Progress In Electromagnetics Research C, Vol. 170, 132–139, 2026

(Received 24 March 2026, Accepted 26 April 2026, Scheduled 11 May 2026)

Study of a Parallel Second-Order Small Slope Approximation Algorithm for Electromagnetic Scattering from the Rough Surface

Xiaoyan Zhang, Luqi Wang*, Hongwei He, and Gang Yu

School of Information and Software Engineering, East China Jiaotong University, Nanchang 330013, Jiangxi, China

ABSTRACT: The second-order small-slope approximation (SSA2) is an analytical approximation method applied to electromagnetic scattering simulations on rough surfaces. However, the conventional serial SSA2 method involves a quadruple integral, and each integration requires a fast Fourier transform (FFT). As a result, the method demands very high memory and suffers from very low computational efficiency. In this paper, a parallel SSA2 method is proposed by combining region decomposition with Message Passing Interface (MPI) programming. The OpenMP technique is also adopted to further improve the algorithm's performance. To avoid the frequent communication overhead caused by spectral shifting in traditional parallel FFT methods, the matrix data is carefully allocated, which effectively removes the related communication bottleneck. In addition, by using the separability of the two-dimensional (2-D) FFT, each 2-D FFT is implemented using two sequential one-dimensional (1-D) FFTs. This design limits inter-node communication to a constant upper bound. Numerical results show that when the number of computer nodes reaches 10, the parallel efficiency of the proposed method exceeds 80%, which confirms the effectiveness of the proposed method.

1. INTRODUCTION

The study of electromagnetic scattering from random rough surfaces (such as oceans and land) is crucial for improving radar detection accuracy and enhancing recognition in complex scenes [1–3]. The electromagnetic simulation models for such problems can be roughly divided into numerical methods and analytical methods. Although numerical methods often provide high precision, their applicability is constrained by the requirement for fine mesh discretization. Initially, traditional numerical methods, such as the method of moments (MoM) and the finite element method (FEM), were predominantly employed for one-dimensional (1-D) scattering problems [4, 5]. Subsequently, with the advent of fast solvers such as the multilevel fast multipole method (MLFMM), the accurate and efficient simulation of scattering from two-dimensional (2-D) rough surfaces became feasible [6, 7]. However, the issue of high computational resource demands persists.

To mitigate these challenges, analytical approximation methods [8] have gained prominence, primarily due to their reliance on coarser grids. Prominent analytical techniques include the Kirchhoff approximation (KA) [9], the small perturbation method (SPM) [10], and the two-scale method (TSM) [11]. Unfortunately, these approaches tend to lose computational accuracy at grazing incidence. In 1994, Voronovich introduced the small slope approximation (SSA) method, which significantly advanced the modeling of scattering from random rough surfaces with small slope roughness over multiple wavebands [12]. Subsequent work demonstrated that the first-order SSA (SSA1) was superior to the KA method [13] and comparable to the Monte Carlo integration method (MCIE) for simulating 1-D scattering in high-sea conditions [14]. Further experimental validation indicated that the second-order SSA (SSA2) outperforms SSA1, particularly under low-grazing angles and high surface roughness [15].

Theoretically, SSA2 excels at modeling scattering from multi-scale 2-D rough surfaces and capillary structures, maintaining acceptable cross-polarization accuracy [16]. However, the method's reliance on fine grids to resolve surface features and its mathematical complexity, including multiple layers of integration and the incorporation of FFTs within the innermost integration layers, result in significant memory demands and reduced computational efficiency. Consequently, traditional serial SSA2 implementations are predominantly suited to smaller rough surfaces [17]. To enhance the computational efficiency, improvements such as advanced FFT algorithms [18, 19] and GPU parallel acceleration techniques [20] have been explored, but these methods remain limited by single-machine architectures and fail to address the challenges posed by large-scale rough surface models or excessive memory demands on a single node.

In this paper, we propose a distributed parallel SSA2 algorithm, leveraging MPI for message passing and OpenMP for additional parallelization. This approach employs a domain decomposition strategy to partition the computational workload across multiple nodes, reducing communication overhead typically associated with traditional parallel FFT spectral shifting. By taking advantage of the separability of 2-D FFTs, the algorithm decomposes each operation into two parallel 1-D FFTs, effectively bounding inter-node communication.

---

* Corresponding author: Luqi Wang (18479358208@163.com).

doi:10.2528/PIERC26032408 Published by THE ELECTROMAGNETIC ACADEMY

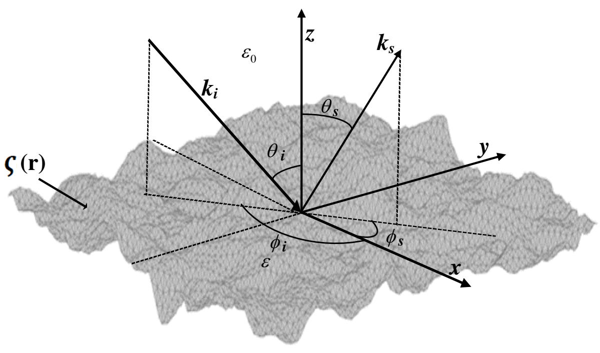


FIGURE 1. 2-D randomly rough surface.

2. THEORY AND METHODOLOGY

In this section, the principles of the serial SSA2 method are initially outlined, followed by an analysis of its characteristics and the primary challenges encountered in its parallel implementation. Subsequently, the parallelization strategies proposed are detailed in Sections 2.2 through II-D.

2.1. Principles and Characteristics of the Serial SSA2

The 2-D randomly-rough surface is depicted in Fig. 1, where $z = \varsigma(\mathbf{r})$ represents the heights of the rough surface [21]. To avoid truncation errors, a tapered wave is incident at angles θ_i and ϕ_i. The method for calculating the scattering amplitude **S** (four polarization channels: VV, VH, HH, HV) of the rough surface by SSA2 is [12]

$$\mathbf{S}(\mathbf{k}_s,\mathbf{k}_i) = \frac{1}{(2\pi)^2}\frac{2(q_1q_{01})^{1/2}}{(q_{01}+q_1)}\int R(\mathbf{r})d\mathbf{r}\times[\mathbf{B}_1(\mathbf{k}_s,\mathbf{k}_i) -\frac{j}{4}\int \mathbf{M}(\mathbf{k}_s,\mathbf{k}_i,\boldsymbol{\xi})\varsigma(\boldsymbol{\xi})e^{j\boldsymbol{\xi}\cdot\mathbf{r}}d\boldsymbol{\xi}] \tag{1}$$

where $\mathbf{B}_1$ is the coefficient matrix of the first order, and $\mathbf{r}$ is the projection of the spatial position vector to the x-y plane. The parameters $\mathbf{k}_i$, q_{01}, and $\mathbf{k}_s$, q_1 correspond to the horizontal (x-y plane) and vertical components of the incident and scattered waves (with scattering angles θ_s and ϕ_s) in air, respectively, and are defined as:

$$\mathbf{k}_i = k\sin\theta_i(\cos\phi_i\hat{x}+\sin\phi_i\hat{y}), \quad q_{01} = k\cos\theta_i \tag{2a}$$
$$\mathbf{k}_s = k\sin\theta_s(\cos\phi_s\hat{x}+\sin\phi_s\hat{y}), \quad q_1 = k\cos\theta_s \tag{2b}$$

Here, $k = \omega\sqrt{\varepsilon_0\mu_0}$, where ω is the angular frequency of the incident wave; ε_0 and μ_0 correspond to the permittivity and permeability of the vacuum, respectively. $\hat{x}$ and $\hat{y}$ are unit vectors.

$$R(\mathbf{r}) = T[\mathbf{r},\varsigma(\mathbf{r})]e^{-j(\mathbf{k}_s-\mathbf{k}_i)\cdot\mathbf{r}+j(q_{01}+q_1)\varsigma(\mathbf{r})} \tag{3}$$

where $T[\mathbf{r},\varsigma(\mathbf{r})]$ represents the amplitude of the taper wave [2], with $\mathbf{r}$ referring to the projection of the spatial position vector onto the x-y plane.

The form of the second-order term $\mathbf{M}(\mathbf{k}_s,\mathbf{k}_i,\boldsymbol{\xi})$ is:

$$\mathbf{M}(\mathbf{k}_s,\mathbf{k}_i,\boldsymbol{\xi}) = \mathbf{B}_2(\mathbf{k}_s,\mathbf{k}_i,\mathbf{k}_S-\boldsymbol{\xi})+\mathbf{B}_2(\mathbf{k}_s,\mathbf{k}_i,\mathbf{k}_i+\boldsymbol{\xi}) +2(q_1+q_{01})\mathbf{B}_1(\mathbf{k}_s,\mathbf{k}_i) \tag{4}$$

where $\mathbf{B}_1$ and $\mathbf{B}_2$ are coefficient matrices of first and second order, respectively. Depending on the polarization mode, these matrices are decomposed into four independent components. Taking vertical polarization as an example, the main polarization expressions of $\mathbf{B}_1$ and $\mathbf{B}_2$ can be expressed as

$$(B_1)_{VV}(\mathbf{k}_s,\mathbf{k}_i) = \frac{\varepsilon_r-1}{(\varepsilon_r q_1+q_2)(\varepsilon_r q_{01}+q_{02})} \times\left(q_2q_{02}\frac{\mathbf{k}_s\cdot\mathbf{k}_i}{k_sk_i}-\varepsilon_r k_sk_i\right) \tag{5}$$

$$(B_2)_{VV}(\mathbf{k}_s,\mathbf{k}_i,\boldsymbol{\xi}) = c_1\left[c_2\left(\frac{\mathbf{k}_s\cdot\boldsymbol{\xi}}{k_s}\frac{\boldsymbol{\xi}\cdot\mathbf{k}_i}{k_i}-\xi^2\frac{\mathbf{k}_s\cdot\mathbf{k}_i}{k_sk_i}\right) +c_3\frac{\mathbf{k}_s\cdot\mathbf{k}_i}{k_sk_i}\right] \tag{6}$$

where

$$c_1 = \frac{(\varepsilon-1)(\omega/c)^2}{(q_1+q_2)(q_{01}+q_{02})} \tag{7a}$$
$$c_2 = \frac{-2(\varepsilon-1)}{(\varepsilon q_{\xi1}+q_{\xi2})} \tag{7b}$$
$$c_3 = q_2+q_{02}+2(q_{\xi1}-q_{\xi2}) \tag{7c}$$
$$q_{02} = \sqrt{\varepsilon k^2-|\mathbf{k}_i|^2} = k\sqrt{\varepsilon-\sin^2\theta_i} \tag{7d}$$
$$q_2 = \sqrt{\varepsilon k^2-|\mathbf{k}_s|^2} = k\sqrt{\varepsilon-\sin^2\theta_s} \tag{7e}$$
$$q_{\xi1} = \begin{cases} \sqrt{k^2-(\mathbf{k}_{sx}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{sy}-\frac{2\pi n}{L_y})^2} \\ k>\sqrt{(\mathbf{k}_{sx}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{sy}-\frac{2\pi n}{L_y})^2} \\ i\sqrt{(\mathbf{k}_{sx}-\frac{2\pi m}{L_x})^2+(\mathbf{k}_{sy}-\frac{2\pi n}{L_y})^2-k^2} \\ k\le\sqrt{(\mathbf{k}_{sx}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{sy}-\frac{2\pi n}{L_y})^2} \end{cases} \tag{7f}$$
$$q_{\xi2} = \begin{cases} \sqrt{k^2-(\mathbf{k}_{ix}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{iy}-\frac{2\pi n}{L_y})^2} \\ k>\sqrt{(\mathbf{k}_{ix}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{iy}-\frac{2\pi n}{L_y})^2} \\ i\sqrt{(\mathbf{k}_{ix}-\frac{2\pi m}{L_x})^2+(\mathbf{k}_{iy}-\frac{2\pi n}{L_y})^2-k^2} \\ k\le\sqrt{(\mathbf{k}_{ix}-\frac{2\pi m}{L_x})^2-(\mathbf{k}_{iy}-\frac{2\pi n}{L_y})^2} \end{cases} \tag{7g}$$

The spatial spectrum $\varsigma(\boldsymbol{\xi})$ is obtained by applying spatial Fourier transform to $\varsigma(\mathbf{r})$ as

$$\varsigma(\boldsymbol{\xi}) = \frac{1}{(2\pi)^2}\int\varsigma(\mathbf{r})e^{-j\boldsymbol{\xi}\cdot\mathbf{r}}d\mathbf{r}. \tag{8}$$

where $\boldsymbol{\xi}$ is the spatial frequency.

After obtaining **S**, the radar cross section (RCS) [22] of the surfaces and their normalized RCS (NRCS) [15] can be further calculated by

$$\sigma_{SSA} = 16\pi^3q_{01}q_1\left\langle|S(\mathbf{k}_s,\mathbf{k}_0)|^2\right\rangle \tag{9}$$

$$\sigma_{SSA} = 4\pi q_{01}q_1 \left\langle \Delta S(\mathbf{k}_s,\mathbf{k}_0)[\Delta S(\mathbf{k}_s,\mathbf{k}_0)]^* \right\rangle \tag{10}$$

with

$$\Delta S(\mathbf{k}_s,\mathbf{k}_0) = S(\mathbf{k}_s,\mathbf{k}_0) - \langle S(\mathbf{k}_s,\mathbf{k}_0)\rangle \tag{11}$$

Here, the symbol * represents conjugation, and $\langle \ldots \rangle$ denotes the average value of the squared scattering amplitudes calculated from multiple rough-surface samples.

From formulas (1), (4), (6), and (8), we can observe that:

(1) The **S**/RCSs of the four polarization modes are calculated simultaneously, so the calculation process of SSA2 requires storing 10 large-sized matrices including $\varsigma(\mathbf{r})$, $\varsigma(\boldsymbol{\xi})$, $\mathbf{M}$ and $\mathbf{B}_2$.

(2) At least four integration layers ($\int \cdot d\mathbf{r} \int \cdot d\xi$) are required for a single incident or scattering direction.

(3) The process of calculating the second-order term $\int M(\mathbf{k}_s,\mathbf{k}_0,\boldsymbol{\xi})\varsigma(\boldsymbol{\xi})e^{j\boldsymbol{\xi}\cdot\mathbf{r}}d\boldsymbol{\xi}$ is as follows:

The zero frequency of the spatial spectrum of **M** is at the center, so after executing formula (8) (achieved by calling the fftw library), $\varsigma(\boldsymbol{\xi})$ also needs to perform a "four-quadrant swap" to place the spectral origin of $\varsigma(\boldsymbol{\xi})$ at the geometric center (this process is represented by FFTshift), so as to calculate $\mathbf{M}\cdot\varsigma$. Next, IFFTshift is applied to the result of $\mathbf{M}\cdot\varsigma$ to perform another four-quadrant swap, thereby restoring the spectrum to its original order before the next step of the IFFT operation. The specific process is as follows:

$$\varsigma(\mathbf{r}) \xrightarrow{\mathit{2DFFT\text{-}FFTshift}} \varsigma(\boldsymbol{\xi}) \rightarrow \mathbf{M}\cdot\varsigma$$
$$\xrightarrow{\mathit{IFFTshift\text{-}2DIFFT}} \int M(\mathbf{k}_s,\mathbf{k}_0,\boldsymbol{\xi})\varsigma(\boldsymbol{\xi})e^{j\boldsymbol{\xi}\cdot\mathbf{r}}d\boldsymbol{\xi}$$

The above features indicate that the computation of **M** and the integration process of $\int M(\mathbf{k}_s,\mathbf{k}_0,\boldsymbol{\xi})\varsigma(\boldsymbol{\xi})e^{j\boldsymbol{\xi}\cdot\mathbf{r}}d\boldsymbol{\xi}$ exhibit inherent limitations in parallelization, as they involve a large amount of data aggregation and frequent inter-node communication. When deployed across multiple nodes in a parallel computing environment, these operations incur considerable communication overhead.

2.2. Regional Decomposition Method

To prevent node communication induced by FFTshift and IFFTshift, the matrix data is carefully allocated to the nodes. To illustrate more specifically, taking matrix

$$\mathbf{a} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_8]^{\mathrm{T}} \tag{12}$$

as an example, where $\mathbf{a}_i = [\mathbf{a}_{i1}, \mathbf{a}_{i2}, \ldots, \mathbf{a}_{i8}]$, $(i = 1, 2, \ldots, 8)$. Assuming that the number of computer nodes (or processes) is 4, and they are represented by 'Node 0', 'Node 1', 'Node 2', and 'Node 3', respectively. The traditional method for decomposing a region along one dimension is to divide the elements of **a** into four subdomains: {$\mathbf{a}_1$, $\mathbf{a}_2$}, {$\mathbf{a}_3$, $\mathbf{a}_4$}, {$\mathbf{a}_5$, $\mathbf{a}_6$} and {$\mathbf{a}_7$, $\mathbf{a}_8$}, as shown in Fig. 2(a). Considering that after FFT operations, the data will exhibit "four-quadrant" symmetry, and the region segmentation strategy shown in Fig. 2(b) is adopted: the matrix data that need to be frequency-shifted are allocated to the same nodes. According to this strategy, the data assigned to 'Node 0', 'Node 1', 'Node 2', and 'Node 3' are {$\mathbf{a}_4$, $\mathbf{a}_5$}, {$\mathbf{a}_3$, $\mathbf{a}_6$}, {$\mathbf{a}_2$, $\mathbf{a}_7$}, {$\mathbf{a}_1$, $\mathbf{a}_8$}, respectively.

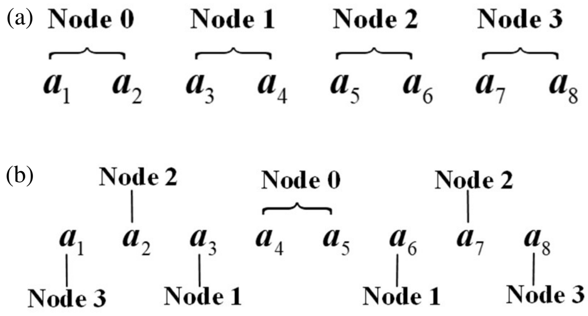


FIGURE 2. The strategy for allocating matrix data to multiple nodes: (a) traditional; (b) proposed.

The proposed strategy facilitates the local execution of the FFTShift operation on each individual node, effectively minimizing the overhead associated with inter-node communication.

2.3. Parallelization Methods for 2-D FFT

Mathematically, based on the separability of the Fourier transform, a 2-D FFT can be decomposed into two 1-D FFT operations. To illustrate this procedure, let a denote the input matrix, assuming it has 8×8 elements. The first stage involves applying the 1-D FFT along the rows (i.e., in the x-axis direction) of **a**, as mathematically expressed in (13a), resulting in an intermediate matrix $\mathbf{a}^{row}$. Subsequently, a second set of 1-D FFT operations is applied along the columns (i.e., in the y-axis direction) of $\mathbf{a}^{row}$, as specified in (13b), thereby producing the final 2-D FFT transformed signal **b**.

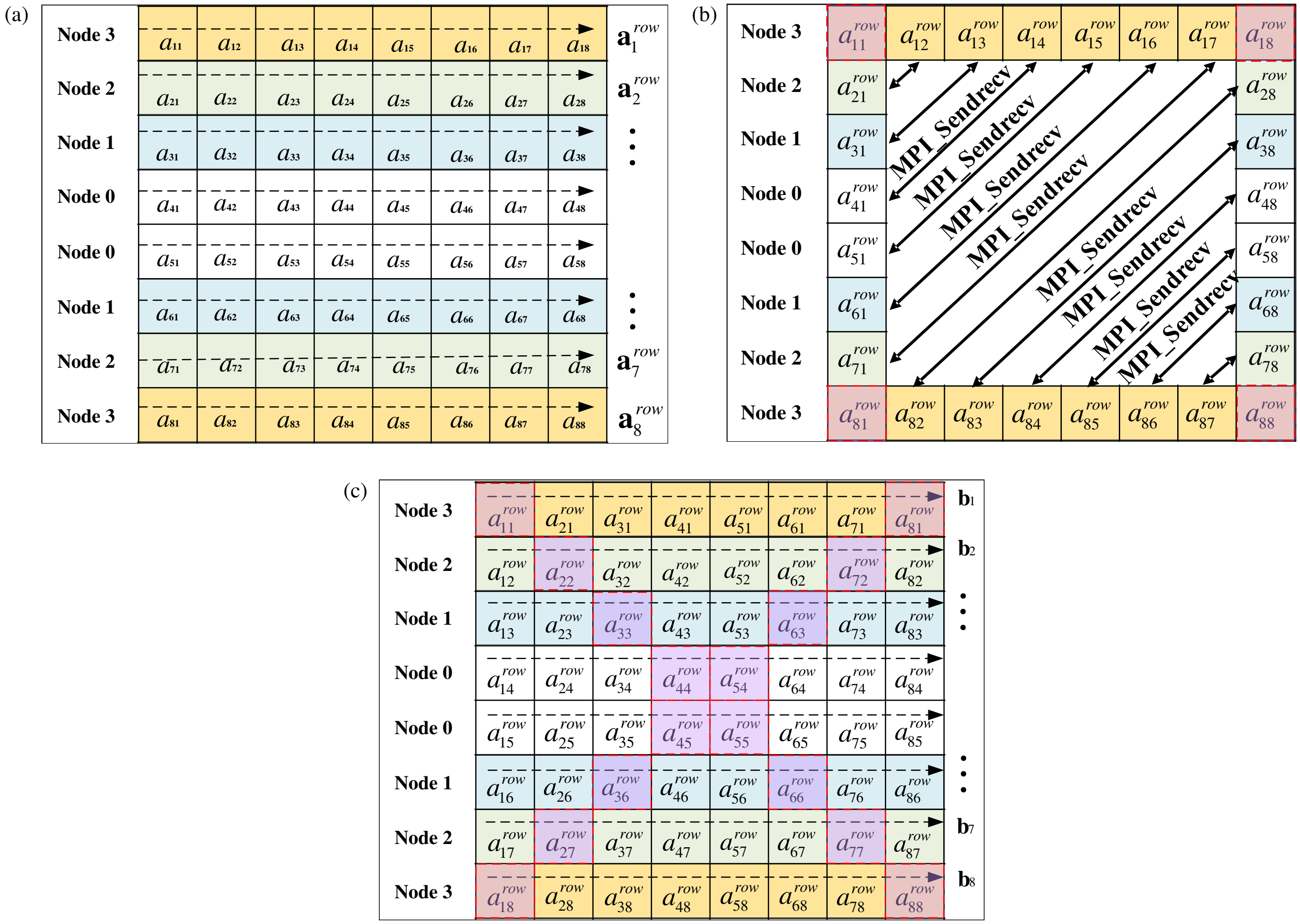
$$\mathbf{a}_i^{row} = FFT_{1d}(\mathbf{a}_i), \quad (i = 1, 2, \ldots, 8) \tag{13a}$$
$$\mathbf{b}_j = FFT_{1d}(\mathbf{a}_j^{row}), \quad (j = 1, 2, \ldots, 8) \tag{13b}$$

The parallelization strategy derived based on Equations (13a) and (13b) is shown in Fig. 3. Specifically, it is divided into the following steps:

Step #1: The elements of **a** in each node are individually computed by (13a) to generate matrix $\mathbf{a}^{row}$ (as shown in Fig. 3(a)). No inter-node communication is required during the calculation, and the resulting matrix $\mathbf{a}^{row}$ is retained locally on each node. For example, in 'Node 3', only {$\mathbf{a}_1^{row}$, $\mathbf{a}_8^{row}$} are stored. To optimize computational performance, the fftw library is utilized to carry out row-wise 1-D FFT throughout this work.

Step #2: *MPI_Sendrecv*, a blocking message passing function in the MPI library, is called at each node to enable the exchange of data required for the column-wise 1-D FFT, as illustrated in Fig. 3(b).

Taking 'Node 3' as an example, when performing column-wise 1-D FFT operations on the elements {$\mathbf{a}_{i1}^{row}$, $\mathbf{a}_{i8}^{row}$} , $(i = 1, 2, \ldots, 8)$, according to formula (13b), it requires the participation of {$\mathbf{a}_{11}^{row}$, $\mathbf{a}_{81}^{row}$} and {$\mathbf{a}_{18}^{row}$, $\mathbf{a}_{88}^{row}$} stored in 'Node 3', as well as {$\mathbf{a}_{21}^{row}$, $\mathbf{a}_{71}^{row}$, $\mathbf{a}_{28}^{row}$, $\mathbf{a}_{78}^{row}$} in 'Node 2', {$\mathbf{a}_{31}^{row}$, $\mathbf{a}_{61}^{row}$, $\mathbf{a}_{38}^{row}$, $\mathbf{a}_{68}^{row}$} in 'Node 1', and {$\mathbf{a}_{41}^{row}$, $\mathbf{a}_{51}^{row}$, $\mathbf{a}_{48}^{row}$, $\mathbf{a}_{58}^{row}$} in 'Node 0'. However, only the elements $\mathbf{a}_{11}^{row}$, $\mathbf{a}_{81}^{row}$, $\mathbf{a}_{18}^{row}$ and



(a)

Node 3	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	$\mathbf{a}_1^{row}$
Node 2	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	$\mathbf{a}_2^{row}$
Node 1	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}	⋮
Node 0	a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}	
Node 0	a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	a_{57}	a_{58}	
Node 1	a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}	a_{67}	a_{68}	⋮
Node 2	a_{71}	a_{72}	a_{73}	a_{74}	a_{75}	a_{76}	a_{77}	a_{78}	$\mathbf{a}_7^{row}$
Node 3	a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}	a_{87}	a_{88}	$\mathbf{a}_8^{row}$

(b)

Node 3	a_{11}^{row}	a_{12}^{row}	a_{13}^{row}	a_{14}^{row}	a_{15}^{row}	a_{16}^{row}	a_{17}^{row}	a_{18}^{row}
Node 2	a_{21}^{row}							a_{28}^{row}
Node 1	a_{31}^{row}							a_{38}^{row}
Node 0	a_{41}^{row}							a_{48}^{row}
Node 0	a_{51}^{row}							a_{58}^{row}
Node 1	a_{61}^{row}							a_{68}^{row}
Node 2	a_{71}^{row}							a_{78}^{row}
Node 3	a_{81}^{row}	a_{82}^{row}	a_{83}^{row}	a_{84}^{row}	a_{85}^{row}	a_{86}^{row}	a_{87}^{row}	a_{88}^{row}

(c)

Node 3	a_{11}^{row}	a_{21}^{row}	a_{31}^{row}	a_{41}^{row}	a_{51}^{row}	a_{61}^{row}	a_{71}^{row}	a_{81}^{row}	$\mathbf{b}_1$
Node 2	a_{12}^{row}	a_{22}^{row}	a_{32}^{row}	a_{42}^{row}	a_{52}^{row}	a_{62}^{row}	a_{72}^{row}	a_{82}^{row}	$\mathbf{b}_2$
Node 1	a_{13}^{row}	a_{23}^{row}	a_{33}^{row}	a_{43}^{row}	a_{53}^{row}	a_{63}^{row}	a_{73}^{row}	a_{83}^{row}	⋮
Node 0	a_{14}^{row}	a_{24}^{row}	a_{34}^{row}	a_{44}^{row}	a_{54}^{row}	a_{64}^{row}	a_{74}^{row}	a_{84}^{row}	
Node 0	a_{15}^{row}	a_{25}^{row}	a_{35}^{row}	a_{45}^{row}	a_{55}^{row}	a_{65}^{row}	a_{75}^{row}	a_{85}^{row}	
Node 1	a_{16}^{row}	a_{26}^{row}	a_{36}^{row}	a_{46}^{row}	a_{56}^{row}	a_{66}^{row}	a_{76}^{row}	a_{86}^{row}	⋮
Node 2	a_{17}^{row}	a_{27}^{row}	a_{37}^{row}	a_{47}^{row}	a_{57}^{row}	a_{67}^{row}	a_{77}^{row}	a_{87}^{row}	$\mathbf{b}_7$
Node 3	a_{18}^{row}	a_{28}^{row}	a_{38}^{row}	a_{48}^{row}	a_{58}^{row}	a_{68}^{row}	a_{78}^{row}	a_{88}^{row}	$\mathbf{b}_8$

FIGURE 3. The proposed 2-D FFT parallel strategy: (a) Step #1, row-wise parallel FFT computation; (b) Step 2, data transfer between nodes (non-communicating regions labeled as "🟥"); (c) Step #3, column-wise parallel FFT execution (non-communicating cells marked as "🟥 🟪") .

$\mathbf{a}_{88}^{row}$ are stored on 'Node 3', so the remaining elements that are distributed on other nodes must be retrieved via inter-node communication. It should be noted that the elements not participating in communication are located along the main diagonal of the matrix. To illustrate more specifically, taking Figs. 3(b) and 3(c) as examples, 48 elements need to be exchanged (12 elements per node), and each node initiates communication with other nodes 3 times (for example, 'Node 0' initiates communication with 'Node 1', 'Node 2', and 'Node 3'). Owing to the use of *MPI_Sendrecv*, the inter-node data exchange is conducted synchronously, and only 4 elements are transferred at a time (accounting for $4/64 = 6.25\%$ of the total data). Taking 'Node 3' and 'Node 2' as an example, the data transmitted simultaneously through *MPI_Sendrecv* function is

$$\begin{aligned}\mathbf{a}_{12}^{row} &\leftrightarrows \mathbf{a}_{21}^{row},\\ \mathbf{a}_{17}^{row} &\leftrightarrows \mathbf{a}_{71}^{row},\\ \mathbf{a}_{28}^{row} &\leftrightarrows \mathbf{a}_{82}^{row},\\ \text{and }\ \mathbf{a}_{78}^{row} &\leftrightarrows \mathbf{a}_{87}^{row}.\end{aligned}$$

Assuming that the number of nodes increases to 8 and that the matrix elements remain unchanged, only $\mathbf{a}_{ii}^{row}(i = 1, 2, \ldots, 8)$ will not be exchanged. The number of matrix elements that need to exchange data increases to 56, but the number of elements that need to be transmitted between nodes is only 7, accounting for 10.94% of the total data. The communication overhead between nodes remains relatively low, indicating that the communication cost associated with the adopted parallelization strategy does not scale significantly with increasing problem size.

Through the above parallel strategy, when the data volume is $N \times N$ elements, the computational complexity of FFT will be reduced from $\mathrm{O}(N^2 \log N)$ to $\mathrm{O}(N^2/N_{node}\log N)$. This is because the data is distributed among N_{node} nodes, and the amount of data that each node needs to calculate is only $N\times(N/N_{node})$ elements. The total data volume is $\mathrm{O}(N^2)$ complex or real numbers. Under the distributed data allocation strategy proposed in this paper, each node stores approximately $\mathrm{O}(N^2/N_{node})$ data.

Step #3: Equation (13b) is executed independently in columns at each node to derive matrix **b**, and this operation is devoid of communication overhead (refer to Fig. 3(c)).

By observing Equation (4), it is found that the calculation of the **M** matrix requires the deformation calculations of $\mathbf{B}_2$ and $\mathbf{B}_1$, and the calculation complexity is very high. Other matrix formulas involve large computational volumes, but the complexity of a single calculation is not high. Therefore, the acceleration effect provided by OpenMP is not significant. To further improve per-node parallel efficiency, OpenMP is introduced. In the implementation, we first call

omp_set_num_threads(n)

to configure the number of threads (n) in the parallel region and then insert the

#pragma omp parallel for

before Equation (1) to partition the loop into multiple sub-tasks, which are distributed by the master thread to worker threads for concurrent execution.

2.4. Data Aggregation of All Nodes

Once all nodes finish their local computations, the function *MPI_Reduce* is called to gather the scattering amplitude results calculated by each node (N_{node} denotes the total number of nodes) to the master node ('Node 0'):

$$S_i(\mathbf{k}_s, \mathbf{k}_i) = \sum_{i=0}^{N_{node}} S_i(\mathbf{k}_s, \mathbf{k}_i) \tag{14}$$

Since the number of unknowns in S is only related to the number of angle scans, the communication overhead can be ignored.

3. NUMERICAL RESULTS

To test the parallel efficiency of the proposed algorithm, a $40 \times 40\,\mathrm{m}^2$ dielectric rough surface with a relative dielectric constant of $\varepsilon_r^* = 68.846773 - 22.99446j$ is taken as an example. The incident taper wave has a frequency of 12 GHz (Ku-band) (the rough surface is $1600 \times 1600\lambda^2$ in electrical size), with a width of $1/6\times \max(L_x,\, L_y)$, where L_x and L_y are the length and width of the surface, and it is incident along $\theta_i = 30°$, $\varphi_i = 0°$. The observation angle is $\phi_s = 0°$, $\theta_s \in [-80°, 80°]$. The sampling interval is set to $\lambda/6$; therefore, the grid size is 4.17×10^{-3} m, and the number is $N_x = N_y = 9600$.

Table 1 lists the configuration of each node in the cluster used for algorithm testing.

TABLE 1. Configuration of each node.

CPU	Intel Xeon E312xx (Sandy Bridge)
N_{node}	1, 4, 8, 10
Memory	32 GB
Operating system	Windows (64-bit)

Figure 4 compares the co-polarization NRCS from serial and parallel SSA2s for 5 surfaces, once again proving that the parallelization method introduces no accuracy loss.

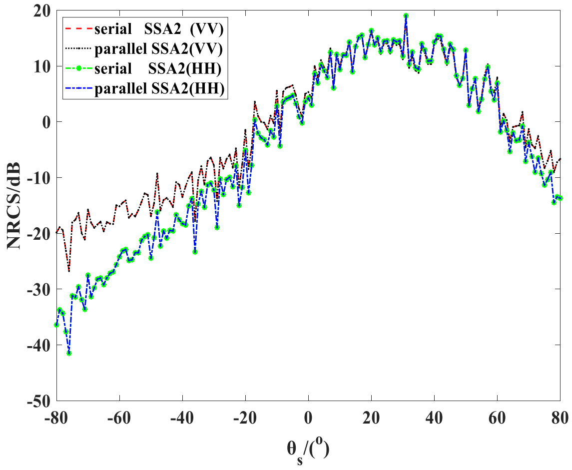


FIGURE 4. Comparison of co-polarization NRCS results between serial and parallel SSA2 methods.

As shown in Figs. 5(a) and (b), the results of the co-polarization and cross-polarization bistatic RCSs for both serial and parallel methods ($N_{node} = 10$) are compared, and they are completely consistent.

The following definitions of acceleration ratio ($S_{N_{node}}$) and parallel efficiency ($E_{N_{node}}$) are employed to evaluate the parallel performance of the proposed parallel SSA2 approach:

$$S_{N_{node}} = T_1/T_{N_{node}} \tag{15a}$$

$$E_{N_{node}} = S_{N_{node}}/N_{node} \times 100\% \tag{15b}$$

where T_1 and $T_{N_{node}}$ represent the computation times of serial and parallel algorithms, respectively.

Table 2 lists a comparison of the actual runtime of serial computing (single node) and single node with multi-thread ($N_{thread} = 16$) and multi-node configurations ($N_{node} = 4$, 8, and 10). Notably, because OpenMP can only accelerate the SSA2 during the computation of the **M** matrix, the performance gain from applying OpenMP parallelization (16 threads) alone to the serial SSA2 is very limited. Even if 16 threads are enabled, the computation time is only reduced to 30% (from 378,206 s to 116,779 s, with a parallel efficiency of only 20.24%). In contrast, the implementation of distributed parallelization strategies has brought significant improvements in computational efficiency, achieving $E_4 = 93.23\%$ ($S_4 = 3.73$), $E_8 = 88.21\%$ ($S_8 = 7.06$), and $E_{10} = 83.62\%$ ($S_{10} = 8.36$).

TABLE 2. Comparison of runtime (Unit: s).

Numerical Methods		N_{node}			
		1	4	8	10
Serial SSA2		378206	–	–	–
Parallel SSA2	OpenMP	116779	–	–	–
	MPI	–	101421	53595	45231
	MPI + OpenMP	–	37262	30088	25507

TABLE 3. Memory usage of the 'Node 0' GB.

$L_x \times L_y$/Frequency ($N_x \times N_y$)	Serial SSA2	The proposed SSA2		
		N_{node}		
		4	8	10
$50 \times 50\,\mathrm{m}^2$/12 GHz ($12000 \times 12000$)	28.13	7.71	3.79	3.36
$40 \times 40\,\mathrm{m}^2$/12 GHz ($9600 \times 9600$)	24.80	4.89	2.43	2.15
$50 \times 50\,\mathrm{m}^2$/8 GHz ($7998 \times 7998$)	17.28	3.41	1.68	1.50
$40 \times 40\,\mathrm{m}^2$/8 GHz ($6402 \times 6402$)	10.97	2.66	1.09	0.96
$50 \times 50\,\mathrm{m}^2$/4 GHz ($4002 \times 4002$)	4.29	0.78	0.43	0.38
$40 \times 40\,\mathrm{m}^2$/4 GHz ($3204 \times 3204$)	2.75	0.56	0.26	0.25

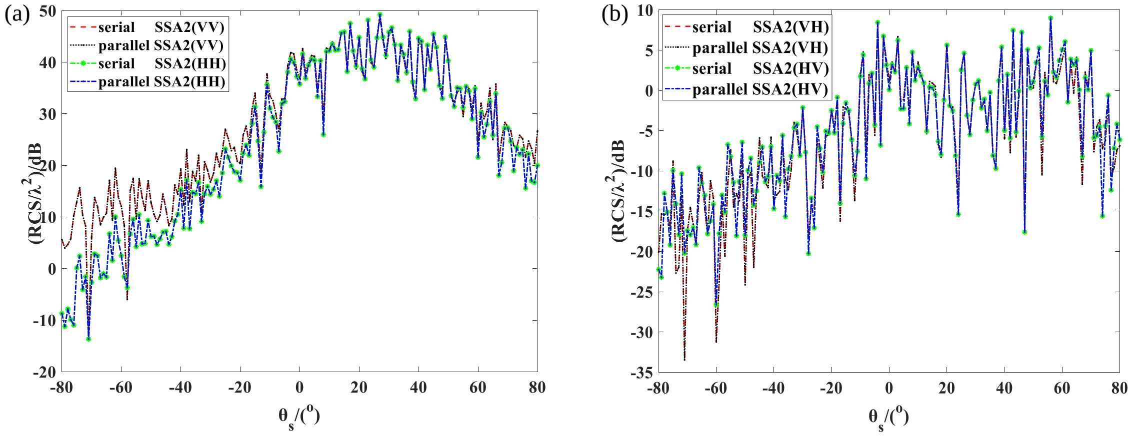


FIGURE 5. Comparison of bistatic RCS results between serial and parallel SSA2 methods. (a) Co-polarization. (b) Cross polarization.

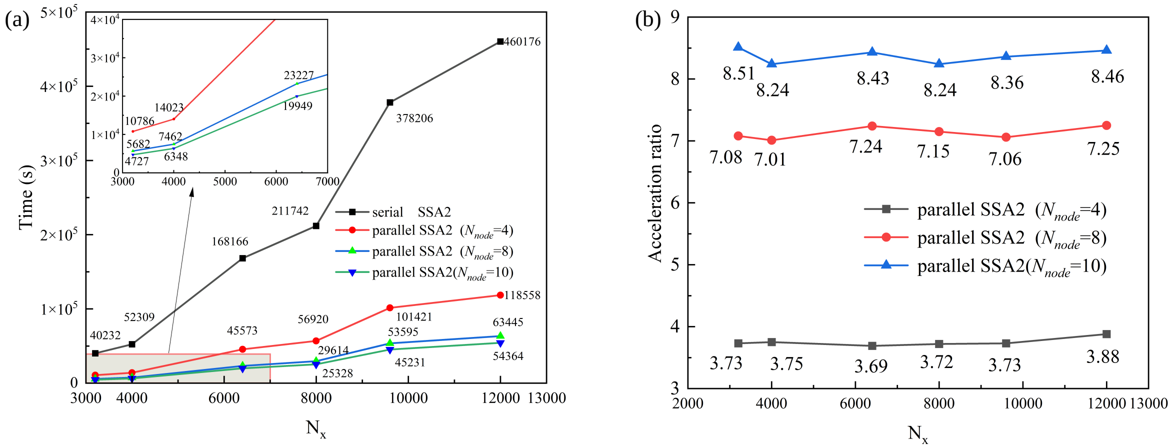


FIGURE 6. The acceleration ratio of the proposed MPI-based SSA2. (a) The time spent in multi-node mode. (b) The acceleration ratios of the multi-node mode.

To further validate the rationality of the proposed parallelization strategy, the memory usages of 'Node 0' on the $40 \times 40\,\mathrm{m}^2$ and a $50 \times 50\,\mathrm{m}^2$ rough surfaces at different incident frequencies are compared (the memory usage of 'Node 0' is usually the largest among all nodes). As shown in Table 3, with a computational grid of $12,000 \times 12,000$ points, the memory usage of the 4-node configuration is 27.4% of that of the single-node case (closely approaching the ideal value of 25%), while the 8-node and 10-node configurations exhibit memory usage of 13.5% (close to the ideal 12.5%) and 11.9% (close to the ideal 10%), respectively. These results indicate that the multi-node parallel system achieves balanced load distribution. As the number of nodes increases, the memory requirement per node decreases significantly, enabling computation over larger rough surface areas.

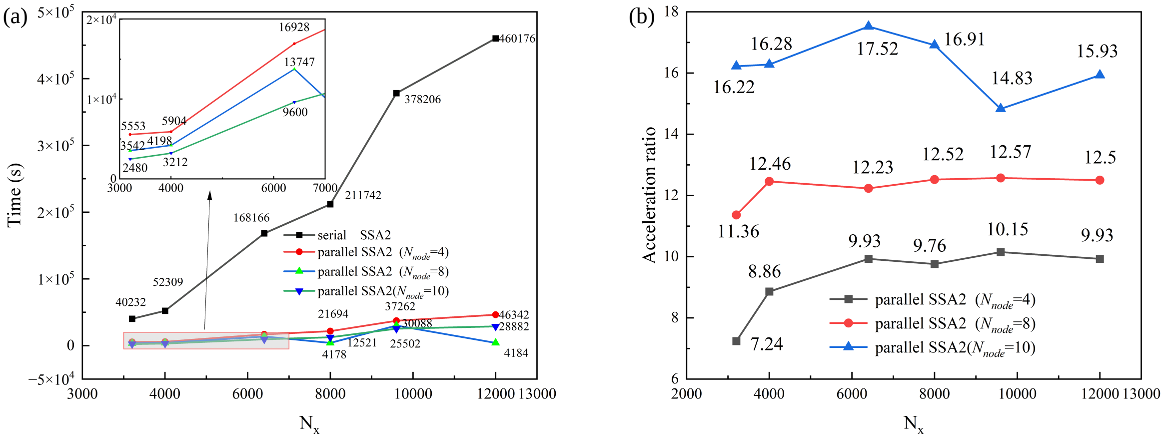


FIGURE 7. The efficiency of the proposed MPI+ OpenMP-based SSA2. (a) The time spent by the multi-node and multi-threading mode. (b) The acceleration ratios of multi-node with $N_{thread} = 16$.

Figure 6(a) illustrates a comparison of computation time across different numbers of nodes and grid sizes, clearly demonstrating that the serial computation time is substantially higher than that of parallel computation. Fig. 6(b) further reveals that the speedup ratios for the 4-node, 8-node, and 10-node configurations are approximately $S_4 = 3.72$, $S_8 = 7.15$, and $S_{10} = 8.24$, respectively, with $E_4 = 93\%$, $E_8 = 89\%$, and $E_{10} = 82\%$.

Figure 7(a) shows that, when 16 threads are enabled in multi-node configurations, the parallel computation time is only 37% of that in the multi-node single-thread configuration. According to Fig. 7(b), the speedup ratios for the 4-node, 8-node, and 10-node configurations under this setup increase to approximately $S_4 = 9.76$, $S_8 = 12.52$, and $S_{10} = 16.91$, respectively. Numerical experiments confirm the effectiveness of the proposed parallel SSA2 algorithm.

4. CONCLUSION

In this paper, a distributed parallel SSA2 algorithm based on MPI is proposed for electromagnetic scattering applications on the rough surface, and OpenMP technology is used to assist in improving the performance of the algorithm. The parallel efficiency tested under a 10-node configuration exceeded 80%. Numerical experiments have demonstrated the effectiveness of the proposed algorithm.

REFERENCES

[1] Gedney, J. J., J. T. Johnson, and R. J. Burkholder, "An analytical formulation for the coherent scattered field of a target above a randomly rough surface," *IEEE Transactions on Antennas and Propagation*, Vol. 73, No. 1, 329–340, 2025.

[2] Sefer, A., A. Yapar, and T. Yelkenci, "Imaging of rough surfaces by RTM method," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 62, 1–12, 2024.

[3] Yoshino, R., Y. Asakura, K. Inagaki, and T. Kawanishi, "300 GHz terahertz wave scattering experiment and simulation from slightly rough surfaces on dielectrics," *IEICE Communications Express*, Vol. 13, No. 2, 43–47, 2024.

[4] Moss, C. D., T. M. Grzegorczyk, H. C. Han, and J. A. Kong, "Forward-backward method with spectral acceleration for scattering from layered rough surfaces," *IEEE Transactions on Antennas and Propagation*, Vol. 54, No. 3, 1006–1016, Mar. 2006.

[5] Ozgun, O. and M. Kuzuoglu, "A domain decomposition finite-element method for modeling electromagnetic scattering from rough sea surfaces with emphasis on near-forward scattering," *IEEE Transactions on Antennas and Propagation*, Vol. 67, No. 1, 335–345, Jan. 2019.

[6] Tian, G., C. Tong, L. Xia, and P. Peng, "Electromagnetic scattering from multiple moving targets above a rough surface," *IEEE Geoscience and Remote Sensing Letters*, Vol. 19, 1–5, 2022.

[7] Hao, J.-W. and X.-Q. Sheng, "Accurate and efficient simulation model for the scattering from a ship on a sea-like surface," *IEEE Geoscience and Remote Sensing Letters*, Vol. 14, No. 12, 2375–2379, Dec. 2017.

[8] Di Martino, G., A. D. Simone, and A. Iodice, "Analytical formulation of scattering from anisotropic power-law spectrum surfaces: Getting rid of the cutoff wavenumber," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 61, 1–13, 2023.

[9] Raines, E. and J. T. Johnson, "A study of the Kirchhoff approximation for the coherent reflection coefficient of a 1-D perfectly conducting rough surface," *IEEE Geoscience and Remote Sensing Letters*, Vol. 21, 1–4, 2024.

[10] Iodice, A. and P. Imperatore, "A simple path to the small perturbation method for scattering from slightly rough dielectric surfaces," *Remote Sensing*, Vol. 16, No. 16, 3035, Aug. 2024.

[11] Di Martino, G., A. D. Simone, A. Iodice, and D. Riccio, "Bistatic scattering from anisotropic rough surfaces via a closed-form two-scale model," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 59, No. 5, 3656–3671, May 2021.

[12] Voronovich, A., "Small-slope approximation for electromagnetic wave scattering at a rough interface of two dielectric half-spaces," *Waves in Random Media*, Vol. 4, No. 3, 337, 1994.

[13] Soriano, G., C.-A. Guérin, and M. Saillard, "Scattering bytwo-dimensional rough surfaces: Comparison between the method ofmoments, Kirchhoff and small-slope approximations," *Waves in Random Media*, Vol. 12, No. 1, 63, Nov. 2001.

[14] Wang, Y. and S. L. Broschat, "A systematic study of the lowest order small slope approximation for a Pierson-Moskowitz spectrum," *IEEE Geoscience and Remote Sensing Letters*, Vol. 8, No. 1, 158–162, Jan. 2011.

[15] Wei, P.-B., M. Zhang, R.-Q. Sun, and X.-F. Yuan, "Scattering studies for two-dimensional exponential correlation textured rough surfaces using small-slope approximation method," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 52, No. 9, 5364–5373, Sep. 2014.

[16] He, Y., Y. Chen, Y. M. Wu, and P. Liu, "An improved two-scale analytical model for bistatic scattering of wind-driven sea surfaces," *IEEE Geoscience and Remote Sensing Letters*, Vol. 22, 1–5, 2025.

[17] Du, Y., J. Yin, S. Tan, J. Wang, and J. Yang, "A numerical study of roughness scale effects on ocean radar scattering using the second-order SSA and the moment method," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 58, No. 10, 6874–6887, Oct. 2020.

[18] Voronovich, A. G. and V. U. Zavorotny, "The transition from weak to strong diffuse radar bistatic scattering from rough ocean surface," *IEEE Transactions on Antennas and Propagation*, Vol. 65, No. 11, 6029–6034, Nov. 2017.

[19] Shan, J., T. Liu, J. Xie, K. Yang, and X. Xu, "FFT-accelerated 3-D near-field radar image generation for objects over rough surfaces," *IEEE Transactions on Antennas and Propagation*, Vol. 73, No. 9, 6800–6813, Sep. 2025.

[20] Jiang, W.-Q., M. Zhang, P.-B. Wei, and X.-F. Yuan, "CUDA-based SSA method in application to calculating EM scattering from large two-dimensional rough surface," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, Vol. 7, No. 4, 1372–1382, Apr. 2014.

[21] Yamac, Y. E. and A. Kizilay, "Scattering analyses of arbitrary roughness from 2-D perfectly conductiveperiodic surfaces with moments method," *Turkish Journal of Electrical Engineering and Computer Sciences*, Vol. 30, No. 3, 730–749, 2022.

[22] Wei, P.-B., M. Zhang, W.-Q. Jiang, and D. Nie, "A new model for sand-ripple scattering based on SSA method and practical ripple profiles," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 54, No. 4, 2450–2459, Apr. 2016.