

# Scattering by rough surface using a hybrid technique combining the multilevel UV method with the sparse matrix canonical grid method

Peng Xu and Leung Tsang<sup>1</sup>

Wireless Communications Research Centre, City University of Hong Kong, Kowloon, Hong Kong

Received 5 January 2005; revised 13 April 2005; accepted 26 April 2005; published 19 August 2005.

[1] A procedure is developed for combining efficiently the multilevel UV method with the sparse matrix canonical grid (SMCG) method in the computation of three-dimensional (3-D) wave scattering from random rough surface. It handles near- and intermediate-field interactions by the multilevel UV method and far-field interactions by the SMCG method. This hybrid UV/SMCG method removes the large memory requirement of both the UV method in the far field and the SMCG method in the near field. The computational complexity for the hybrid method is  $O(N \log N)$ . The tradeoffs in computer memory requirement and CPU time between the UV part and the SMCG part are examined for Gaussian random rough surfaces with Gaussian correlation function and with exponential correlation function. For a surface area of  $44 \times 44$  square wavelengths with RMS of 1 wavelength and 123,904 surface unknowns, the UV/SMCG method requires CPU of 52.2 s per iteration with total CPU of 46.6 min for 22 iterations on a single processor of CPU speed of 2.66 GHz.

**Citation:** Xu, P., and L. Tsang (2005), Scattering by rough surface using a hybrid technique combining the multilevel UV method with the sparse matrix canonical grid method, *Radio Sci.*, 40, RS4012, doi:10.1029/2005RS003242.

## 1. Introduction

[2] The solutions of wave equations for rough surface scattering can be calculated by using integral equations with the method of moments (MoM). For such problems, the common methodology is to use iterative solvers such as the conjugate gradient (CG) method. The bottleneck is to calculate the matrix-vector multiplication when analyzing large-scale scattering problems. Several fast numerical algorithms have been developed to accelerate such matrix-vector multiplication. They are the sparse matrix canonical grid method (SMCG) [Tsang *et al.*, 1993, 1994; Johnson, 1998], the fast multipole method (FMM) [Wagner *et al.*, 1997], the forward and backward method with an acceleration scheme [Torrungrueng *et al.*, 2000; Torrungrueng and Johnson, 2001], the multilevel UV method [Tsang *et al.*, 2004; Tsang and Li, 2004; Xu and Tsang, 2004], etc. In these methods, the computational complexity and the

memory requirement are reduced to  $O(N \log N)$  or  $O(N)$ . The SMCG and the FMM have been extended to the case of 3-D problem with penetrable surfaces. However, the method of spectral acceleration has not been extended to 3-D problem of surfaces with moderate dielectric constant.

[3] In the SMCG, the interaction between two points on the surface is distinguished as either strong (near) interaction or weak (far) interaction. The near interactions among the source points and field points are computed using the direct multiplication. The far interactions are computed by the fast Fourier transform (FFT) through a Taylor series expansion of the Green's function about a flat surface. For a moderate neighborhood distance, the near interactions impedance matrix has large memory requirement. When the RMS height of the rough surface increases, the number of terms in the Taylor series expansion increases, and the SMCG becomes inefficient. Li *et al.* [2001] made improvements to the SMCG method with a multilevel expansion of the SMCG method. It can reduce near interactions by having a smaller neighborhood distance. It reduces the number of FFT operations at the expense of replacing the 2-D FFTs by 3-D FFTs. However, its memory requirement and CPU time in far interactions are much

<sup>1</sup>Also at Department of Electrical Engineering, University of Washington, Seattle, Washington, USA.

more than that of the original SMCG. It also results in slow convergence as it uses a small neighborhood distance.

[4] Recently, the singular value decomposition (SVD) type algorithm [Kapur and Long, 1997] has attracted considerable attention. Implementations have been made on single level [Burkholder and Lee, 2004], or multilevel [Tsang et al., 2004; Tsang and Li, 2004; Xu and Tsang, 2004; Gope and Jandhyala, 2004; Wang et al., 2005]. The QR decomposition method [Kapur and Long, 1997] uses directly the SVD to compress each block of the original impedance matrix. However, it becomes inefficient when the block's size increases. In the multilevel UV method [Tsang et al., 2004; Tsang and Li, 2004; Xu and Tsang, 2004], we have used fast coarse-coarse sampling to search the rank of each block, and construct the matrices U and V using interpolation. Generally, the rank increases with the size of the block and the slope of the surface and decreases with the separation distance between the transmitting group and the receiving group associated with the block. In the 2-D problem the block's size at the  $i$ th level is twice of that at the  $(i - 1)$ th level, and the separation distance at the  $i$ th level is twice of that at the  $(i - 1)$ th level too. Then the effects on rank of the size and the separation distance of each multilevel block can cancel out, and the ranks at different levels are roughly of the same order [Xu and Tsang, 2004]. However, in 3-D problem the block's size at the  $i$ th level is four times of that at the  $(i - 1)$ th level, and the separation distance remains twice. Hence the rank will increase with the increase of level number. The memory requirement at higher levels becomes larger in spite of considerable compression.

[5] In this paper, we combine the multilevel UV method with the SMCG method by calculating the near and intermediate interactions with the UV method and the far interactions with the SMCG method. By using the UV method, we can have a larger neighborhood distance. The memory requirement and the computational steps of each block with dimension of  $D_i \times D_i$  at the  $i$ th level can be decreased from  $D_i^2$  to  $2r_i D_i$ , where  $r_i$  is the block's rank. The far interactions can be computed by the SMCG method with a small number of Taylor series terms. As the SMCG method requires only a translational invariant Green's function, the memory requirement associated with the far interactions is small, and the computational steps of each block in flat surface matrix are of the order of  $(2D_i \log D_i^2 + 4D_i)$ , which is less than  $2r_i D_i$  at a higher level. The memory requirement of the hybrid UV/SMCG method is less than that of the UV method or of the SMCG method. This means that a larger surface area problem with a larger RMS height can be solved by the hybrid UV/SMCG method.

[6] In section 2, the formulation of the problem of wave scattering impinging upon a rough surface is

given. In section 3, the UV/SMCG procedure is presented. In section 4, the computational complexity and the memory requirement are derived. In section 5, numerical results are illustrated for Gaussian random rough surfaces with Gaussian correlation function and with exponential correlation function. Results are compared with the small perturbation method (SPM) and the Kirchhoff approximation (KA). It is shown that for exponential correlation function, the KA is neither applicable at low frequency nor at high frequency. The tradeoffs in memory requirement and CPU time between the UV part and the SMCG part are examined. For a surface area of  $44 \times 44$  square wavelengths with RMS of 1 wavelength and 123,904 surface unknowns, the hybrid UV/SMCG method requires CPU of 52.2 s per iteration with total CPU of 46.6 minutes for 22 iterations on a single processor of CPU speed of 2.66 GHz. The total CPU includes the onetime UV decomposition. The memory requirement is 1,550 MB. All the simulation results of this paper are based on a single processor.

## 2. Wave-Scattering Formulation

[7] Consider a 3-D problem of a tapered scalar plane wave  $\psi_{\text{inc}}(x, y, z)$  [Tsang et al., 2001, equation 6.1.1] impinging upon a 2-D random rough surface with Dirichlet boundary condition. The surface is confined to the surface area  $L_x \times L_y$ , and is with a height profile  $z = f(x, y)$ . Let  $\vec{r}' = \hat{x}x' + \hat{y}y' + \hat{z}f(x', y')$  denote a field point and  $\vec{r} = \hat{x}x + \hat{y}y + \hat{z}f(x, y)$  denote a source point on the rough surface. Then the surface integral equation is [Tsang et al., 2001, equation 6.1.5]

$$\psi_{\text{inc}}(\vec{r}') = \iint dx dy G_0(x, y, f(x, y); x', y', f(x', y')) \cdot U(x, y), \quad (1)$$

where  $G_0 = \frac{\exp(ik|\vec{r} - \vec{r}'|)}{4\pi|\vec{r} - \vec{r}'|}$  is the free space Green's function and the unknown surface field is  $U(x, y)$ .

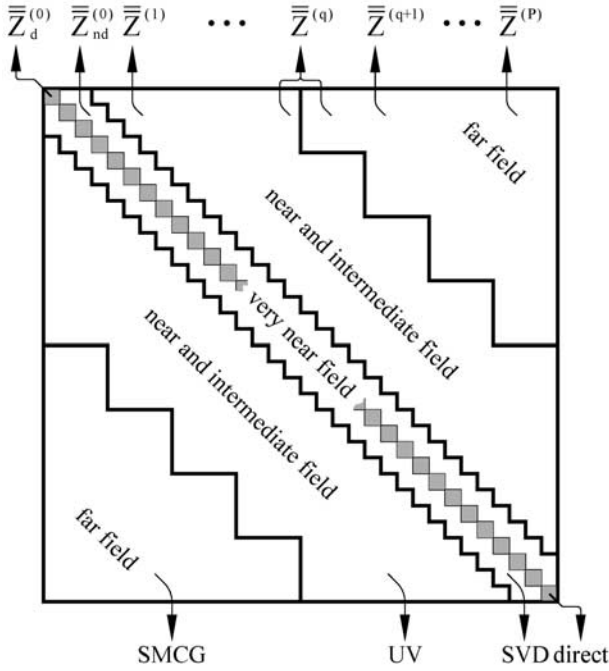
[8] The MoM is used to discretize the integral equation. We use the pulse basis function and point-matching method. The resulting matrix equation is

$$\overline{\overline{Z}} \cdot \overline{\overline{u}} = \overline{\overline{b}}. \quad (2)$$

Unless stated otherwise, we have used 64 points per square wavelength in the discretization.

## 3. Hybrid UV/SMCG Method Descriptions

[9] In the hybrid UV/SMCG method, the multilevel partitioning and decomposition into very near, near and



**Figure 1.** Multilevel structure of the hybrid UV/SMCG method.

intermediate, and far field are illustrated in Figure 1. As in the previous paper [Tsang et al., 2004], the matrix  $\bar{\bar{Z}}$  is decomposed into the sum of  $(P + 1)$  matrices

$$\bar{\bar{Z}} = \bar{\bar{Z}}^{(0)} + \bar{\bar{Z}}^{(1)} + \bar{\bar{Z}}^{(2)} + \dots + \bar{\bar{Z}}^{(P)}. \quad (3)$$

Matrix  $\bar{\bar{Z}}^{(0)}$  includes all the interactions among neighboring groups (including self group) at the 1st level. Matrix  $\bar{\bar{Z}}^{(1)}$  includes all the interactions among neighboring groups at the 2nd level excluding those that have been included by  $\bar{\bar{Z}}^{(0)}$ . Similarly, Matrix  $\bar{\bar{Z}}^{(i)}$  includes all the interactions among neighboring groups at the  $(i + 1)$ th level excluding those that have been included by  $\bar{\bar{Z}}^{(i-1)}$ . Let the size of the 1st level group be  $\sqrt{M} \times \sqrt{M}$  in  $x$  and  $y$  directions, respectively, and the block  $\bar{\bar{Z}}_{m_i n_i}^{(i)}$  in  $\bar{\bar{Z}}^{(i)}$  represent the interactions between two nonneighbor groups  $m_i$  and  $n_i$  of the level  $i$ , then

$\bar{\bar{Z}}_{m_1 n_1}^{(0)}$  is with dimensions of  $M \times M$ ;

$\bar{\bar{Z}}_{m_1 n_1}^{(1)}$  is with dimensions of  $M \times M$ ;

$\bar{\bar{Z}}_{m_2 n_2}^{(2)}$  is with dimensions of  $4M \times 4M$ ;

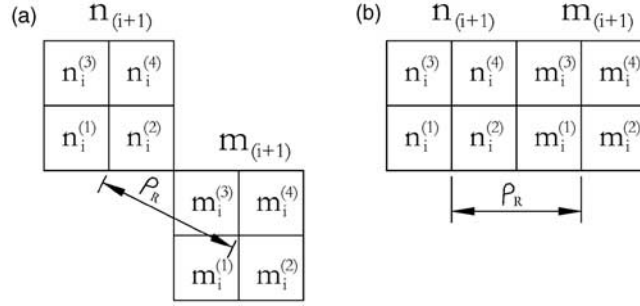
and  $\bar{\bar{Z}}_{m_i n_i}^{(i)}$  is with dimensions of  $4^{(i-1)}M \times 4^{(i-1)}M$ .

The total number of unknowns for the surface scattering problem is

$$N = \left(2^{(P+1)}\sqrt{M}\right)^2. \quad (4)$$

The surface area  $L_x \times L_y$  is generated in a square area with  $2^{(P+1)}\sqrt{M}$  points in  $x$  direction and  $2^{(P+1)}\sqrt{M}$  points in  $y$  direction.

[10] As shown in Figure 1 the impedance matrix  $\bar{\bar{Z}}$  is divided into three parts: (1) very near field interactions of separation distance of about  $\sqrt{M} \cdot \Delta x$  corresponding to  $\bar{\bar{Z}}^{(0)}$ , (2) near- and intermediate-field interactions, and (3) far-field interactions. The tradeoff between intermediate field and far field is dependent on the RMS height. In the SMCG method, the strong and weak interactions are distinguished by the neighborhood distance  $r_d$ . If the horizontal separation  $\rho_R$  between the source and field points is such that  $\rho_R < r_d$ , the interaction is defined as a strong interaction. Outside the neighborhood distance, the interaction is labeled “weak”. However, in the hybrid UV/SMCG method, we operate on the group-group interactions. Then we define  $\rho_R$  as the minimum distance between the two groups as shown in Figure 2. There are two kinds of neighbor groups [Tsang et al., 2004; Tsang and Li, 2004]. One kind is sharing only one common point (Figure 2a). The other kind is sharing one common edge (Figure 2b). Note that in Figure 2, there are four  $i$ th-level subgroups in each  $(i + 1)$ th-level group, the block  $\bar{\bar{Z}}_{m_i n_i}^{(i)}$  in  $\bar{\bar{Z}}^{(i)}$  are interactions between two nonneighbor  $i$ th-level groups  $m_i$  and  $n_i$ , and the groups  $m_i$  and  $n_i$  belong to two neighbor  $(i + 1)$ th-level groups  $m_{(i+1)}$  and  $n_{(i+1)}$ , respectively. If the groups  $m_i$  and  $n_i$  are neighbor such as  $m_i^{(3)}$  and  $n_i^{(2)}$  shown in Figure 2, their interactions are not belong to  $\bar{\bar{Z}}^{(i)}$ . In Figure 2 we show a  $\rho_R$  between two non-neighbor groups  $m_i^{(2)}$  and  $n_i^{(1)}$  only as an example. When  $\rho_R < r_d$ , the blocks are defined as in the near- and intermediate-field interactions, and they are compressed by the UV method. Otherwise, they are defined as in the far field, and they are expanded in a Taylor’s series and transformed into 2-D Toeplitz matrices. The block-vector multiplication is then calculated by 2-D FFT. We select  $r_d = 2(2^{(q-1)}\sqrt{M})\Delta x$  to be much greater than the RMS height. Then all the impedance blocks between groups in  $\bar{\bar{Z}}^{(1)}, \bar{\bar{Z}}^{(2)}, \dots, \bar{\bar{Z}}^{(q-1)}$  are calculated by the UV method. All the impedance blocks between groups in  $\bar{\bar{Z}}^{(q+1)}, \dots, \bar{\bar{Z}}^{(P)}$  are calculated by the SMCG method. The impedance blocks between groups in  $\bar{\bar{Z}}^{(q)}$  are partly calculated by the UV method, and partly computed by the SMCG method as shown in Figure 1. Let  $i = q$ , there are 15 blocks belonging to  $\bar{\bar{Z}}^{(q)}$  as shown in Figure 2a, the 7 impedance



**Figure 2.** Illustration of horizontal separation between two  $i$ -th level nonneighbor groups  $m_i^{(2)}$  and  $n_i^{(1)}$  associated with the block in  $\bar{Z}^{(i)}$ . (a) Two  $(i+1)$ -th level neighbor groups sharing one common point;  $\rho_R = \sqrt{5}(2^{(i-1)}\sqrt{M})\Delta x$ . (b) Two  $(i+1)$ -th level neighbor groups sharing one common edge;  $\rho_R = 2(2^{(i-1)}\sqrt{M})\Delta x$ .

blocks  $\bar{Z}_{m_q^{(1)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(1)}n_q^{(4)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(4)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(4)}n_q^{(1)}}^{(q)}$ , and  $\bar{Z}_{m_q^{(4)}n_q^{(3)}}^{(q)}$  belong to far field because their  $\rho_R$  s are all equal or larger than  $r_d = 2(2^{(q-1)}\sqrt{M})\Delta x$ . However, the other 8 impedance blocks  $\bar{Z}_{m_q^{(1)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(1)}n_q^{(2)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(2)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(4)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(4)}n_q^{(2)}}^{(q)}$ , and  $\bar{Z}_{m_q^{(4)}n_q^{(4)}}^{(q)}$  belong to near and intermediate field because their  $\rho_R$  s are all less than  $r_d$ . The impedance block  $\bar{Z}_{m_q^{(3)}n_q^{(2)}}^{(q)}$  has been included in a lower-level interaction and will be excluded from  $\bar{Z}^{(q)}$ . In Figure 2b there are 12 blocks, the 4 impedance blocks  $\bar{Z}_{m_q^{(2)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(4)}n_q^{(1)}}^{(q)}$ , and  $\bar{Z}_{m_q^{(4)}n_q^{(3)}}^{(q)}$  belong to far field, while the other 8 impedance blocks  $\bar{Z}_{m_q^{(1)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(1)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(2)}n_q^{(2)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(4)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(1)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(3)}n_q^{(3)}}^{(q)}$ ,  $\bar{Z}_{m_q^{(4)}n_q^{(2)}}^{(q)}$ , and  $\bar{Z}_{m_q^{(4)}n_q^{(4)}}^{(q)}$  belong to near and intermediate field. The rest of the impedance blocks are excluded from  $\bar{Z}^{(q)}$  because they have been included in a lower-level interaction.

[11] In term of matrix notation, every impedance block belonging to far field is decomposed into the flat surface matrix  $\bar{Z}_{FS}$  and the weak matrices  $\bar{Z}_W$ . Then equation (3) becomes

$$\begin{aligned} \bar{\bar{Z}} &= \bar{\bar{Z}}^{(0)} + \left( \bar{\bar{Z}}_{UV}^{(1)} + \bar{\bar{Z}}_{UV}^{(2)} + \cdots + \bar{\bar{Z}}_{UV}^{(q-1)} + \bar{\bar{Z}}_{UV}^{(q)} \right) \\ &\quad + \left( \bar{\bar{Z}}_{SM}^{(q)} + \bar{\bar{Z}}_{SM}^{(q+1)} + \cdots + \bar{\bar{Z}}_{SM}^{(P)} \right) \\ &= \bar{\bar{Z}}^{(0)} + \left( \bar{\bar{Z}}_{UV}^{(1)} + \bar{\bar{Z}}_{UV}^{(2)} + \cdots + \bar{\bar{Z}}_{UV}^{(q-1)} + \bar{\bar{Z}}_{UV}^{(q)} \right) \\ &\quad + \left( \bar{\bar{Z}}_{FS}^{(q)} + \bar{\bar{Z}}_{FS}^{(q+1)} + \cdots + \bar{\bar{Z}}_{FS}^{(P)} \right) \\ &\quad + \left( \bar{\bar{Z}}_W^{(q)} + \bar{\bar{Z}}_W^{(q+1)} + \cdots + \bar{\bar{Z}}_W^{(P)} \right), \end{aligned} \quad (5)$$

where  $\bar{\bar{Z}}_W^{(i)} = \sum_{l=1}^{N_T} \bar{\bar{Z}}_{W,l}^{(i)}$ ,  $i = q, q+1, \dots, P$ ,  $N_T$  is the number of terms in the Taylor series expansion.

Let

$$\bar{\bar{Z}}_{UV} = \left( \bar{\bar{Z}}_{UV}^{(1)} + \bar{\bar{Z}}_{UV}^{(2)} + \cdots + \bar{\bar{Z}}_{UV}^{(q-1)} + \bar{\bar{Z}}_{UV}^{(q)} \right) \quad (6a)$$

$$\bar{\bar{Z}}_{FS} = \left( \bar{\bar{Z}}_{FS}^{(q)} + \bar{\bar{Z}}_{FS}^{(q+1)} + \cdots + \bar{\bar{Z}}_{FS}^{(P)} \right) \quad (6b)$$

$$\bar{\bar{Z}}_W = \left( \bar{\bar{Z}}_W^{(q)} + \bar{\bar{Z}}_W^{(q+1)} + \cdots + \bar{\bar{Z}}_W^{(P)} \right). \quad (6c)$$

[12] The impedance matrix  $\bar{\bar{Z}}^{(0)}$  is further decomposed into the block-diagonal part  $\bar{\bar{Z}}_d^{(0)}$  which represents the interactions between the elements of the self group and the block-nondiagonal part  $\bar{\bar{Z}}_{nd}^{(0)}$  as shown in Figure 1. The block-diagonal part  $\bar{\bar{Z}}_d^{(0)}$  is computed by using the Green's function directly. Its inverse is used as a preconditioner. This does not need extra memory nor extra CPU time [Song *et al.*, 1997]. The block-nondiagonal part  $\bar{\bar{Z}}_{nd}^{(0)}$  is compressed by using the SVD. For blocks in  $\bar{\bar{Z}}_{UV}$ , we make improvement in constructing matrix  $V$  so that the UV decomposition can be applied to the rougher surface with exponential correlation function. Consider a block  $\bar{A} = \bar{\bar{Z}}_{m,n}^{(i)}$  with dimensions of  $D_i \times D_i = 4^{(i-1)}M \times 4^{(i-1)}M$  in  $\bar{\bar{Z}}_{UV}^{(i)}$ . Let the rank be  $r_i$  with  $r_i \ll D_i$ . The rank is determined by using fast coarse-coarse sampling and then using SVD on the coarse-sampled matrix. The accuracy is controlled by a threshold. We choose the threshold to be 1.e-5 for rough surface with Gaussian correlation function and 1.e-6 for rough surface with exponential correlation function because surface with

exponential correlation function is rougher. The block  $\bar{\bar{A}}$  can be decomposed by the UV method using [Tsang *et al.*, 2004]

$$\bar{\bar{A}}_{D_i \times D_i} = \bar{\bar{U}}_{D_i \times r_i} \bar{\bar{V}}_{r_i \times D_i} \quad (7)$$

$$\bar{\bar{U}}_{r_i \times r_i} \bar{\bar{V}}_{r_i \times D_i} = \bar{\bar{R}}_{r_i \times D_i}. \quad (8)$$

First  $\bar{\bar{U}}_{D_i \times r_i}$  is selected which consists of interactions between  $D_i$  points in group  $m_i$  and  $r_i$  points in group  $n_i$ . The  $r_i$  points are distributed uniformly in group  $n_i$  in both  $x$  and  $y$  directions. Next  $\bar{\bar{R}}_{r_i \times D_i}$  is selected and consists of interactions between  $r_i$  points in group  $m_i$  and  $D_i$  points in group  $n_i$ . The  $r_i$  points are distributed uniformly in group  $m_i$  in both  $x$  and  $y$  directions. Then  $\bar{\bar{U}}_{r_i \times r_i}$  is selected which consists of interactions between  $r_i$  points in group  $m_i$  and  $r_i$  points in group  $n_i$ . Both sets of  $r_i$  points are distributed uniformly in both  $x$  and  $y$  directions. Finally  $\bar{\bar{V}}_{r_i \times D_i}$  is determined by solving equation (8) by Crout's method with partial pivoting, which is much more accurate than the inverse of  $\bar{\bar{U}}_{r_i \times r_i}$  without extra memory and extra CPU time.

[13] The iterative procedure is based on updating the right-hand side (RHS) with the weak matrix. For the first-order and the higher-order solutions

$$\left(\bar{\bar{Z}}_d^{(0)}\right)^{-1} \left(\bar{\bar{Z}}_d^{(0)} + \bar{\bar{Z}}_{nd}^{(0)} + \bar{\bar{Z}}_{UV} + \bar{\bar{Z}}_{FS}\right) \bar{u}^{(1)} = \left(\bar{\bar{Z}}_d^{(0)}\right)^{-1} \bar{b} \quad (9)$$

$$\begin{aligned} \left(\bar{\bar{Z}}_d^{(0)}\right)^{-1} \left(\bar{\bar{Z}}_d^{(0)} + \bar{\bar{Z}}_{nd}^{(0)} + \bar{\bar{Z}}_{UV} + \bar{\bar{Z}}_{FS}\right) \bar{u}^{(n+1)} \\ = \left(\bar{\bar{Z}}_d^{(0)}\right)^{-1} \bar{b}^{(n+1)} \end{aligned} \quad (10)$$

$$\bar{b}^{(n+1)} = \bar{b} - \bar{\bar{Z}}_w \bar{u}^{(n)}. \quad (11)$$

The weak matrix–vector multiplication has been moved to the RHS and is computed only once in each order. The consequence is that it requires more iteration steps to converge for a small  $r_d$  than a large  $r_d$ . In the hybrid UV/SMCG method,  $r_d$  is chosen large enough to ensure that only a small number of Taylor's expansion terms is required. We find it more efficient by updating the RHS with the weak matrix.

#### 4. Computational Complexity and Memory Requirement

[14] In this section we derive the computational complexity and memory requirement.

[15] We will discuss preprocessing CPU and the CPU for each order and the CPU for each iteration. Preprocessing CPU includes the determination of the rank and the UV decomposition for the near and intermediate field. Preprocessing is done only once.

[16] Note that we put the  $\bar{\bar{Z}}_w$  on the RHS, and we update the RHS for each order. We let  $N_O$  be the total orders of solutions. For each order we need to calculate the product of  $\bar{\bar{Z}}_w$  and column vector once. For each order, we also need to solve the matrix equation (10) once using iterative approach. For the  $j$ th order, let  $N_j$  denote the number of iterations. Then  $N_{all} = \sum_{j=1}^{N_O} N_j$  be the total number of iterations. In the CPU tables, we list  $N_O$  and  $N_{all}$ .

##### 4.1. Preprocessing

[17] Each block in  $\bar{\bar{Z}}_{nd}^{(0)}$  is decomposed into  $\bar{\bar{Q}}\bar{\bar{\Sigma}}\bar{\bar{R}}$  directly by the SVD, where  $\bar{\bar{\Sigma}} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_M)$ . Let its rank be  $r_0$ , each block is compressed with  $\bar{\bar{U}} \cdot \bar{\bar{V}} = \bar{\bar{Q}}(:, 1:r_0) \cdot (\bar{\bar{\Sigma}}(1:r_0, 1:r_0) \bar{\bar{R}}(1:r_0, :))$ . Then the number of computational steps includes  $21M^3$  in the SVD [Golub and Van Loan, 1996] and  $r_0M$  in calculating  $\bar{\bar{V}}_{r_0 \times M}$ . For total  $4(2 \cdot 4^{(P+1)} - 3 \cdot 2^{(P+1)} + 1)$  nondiagonal blocks in  $\bar{\bar{Z}}_{nd}^{(0)}$ , as the original impedance matrix is symmetric, only half of the blocks are decomposed, then the number of computational complex for preprocessing  $\bar{\bar{Z}}_{nd}^{(0)}$  is

$$\begin{aligned} 4 \left( 2 \cdot 4^{(P+1)} - 3 \cdot 2^{(P+1)} + 1 \right) \cdot (21M^3 + r_0M) / 2 \\ = 2(21M^2 + r_0) \left( 2 - \frac{3}{L} + \frac{1}{L^2} \right) N, \end{aligned} \quad (12)$$

where  $L = \sqrt{\frac{N}{M}} = 2^{(P+1)}$ .

[18] Consider a block  $\bar{\bar{Z}}_{m,n}^{(i)}$  with dimensions of  $D_i \times D_i$  in  $\bar{\bar{Z}}_{UV}^{(i)}$ . Coarse sampling means that we select the number of points to be slightly large than the rank. This means we have a priori knowledge of roughly what is the rank, on the basis of the numerical experiments carried out. Suppose we pick the number of points to be about twice that of the rank, then there are  $\frac{8}{3}(2r_i)^3 = \frac{64r_i^3}{3}$  steps to search rank by requiring  $\bar{\bar{\Sigma}}$  [Golub and Van Loan, 1996], and  $\frac{r_i}{3}(r_i^2 - 1) + r_i^2 D_i$  steps to determinate  $\bar{\bar{V}}$  in (8) by Crout's method. Then the number of computational steps for preprocessing the block  $\bar{\bar{Z}}_{m,n}^{(i)}$  is

$$\left( \frac{64}{3} r_i^3 + \frac{r_i}{3} (r_i^2 - 1) + r_i^2 D_i \right) = \left( \frac{65}{3} r_i^3 - \frac{r_i}{3} + r_i^2 D_i \right).$$

[19] For total  $12(9 \cdot 4^{(P+1-i)} - 14 \cdot 2^{(P+1-i)} + 5)$  blocks in  $\overline{\overline{Z}}^{(i)}$  ( $0 < i < q$ ). The number of computational steps for preprocessing  $\overline{\overline{Z}}_{UV}^{(i)}$  is

$$\begin{aligned} & \frac{12(9 \cdot 4^{(P+1-i)} - 14 \cdot 2^{(P+1-i)} + 5)}{2} \cdot \left( \frac{65}{3} r_i^3 - \frac{r_i}{3} + r_i^2 D_i \right) \\ &= (130r_i^3 - 2r_i) \left( 9 \frac{L^2}{4^i} - 14 \frac{L}{2^i} + 5 \right) \\ &+ \frac{3}{2} r_i^2 \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) N. \end{aligned} \quad (13)$$

As the exact rank is not required, once a rank table is obtained, we do not need to search the ranks for all the blocks. We can assume that the blocks with the same separation between a pair of groups, such as the blocks  $\overline{\overline{Z}}_{m_1^{(1)}n_1^{(1)}}^{(i)}$  and  $\overline{\overline{Z}}_{m_2^{(2)}n_2^{(2)}}^{(i)}$  as shown in Figure 2b, have the same rank [Tsang et al., 2004]. Then the steps in (13) will only have the last term.

[20] For  $i = q$ , there are only  $32(2 \cdot 4^{(P+1-q)} - 3 \cdot 2^{(P+1-q)} + 1)$  blocks in  $\overline{\overline{Z}}^{(q)}$  calculated with the UV decomposition. The preprocessing number of their computational steps is

$$\begin{aligned} & \frac{32(2 \cdot 4^{(P+1-q)} - 3 \cdot 2^{(P+1-q)} + 1)}{2} \cdot \left( \frac{65}{3} r_q^3 - \frac{r_q}{3} + r_q^2 D_i \right) \\ &= \frac{16}{3} (65r_q^3 - r_q) \left( 2 \frac{L^2}{4^q} - 3 \frac{L}{2^q} + 1 \right) \\ &+ 4r_q^2 \left( 2 - 3 \frac{2^q}{L} + \frac{4^q}{L^2} \right) N. \end{aligned} \quad (14)$$

The preprocessing CPU in SMCg part is negligible. Thus the total number of computational steps for preprocessing is

$$\begin{aligned} T_{\text{pre}} &= 2(21M^2 + r_0) \left( 2 - \frac{3}{L} + \frac{1}{L^2} \right) N \\ &+ \sum_{i=1}^{q-1} \left[ (130r_i^3 - 2r_i) \left( 9 \frac{L^2}{4^i} - 14 \frac{L}{2^i} + 5 \right) \right. \\ &+ \left. \frac{3}{2} r_i^2 \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) N \right] + \frac{16}{3} (65r_q^3 - r_q) \\ &\cdot \left( 2 \frac{L^2}{4^q} - 3 \frac{L}{2^q} + 1 \right) + 4r_q^2 \left( 2 - 3 \frac{2^q}{L} + \frac{4^q}{L^2} \right) N. \end{aligned} \quad (15)$$

## 4.2. Computational Complexity and Memory Requirement for Very Near Field and Near and Intermediate Field

[21] For  $4^{(P+1)}$  diagonal blocks in  $\overline{\overline{Z}}^{(0)}$ , the number of their computational steps for each iteration is

$$4^{(P+1)} M^2 = MN. \quad (16)$$

[22] For  $4(2 \cdot 4^{(P+1)} - 3 \cdot 2^{(P+1)} + 1)$  nondiagonal blocks in  $\overline{\overline{Z}}^{(0)}$ , the number of their computational steps is

$$\begin{aligned} & 4(2 \cdot 4^{(P+1)} - 3 \cdot 2^{(P+1)} + 1) \cdot 2r_0 M \\ &= 8r_0 \left( 2 - \frac{3}{L} + \frac{1}{L^2} \right) N. \end{aligned} \quad (17)$$

[23] For  $12(9 \cdot 4^{(P+1-i)} - 14 \cdot 2^{(P+1-i)} + 5)$  blocks in  $\overline{\overline{Z}}_{UV}^{(i)}$  ( $0 < i < q$ ), the number of computational steps for  $\overline{\overline{Z}}_{UV}^{(i)}$  is

$$\begin{aligned} & 12(9 \cdot 4^{(P+1-i)} - 14 \cdot 2^{(P+1-i)} + 5) \cdot 2r_i D_i \\ &= 6r_i \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) N. \end{aligned} \quad (18)$$

For  $32(2 \cdot 4^{(P+1-q)} - 3 \cdot 2^{(P+1-q)} + 1)$  blocks in  $\overline{\overline{Z}}^{(q)}$  calculated with the UV decomposition, the number of the computational steps is

$$\begin{aligned} & 32(2 \cdot 4^{(P+1-q)} - 3 \cdot 2^{(P+1-q)} + 1) \cdot 2r_q D_q \\ &= 16r_q \left( 2 - 3 \frac{2^q}{L} + \frac{4^q}{L^2} \right) N. \end{aligned} \quad (19)$$

[24] The total number of computational steps for very near field and near- and intermediate-field interactions is

$$\begin{aligned} & \left[ M + 8r_0 \left( 2 - \frac{3}{L} + \frac{1}{L^2} \right) + 6 \sum_{i=1}^{q-1} r_i \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) \right. \\ & \left. + 16r_q \left( 2 - 3 \frac{2^q}{L} + \frac{4^q}{L^2} \right) \right] N \\ &= \left[ \left( M + 16r_0 + 54 \sum_{i=1}^{q-1} r_i + 32r_q \right) \right. \\ & \left. - \left( 24r_0 + 84 \sum_{i=1}^{q-1} r_i 2^i + 48r_q 2^q \right) \right. \\ & \left. / L + \left( 8r_0 + 30 \sum_{i=1}^{q-1} r_i 4^i + 16r_q 4^q \right) / L^2 \right] N. \end{aligned} \quad (20)$$

Since the original impedance matrix is symmetric, only half of the blocks are stored. Then the memory requirement is of the order

$$\left[ \left( \frac{M}{2} + 8r_0 + 27 \sum_{i=1}^{q-1} r_i + 16r_q \right) - \left( 12r_0 + 42 \sum_{i=1}^{q-1} r_i 2^i + 24r_q 2^q \right) \right] / L + \left( 4r_0 + 15 \sum_{i=1}^{q-1} r_i 4^i + 8r_q 4^q \right) / L^2 \Big] N. \quad (21)$$

### 4.3. Computational Complexity and Memory Requirement Analysis for Far Field

[25] Equations (9)–(10) are solved by updating the RHS. For each order of solution, we need to solve (10) with  $N_j$  iterations and compute the RHS once. The respective CPU and memory requirement are as follows.

[26] For solving equation (10), we derive the CPU required for computing the product of flat surface block and column vector. Consider a block  $\overline{\overline{Z}}_{m,n_i}^{(i)}$  with dimensions of  $D_i \times D_i$  in  $\overline{\overline{Z}}_{\text{FS}}^{(i)}$ . It has three 2-D FFT operations. As the 2-D Toeplitz matrices associated with the different blocks are the same, such as blocks  $\overline{\overline{Z}}_{m_1^{(i)}n_1^{(i)}}^{(i)}$  and  $\overline{\overline{Z}}_{m_2^{(i)}n_1^{(i)}}^{(i)}$  as shown in Figure 2a, we can calculate and store only one Toeplitz matrix with dimensions of  $2\sqrt{D_i} \times 2\sqrt{D_i}$  for these different blocks. Therefore the CPU of  $\overline{\overline{Z}}_{m,n_i}^{(i)}$  consists of only two 2-D FFT operations and a multiplication, and is of the order

$$2D_i \log D_i^2 + 4D_i = (8i - 4 + 4 \log M) D_i. \quad (22)$$

The memory requirement is negligible as sizes of Toeplitz matrices are much less than that of corresponding blocks. Also most Toeplitz matrices are the same. We need only to store  $4D_i$  elements in only one of these identical Toeplitz matrices. On the other hand, if we apply the UV method to far field, its computational steps and memory requirement will be  $2r_i D_i$ , which is dependent on the rank  $r_i$ . The rank is such that

$$2r_i D_i \gg (8i - 4 + 4 \log M) D_i \quad (23)$$

for higher-level blocks in 3-D scattering problem. The memory requirement in the UV/SMCG method is less than that in the UV method.

[27] There are  $4(11 \cdot 4^{(P+1-q)} - 18 \cdot 2^{(P+1-q)} + 7)$  blocks in  $\overline{\overline{Z}}_{\text{FS}}^{(q)}$  for 2-D FFT. The number of computational steps in  $\overline{\overline{Z}}_{\text{FS}}^{(q)}$  is of order

$$4 \left( 11 \cdot 4^{(P+1-q)} - 18 \cdot 2^{(P+1-q)} + 7 \right) 4^{(q-1)} \cdot M(8q - 4 + 4 \log M) = 4(2q - 1 + \log M) \cdot \left( 11 - 18 \frac{2^q}{L} + 7 \frac{4^q}{L^2} \right) N. \quad (24)$$

The number of computational steps in  $\overline{\overline{Z}}_{\text{FS}}^{(i)}$  ( $i > q$ ), is of order

$$12 \left( 9 \cdot 4^{(P+1-i)} - 14 \cdot 2^{(P+1-i)} + 5 \right) 4^{(i-1)} \cdot M(8i - 4 + 4 \log M) = 12(2i - 1 + \log M) \cdot \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) N. \quad (25)$$

[28] The total number of computational steps in the far field is then

$$\left[ 4(2q - 1 + \log M) \left( 11 - 18 \frac{2^q}{L} + 7 \frac{4^q}{L^2} \right) + 12 \sum_{i=q+1}^P (2i - 1 + \log M) \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) \right] N = \left[ 108(P^2 - q^2) - 296P + 88q + \frac{1280}{3} + 4(27P - 27q - 26) \log M + 24(22q - 39 + 11 \cdot \log M) \frac{2^q}{L} - \left( 104q - \frac{316}{3} + 52 \log M \right) \frac{4^q}{L^2} \right] N. \quad (26)$$

### 4.4. Computational Complexity for the Weak Matrix Vector Multiplication

[29] For the updating of RHS in equation (11), we have the CPU for computing the product of weak block and column vector. Consider a block  $\overline{\overline{Z}}_{m,n_i}^{(i)}$  with dimensions of  $D_i \times D_i$  in  $\overline{\overline{Z}}_{\text{W}}^{(i)}$ . It consists of  $N_T$  terms of Taylor series. There are 3 parts in 1st term, 5 parts in 2nd terms, and so on. The total number of 2-D Toeplitz matrices is  $N_T(N_T + 2)$ . It seems that the number of computational steps is  $N_T(N_T + 2)(2D_i \log D_i^2 + 4D_i + 2D_i)$ . However, in fact, many 2-D FFT operations and premultiplications are duplicates. Thus the actual number of computational steps is

$$D_i \log D_i^2 + N_T(N_T + 2)4D_i + N_T(4D_i \log D_i^2 + 3D_i) + N_T^2(D_i \log D_i^2 + D_i). \quad (27)$$

For this paper, let  $N_T = 3$ . Equation (27) is equal to

$$(44 \log D_i + 78)D_i = 4^{(i-1)}M(88i - 10 + 44 \log M). \quad (28)$$

The total number of computational steps in updating the RHS is then

$$\begin{aligned} T_W &= \left[ (88q - 10 + 44 \log M) \left( 11 - 18 \frac{2^q}{L} + 7 \frac{4^q}{L^2} \right) \right. \\ &\quad + 3 \sum_{i=q+1}^P (88i - 10 + 44 \log M) \\ &\quad \left. \cdot \left( 9 - 14 \frac{2^i}{L} + 5 \frac{4^i}{L^2} \right) \right] N \\ &= \left[ 1188(P^2 - q^2) - 2338P + 50q + \frac{11428}{3} \right. \\ &\quad + 4(297P - 297q - 286) \log M \\ &\quad + 12(484q - 671 + 242 \cdot \log M) \frac{2^q}{L} \\ &\quad \left. - \left( 1144q - \frac{2150}{3} + 572 \log M \right) \frac{4^q}{L^2} \right] N. \quad (29) \end{aligned}$$

#### 4.5. Total Cost Function

[30] By using (15), (20), (26) and (29), the total number of all computational steps is

$$\begin{aligned} T_{\text{pre}} + N_{\text{all}} \cdot N &\left[ \left( M + 16r_0 + 54 \sum_{i=1}^{q-1} r_i + 32r_q \right) \right. \\ &\quad + 108(P^2 - q^2) - 296P + 88q + \frac{1280}{3} \\ &\quad + 4(27P - 27q - 26) \log M - \left( 24r_0 + 84 \sum_{i=1}^{q-1} r_i 2^i \right. \\ &\quad \left. + 48r_q 2^q - 24(22q - 39 + 11 \cdot \log M) 2^q \right) / L \\ &\quad + \left( 8r_0 + 30 \sum_{i=1}^{q-1} r_i 4^i + 16r_q 4^q - \left( 104q - \frac{316}{3} \right. \right. \\ &\quad \left. \left. + 52 \log M \right) 4^q \right) / L^2 \left. \right] + N_O \cdot T_W. \quad (30) \end{aligned}$$

In the case of large  $N$ , the CPU of equation (30) approaches  $O(N \log N)$ . The total memory requirement is as in (21).

## 5. Numerical Results and Discussion

[31] In this section, we illustrate the numerical simulation results of the bistatic scattering coefficients. Sim-

ulations are based on two kinds of random rough surfaces: Gaussian and exponential correction functions, respectively. In the past, the Gaussian correlation function has been studied extensively. However, for equal RMS height and correlation length, the Gaussian correlation function surface generally is much smoother than surface with exponential correlation function. Surfaces with Gaussian correlation function have backscattering cross section of land surfaces that are many decibels below measurements. Surface with exponential correlation function has many fine-scale irregular features. Results of scattering from exponential correlation function surfaces are much closer to measurements. The existence of fine irregular features means that the fine discretization and the accurate evaluation of near-field impedance matrix are required to produce accurate results [Zhou *et al.*, 2004]. The remote sensing user communities are more interested in results based on exponential correlation function.

[32] In the simulations, the tapering parameter of the incident wave is selected to be 1/3. The incident angle is 30° from normal for all cases. The stabilized biconjugate gradient (BICG) iterations are stopped when the error ( $L^2$  norm) is less than 1%.

[33] We illustrate results for the bistatic scattering coefficients  $\sigma(\theta_s, \phi_s)$ , defined by equation 6.1.16 of Tsang *et al.* [2001]. Energy conservation test is that  $\int_0^{\pi/2} d\theta_s \sin \theta_s \int_0^{2\pi} d\phi_s \sigma(\theta_s, \phi_s) = 1$ . All the energy tests conducted in this paper satisfy energy conservation to less than 0.7%. The bistatic scattering coefficients are plotted in the plane of incidence with specular scattering at 30°, and backscattering at -30°.

### 5.1. Comparisons Based on Various Methods for Surfaces With Gaussian Correlation Function

[34] Comparisons are made among the UV/SMCG method, the UV method, and the SMCG method. The results of bistatic scattering coefficients are in good agreement. We make comparisons of CPU and memory requirement. Note that in UV/SMCG, we put the  $\bar{Z}_W$  on the RHS, and we update the RHS for each order. This will be labeled “the approach of updating RHS”. Tables 1, 2, and 3 show the tradeoff of computer memory requirements and CPU time for different methods and neighborhood distances for surfaces with  $L_x = L_y = 32\lambda$ ,  $L_x = L_y = 40\lambda$ , and  $L_x = L_y = 48\lambda$ , respectively. The surface areas are  $1024\lambda^2$ ,  $1600\lambda^2$ , and  $2304\lambda^2$  with numbers of unknowns  $N$  of 65,536, 102,400, and 147,456, respectively. The RMS heights of the rough surfaces are  $h = 0.5\lambda$ , with correlation lengths of  $l = 0.707\lambda$ . In Table 2 we do not have the result by the SMCG method because the problem is too large for the SMCG on a single processor. In Table 3 we do not have

**Table 1.** Comparison of CPU Time and Memory Usage Among Three Approaches of the Hybrid UV/SMCG, the UV, and the SMCG for Gaussian Surfaces With Gaussian Correlation Function<sup>a</sup>

Method	Memory Requirement, MB	Preprocessing Time, s	Average Time Per Iteration, s	Number of Iterations	Time in CG, s	Total Time, min
UV/SMCG	575	340	22.4	33, <sup>b</sup> 12, <sup>c</sup> 3 <sup>d</sup>	1,077	23.6
UV	786	376	29.5	32 <sup>c</sup>	944	22.0
SMCG	1,790	133	152.6	57 <sup>c</sup>	8,696	147.2

<sup>a</sup>Here  $h = 0.5\lambda$ ,  $l = 0.707\lambda$ ,  $L_x = L_y = 32\lambda$ ,  $\sqrt{M} = 8$ ,  $P = 4$ ,  $q = 3$ , and  $N = 65,536$ .

<sup>b</sup>Corresponds to first-order solution.

<sup>c</sup>Corresponds to second-order solution.

<sup>d</sup>Corresponds to third-order solution.

<sup>e</sup>Not based on updating right-hand side.

results by the SMCG nor by the UV method. We only have results by the hybrid UV/SMCG. In Table 1, the neighborhood distance is  $r_d = 6\lambda$  with 6 Taylor series expansions terms for the SMCG. For Table 1, as  $h = 0.5\lambda$  and  $l = 0.707\lambda$ , the surface is very rough because of large slope. However, the RMS height is only 0.5. Thus using 6 Taylor series terms with a moderate neighborhood distance, the SMCG can be applied. We compare the results of surface unknowns by the three methods in Figure 3. Figures 3a and 3b show the real and imaginary parts, respectively, between the UV/SMCG and the UV, the difference ( $L^2$  norm) is 2.7%; Figures 3c and 3d show the real and imaginary parts, respectively, between the UV/SMCG and the SMCG, the difference is 4.0%. The results of surface unknowns from the three methods are in agreement, and their differences can be reduced when the tolerance of BICG decreases. For the hybrid UV/SMCG method, the neighborhood distances are  $r_d = 2(2^{q-1}\sqrt{M})\Delta x$  of  $8\lambda$  and  $10\lambda$  with 3 expansion terms in Tables 1 and 2, respectively. In the results of Table 3 based on the hybrid UV/SMCG method, two  $r_d$ s of  $6\lambda$  and  $12\lambda$  are chosen with 3 expansion terms. In Table 1, which is the case for  $N = 65,536$ , it can be seen that the memory requirement and preprocessing time for the UV method are larger than that of the UV/SMCG method. This is because constructing matrices U and V of far

interactions in the UV requires more memory and CPU. On the other hand, this is not needed in the far field because of the use of 2-D FFT in the UV/SMCG method. The CPU per iteration for the UV/SMCG is less than that for the UV. This can be explained by the fact that the computational steps decrease when the UV is replaced by 2-D FFT in the far field. The ranks for this case increase from 16 of the 1st level to 36 of the highest level. Substituting into (23), we get  $2r_4 = 72 > (8 \cdot 4 - 4 + 4 \log M) = 52$  at the highest level. However, because of the updating on the RHS for the UV/SMCG method, it requires more iterations than the UV method. For example, the first number of 33 corresponds to the iteration number needed to solve (9), and the 2nd and 3rd numbers corresponds to the iteration numbers for solving (10) with  $n = 1$  and  $n = 2$ . The increase of the number of iterations explains why the total CPU times are almost the same between the UV/SMCG and the UV method. Table 1 also shows that both memory and CPU time for the SMCG are large than that for the UV/SMCG or for the UV method. This is because of large memory associated with near field in the SMCG method. In the SMCG method based on a single processor, we can only store part of the strong interactions. The other part of the strong interactions has to be computed repeatedly in each iteration.

**Table 2.** Comparison of CPU Time and Memory Usage Between Two Approaches of the Hybrid UV/SMCG and the UV for Gaussian Surfaces With Gaussian Correlation Function<sup>a</sup>

Method	Memory Requirement, MB	Preprocessing Time, s	Average Time Per Iteration, s	Number of Iterations	Time in CG, s	Total Time, min
UV/SMCG	996	932	39.1	29, 8	1,446	39.6
UV	1350	993	47.8	29 (48) <sup>b</sup>	1,387	39.7
SMCG <sup>c</sup>	NA	NA	NA	NA	NA	NA

<sup>a</sup>Here  $h = 0.5\lambda$ ,  $l = 0.707\lambda$ ,  $L_x = L_y = 40\lambda$ ,  $\sqrt{M} = 10$ ,  $P = 4$ ,  $q = 3$ , and  $N = 102,400$ .

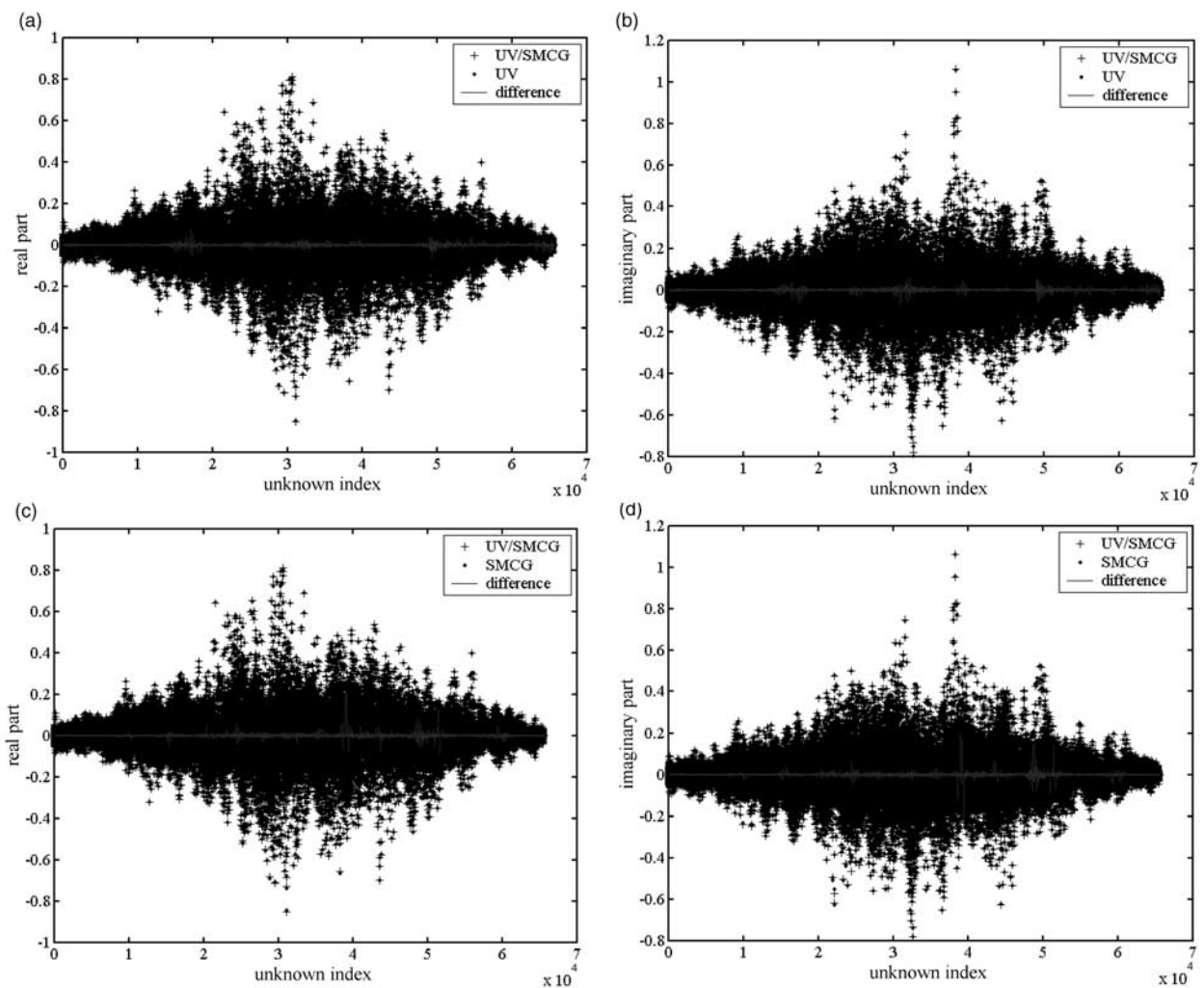
<sup>b</sup>The number in parentheses is the number of iterations without preconditioner.

<sup>c</sup>Not available (NA) because of the limitation of computer resources.

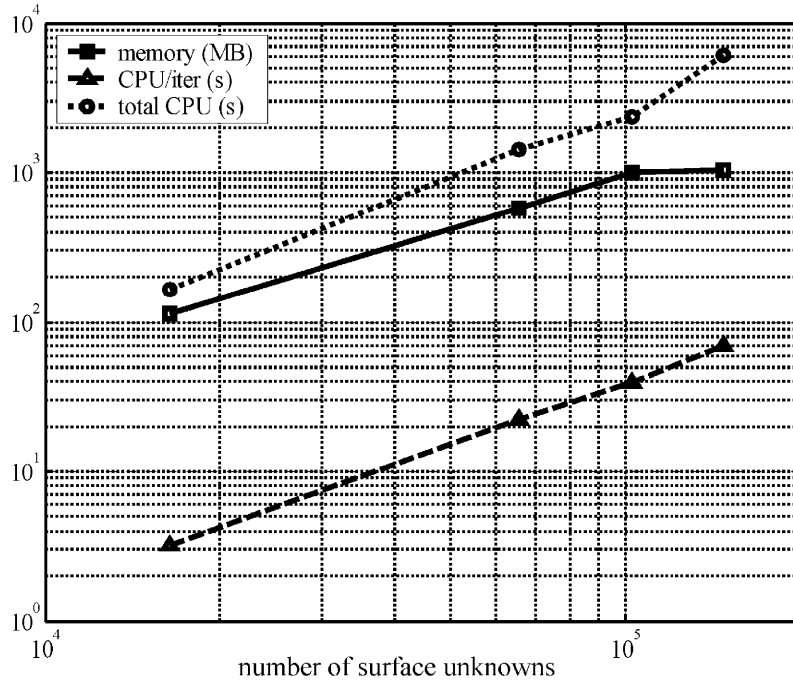
**Table 3.** Comparison of CPU Time and Memory Usage by the UV/SMCG Method With Two Different Neighborhood Distance  $r_d = 2(2^{q-1}\sqrt{M})\Delta x$  for Gaussian Surfaces With Gaussian Correlation Function<sup>a</sup>

Method	Memory Requirement, MB	Preprocessing Time, s	Average Time Per Iteration, s	Number of Iterations	Time in CG, s	Total Time, min
UV/SMCG $q = 2, r_d = 6\lambda$	1,033	2,399	69.3	31, 16, 6	3,674	101.2
UV/SMCG $q = 3, r_d = 12\lambda$	1,611	2,493	76.3	30, 8	2,901	89.9
UV	NA	NA	NA	NA	NA	NA
SMCG	NA	NA	NA	NA	NA	NA

<sup>a</sup>Here  $h = 0.5\lambda, l = 0.707\lambda, L_x = L_y = 48\lambda, \sqrt{M} = 12, P = 4,$  and  $N = 147,456.$



**Figure 3.** Comparisons of surface unknowns solutions between the UV and the UV/SMCG and between the SMCG and the UV/SMCG. Here  $h = 0.5\lambda, l = 0.707\lambda, L_x = L_y = 32\lambda, \sqrt{M} = 8, P = 4, q = 3,$  and  $N = 65,536.$



**Figure 4.** Memory usage (solid curve), CPU time per iteration (dashed curve), and total CPU time (dotted curve) as a function of number of unknowns for Gaussian surfaces with Gaussian correlation function.

[35] The advantage of putting the matrix on the RHS is that the product of weak matrix and column vector is only calculated once for each order of solution. The disadvantage is that the convergence is slower as the total number of iterations increases. In the hybrid UV/SMCG method, the neighborhood distance is considerably larger because of the use of UV method for intermediate distance. The convergence with larger neighborhood distance is much faster than that with smaller neighborhood distance. Since the number of iterations is reduced, it is advantageous to put the weak matrix on the RHS. On the other hand, for the SMCG, the neighborhood distance is as small as possible in order to reduce the near-field computation. As a result, the number of iterations increases. Thus it is more advantageous to put the weak matrix on the LHS for the SMCG.

[36] In Table 2, with a larger  $N = 102,400$ , we can see that more memory is saved by using the UV/SMCG method than by the UV method. As the neighborhood distance  $r_d$  increases, the number of iterations decrease for the UV/SMCG method. In this paper all simulations are computed by using the block-diagonal preconditioner. It is effective in reducing the number of iterations. As an example, we show the number of iterations without preconditioner in Table 2. It can be seen that the number of iterations decreases from 48 to 29 by using this preconditioner.

[37] In Table 3, with  $N = 147,456$ , the problem becomes too large to apply to even the UV method on a single processor. Nevertheless, the UV/SMCG method continues to work well. Table 3 shows the tradeoff between the neighborhood distances. With smaller  $r_d$ , the near and intermediate interactions by the UV part become fewer and the far interactions become more. Its memory requirement becomes less. However, it requires more iterations for convergence.

## 5.2. Comparisons of CPU Time and Memory Usage for Different Numbers of Unknowns by the UV/SMCG Method

[38] In Figure 4, we illustrate the CPU time and memory usage as a function of number of surface

**Table 4.** Comparison of CPU Time and Memory Usage for Different Number of Unknowns

Number of Unknowns	Memory Requirement, MB	Average Time Per Iteration, s	Total Time, s	$r_d$ ( $\lambda$ )
16,384	114	3.2	164	8
65,536	575	22.4	1,417	8
102,400	996	39.1	2,378	10
147,456	1,033	69.3	6,073	6

**Table 5.** Comparison of CPU Time and Memory Usage Among Three Approaches of the Hybrid UV/SMCG, the UV, and the SMCG for Gaussian Surfaces With Exponential Correlation Function<sup>a</sup>

Method	Memory Requirement, MB	Preprocessing Time, s	Average Time Per Iteration, s	Number of Iterations	Time in CG, s	Total Time, min
UV/SMCG	683	349	24.1	42, 16, 11	1,664	33.6
UV	926	392	32.0	42	1,344	28.9
SMCG	1,790	132	152.2	68	10,350	174.7

<sup>a</sup>Parameters are the same as in Table 1.

unknowns from 16,384 to 147,456. Surfaces are with Gaussian correlation functions,  $h = 0.5\lambda$ ,  $l = 0.707\lambda$ . The  $r_d$ s are chosen as small as possible. They are listed in Table 4. The memory requirement and the CPU per iteration shows  $O(N\log N)$  dependence. The memory and the total CPU show a tradeoff by varying the neighborhood distance  $r_d$ . For a smaller  $r_d$ , the memory is smaller and the CPU is larger because of slower convergence.

### 5.3. Larger RMS Height for Surfaces With Gaussian Correlation Function

[39] The hybrid can also apply to surfaces with a larger RMS height of  $h = 1.0\lambda$ ,  $l = 3.0\lambda$ ,  $L_x = L_y = 44\lambda$ ,  $N = 123,904$ . As the RMS is larger, we use  $r_d = 22\lambda$  with 6 Taylor expansion terms. The memory usage is 1,550 MB with total CPU of 46.6 minutes and CPU per iteration of 52.2 s with 22 iterations. Neither the UV method nor the SMCG method can solve this case on a single processor.

### 5.4. Results for Surfaces With Exponential Correlation Function

[40] The UV/SMCG method is also applied to random rough surfaces with exponential correlation function. Here the CPU and memory requirement in Tables 5 and 6 correspond to Tables 1 and 2, respectively but with exponential correlation functions. Their results of bistatic scattering coefficients are in good agreement on the basis of the UV/SMCG, UV, and SMCG methods. From Tables 5 and 6, we find that the memory usage and CPU per iteration both by the UV/SMCG method and by

the UV method are slightly larger than that for the case of Gaussian correlation function. However, the SMCG method does not show any increase. This is because exponential correlation function has a rougher surface. Because the ranks are higher than that of Gaussian correlation function, the UV interpolation needs slightly denser sampling.

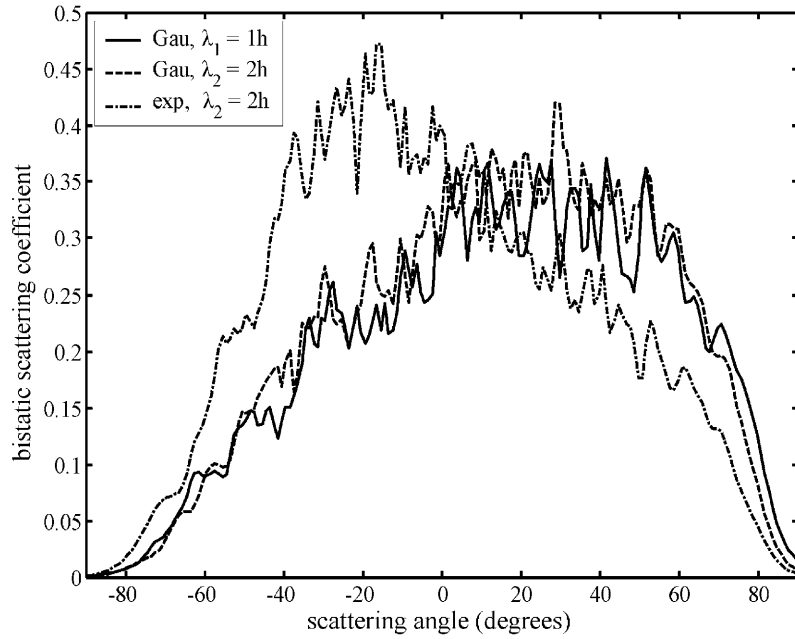
### 5.5. Results of Bistatic Scattering Coefficients Averaged Over Many Realizations

[41] We next study the physics of the bistatic scattering coefficients averaged over many realizations by the hybrid approach in Figure 5. The cases are with same RMS height  $h = 1.0$  and same correlation length  $l = 3.0$  but with different incident wavelengths and different types of surfaces. In microwave remote sensing, it is common to measure scattering by the same surface at several frequencies. Thus we show the scattering by fixing RMS height and correlation length and varying the incident wavelengths. There are two cases with Gaussian correlation function with wavelengths of  $\lambda_1 = h$  and  $\lambda_2 = 2h$ , and with surface area of  $L_x \times L_y = 44\lambda_1 \times 44\lambda_1$ ,  $48\lambda_2 \times 48\lambda_2$ , respectively. The third case is surface with exponential correlation function with wavelength of  $\lambda_2 = 2h$ . The surface area is  $L_x \times L_y = 48\lambda_2 \times 48\lambda_2$ . One can see that the forward scattering become stronger with the increase of wavelength for surfaces with Gaussian correlation function. The bistatic scattering coefficients of exponential correlation function are different from that of Gaussian correlation function. It exhibits larger backscattering while Gaussian correlation

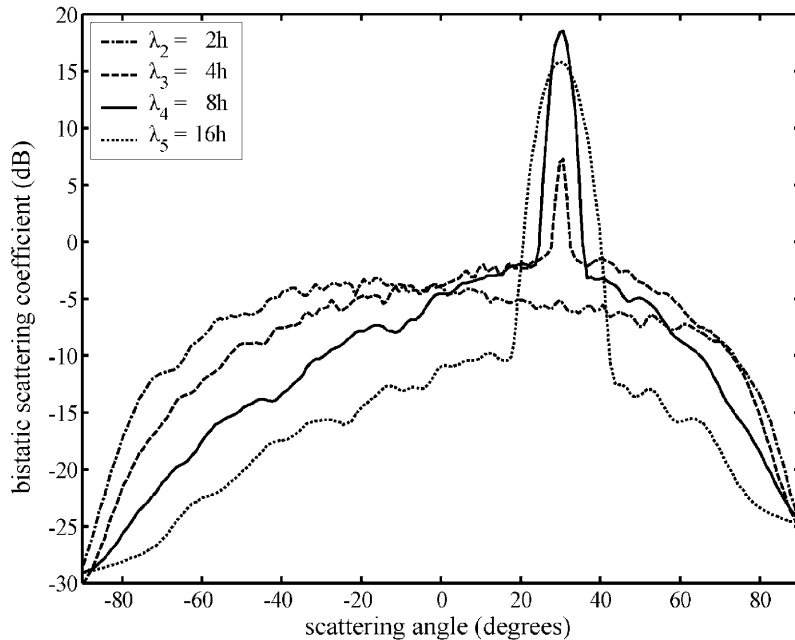
**Table 6.** Comparison of CPU Time and Memory Usage Between Two Approaches of the Hybrid UV/SMCG and the UV for Gaussian Surfaces With Exponential Correlation Function<sup>a</sup>

Method	Memory Requirement, MB	Preprocessing Time, s	Average Time Per Iteration, s	Number of Iterations	Time in CG, s	Total Time, min
UV/SMCG	1,183	941	42.3	39, 13	2,198	56.4
UV	1,564	1,009	52.3	38	1,989	50.0
SMCG	NA	NA	NA	NA	NA	NA

<sup>a</sup>Parameters are the same as in Table 2.



**Figure 5.** Average bistatic scattering coefficients over many realizations. For all cases,  $h = 1.0$ ,  $l = 3.0$ , and  $\theta_i = 30^\circ$ . The solid curve is Gaussian correlation function,  $\lambda_1 = 1h$ ,  $L_x = L_y = 44\lambda_1$ , 100 realizations; the dashed curve is Gaussian,  $\lambda_2 = 2h$ ,  $L_x = L_y = 48\lambda_2$ , 150 realizations; and the dash-dotted curve is exponential correlation function,  $\lambda_2 = 2h$ ,  $L = L_y = 48\lambda_2$ , 150 realizations.



**Figure 6.** Frequency dependence for surface with exponential correlation function over many realizations:  $h = 1.0$ ,  $l = 3.0$ ,  $\theta_i = 30^\circ$ , and  $N = 147,456$ .

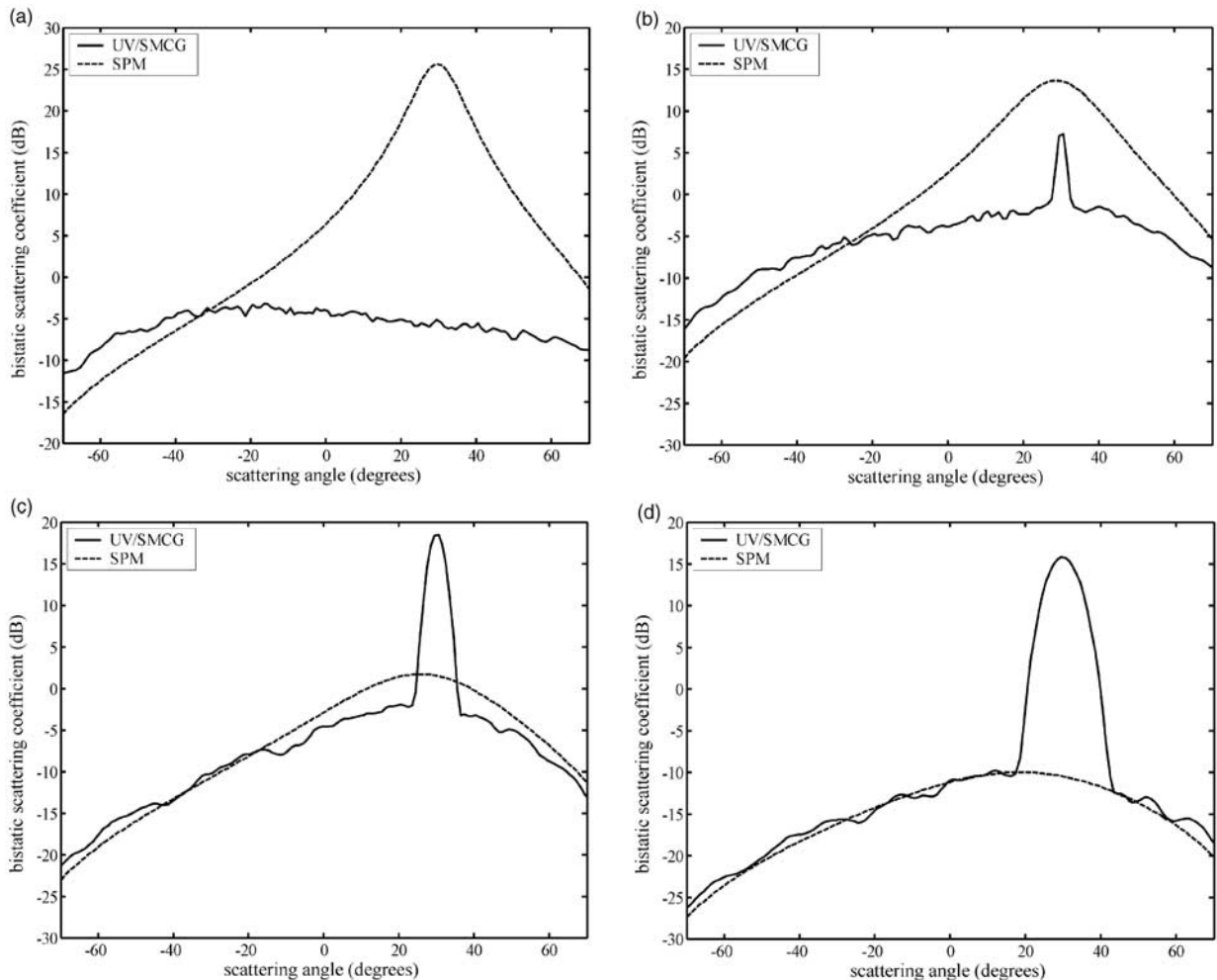
**Table 7.** Comparison of Backscattering Coefficients Over Many Realizations From Surface With Exponential Correlation Function,  $h = 1$ ,  $l = 3$ , and  $N = 147,456$

Wavelength	Surface Area	Number of Sampling Points	Number of Realizations	Backscattering Coefficient
$\lambda_2 = 2h$	$48\lambda_2 \times 48\lambda_2$	$64/\lambda_2^2$	150	0.3814
$\lambda_3 = 4h$	$36\lambda_3 \times 36\lambda_3$	$114/\lambda_3^2$	162	0.2521
$\lambda_4 = 8h$	$18\lambda_4 \times 18\lambda_4$	$455/\lambda_4^2$	83	0.0937
$\lambda_5 = 16h$	$9\lambda_5 \times 9\lambda_5$	$1820/\lambda_5^2$	100	0.0266

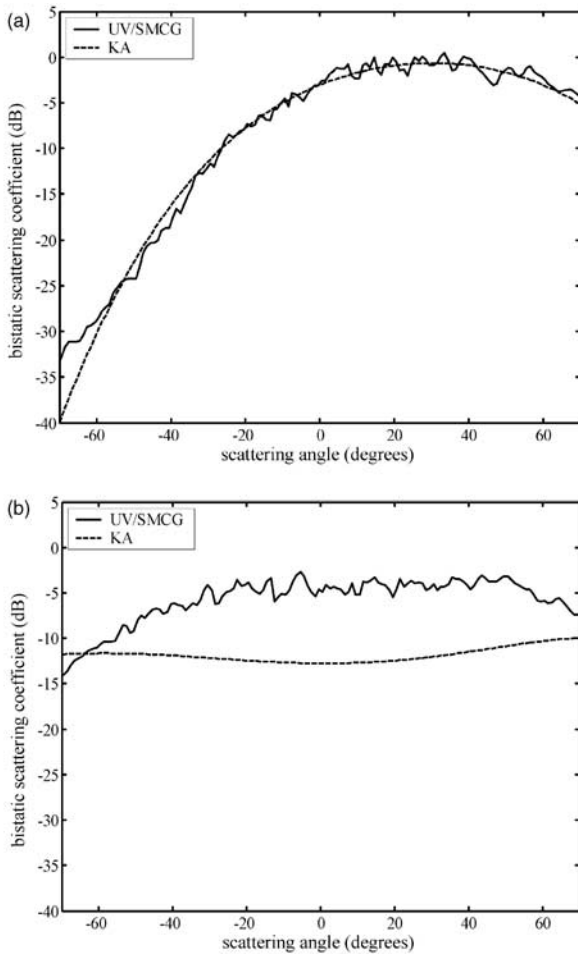
function surfaces exhibit strong forward scattering. For surfaces with same RMS height and correlation length, exponential correlation function can exhibit backscattering enhancement when Gaussian correlation function surfaces do not.

[42] Figure 6 shows the bistatic scattering coefficients from surface with exponential correlation function over many realizations at different frequencies.

[43] In Table 7, we also show the backscattering coefficients of surfaces with exponential correlation



**Figure 7.** Comparison of bistatic scattering coefficients from surfaces with exponential correlation function between the UV/SMCG and the SPM at different frequencies. Parameters are the same as in Figure 6: (a)  $\lambda_2 = 2h$ , (b)  $\lambda_3 = 4h$ , (c)  $\lambda_4 = 8h$ , and (d)  $\lambda_5 = 16h$ .



**Figure 8.** Comparison of bistatic scattering coefficients between the UV/SMCG and the KA: (a) Gaussian correlation function and (b) exponential correlation function. Parameters are  $h = 1.0$ ,  $l = 6.0$ ,  $\lambda_2 = 2h$ ,  $L_x = L_y = 48\lambda_2$ ,  $N = 147,456$ , and 50 realizations.

function associated with Figure 6. Four frequencies are used with frequency doubled each time. The results are consistent with *Torrungrueng and Johnson* [2001]. For this case, the scattering at low frequency is quadratic with frequency. As frequency increases, backscattering is close to linear with frequency.

### 5.6. Comparisons With SPM and KA

[44] We next compare the results of the bistatic scattering coefficients with the small perturbation method (SPM) and the Kirchhoff approximation (KA). Analytical formulas for the SPM and the KA are given in Appendix A. In Figure 7, we make comparisons for surfaces with exponential correlation function. Results

indicate that the SPM agree with simulated results at low frequency but not at high frequency. We also see a distinct coherent wave peak in the specular direction of  $30^\circ$  at lower frequencies. However, it disappears at the highest frequency because of larger roughness. Note that the results for the SPM are for the incoherent wave only. In Figure 8, we show the comparisons between the UV/SMCG and the KA simulated results for surfaces with Gaussian correlation function and exponential correlation function. The parameters are  $l = 6h$ , and  $\lambda = 2h$ . Results indicate that the KA results agree with simulated results for Gaussian correlation function but not for exponential correlation function. Surface with Gaussian correlation function for this case has a large radius of curvature that accounts for the agreement. The simulations show that the SPM is applicable to small RMS height. On the other hand, for the case of exponential correlation function, the KA is neither applicable at low frequency nor at high frequency. This is contrary to the traditional belief that the KA is applicable at high frequency. This is because for exponential correlation function, the spectral density decreases slowly with spatial frequency making it effectively a multiscale surface.

## 6. Conclusion

[45] In this paper, we have presented the hybrid method that combines efficiently the multilevel UV method with the SMCG method. It needs less memory than both the UV method and the SMCG method but with same high computation efficiency as the multilevel UV method. Then it can be applied to larger problem with larger RMS height. The SMCG is FFT based. The UV is rank based. Both concepts can be extended and are being extended to the vector electromagnetic and the penetrable case. The UV method is inherently parallel in nature. The SMCG is FFT based and has been parallelized. We are also presently implementing the algorithm on parallel computation.

## Appendix A: Small Perturbation Method and Kirchhoff Approximation

[46] In this appendix, we list the results for the incoherent wave bistatic scattering coefficients for 3-D scalar waves. The derivation can be conducted in a manner similar to *Tsang et al.* [2000].

[47] The bistatic scattering coefficient for the incoherent wave using the small perturbation method (SPM) is

$$\begin{aligned} \sigma_{incoh}(\theta_s, \varphi_s) = & 4k^4 \cos \theta_i \cos^2 \theta_s \mathcal{W}(k \sin \theta_s \cos \varphi_s \\ & - k \sin \theta_i \cos \varphi_i, k \sin \theta_s \sin \varphi_s \\ & - k \sin \theta_i \sin \varphi_i), \end{aligned} \quad (A1)$$

where  $W$  is the spectral density and is

$$W(k_x, k_y) = \begin{cases} \frac{h^2 l^2}{2\pi} \left(1 + (k_x^2 + k_y^2) l^2\right)^{-\frac{3}{2}} & \text{for exponential correlation function} \\ \frac{h^2 l^2}{4\pi} \exp\left(-\frac{(k_x^2 + k_y^2) l^2}{4}\right) & \text{for Gaussian correlation function.} \end{cases} \quad (\text{A2})$$

[48] The bistatic scattering coefficient for the incoherent wave under the Kirchhoff approximation (KA) is

$$\sigma_{incoh}(\theta_s, \varphi_s) = \frac{k^2 \cos^2 \theta_s}{\cos \theta_i} Q(k_x, k_y), \quad (\text{A3})$$

where

$$Q(k_x, k_y) = \frac{\exp(-k_{dz}^2 h^2)}{2\pi} \left| \frac{k_{ix} k_{dx} + k_{iy} k_{dy} - k_{iz} k_{dz}}{k_z k_{dz}} \right|^2 \cdot \sum_{m=1}^{\infty} \frac{(k_{dz}^2 h^2)^m}{m!} P_m(k_{dx}, k_{dy}) \quad (\text{A4})$$

$$P_m(k_{dx}, k_{dy}) = \begin{cases} m! (m^2 + (k_{dx}^2 + k_{dy}^2) l^2)^{-\frac{3}{2}} & \text{for exponential correlation function} \\ \frac{l^2}{2m} \exp\left(-\frac{(k_{dx}^2 + k_{dy}^2) l^2}{4m}\right) & \text{for Gaussian correlation function} \end{cases} \quad (\text{A5})$$

$$\begin{cases} k_x = k \sin \theta_s \cos \varphi_s \\ k_y = k \sin \theta_s \sin \varphi_s \\ k_z = k \cos \theta_s \end{cases}, \quad \begin{cases} k_{ix} = k \sin \theta_i \cos \varphi_i \\ k_{iy} = k \sin \theta_i \sin \varphi_i \\ k_{iz} = k \cos \theta_i \end{cases},$$

$$\begin{cases} k_{dx} = k_{ix} - k_x \\ k_{dy} = k_{iy} - k_y \\ k_{dz} = -k_{iz} - k_z. \end{cases}$$

[49] **Acknowledgment.** The research in this paper was supported by Hong Kong RGC Central Allocation grant 8730017, the U.S. Office of Naval Research, and NASA.

## References

- Burkholder, R. J., and J. F. Lee (2004), Fast dual-MGS block-factorization algorithm for dense MoM matrices, *IEEE Trans. Antennas Propag.*, 52(7), 1693–1699.
- Golub, G. H., and C. F. Van Loan (1996), *Matrix Computations*, 3rd ed., pp. 253–254, Johns Hopkins Univ. Press, Baltimore, Md.
- Gope, D., and V. Jandhyala (2004), Oct-tree based multilevel low-rank decomposition algorithm for rapid 3D parasitic extraction, *IEEE Trans. Comput. Aided Design Integrated Circuits Syst.*, 23(11), 1575–1580.
- Johnson, J. T. (1998), A numerical study of low-grazing-angle backscatter from ocean-like impedance surfaces with the canonical grid method, *IEEE Trans. Antennas Propag.*, 46(1), 114–120.
- Kapur, S., and D. E. Long (1997), IES3: A fast integral equation solver for efficient 3-dimensional extraction, paper presented at International Conference on Computer-Aided Design 1997, Inst. of Electr. and Electron. Eng., San Jose, Calif.
- Li, S. Q., C. H. Chan, M. Y. Xia, B. Zhang, and L. Tsang (2001), Multilevel expansion of the sparse-matrix canonical grid method for two-dimensional random rough surfaces, *IEEE Trans. Antennas Propag.*, 49(1), 1579–1589.
- Song, J., C. C. Lu, and W. C. Chew (1997), Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects, *IEEE Trans. Antennas Propag.*, 45(10), 1488–1493.
- Torrungrueng, D., and J. T. Johnson (2001), Numerical studies of backscattering enhancement of electromagnetic waves from two-dimensional random rough surfaces with the forward-backward/novel spectral acceleration method, *J. Opt. Soc. Am. A Opt. Image Sci.*, 18(10), 2518–2526.
- Torrungrueng, D., H. T. Chou, and J. T. Johnson (2000), A novel acceleration algorithm for the computation of scattering from two-dimensional large-scale perfectly conducting random rough surfaces with the forward-backward method, *IEEE Trans. Geosci. Remote Sens.*, 38(4), 1656–1668.
- Tsang, L., and Q. Li (2004), Wave scattering with UV multilevel partitioning method: Volume scattering by discrete scatterers, *Microwave Opt. Technol. Lett.*, 41(5), 354–361.
- Tsang, L., C. H. Chan, and K. Pak (1993), Monte Carlo simulation of a two-dimensional random rough surface using the sparse-matrix flat-surface iterative approach, *Electron. Lett.*, 29(13), 1153–1154.
- Tsang, L., C. H. Chan, K. Pak, and H. Sangani (1994), A BMIA/FFT algorithm for the Monte Carlo simulations of large scale random rough surface scattering: Application to grazing incidence, in *IEEE Antennas and Propagation Society International Symposium 1994*, vol. 3, pp. 2028–2031, IEEE Press, Piscataway, N. J.
- Tsang, L., J. A. Kong, and K. H. Ding (2000), *Scattering of Electromagnetic Waves*, vol. 1, *Theories and Applications*, pp. 397–416, Wiley-Interscience, Hoboken, N. J.
- Tsang, L., J. A. Kong, K. H. Ding, and C. O. Ao (2001), *Scattering of Electromagnetic Waves*, vol. 2, *Numerical Simulations*, Wiley-Interscience, 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.
- Wagner, R. L., J. Song, and W. C. Chew (1997), Monte Carlo simulation of electromagnetic scattering from two-dimensional random rough surface, *IEEE Trans. Antennas Propag.*, 45(2), 235–245.
- Wang, H. G., C. H. Chan, and L. Tsang (2005), A new multi-level Green's function interpolation method for large scale low frequency EM simulations, *IEEE Trans. Comput. Aided Design Integrated Circuits Syst.*, in press.
- Xu, P., and L. Tsang (2004), Propagation over terrain and urban environment using the multilevel UV method and a hybrid UV/SDFMM method, *IEEE Antennas Wireless Propag. Lett.*, 3, 336–339.
- Zhou, L., L. Tsang, V. Jandhyala, Q. Li, and C. H. Chan (2004), Emissivity simulations in passive microwave remote sensing with 3-D numerical solutions of Maxwell equations, *IEEE Trans. Geosci. Remote Sens.*, 42(8), 1739–1748.
- 
- L. Tsang and P. Xu, Wireless Communications Research Centre, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong. (tsang@ee.washington.edu)