bar{U}_{1,3,2}^{(2)}$ and $\bar{V}_{1,3,2}^{(2)}$ is $2rM + 2rM = 4rM$.

[44] 8. Compute $\bar{\bar{Z}}_{1,4,2}^{(2)}\bar{b} = \bar{U}_{1,4,2}^{(2)}(\bar{V}_{1,4,2}^{(2)}\bar{b})$. The number of computational steps is $2rM + 2rM = 4rM$. The memory for storing $\bar{U}_{1,4,2}^{(2)}$ and $\bar{V}_{1,4,2}^{(2)}$ is $2rM + 2rM = 4rM$.

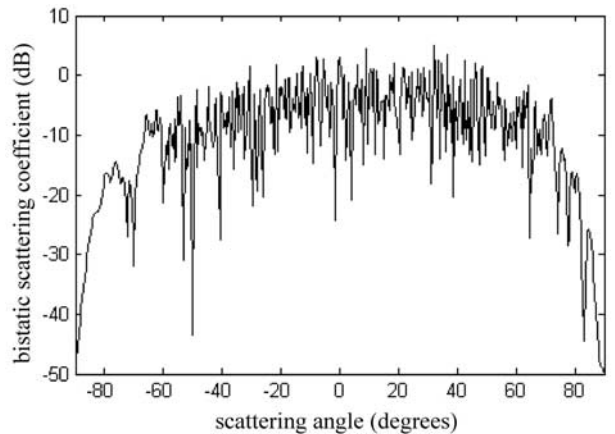


Figure 4. Bistatic scattering coefficient of the rough surface. Here $\theta_i = 11.46^\circ$.

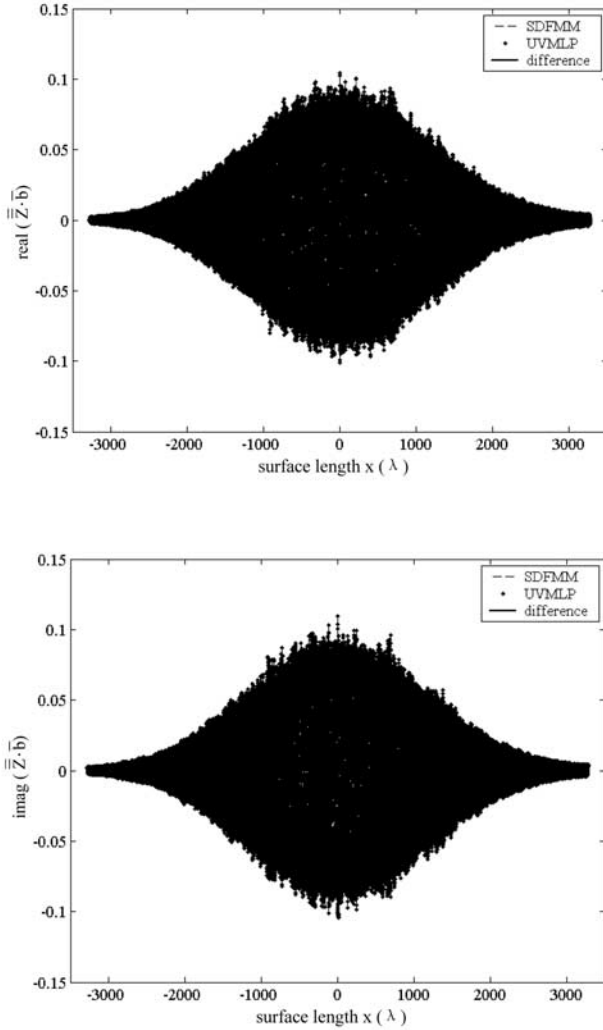


Figure 5. Comparison of one matrix-vector product using UVMLP and SDFMM. (top) Real part. (bottom) Imaginary part.

[45] 9. Compute $\overline{\overline{Z}}_{2,2,4_2}^{(2)} \overline{\overline{b}} = \overline{\overline{U}}_{2,2,4_2}^{(2)} (\overline{\overline{V}}_{2,2,4_2}^{(2)} \overline{\overline{b}})$. The number of computational steps is $2rM + 2rM = 4rM$. The memory for storing $\overline{\overline{U}}_{2,2,4_2}^{(2)}$ and $\overline{\overline{V}}_{2,2,4_2}^{(2)}$ is $2rM + 2rM = 4rM$.

[46] 10. For the second-level upper matrix $\overline{\overline{Z}}_{2,2,4_2}^{(2)}$, there are $(L/2^2 - 1)$ submatrices similar to $\overline{\overline{Z}}_{1,2,3_2}^{(2)}$, $\overline{\overline{Z}}_{1,2,4_2}^{(2)}$ and $\overline{\overline{Z}}_{2,2,4_2}^{(2)}$, respectively. Thus the number of computational steps for $\overline{\overline{Z}}_{2,2,4_2}^{(2)} \overline{\overline{b}}$ is $(L/2^2 - 1)(4rM + 4rM + 4rM) = 3r(N - 4M)$. The memory requirement is $(L/2^2 - 1)(4rM + 4rM + 4rM) = 3r(N - 4M)$.

[47] 11. Compute the second-level lower matrix $\overline{\overline{Z}}_{2,2,4_2}^{(2)L} \overline{\overline{b}}$. The number of computational steps is $(L/2^2 - 1)(4rM + 4rM + 4rM) = 3r(N - 4M)$. The memory requirement is $(L/2^2 - 1)(4rM + 4rM + 4rM) = 3r(N - 4M)$.

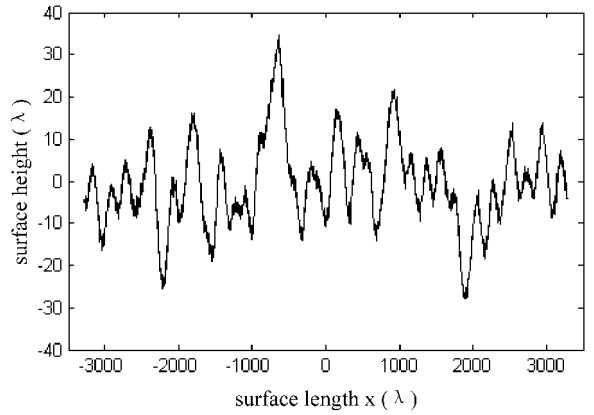


Figure 6. A single realization of a composite rough surface profile. $\Delta x = 0.1\lambda$ and $N = 65,536$.

[48] 12. With the similar manner, compute $\overline{\overline{Z}}_{1,3_i}^{(i)} \overline{\overline{b}} = \overline{\overline{U}}_{1,3_i}^{(i)} (\overline{\overline{V}}_{1,3_i}^{(i)} \overline{\overline{b}})$ in the i th-level upper matrix. The number of computational steps is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$. The memory requirement is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$.

[49] 13. Compute $\overline{\overline{Z}}_{1,4_i}^{(i)} \overline{\overline{b}} = \overline{\overline{U}}_{1,4_i}^{(i)} (\overline{\overline{V}}_{1,4_i}^{(i)} \overline{\overline{b}})$. The number of computational steps is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$. The memory requirement is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$.

[50] 14. Compute $\overline{\overline{Z}}_{2,4_i}^{(i)} \overline{\overline{b}} = \overline{\overline{U}}_{2,4_i}^{(i)} (\overline{\overline{V}}_{2,4_i}^{(i)} \overline{\overline{b}})$. The number of computational steps is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$. The memory requirement is $r(2^{i-1}M) + r(2^{i-1}M) = 2^i rM$.

[51] 15. There are $(L/2^i - 1)$ submatrices similar to $\overline{\overline{Z}}_{1,3_i}^{(i)}$, $\overline{\overline{Z}}_{1,4_i}^{(i)}$ and $\overline{\overline{Z}}_{2,4_i}^{(i)}$, respectively. Thus the number of computational steps for $\overline{\overline{Z}}_{2,4_i}^{(i)} \overline{\overline{b}}$ is $(L/2^i - 1)(2^i rM + 2^i rM +$

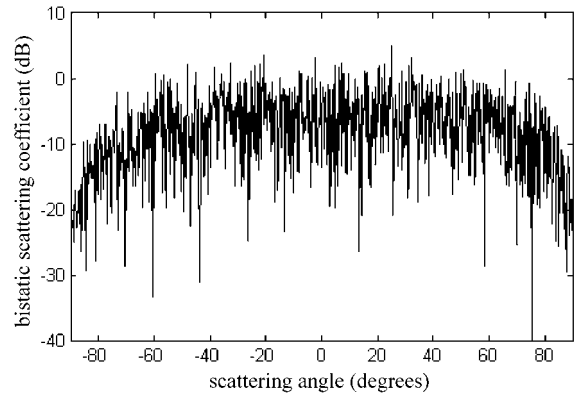


Figure 7. Bistatic scattering coefficient for one realization, using the composite rough surface as shown in Figure 6. Here $\theta_i = 11.46^\circ$.

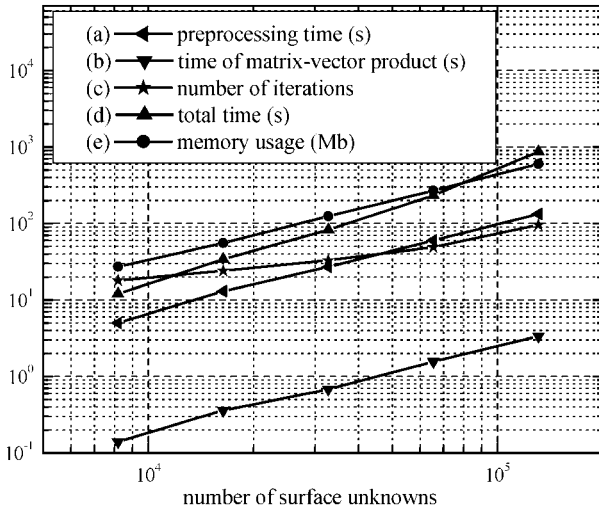


Figure 8. Some variables as a function of number of surface unknowns. (a) Preprocessing time of UV decomposition with sampling procedure. (b) Matrix-vector product time of UV decomposition. (c) Number of iterations. (d) Total CPU time to solve the matrix equation for the induced surface current. (e) Memory usage for storing the UV decomposition.

$2^i r M) = 3r(N - 2^i M)$. The memory requirement is $(L/2^i - 1)(2^i r M + 2^i r M) = 3r(N - 2^i M)$.

[52] 16. Compute $\bar{Z}^{(i)L} \bar{b}$. The number of computational steps is $(L/2^i - 1)(2^i r M + 2^i r M + 2^i r M) = 3r(N - 2^i M)$. The memory requirement is $(L/2^i - 1)(2^i r M + 2^i r M + 2^i r M) = 3r(N - 2^i M)$. The total number of steps is

$$\begin{aligned} N_{total} &= (3MN - 2M^2) + 2 \sum_{i=1}^p 3r(N - 2^i M) \\ &= (3MN - 2M^2) + 6rpN - 12rM(2^p - 1) \\ &= M(3N - 2M + 12r) + 6rN[\log_2(N/M) - 2]. \end{aligned} \quad (38)$$

Table 3. Rank Table of UVMLP^a

Level i	Upper Matrix		Lower Matrix	
	Maximum Rank in Level i	Minimum Rank in Level i	Maximum Rank in Level i	Minimum Rank in Level i
1	3	3	3	3
2	3	3	3	3
3	3	3	3	3
4	3	3	3	3
5	4	3	4	3
6	6	3	6	3
7	6	3	6	3
8	6	3	6	3
9	5	3	5	3
10	4	3	4	3
11	3	3	3	3
12	4	3	4	3

^aSimulation parameters are: $h = 10.0\lambda$, $l = 100.0\lambda$, $\Delta x = 0.1\lambda$, and $\theta_i = 11.46^\circ$. The number of elements in the first-level group is $M = 16$.

Since $M, r \ll N$, the total number of steps is of $O(N \log_2 N)$. The total memory requirement is

$$\begin{aligned} Memory_{total} &= (3MN - 2M^2) + 2 \sum_{i=1}^p 3r(N - 2^i M) \\ &= M(3N - 2M + 12r) \\ &\quad + 6rN[\log_2(N/M) - 2]. \end{aligned}$$

It is also of $O(N \log_2 N)$.

5. Results and Discussion

[53] In this section, we illustrate results of scattering from random rough surfaces with Gaussian correlation function. All cases are based on TE case except that Figure 9 (bottom) is TM case. First, we illustrate a case of RMS height $h = 2.0\lambda$, correlation length $l = 20.0\lambda$, incident angle $\theta_i = 11.46^\circ = 0.2rad$; $\Delta x = \lambda/10$, the number of unknowns $N = 4096$, and the surface length

Table 2. Comparison of CPU Time and Memory Usage for Different Number of Unknowns^a

Number of Unknowns	Preprocessing Time, s	Time per Matrix Vector Product, s	Time per Iteration, s	Number of Iterations	Total Time, s	Memory Usage, Mbytes
8192	5	0.14	0.33	18	12	27.2
16,384	13	0.36	0.75	24	34	55.6
32,768	27	0.68	1.51	33	83	124.4
65,536	60	1.56	3.26	49	233	268.9
131,072	133	3.36	7.41	96	870	602.0

^aSimulation parameters are: $h = 10.0\lambda$, $l = 100.0\lambda$, $\Delta x = 0.1\lambda$, and $\theta_i = 11.46^\circ$.

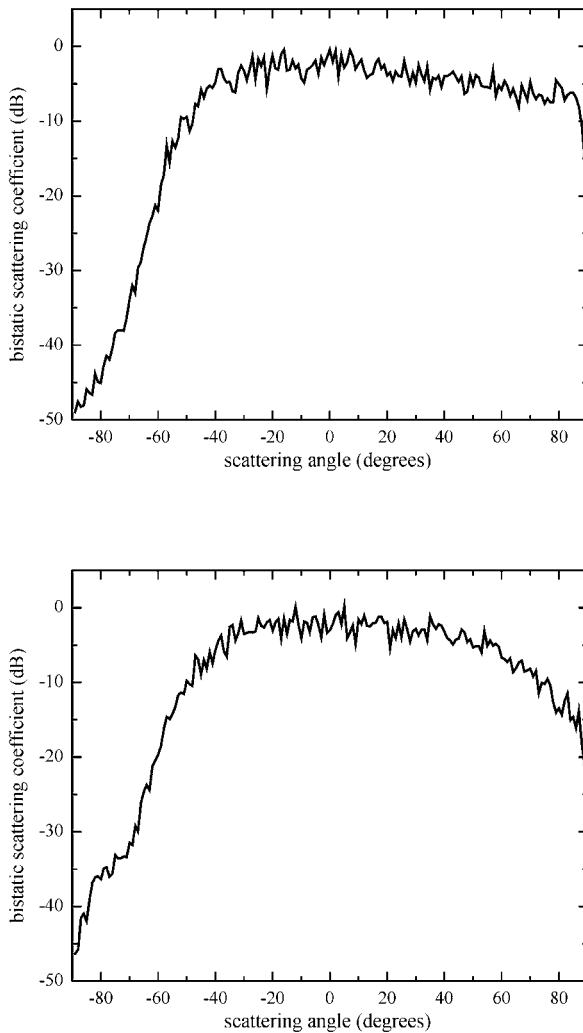


Figure 9. Bistatic scattering coefficient from the rough surface of 20 realizations at $\theta_i = 80^\circ$ with $h = 3.0\lambda$, $l = 6.0\lambda$, and $L = 6,553.6\lambda$. (top) TE polarization. (bottom) TM polarization.

$L = 410\lambda$. We apply Conjugate Gradient method (CGM) as the iterative procedure. We use a single processor of 2.4 GHz and 1 G RAM memory. To show the validity of our method, we compare the results with that of using equation (3). Figure 3 illustrates the comparison of results of surface unknowns solved by UVMLP and the original impedance matrix. These figures show that the results of UVMLP method are in a good agreement with the exact solution. We also show the bistatic scattering in Figure 4.

[54] Next we compare the result of matrix-vector product $\bar{c} = \bar{Z}\bar{b}$ between UVMLP and SDFMM, where \bar{Z} is the impedance matrix, \bar{b} is the incident wave vector,

as shown in equation (3). The parameters are $h = 0.1\lambda$, $l = 1.0\lambda$, $\Delta x = \lambda/10$ with $N = 65536$. The surface length L is more than 6500λ . The incidence angle is $\theta_i = 11.46^\circ$. Figure 5 illustrates the matrix-vector product results between UVMLP and SDFMM. The two results are in a good agreement.

[55] Figure 6 shows a composite rough surface profile. The composite rough surface has a small-scale roughness ($h_1 = 1.0\lambda$ and $l_1 = 3.0\lambda$) superimposed on a large-scale roughness ($h_2 = 10.0\lambda$ and $l_2 = 100.0\lambda$). $N = 65536$, $\Delta x = \lambda/10$, $\theta_i = 11.46^\circ$. Figure 7 shows the bistatic scattering coefficient for one realization, using the composite rough surface as shown in Figure 6.

[56] In Figure 8, we illustrate the CPU time of UV decomposition preprocessing, matrix-vector product, the number of iterations, the total CPU time and the memory requirement as a function of number of surface unknowns. For these examples, the incident angle is $\theta_i = 11.46^\circ$ with $h = 10.0\lambda$, $l = 100.0\lambda$, $\Delta x = \lambda/10$. The number of unknowns, N , ranges from 8192 to 131072. We list in Table 2 the CPU and memory data of Figure 8 for convenience.

[57] Consider the case with $h = 10.0\lambda$, $l = 100.0\lambda$, $\Delta x = \lambda/10$. Incident angle is $\theta_i = 11.46^\circ$. $N = 131072$ and L is more than 13100λ . The total solution time is 14.5 min for 96 iterations. The preprocessing costs 133 s and it takes about 7.4 s per iteration. Table 3 illustrates the maximum and minimum rank of the blocks in each level. We see that the rank is low and is roughly of same order for all the levels of impedance matrices. We note that the case from 65536 unknowns to 131,072 unknowns has a total CPU close to 4 times because the number of iterations has doubled. In the future, we will study the reduction of iterations by near-field preconditioning [Huang et al., 2004].

[58] We next study the case of large incidence angle and the case of TE and TM polarizations. In Figure 9, we illustrate the bistatic scattering coefficients of a TE case and a TM case, respectively, from a random rough surface for $\theta_i = 80^\circ$, $h = 3.0\lambda$, $l = 6.0\lambda$, $\Delta x = \lambda/10$ with $N = 65536$. The results are averaged over 20 realizations. The average CPU times per iteration are 5.11 and 5.33 s, and the average numbers of conjugate gradient iterations are 146 and 143 for the TE and TM case, respectively.

Table 4. Average CPU Time Over 20 Realizations at $\theta_i = 80^\circ$ With $h = 3.0\lambda$, $l = 6.0\lambda$, and $L = 6553.6\lambda^a$

Polarization	Preprocessing Time, s		Time Per Iteration, s	Number of Iterations	Total Time, s
	Near Field	UV Decomposition			
TE	12	82	5.11	146	840
TM	13	85	5.33	143	960

^aThe tolerances of residuals (L^2 norm) are less than 1%.

Table 5. Rank Table of a Realization at $\theta_i = 80^\circ$ With $h = 3.0\lambda$, $l = 6.0\lambda$, and $L = 6553.6\lambda$

Level	TE				TM			
	Upper Matrix		Lower Matrix		Upper Matrix		Lower Matrix	
	i	rank _{max}	rank _{min}	rank _{max}	rank _{min}	rank _{max}	rank _{min}	rank _{max}
1	7	3	7	3	7	3	7	3
2	10	3	10	3	11	3	11	3
3	14	3	14	3	11	4	11	4
4	10	3	10	3	11	4	10	4
5	8	4	8	4	9	4	8	4
6	6	4	6	4	7	4	7	4
7	6	4	6	4	6	4	6	4
8	5	4	5	4	5	4	5	4
9	5	4	5	4	5	4	5	3
10	5	4	5	4	5	4	5	4
11	6	5	6	5	5	3	5	5

The CPU are shown in Table 4. The tolerances of residuals (L^2 norm) are less than 1%. There are only slight difference of CPU and number of iterations from realization to realization. Comparing with the case of $h = 10.0\lambda$, $l = 100.0\lambda$, we find that the CPU per iteration is more, because the surface is rougher and the slope is bigger. The rank table is given in Table 5 and shown an increase in rank. We have also compared between \bar{b} and the direct product of original Z_{mn} and our solution W_n in (3). Both are in a good agreement. For cases of moderate RMS height, as done in Figure 5, we have made computations using the SDFMM code we have written using page 212 to 242 of [Tsang et al., 2001]. Since this is based on our own version of SDFMM, the CPU comparison with UV may not be a fair comparison. Within this context of uncertainty, the present UV method is substantially faster. Also, for the case of large RMS height as done in Figure 9, we cannot run the SDFMM code because of the large memory requirement in near field for a single processor.

6. Conclusions

[59] The UVMLP method is presented for rapid solution of integral equation. It achieves the reduction of the CPU and memory requirement for matrix-vector multiplication. On the other hand, the forward and backward method [Chou and Johnson, 1998] emphasizes on reduction of number of iterations. A recent method of characteristic basis functions [Sun et al., 2003] emphasizes on the drastic reduction in basis functions. In addition, the UVMLP method operates on submatrixes of the impedance matrix using tabulated values of Green's values, thus it is applicable to all types of Green's functions. The method has also been applied to volume scattering by large number of cylinders of moderate size where the UV

method is directly applied to the Green's function of higher-order partial waves [Tsang and Li, 2004].

Appendix A: Upper Bound of Rank From Green's Function Analysis

[60] Let the transmitting points of a group be at x_n , $n = 1, 2, \dots, N_t$ and the receiving points of a group be at x_m , $m = 1, 2, \dots, N_r$. Without loss of generality, let $x_m > x_n$.

[61] Then, using the steepest descent Green's function [Tsang et al., 2001, p. 214], $g(\bar{r}_m, \bar{r}_n)$ is a $N_r \times N_t$ matrix, and

$$g(\bar{r}_m, \bar{r}_n) = \frac{i}{4\pi} \sum_{q=1}^Q \exp(ik(x_m - x_n) \sin \alpha_q - ik(z_m - z_n) \cos \alpha_q) \Delta \alpha_q, \quad (A1)$$

where α_q is the discretization of the steepest descent path, and Q is the number of discretization angles. The choice of Q is based on the rule defined for steepest path.

[62] We can define the m th element of the n th column of the $g(\bar{r}_m, \bar{r}_n)$ matrix by

$$(\bar{g}_n)_m = g(\bar{r}_m, \bar{r}_n). \quad (A2)$$

[63] Let the q th column vector \bar{a}_q be defined such that its m th element (i.e., the m th row) of the vector \bar{a}_q is

$$(\bar{a}_q)_m = \frac{i}{4\pi} \exp(ikx_m \sin \alpha_q - ikz_m \cos \alpha_q) \Delta \alpha_q. \quad (A3)$$

[64] Then, we have

$$(\bar{g}_n)_m = \frac{i}{4\pi} \sum_{q=1}^Q \exp(-ikx_n \sin \alpha_q + ikz_n \cos \alpha_q) (\bar{a}_q)_m. \quad (A4)$$

Thus

$$\bar{g}_n = \sum_{q=1}^Q C_{nq} \bar{a}_q, \quad (A5)$$

where

$$C_{nq} = \exp(-ikx_n \sin \alpha_q + ikz_n \cos \alpha_q). \quad (A6)$$

[65] This means the columns of the impedance matrix block, $g(\bar{r}_m, \bar{r}_n)$, can be expressed as linear combination of Q column vectors. This puts the upper bound of the rank of $g(\bar{r}_m, \bar{r}_n)$ at Q . On the other hand, the rank table in Table 3 shows the rank is less than Q .

[66] **Acknowledgment.** The research in this paper was supported by the City University of Hong Kong research grant 9380034 and Hong Kong RGC Competitive Earmarked Research Grant (CERG) 9040715 and RGC central allocation grant 8730017.

References

- Chan, C. H., and L. Tsang (1995), A sparse matrix canonical grid method for scattering by many scatterers, *Microwave Opt. Technol. Lett.*, 8, 114–118.
- Chou, H. T., and J. T. Johnson (1998), A novel acceleration algorithm for the computation of scattering from rough surfaces with the forward-backward method, *Radio Sci.*, 33, 1277–1287.
- Golub, G. H., and C. F. Van Loan (1989), *Matrix Computations*, 2nd ed., Johns Hopkins Univ. Press, Baltimore, Md.
- Huang, C. C., L. Tsang, C. H. Chan, and K. H. Ding (2004), Multiple scattering among vias in planar waveguides using pre-conditioned SMCG method, *IEEE Trans. Microwave Theory Tech.*, 52(1), 20–28.
- Jandhyala, V., B. Shanker, E. Michielssen, and W. C. Chew (1998), A fast algorithm for the analysis of scattering by dielectric rough surfaces, *J. Opt. Soc. Am. A*, 15, 1877–1885.
- Kapur, S., and D. E. Long (1997), IES3: A fast integral equation solver for efficient 3-dimensional extraction, in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 448–455, Inst. of Electr. and Electron. Eng., London.
- Li, S. Q., C. H. Chan, L. Tsang, Q. Li, and L. Zhou (2000), Parallel implementation of the sparse matrix/canonical grid method for the analysis of two-dimensional random rough surface (three-dimensional scattering problem) on a Beowulf system, *IEEE Trans. Geosci. Remote Sens.*, 38, 1600–1608.
- Michielssen, E., and W. C. Chew (1996), The fast steepest descent algorithm for analyzing scattering from two dimensional objects, *Radio Sci.*, 31, 1215–1224.
- Rohklin, V. (1990), Rapid solution of integral equations of scattering theory in two dimensions, *J. Comput. Phys.*, 36, 414–439.
- Sun, Y. F., C. H. Chan, R. Mittra, and L. Tsang (2003), Characteristic basis functions for solving large-scale problem in dense medium scattering, *IEEE Int. Symp. Antennas Propag.*, 1, 22–27.
- Tsang, L., and Q. Li (2004), Wave scattering with UV multilevel partitioning method: Volume scattering by discrete scatterers, *Microw. Opt. Tech. Lett.*, 41(5), 354–361.
- Tsang, L., C. H. Chan, K. Pak, and H. Sangani (1995), Monte Carlo simulations of large scale problems of random rough surface scattering and applications to grazing incidence with the BMIA/Canonical Grid Method, *IEEE Trans. Antennas Propag.*, 43, 851–859.
- Tsang, L., J. A. Kong, K. H. Ding, and C. O. Ao (2001) *Scattering of Electromagnetic Waves*, vol. 2, *Numerical Simulations*, Wiley Intersci., Hoboken, N. J.
- Tsang, L., Q. Li, P. Xu, D. Chen, and V. Jandhyala (2004), Wave scattering with UV multilevel partitioning method: 2. Three-dimensional problem of nonpenetrable surface scattering, *Radio Sci.*, 39, RS5011, doi:10.1029/2003RS003010.

D. Chen, L. Tsang, and P. Xu, Department of Electronic Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong, China. (eeltsang@cityu.edu.hk)

V. Jandhyala and Q. Li, Department of Electrical Engineering, Box 352500, University of Washington, Seattle, WA 98195-2500, USA.