Efficient and Accurate Electromagnetic Angular Sweeping of Rough Surfaces by MPI Parallel Randomized Low-Rank Decomposition

Si-Lu Huang, Wei Song, Yi-Zhuo Wang, Yu-Mao Wu, Xiao-Min Pan, and Xin-Qing Sheng

***Abstract*—A message passing interface (MPI) parallel scheme on distributed memory platforms is developed for the low-rank decomposition to accommodate the memory requirement during angular sweeping of a rough surface in terms of tapered wave incidence. Numerical examples, including that conducted on a 160×160 square wavelength rough surface, are carried out to demonstrate the performance of the proposed MPI angular sweeping with respect to accuracy, efficiency, scalability, and the peak memory requirement.**

***Index Terms*—Distributed memory parallel platform, message passing interface (MPI) parallelization, randomized low-rank decomposition, rank deficiency.**

I. Introduction

THE electromagnetic simulation of scattering from a random rough surface plays a fundamental role in many areas, such as remote sensing, target recognition, and radar surveillance [1]–[15]. The so-called analytical methods, such as the Kirchhoff approximation and the small perturbation method [16], are effective for simulating scattering from targets with rough surfaces. However, their accuracy is sometimes uncontrollable, and full-wave simulation may be required to calibrate them. In full-wave simulations, the randomness can be treated by the Monte Carlo (MC) method [13], [17], the polynomial chaos expansion method [18]–[20], or other approaches [21]. Among them, the MC method performs well when the surface variation is large.

The electromagnetic wave scattering by a large rough surface may be highly dependent on the direction of the incident wave. Angular sweeping is, thus, important to reveal the scattering characteristics of rough surfaces. Suppose the number of sweeping angles is n and the number of the random surfaces in the MC is $N_{\rm mc}$, the resource-demanding deterministic computation will be carried out $nN_{\rm mc}$ times repeatedly for a common implementation of the sweeping. Without loss of generality, assuming that the expense of the deterministic computation for one realization of the rough surface is $C_{\rm d}$ corresponding to a single incident angle, the total cost of the simulation can reach $\mathcal{O}(nN_{\rm mc}C_{\rm d})$ to obtain the scattering characteristics of the rough surfaces in terms of the same statistical roughness properties. The efficiency of a sweeping can, thus, be improved by either decreasing n, $N_{\rm mc}$ or reducing the cost of each single deterministic computation $C_{\rm d}$. Many efforts have been devoted to reduce $C_{\rm d}$ [22]–[24] as well as $N_{\rm mc}$. However, to the best of the authors' knowledge, seldom can be found in terms of n. Encouraged by the angular sweeping algorithm first proposed in [25], this work presents our attempt to improve the efficiency of the MC simulation on large-scale rough surfaces where solve each single deterministic computation. Although the skeletonization has been proved efficient in angular sweeping in terms of plane waves and Gaussian beams, its capability in accelerating the simulation in terms of tapered waves is not well documented. As it is known, tapered waves are always employed to truncate the infinite rough surfaces into finite ones. It is not clear how the truncation would impact the accuracy and performance of the skeletonization for the case of rough surface simulations. Inherently, it can be revealed by numerical studies as well as theoretical analysis. This work launches such an investigation numerically.

Essentially, the skeletonization approach proposed in [25] is based on randomized low-rank matrix decomposition [26]–[28]. Suppose $N_{\rm unk}$ is the number of unknowns for the deterministic computations, V is a right-hand-side (RHS) matrix of size $N_{\rm unk} \times n$, consisting of n RHS vectors. The randomized decomposition first estimates l, the rank of V, and then utilizes the standard procedure, i.e., pivoted QR, to conduct a decomposition on a size-reduced matrix. It is revealed that the skeletonization is cheap in runtime but expensive in memory usage [26]–[28]. The peak memory requirement of the randomized decomposition can be as high as $\mathcal{O}(N_{\rm unk} \cdot n)$. Such a memory requirement can hardly be satisfied on a single computer/server if $N_{\rm unk} \cdot n$ is large [25], [29], [30]. Compared with shared memory computers [31], distributed memory computing platforms are more

Manuscript received July 4, 2019; revised January 9, 2020 and March 3, 2020; accepted March 7, 2020. Date of publication March 30, 2020; date of current version May 7, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61771053 and Grant U-1730102, and in part by the Key Laboratory for Information Science of Electromagnetic Waves under Grant EMW201908. *(Corresponding author: Xiao-Min Pan.)*

Si-Lu Huang, Wei Song, Xiao-Min Pan, and Xin-Qing Sheng are with the Center for Electromagnetic Simulation, School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China (e-mail: 332518787@qq.com; wsong@bit.edu.cn; xmpan@bit.edu.cn; xsheng@bit.edu.cn).

Yi-Zhuo Wang is with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: frankwyz@bit.edu.cn).

Yu-Mao Wu is with the Key Laboratory for Information Science of Electromagnetic Waves, Fudan University, Shanghai 200433, China (e-mail: yumaowu@fudan.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2020.2981124

suitable for handling this difficulty because more memory space can be provided by simply adding more computing nodes [32]. MapReduce and the message passing interface (MPI) are two commonly employed parallel models for distributed memory platforms. Parallelization based on the MapReduce programming model [32] has preliminarily shown the capability of parallel low-rank decomposition. In this work, we focus on the MPI parallelization of the randomized low-rank matrix decomposition, which can be integrated smoothly with the MPI parallel multilevel fast multipole algorithm (MLFMA) [33]–[37].

Based on our preliminary studies [38], [39], this work develops a comprehensive MPI parallelization of the skeletonization, which is one of the major contributions of this work. In particular, the detailed implementation of the parallelization not available in [38] and [39] is reported in this work. Another contribution of the work, we think, is the detailed investigation of the procedure of estimating l. The studies in [38] and [39] show that the bottleneck of the MPI parallel skeletonization lies in the specific procedure employed to estimate l. Without a detailed investigation of the procedure, an efficient MPI parallelization can hardly be implemented.

The rest of this article is organized as follows. Section II introduces the formulation employed to handle the angular sweeping of each deterministic scattering problem and the efficient sequential implementation of the randomized low-rank decomposition. Section III-B discusses a modified QR decomposition (denoted by the Rank-QR) to estimate l and the proposed a parallel approach for it. Section IV presents numerical experiments demonstrating the performance of the proposed parallel randomized decomposition with respect to accuracy, efficiency, scalability, and the peak memory requirement. Section V concludes this article.

II. Angular Sweeping of a Random Surface

A. Tapered Incident Wave and Statistical Scattering

Consider the problem of electromagnetic wave scattering by a 2-D large-scale random rough surface. The height profile $z = f(x, y)$ of the rough surface is characterized by a random process. To confine the objective infinite rough surface to a finite area $L_x \times L_y$ and to avoid unwanted edge effects, the incident field is tapered as [12]

$$\mathbf{E}^{\text{inc}}(x,y,z) = (\hat{\theta}\cos\gamma + \hat{\phi}\sin\gamma)\exp(-t) \exp[-jk_0(\sin\theta_i\cos\phi_i x + \sin\theta_i\sin\phi_i y - \cos\theta_i z)(1+\omega)] \tag{1}$$

where $\hat{\theta}$ and $\hat{\phi}$ denote the polarization along θ and ϕ, respectively, $t = t_x + t_y$, $t_x = \frac{(\cos\theta_i\cos\phi_i x + \cos\theta_i\sin\phi_i y + \sin\theta_i z)^2}{g^2\cos^2\theta_i}$, $t_y = \frac{(-\sin\phi_i x + \cos\phi_i y)^2}{g^2}$, $\omega = \frac{1}{k_0^2}[\frac{(2t_x-1)}{g^2\cos^2\theta_i} + \frac{2t_y-1}{g^2}]$, and g is the parameter that controls the tapering of the incident wave. γ is the polarization angle, k_0 is the wavenumber in free space, and θ_i and ϕ_i are the incident angles. According to [40], the tapering parameter g as well as the surface lengths L_x and L_y could be determined as $\min(L_x, L_y) \geq 15 l_c$ and $\min(L_x, L_y) \geq 4\,g$, where $g_{\min} = \frac{6}{(\cos\theta_i)^{1.5}}$ and l_c is the correlation length of the Gaussian random surface.

To characterize the scattering from the rough surface, the average normalized radar cross-section (NRCS) is often employed, which is usually defined as [41], [42]

$$\sigma_{\alpha\beta} = 4\pi \frac{|\langle E_\alpha^{\text{scat}}\rangle|^2}{2\psi P_\beta^{\text{inc}}} \ (\alpha, \beta = p_H, p_V) \tag{2}$$

where p_H and p_V are the horizontal and vertical polarizations, respectively. P_β^{inc} is the real incident power, and ψ is the characteristic impedance of the background. The symbol $\langle\cdot\rangle$ indicates an ensemble average over realizations of the stochastic process for the model.

B. Fast Angular Sweeping

For simplicity, assuming the rough surface is a perfect electrical conductor, the associated method of moments (MoM) matrix system with multiple excitations has the form

$$Z \cdot I(\kappa) = V(\kappa) \tag{3}$$

where κ is the set of parameters used to describe an excitation, and Z is the impedance matrix in the MoM. The matrix $I(\kappa)$ is the corresponding solution matrix. We will drop κ in the following discussion for simplicity.

As mentioned previously, the efficiency of the solution of (3) is one of the major difficulties in angular sweeping. To solve this problem, we can conduct a randomized low-rank decomposition [27] of V in (3) as $V = V^{\text{S}} \cdot P$ and then solve the multiple-RHS system as [25]

$$I = I^{\text{S}} \cdot P \text{ and } Z \cdot I^{\text{S}} = V^{\text{S}} \tag{4}$$

where V^{S} is the $m \times l$ skeletonized RHS matrix with $l \leq \min(m, n)$. The l RHS vectors in V^{S} are referred to as the skeleton RHS vectors or *skeleton RHSs* for conciseness in this work. Matrix P is the $l \times n$ projection matrix. Matrix I^{S} is the $m \times l$ skeletonized solution matrix, consisting of the solution vectors related to the skeleton RHSs. To accelerate the iterative solution, an algebraic preconditioner [43], [44] can be employed.

It has been shown in [25] that the abovementioned skeletonization scheme was strictly controlled by a threshold during the factorization when applying to the plane wave sweeping of deterministic target. As it is known, due to the employing of the tapered wave, the infinite rough surface is truncated into a finite one. Investigations are necessary to reveal how such truncation would impact the accuracy control and, thus, the performance of the skeletonization scheme is not known yet. Generally, such investigations can be carried out both numerically or analytically. In this work, we launch such study numerically. The theoretical proof of it will be presented in our future work.

C. Main Idea of Skeletonization

To realize the skeletonization in (4), an efficient algorithm to take advantage of randomness is employed. Here, we give a brief introduction of it. It is assumed that $V(i_1 : i_2, j_1 : j_2)$ refers to a submatrix formed by extracting rows indexed from i_1 to i_2 and columns indexed from j_1 to j_2 of V. Thus, $V(:, j)$ is a column vector corresponding to the jth column of V. The skeletonization begins with generating a sampling matrix Y of

dimension $m_2 \times n$ by the random linear combination of the rows of V, where m_2 is the greatest integer less than or equal to m, which is a power of two [27]. In particular, Y is obtained by

$$Y^{m_2 \times n} = \Omega^{m_2 \times m} \cdot V^{m \times n} \tag{5}$$

where Ω consists of uniformly randomly selected rows of the product of the discrete Fourier transform matrix and a random diagonal matrix, as pointed out in [27]. Using the algorithm in [45], which is based on the fast Fourier transform, the matrix-vector multiplication (MVM) in (5) can be quite efficient in practical applications.

After the randomized sampling matrix Y is obtained, we can approximate l, the row rank of V, by using a modified QR decomposition, which will be discussed in Section III-A. With the estimated l, the low-rank decomposition can be then carried out on a matrix of size $l \times n$ by the standard algorithm, i.e., pivoted QR in this work. From previous studies [25], [39], it is clear that the skeletonization is very efficient in terms of CPU time but requires about $\mathcal{O}(mn)$ memory space. This highlights the need for developing parallel randomized matrix decomposition on distributed memory platforms that can meet the memory requirement.

Algorithm 1: The sequential Rank-QR.

Input: $X^{n \times m_2}, \varepsilon_{\text{LRD}}, \sigma_{\max}$
Output: l
1 $\sigma = \|X(:,1)\|_2$;
2 $i = 0$;
3 **while** $\sigma > \sigma_{max} * \varepsilon_{LRD}$ **do**
4 $\quad i = i + 1$;
5 $\quad$ ==Update column $X(:,i)$ as:
6 $\quad$ **for** $j = 1$; $j \leq i - 1$; j ++ **do**
7 $\quad\quad$ Produce v^j as, $v^j(1) = 1$ and $v^j(2 : n - j + 1) \leftarrow X(j + 1 : n, j)$;
8 $\quad\quad$ Update $X(j : n, i)$ by **happ**$[X(j : n, i : i),\ v^j]$;
9 $\quad$ **end**
10 $\quad$ Compute v^i by **htrans**$[v^i,\ X(i : n, i)]$;
11 $\quad$ Update $X(i : n, i)$ by **happ**$[X(i : n, i : i),\ v^i]$;
12 $\quad$ Store the Householder vector v^i as Eq. (7);
13 $\quad \sigma = |X(i,i)|$;
14 **end**
15 $l = i$;

III. Rank-QR and Its MPI Parallelization

The estimation of l is essential to make the skeletonization in (4) efficient. But the procedure can become the bottleneck during the decomposition when the size of the matrix is large [25], [39]. The solution is to carry out the decomposition on distributed memory platforms. In the following, The modified QR is referred to as Rank-QR as its aim is to estimate the rank. For convenience, the traditional QR is denoted by Trad-QR for short. In this section, we will explain the Rank-QR in sufficient detail and then present our proposed efficient MPI parallel strategy for it.

A. Sequential Rank-QR

Similar to Trad-QR, Rank-QR assumes the column rank is of interest. However, the skeletonization discussed here wants the row rank of V. For this reason, the task of estimating l is conducted on $(Y^{m_2 \times n})^T$, denoted by $X^{n \times m_2}$ or X in short in what follows, where the superscript T means a matrix transpose operation.

The Rank-QR is summarized in Algorithm 1, where the definitions of functions **htrans** and **happ** are presented in the Appendix. In Algorithm 1, ε_{LRD} is the threshold to control the accuracy of the randomized low-rank decomposition, and $\sigma_{\max}$ is the maximum of the two-norms of all columns in V. To clearly show the difference between Rank-QR and Trad-QR, an implementation of Trad-QR is also listed in Algorithm 5 in the Appendix.

Rank-QR differs from Trad-QR in the following aspects. First, the update procedure is different. In Rank-QR, the ith update procedure is conducted by applying H^i, the Householder matrix associated with the ith column $X(:,i)$, to one column of X through a call of **happ**, as depicted by Line 8 and Line 11 in Algorithm 1. In contrast, Trad-QR implements the update by applying matrix H^i to the associated submatrix X as shown in Algorithm 5. Second, Trad-QR explicitly computes and stores Q and R, while in Rank-QR, the Q and R matrices are not explicitly computed nor stored since the purpose of the latter decomposition is only to find the rank. The third difference lies in the stopping criterion for the loop. In Rank-QR, the factorization stops when the singular value becomes smaller than the required accuracy, while the computation will not stop until the entire matrix is factorized in Trad-QR.

If X is a full-rank matrix, the computational cost of Rank-QR is identical to that of Trad-QR. However, Rank-QR is more efficient than Trad-QR in estimating l if X is rank deficient because the factorization terminates as soon as l is determined. Therefore, the computations associated with the remaining $(m_2 - l)$ columns in X are not conducted as they become unnecessary and irrelevant.

B. MPI Parallel Rank-QR

Although the parallelization on some procedures of the randomized decomposition is quite straightforward [38] or is readily available in ScaLAPACK [46], it is not the case for the Rank-QR procedure due to the unique characteristics of it. In the following, we will present our parallelization of Rank-QR, where the number of processes is assumed to be n_p.

To overcome the memory bottleneck, it is mandatory to distribute X among different processes. As discussed in Section III-A, the Householder vector v^i cannot be generated before the preceding $(i - 1)$ rounds of the Householder transformations are applied to the column $X(:,i)$. If X is partitioned in a column-major manner, the QR factorization cannot take advantage of the parallelization due to the inherent column-major sequence of the factorization, as described in Algorithm 1. With this limitation, we have to distribute X in a row-major manner by assigning

Algorithm 2: The MPI Block-Wise Rank-QR.

Input: $X^{n\times m_2}, \varepsilon_{\text{LRD}}, \sigma_{\max}, n_u$
Output: l

```
1  σ = ‖X(:,1)‖_2;
2  m_s = 1;
3  flag = 1;
4  while (σ > σ_max * ε_LRD) && (flag == 1) do
5      m_t = min(m_s + n_u − 1, m_2);
6      ==Update the submatrix X(:, m_s : m_t) as:
7      for j = 1; j ≤ m_s − 1; j ++ do
8          Produce v^j as, v^j(1) = 1 and
             v^j(2 : n − j + 1) ← X(j + 1 : n, j);
9          Update X(j : n, m_s : m_t) by
             happ[X(j : n, m_s : m_t), v^j] ;
10     end
11     ==Compute the Householder transformation as:
12     for q = m_s; q ≤ m_t; q ++ do
13         if flag == 1 then
14             Compute v^q by htrans[v^q, X(q : n, q)] ;
15             Update X(q : n, q : m_t) by
                 happ[X(q : n, q : m_t), v^q] ;
16             Store v^q as Eq. (7);
17             σ = |X(q, q)| ;
18             if σ ≤ σ_max * ε_LRD then
19                 flag = 0;
20             end
21         end
22     end
23     m_s = m_s + n_u;
24 end
25 l = q;
```

n/n_p rows of X to each process. However, the partition leads to two types of global communication.

One type of communication occurs in the function **htrans** because each process can access only its own share of X from the local memory. With the assumption that the rank is l, the total number of these communications is proportional to l. The second type corresponds to the MVMs during the updating procedure in Algorithm 1. Specifically, column $X(:,i)$ involves at least $(i-1)$ communications because the number of Householder transformations is $(i-1)$. Therefore, the total number of global communications of the second type will be at least $l(l-1)/2$. It would, thus, be challenging to complete the decomposition within an acceptable time when n_p becomes large, despite the efficiency of the sequential Rank-QR. From abovementioned analysis, it is clear that the second type of communication requires a larger number of data transfers than the first one and, therefore, the poor efficiency is mainly caused by the former. For this reason, we will focus on the second type of communication in the following.

To alleviate this difficulty, we propose a solution by updating X in a blockwise manner, as shown in Algorithm 2. For simplicity, Algorithm 2 does not present the MPI communications. In the blockwise approach, the updating procedure proceeds with n_u columns each time, as depicted by Line 9 and Line 15 in Algorithm 2. After n_u columns in one block have been successfully handled, the factorization proceeds to the next block. Here, n_u is a user-defined parameter. If $n_u = m_2$, Rank-QR becomes identical to Trad-QR. Generally, the number of communications of the second type is $l(l+n_u-2)/2n_u$ if $l > n_u$ in the blockwise factorization.

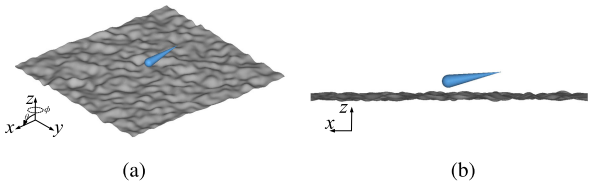
It goes without saying that the number of blocks is equal to $n_b = \lceil l/n_u \rceil$. The columns contained in the computed blocks that should be updated can be classified into *relevant* and *irrelevant* columns. The former contributes to l, whereas the latter does not. The numbers of relevant and irrelevant columns equal l and $n_{ir} = \lceil l/n_u \rceil n_u - l$, respectively. Because the corresponding computations are unwanted, the number of irrelevant columns should be decreased to improve the efficiency. On the one hand, a larger n_u may give rise to a larger number of irrelevant columns and, in turn, a less efficient computation, just as in Trad-QR. From this point of view, we should use a small n_u. On the other hand, the larger n_u is, the fewer communications required. A large n_u is preferable in terms of reducing communications. The most optimal case is to set $n_u = l$. However, l is unknown in advance. Therefore, it is difficult to find a universal rule for choosing n_u. Fortunately, for many applications, it is possible to achieve a good estimation of the upper bound of l before the decomposition starts. For example, when selecting skeleton excitations [25], [30], the upper bound of l can be estimated from the truncation number in Addition Theorem. In these circumstances, the parallel Rank-QR can be efficient because finding a suitable n_u is always a relatively easy task. In addition, as will be shown in Section IV, the performance of the proposed approach is insensitive to n_u, at least for the matrices being investigated in this work. The parallel efficiency can be very high when n_u varies within a wide range.

The blockwise approach proposed here is different from the blocked QR factorization discussed in [47], where a standard QR factorization generating an orthogonal matrix and an upper triangular matrix was discussed. However, the approach here aims to estimate the rank of the matrix in question.

It should be noted that the Householder matrix H^i is not explicitly computed or stored [48]. Simultaneously, the Householder vector v^i can be stored in the lower triangular position of X. The memory usage can, thus, be reduced by a factor of about n_p as each process contains only n/n_p rows of X. It is evident that the proposed parallel strategy has fully exploited the potential of a distributed platform in overcoming the memory bottleneck.

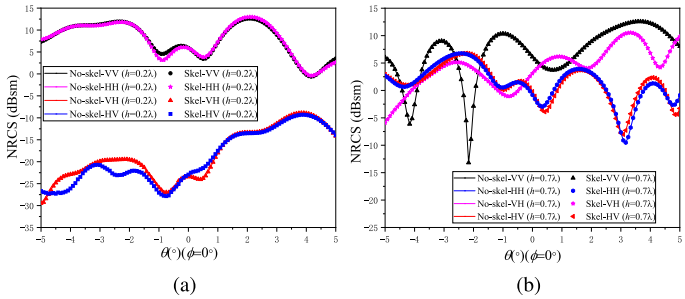
IV. Numerical Results

In this section, the performance of the parallel scheme is investigated by the computations perfectly electrically conducting (PEC) random rough surface models with or without a target positioned above the surfaces, as shown in Fig. 1(a). To show the versatility of the proposed method, we employ two kinds of rough surfaces, i.e., Gaussian rough surfaces and Pierson–Moskowitz (PM) spectrum rough surfaces. The models are homogeneously meshed and the number of unknowns is



(a) (b)

Fig. 1. Illustration of a PEC cone-sphere on a PEC rough surface model (only the Gaussian surface is shown here). (a) Perspective view. (b) Side view.



(a) (b)

Fig. 2. Monostatic NRCS results for VV-, HH-, and cross polarization from the Gauss-RS model obtained with and without the MPI parallel randomized decomposition, denoted by "Skel" and "No-skel", respectively. Here, we set $\theta \in [-5^\circ, 5^\circ]$ with $\phi = 0^\circ$. (a) $h = 0.2\,\lambda$. (b) $h = 0.7\,\lambda$.

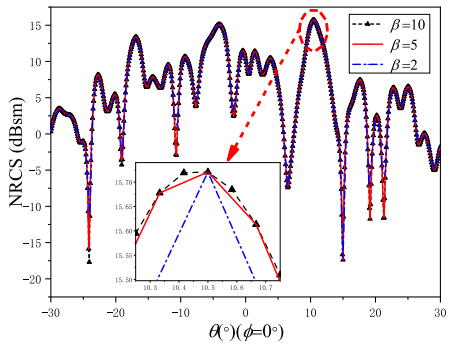


Fig. 3. VV-polarized monostatic NRCS results from the Gauss-CS model with different values of β. Here, we set $\theta \in [-30^\circ, 30^\circ]$ with $\phi = 0^\circ$.

denoted by N_{unk} in this section. It should be pointed out that we employ the strategy of finding skeleton Rao–Wilton–Glisson functions [30] to generate a size-reduced excitation matrix (of dimension $m \times n$ with $m < N_{\text{unk}}$) that feeds into the randomized low-rank decomposition. All the skeletonizations are conducted upon $\varepsilon_{\text{LRD}} = 10^{-4}$, consistent with common engineering practices [25], [30].

The associated computations are performed on a computing platform where each node is equipped with 94 GB of memory and two 6-core X5650 CPUs. In the computations, at most 8 nodes (96 CPU cores) are employed and the decomposition incorporates with the MPI parallel MLFMA to obtain the NRCS results [39].

In what follows, we begin with studying the accuracy of the proposed method in terms of the tapered incident waves in Section IV-A. Then, we discuss the parallelization of the Rank-QR in Section IV-B, including the selection of n_u, the weak scalability as well as the strong scaling. Finally, the capability of the proposed method is studied in Sections IV-C.

A. Accuracy

First of all, computations are conducted on the two Gaussian rough surfaces (Gauss-RSs). Except for the root-mean-square height (h), all other parameters of the surfaces are identical. The incident angles are less than 30° for both θ and ϕ, and accordingly, the area of each rough surface is $L_x \times L_y = 32\lambda \times 32\lambda$, with λ denoting the wavelength. The tapering parameter is $g = L_x/4 = 8\lambda$. Each rough surface has the correlation lengths of $l_x = l_y = 1.2\lambda$. The h's are $0.2\,\lambda$ and $0.7\,\lambda$, respectively. Both surfaces are homogeneously meshed leading to $N_{\text{unk}} =$ 353 876 and 706 914. Accordingly, $m =$ 63 102 and 110 973. Fig. 2 gives the monostatic NRCS for VV-, HH-, and cross-polarization within the angular range $\theta \in [-5^\circ, 5^\circ]$, $\phi = 0^\circ$. As shown in the figures, results obtained with and without MPI parallel skeletonization ($n_p = 12$) agree very well for both cases.

Theoretically, the cross-polarized NRCS results should be identical according to the reciprocity relation. Because of the error caused by tapered wave truncation, our numerical results (not presented here) show that the reciprocity relation is observed in the sense of the statistical respect but not for each single realization of the rough surface. This may be one reason leading to the difference between the cross-polarized NRCSs. Additionally, the coarse mesh used for the computation could give rise to numerical errors of MoM and MLFMA, which could also result in the deviation of cross-polarized NRCS results.

The accuracy is then investigated in terms of different n's. According to [25], the sampling rate is computed by βL, with L being the truncation number in Addition Theorem and β a superparameter. To reach a more strict study, a PEC cone-sphere is placed at an altitude of 1 λ above the Gauss-RS with $h = 0.2\,\lambda$, as shown in Fig. 1(b), which is denoted by *Gauss-CS* in the following. The main body of the cone-sphere is identical to its counterpart in [49] but with the largest dimension of 6.8 λ. For this example, $N_{\text{unk}} =$ 362 021 and $m =$ 64 115.

We compare the VV-polarized monostatic NRCS results from the Gauss-CS model with different values of β, namely, 2, 5, and 10, as presented in Fig. 3. Here, we set $\theta \in [-30^\circ, 30^\circ]$ and $\phi = 0^\circ$. Accordingly, the angular steps are $1^\circ/2$, $1^\circ/6$, and $1^\circ/12$, where n reaches 121, 361, and 721. As indicated by Fig. 3, all cases capture the major oscillations of the NRCS results. However, the case $\beta = 10$ delivers a higher spatial resolution than the cases $\beta = 5$ and $\beta = 2$. It should be noted that all computations consume almost the same amount of runtime because of two facts. One fact is that the number of skeleton RHSs required to be solved iteratively is a constant of 40 [25] for computations with different β's. The other fact is that the runtime for the skeletonization is negligible in comparison with the total solution time. As a result, the case $\beta = 10$ can reach a higher resolution without additional runtime. Therefore, $\beta = 10$ is a suitable choice with respect to the tapered incident waves and is employed in this work.

TABLE I
RUNTIME (S) OF THE RANK-QR AS A FUNCTION OF n_u FOR THE GAUSS-CS MODEL WHERE THE EXCITATION MATRIX IS OF DIMENSION $64\,115 \times 14\,641$

n_u	n_b	n_{ir}	Time (s)
14 641	1	14 572	16.4
160	1	91	1.6
80	2	91	1.5
10	7	1	1.5
1	69	0	1.8

n_b denotes the number of blocks.
n_{ir} denotes the number of irrelevant columns.
The estimated rank l is 69.
In the computations, $n_p = 24$.

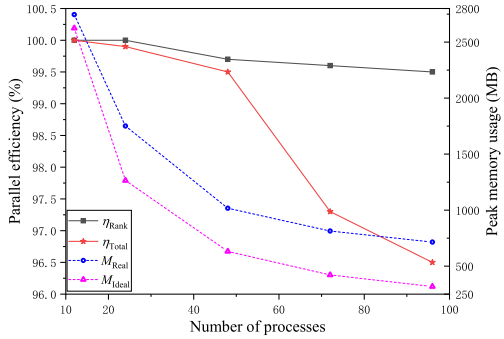


Fig. 4. Parallel efficiency and peak memory requirement (MB) of individual process for the strong scaling study where $m \times n = 64\,115 \times 14\,641$.

TABLE II
RUNTIME (S) AND THE PEAK MEMORY REQUIREMENT (MB) OF INDIVIDUAL PROCESS FOR THE WEAK SCALING STUDY

n_p	θ	n	k	Time (s)		Mem. (MB)
				T_r	T_t	
12	$[-5^\circ, 5^\circ]$	3 025	27	0.5	4.3	912
24	$[-10^\circ, 10^\circ]$	6 025	45	0.6	4.6	912
48	$[-20^\circ, 20^\circ]$	12 025	87	0.7	5.2	913
72	$[-30^\circ, 30^\circ]$	18 025	130	1.0	5.7	917

T_r denotes runtime of Rank-QR.
T_t denotes runtime of the whole low-rank decomposition.
In the computations, n keeps proportional to n_p while $n_u = 30$.

B. Parallelization

1) Selection of Parameter n_u in Parallel Rank-QR: The performance of the blockwise approach discussed in Section III-B and the selection of n_u are studied by computations of the Gauss-CS model. We generate the excitation matrix by choosing the directions of the incident waves within the range of $\theta \in [-5^\circ, 5^\circ], \phi \in [-5^\circ, 5^\circ]$ with angular steps of $\Delta\theta = \Delta\phi = 1^\circ/12$. Therefore, the dimension of the associated excitation matrix is $m \times n = 64\,115 \times 14\,641$. In particular, we will reach a rule for selecting n_u according to the runtime of the parallel Rank-QR by varying n_u. By setting $D = 32\lambda$, the estimated number of skeleton RHSs is no larger than 160 according to the truncation number in addition theorem [50]. We set a relatively large n_p in comparison with 160, i.e., $n_p = 24$, in the computations to realize a strict investigation of the parallel Rank-QR.

Table I enumerates the computational configurations as well as the runtime. Here, the estimated rank l is 69. As shown in Table I, the performance of the proposed approach varies obviously with n_u only when n_u is relatively large or is very small. In other words, n_u can deliver good performance within a wide range, namely, $[10, 80]$. The reason lies in the MPI parallelization. Because the irrelevant tasks are distributed among the processes, the associated runtime is reduced. Consequently, the computation performance is not appreciably affected by n_u. As a result, the choice of n_u is not critical. For the other examples not presented here, a similar conclusion can be reached.

2) Parallelization Scalability and Memory Requirement: In the context of high-performance computing, there are two common notions of scalability. The first is strong scaling, which is defined as how the solution time varies with n_p for a fixed total problem size. The second is weak scaling, which is defined as how the solution time varies with n_p for a fixed problem size per process. Since the peak memory requirement is the most important factor in our study, the peak memory usage for a single process is investigated as well, which is denoted by M_{Real}. For a comprehensive view, the ideal memory usage is estimated and denoted by M_{Ideal}. Meanwhile, we denote η_{Rank} as the parallel efficiency of the Rank-QR and η_{Total} as the parallel efficiency of the total randomized low-rank decomposition. The parallel efficiency is computed by $\frac{n_r T_{n_r}}{n_p T_{n_p}} \times 100\%$, where T_{n_r} and T_{n_p} are the runtimes when n_r and n_p processes are employed, respectively. Because of the complicated hardware architecture of the computing node, the cache hit rate will impact the MPI parallel efficiency significantly. To simplify the associated analysis, we set n_r to 12.

The strong scaling is studied by computations of the Gauss-CS model, where the same excitation matrix as described in Section IV-B1 is used. According to Section IV-B1, we set n_u to 40. Here, n_p varies from 12 to 96. When $n_p = 12$, the Rank-QR costs 3.0 s while the whole decomposition completes in 24 s. The peak memory usage is approximately 2745 MB in each process. Fig. 4 presents the parallel efficiencies and peak memory requirements for the computations. As presented in the figure, η_{Rank} is over 99% for all computations. Due to this fact, η_{Total} is about 96% when 96 processes are employed. The study shows that the parallelization performs well in terms of strong scaling. Simultaneously, the peak memory usage in an individual process decreases significantly with the increase in n_p. In particular, it is only approximately 716 MB when 96 processes are employed. With the remarkable memory reduction, the memory bottleneck can be successfully addressed by the distributed memory parallelization. It is indicated that the actual memory requirement is always larger than the ideal estimation. The difference arises from the additional memory required to establish the MPI parallel environment.

Weak scaling and the memory requirement are studied by multiple computations of the Gauss-CS model, ensuring that the size of the submatrix assigned to each process remains constant. In particular, m is fixed at 64 115, while n is increased in direct proportion to n_p, as shown in Table II. The excitation matrices are generated by keeping $\phi \in [-1^\circ, 1^\circ]$ while changing the range

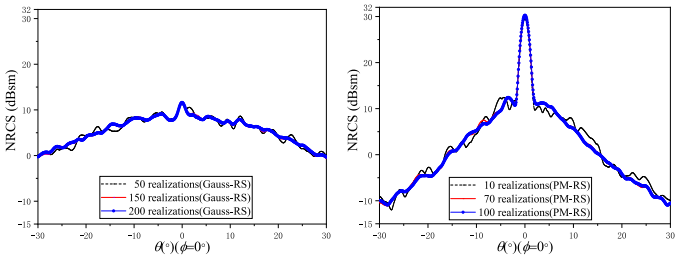


Fig. 5. VV-polarized average monostatic NRCS results ($n = 721$). Here, we set $\theta \in [-30^\circ, 30^\circ]$ with $\phi = 0^\circ$. (a) Gauss-RS model. (b) PM-RS model.

TABLE III
RUNTIME AND THE PEAK MEMORY REQUIREMENT FOR THE LARGE-SCALE ROUGH SURFACE MODELS

	Time		Mem.
	T_r	T_i	(MPI Rank-QR)
Large-PM-RS-3	41.4 s	8.7 h	5.7 GB
Large-PM-RS-5	44.3 s	10.1 h	5.7 GB

T_r and T_i denote runtime of Rank-QR and iteration procedure, respectively.
The memory usage of the iteration procedure is approximately the same as that of MPI rank-QR.
In the computations, $n_u = 175$ and $n_p = 60$.

of the incident angle along θ. The angular steps are fixed as $1^\circ/12$, and $n_u = 30$. As indicated by Table II, the runtime grows when n_p increases because of the growth of k. Simultaneously, the peak memory requirement of the parallel skeletonization remains unchanged. In particular, the peak memory usages in terms of each single process are approximately 912 MB when $n_p = 12$ and 917 MB when $n_p = 72$. This set of experiments validates the good weak scaling of the MPI parallel Rank-QR. In addition, the peak memory requirement will no longer be a bottleneck.

C. Stochastic Characteristics of Scattering From Rough Surface

Here, we investigate the scattering properties of Gaussian rough surfaces and PM rough surfaces (PM-RSs) by calculating the NRCS within the angular range $\theta \in [-30^\circ, 30^\circ]$ $(\phi = 0^\circ)$. The Gauss-RS model with $h = 0.2\,\lambda$ in Section IV-A is employed. The PM-RS corresponds to a sea surface under the wind speed is 3 m/s, where $L_x \times L_y = 32\lambda \times 32\lambda$, $h = 0.1\,\lambda$, $N_{\text{unk}} = 315\,122$, and $m = 71\,517$. According to the study in Section IV-A, we choose $\beta = 10$, $\Delta\theta = 1^\circ/12$ and $n_p = 12$.

It is known that a number of realizations should be taken in the MC simulation to obtain the stochastic of the scattering. To measure convergence in terms of the number of realizations (samples), namely, N_{mc}, we present the average NRCS results against N_{mc} in Fig. 5. As shown in the figure, the peak NRCS of the Gauss-RS is lower than that of the PM-RS. More specifically, it is approximately 12 dB for the former but 30 dB for the latter. The different behaviors of the NRCS can be explained by the roughness of the two surfaces. In our configuration, the Gauss-RS is rougher than the PM-RS. Consequently, in comparison with the latter, the former should deliver stronger scattering associated with directions off the incidence angle while weaker backward scattering. Additionally, it is expected that the rougher the surface, the larger N_{mc} is required to obtain a converged stochastic scattering. The expectation is validated by our computations. As shown in the figure, the PM-RS can reach the convergent NRCS results in about 100 realizations while the Gauss-RS needs about 200 samples. The overall computation costs approximately 77 h for the Gauss-RS (200 realizations) and 56 h for the PM-RS (100 realizations). Without the proposed method, the runtime should be increased by a factor about $721/40 \approx 18$ for both models since the repetition associated with n is reduced from 721 to 40 in our simulations. From this set of computations, we can find that improving the efficiency on each realization is essential to obtain the stochastic scattering in acceptable time.

The capability of the proposed parallelization is further indicated by computations on the NRCS for a wide angular range, specifically, $\theta \in [-60^\circ, 60^\circ]$ $(\phi = 0^\circ)$. Accordingly, the area of the rough surfaces is increased to $L_x \times L_y = 160\,\lambda \times 160\,\lambda$. The tapering parameter of the incident is $g = L_x/4 = 40\,\lambda$. Two PM-RS models are employed with the wind speed of 3 m/s ($h = 0.1\,\lambda$) and 5 m/s ($h = 0.3\,\lambda$). We denote these two models by *large-PM-RS-3* and *large-PM-RS-5*, respectively. According to Addition Theorem and the study in previous sections, we set $\Delta\theta$ to be $1^\circ/60$, where n reaches 7201. As a result, for the large-PM-RS-3 case, $N_{\text{unk}} = 7865\,152$ and $m \times n = 1\,487\,269 \times 7201$, meanwhile, for the large-PM-RS-5 case, $N_{\text{unk}} = 8\,116\,327$ and $m \times n = 1\,490\,754 \times 7201$. By setting $n_u = 175$, 290 skeleton RHSs are figured out for both models. Table III presents the resources required by the Rank-QR and the iteration procedure when $n_p = 60$. On one hand, although it is reduced by a factor about $7201/290 \approx 25$ with the aid of the skeletonization, the runtime of the iteration procedure is still relatively long, reaching 8.7 and 10.1 h, respectively, due to the large scale nature of the computations. These results further validate the necessity of acceleration technique during angular sweeping of large-scale rough surfaces. On the other hand, the runtime required by the parallel Rank-QR is approximately 1 min, which makes the MPI skeletonization efficient in runtime. Meanwhile, the peak memory usage of an individual process is only approximately 5.7 GB. The memory bottleneck is successfully alleviated by our MPI parallelization.

Fig. 6 shows the NRCS results by the large-PM-RS-3 model and large-PM-RS-5 model with $\theta \in [-60^\circ, 60^\circ]$ and $\phi = 0^\circ$. The trend of the NRCS is shown by the moving average approach to smooth out the fluctuation, as shown in Fig. 6(b). The number of sampling angles in the moving window is 80, a little larger than 1°. Comparing with the large-PM-RS-3 case, the large-PM-RS-5 case is rougher. Thus, the former has stronger backward scattering. For both cases, the NRCS results oscillate fast, which leads to the fact that a much finer angular step is required for large-scale rough surface to capture the oscillation.

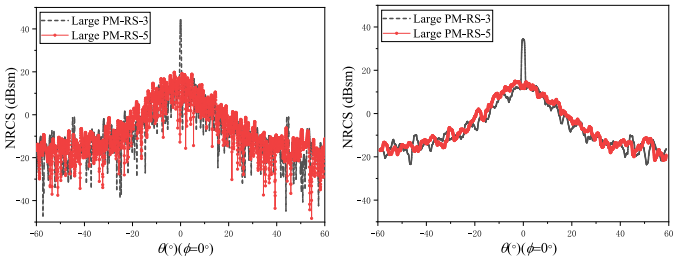


Fig. 6. VV-polarized monostatic NRCS results by the large PM-RS-3 model and the large PM-RS-5 model. Here, we set $\theta \in [-60^\circ, 60^\circ]$ with $\phi = 0^\circ$. (a) NRCS results. (b) Moving average of NRCS results.

V. Conclusion

An MPI parallel Rank-QR, is developed based on distributed memory platforms to address the memory bottleneck during the angular sweeping of rough surfaces. The applicability of the proposed method in terms of a tapered wave incidence is demonstrated. Numerical experiments show that the proposed parallelization can solve the memory bottleneck in the randomized decomposition for a large-scale matrix with high parallel efficiency. Particularly, for the computation of a rough surface that is $L_x \times L_y = 160\,\lambda \times 160\,\lambda$, the peak memory usage in a single process is approximately 5.7 GB when 60 processes are employed.

Appendix

In the following Algorithms, I_n is the $n \times n$ identity matrix, $\|\cdot\|_2$ gives the two-norm of a vector and † denotes the conjugate transpose operation.

1) **htrans**$[v^i,\ X(i:n,i)]$: compute the Householder vector v^i corresponding to the $X(i:n,i)$.
2) **happ**$[X(i:n, j_1:j_2),\ v^i]$: apply the Householder transformation associated with v^i to X to obtain an updated X.

Algorithm 3: **htrans**$[v^i,\ X(i:n,i)]$.

Input: $X(i:n,i)$
Output: v^i

$n_i = n - i + 1$;
$v^i(1) = 1$;
$v^i(2:n_i) \leftarrow \frac{X(i+1:n,i)}{X(i,i) - \|X(i:n,i)\|_2}$;

Algorithm 4: **happ** $[X(i:n, j_1:j_2), v^i]$.

Input: $v^i, X(i:n, j_1:j_2)$
Output: $X(i:n, j_1:j_2)$

$$X(i:n, j_1:j_2) \leftarrow [I_{n-i+1} - \frac{2}{(v^i)^\dagger \cdot v^i} v^i \cdot (v^i)^\dagger] \cdot X(i:n, j_1:j_2) \quad (6)$$

In Algorithm 5, matrix R is the upper triangular part of the updated X.

Algorithm 5: The Sequential Trad-QR.

Input: $X^{n\times m_2}$
Output: $Q^{n\times n}$ and $R^{n\times m_2}$

for $i = 1;\ i \le m_2;\ i$ ++ **do**
  Compute v^i by **htrans**$[v^i, X(i:n,i)]$;
  Update X by **happ**$[X(i:n, i:m_2),\ v^i]$;
  Store the Householder vector v^i as:
  if $i < n$ **then**

$$X(i+1:n,i) \leftarrow v^i(2:n-i+1) \quad (7)$$

  end
end
Compute $Q^{n\times n}$ as:

$$Q = \prod_{i=1}^{m_2} H^i, \text{ where } H^i = I - \frac{2}{(v^i)^\dagger \cdot v^i} v^i \cdot (v^i)^\dagger \quad (8)$$

References

[1] T. Qiao *et al.*, "Sea surface radar scattering at L-Band based on numerical solution of Maxwell's equations in 3-D (NMM3D)," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 6, pp. 3137–3147, Jun. 2018.
[2] Y. Liu, K. Chen, Y. Liu, J. Zeng, P. Xu, and Z. Li, "On angular features of radar bistatic scattering from rough surface," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 6, pp. 3223–3235, Jun. 2017.
[3] L. X. Guo and R. W. Xu, "An efficient multiregion FEM-BIM for composite scattering from an arbitrary dielectric target above dielectric rough sea surfaces," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 7, pp. 3885–3896, Jul. 2015.
[4] G. Yang and Y. Du, "A robust preconditioned GMRES method for electromagnetic scattering from dielectric rough surfaces," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 9, pp. 3396–3408, Sep. 2012.
[5] M. Zribi, A. L. Morvan-Quemener, M. Dechambre, and N. Baghdadi, "Numerical backscattering analysis for rough surfaces including a cloddy structure," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 5, pp. 2367–2374, May 2010.
[6] B. Guan, J. F. Zhang, X. Y. Zhou, and T. J. Cui, "Electromagnetic scattering from objects above a rough surface using the method of moments with half-space Green's function," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 10, pp. 3399–3405, Oct. 2009.
[7] Y. Ren, Z. Nie, Y. Zhao, and W. Ma, "Sparsification of the impedance matrix in the solution of the integral equation by using the maximally orthogonalized basis functions," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 7, pp. 1975–1981, Jul. 2008.
[8] H. J. He and L. X. Guo, "A multihybrid FE-BI-KA technique for 3-D electromagnetic scattering from a coated object above a conductive rough surface," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 2009–2013, Dec. 2016.
[9] X. Qi, Z. Nie, X. Que, Y. Wang, and Y. Yang, "An efficient method for analysis of EM scattering from objects straddling the interface of a half-space," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 12, pp. 2014–2018, Dec. 2016.
[10] B. Liu, Z. Y. Li, and Y. Du, "A fast numerical method for electromagnetic scattering from dielectric rough surfaces," *IEEE Trans. Antennas Propag.*, vol. 59, no. 1, pp. 180–188, Jan. 2011.
[11] L. Kuang and Y. Q. Jin, "Bistatic scattering from a three-dimensional object over a randomly rough surface using the FDTD algorithm," *IEEE Trans. Antennas Propag.*, vol. 55, no. 8, pp. 2302–2312, Aug. 2007.
[12] L. Tsang, C. H. Chan, and K. Pak, "Backscattering enhancement of a two-dimensional random rough surface (three-dimensional scattering) based on Monte Carlo simulations," *J. Opt. Soc. Amer. A*, vol. 11, no. 2, pp. 711–715, 1994.

[13] K. Pak, L. Tsang, L. Li, and C. H. Chan, "Combined random rough surface and volume scattering based on Monte Carlo simulations of solutions of Maxwell's equations," *Radio Sci.*, vol. 28, no. 3, pp. 331–338, 1993.

[14] G. Chen, H. X. Dang, X. M. Tan, H. Chen, and T. J. Cui, "Scattering properties of electromagnetic waves from randomly oriented rough metal plate in the lower Terahertz region," *J. Radars*, vol. 7, no. 1, pp. 75–82, 2018.

[15] Y. Fan and Y. Jie, "A modified two-scale microwave scattering model for a dielectric randomly rough surface," *J. Radars*, vol. 4, no. 5, pp. 560–570, 2015.

[16] L. Tsang, J. A. Kong, and K. H. Ding, *Scattering of Electromagnetic Waves, Theories and Applications*. New York, NY, USA: Wiley, 2000.

[17] G. Yang and Y. Du, "An optimized Monte Carlo procedure and its application in electromagnetic scattering from rough surfaces," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2607–2616, May 2014.

[18] Z. Zubac, D. D. Zutter, and D. V. Ginste, "Scattering from two-dimensional objects of varying shape combining the multilevel fast multipole method (MLFMM) with the stochastic galerkin method (SGM)," *IEEE Antennas Wireless Propag. Lett,*, vol. 13, pp. 1275–1278, 2014.

[19] Z. Zubac, L. Daniel, D. D. Zutter, and D. V. Ginste, "A cholesky-based SGM-MLFMM for stochastic full-wave problems described by correlated random variables," *IEEE Antennas Wireless Propag. Lett.*, vol. 16, pp. 776–779, 2017.

[20] U. K. Khankhoje and S. Padhy, "Stochastic solutions to rough surface scattering using the finite element method," *IEEE Trans. Antennas Propag.*, vol. 65, no. 8, pp. 4170–4180, Aug. 2017.

[21] G. Silva-Oelker, R. Aylwin, C. Jerez-Hanckes, and P. Fay, "Quantifying the impact of random surface perturbations on reflective gratings," *IEEE Trans. Antennas Propag.*, vol. 66, no. 2, pp. 838–847, Feb. 2018.

[22] J. Shaeffer, "Direct solve of electrically large integral equations for problem sizes to 1 M unknowns," *IEEE Trans. Antennas Propag.*, vol. 56, no. 8, pp. 2306–2313, Aug. 2008.

[23] X.-M. Pan, W.-C. Pi, M.-L. Yang, Z. Peng, and X.-Q. Sheng, "Solving problems with over one billion unknowns by the MLFMA," *IEEE Trans. Antennas Propag.*, vol. 60, no. 5, pp. 2571–2574, May 2012.

[24] B. Michiels, J. Fostier, I. Bogaert, and D. De Zutter, "Full-wave simulations of electromagnetic scattering problems with billions of unknowns," *IEEE Trans. Antennas Propag.*, vol. 63, no. 2, pp. 796–799, Feb. 2015.

[25] X. M. Pan and X. Q. Sheng, "Accurate and efficient evaluation of spatial electromagnetic responses of large scale targets," *IEEE Trans. Antennas Propag.*, vol. 62, no. 9, pp. 4746–4753, Sep. 2014.

[26] H. Cheng, Z. Gimbutas, P. G. Martinsson, and V. Rokhlin, "On the compression of low rank matrices," *SIAM J. Sci. Comput.*, vol. 26, no. 4, pp. 1389–1404, 2005.

[27] E. Liberty, F. Woolfe, P. G. Martinsson, V. Rokhlin, and M. Tygert, "Randomized algorithms for the low-rank approximation of matrices," *Proc. Nat. Acad. Sci. USA*, vol. 104, pp. 20 167–20 172, 2007.

[28] N. Halko, P. G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

[29] X. M. Pan, M. J. Gou, and X. Q. Sheng, "Prediction of radiation pressure force exerted on moving particles by the two-level skeletonization," *Opt. Express*, vol. 22, no. 8, pp. 10032–10045, 2014.

[30] X. M. Pan and X. Q. Sheng, "Fast solution of linear systems with many right hand sides based on skeletonization," *IEEE Antennas Wireless Propag. Lett.*, vol. 15, no. 1, pp. 301–304, 2016.

[31] A. Lucas, M. Stalzer, and J. Feo, "Parallel implementation of fast randomized algorithms for low rank matrix decomposition," *Parallel Process. Lett.*, vol. 24, no. 1, 2012, Art. no. 1450004.

[32] N. P. Halko, "Randomized methods for computing low-rank approximations of matrices," Ph.D. dissertation, USA, 2012.

[33] S. Velamparambil and W. C. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 53, no. 8, pp. 2719–2727, Aug. 2005.

[34] X. M. Pan and X. Q. Sheng, "A highly efficient parallel approach of multilevel fast multipole algorithm," *J. Electromagn. Waves Appl.*, vol. 20, no. 8, pp. 1081–1092, 2006.

[35] X. M. Pan and X. Q. Sheng, "A sophisticated parallel MLFMA for scattering by extremely large targets," *IEEE Antennas Propag. Mag.*, vol. 50, no. 3, pp. 129–138, Jun. 2008.

[36] O. Ergul and L. Gurel, "A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 57, no. 6, pp. 1740–1750, Jun. 2009.

[37] J. M. Taboada, M. Araujo, J. M. Bertolo, L. Landesa, F. Obelleiro, and J. L. Rodriguez, "MLFMA-FFT parallel algorithm for the solution of large-scale problems in electromagnetics," *Progress Electromagn. Res.*, vol. 105, pp. 15–30, 2010.

[38] S. L. Huang, H. Xu, X. M. Pan, and X. Q. Sheng, "Fast solution of linear systems with many right hand sides using MPI parallel interpolative decomposition," in *Proc. 11th Int. Symp. Antennas, Propag. EM Theory*, 2016, pp. 523–526.

[39] X. M. Pan, S. L. Huang, and X. Q. Sheng, "Wide angular sweeping of dynamic electromagnetic responses from large targets by MPI parallel skeletonization," *IEEE Trans. Antennas Propag.*, vol. 66, no. 3, pp. 1619–1623, Mar. 2018.

[40] H. X. Ye and Y. Q. Jin, "Parameterization of the tapered incident wave for numerical simulation of electromagnetic scattering from rough surface," *IEEE Trans. Antennas Propag.*, vol. 53, no. 3, pp. 1234–1237, Mar. 2005.

[41] R. L. Wagner, J. Song, and W. C. Chew, "Monte Carlo simulation of electromagnetic scattering from two-dimensional random rough surfaces," *IEEE Trans. Antennas Propag.*, vol. 45, no. 2, pp. 235–245, Feb. 1997.

[42] J. W. Hao and X. Q. Sheng, "Accurate and efficient simulation model for the scattering from a ship on a sea-like surface," *IEEE Geosci. Remote Sens. Lett.*, vol. 14 no. 12, pp. 2375–2379, Dec. 2017.

[43] X. M. Pan and X. Q. Sheng, "Improved algebraic preconditioning for MoM solutions of large-scale electromagnetic problems," *IEEE Antennas Wireless Propag. Lett.*, vol. 13, pp. 106–109, 2014.

[44] X. M. Pan and X. Q. Sheng, "Sparse approximate inverse preconditioner for multiscale dynamic electromagnetic problems," *Radio Sci.*, vol. 49, no. 11, pp. 1041–1051, 2014.

[45] H. V. Sorensen and C. S. Burrus, "Efficient computation of the DFT with only a subset of input or output points," *IEEE Trans. Signal Process.*, vol. 41, no. 3, pp. 1184–1200, Mar. 1993.

[46] J. Choi *et al.*, "ScaLAPACK: A portable linear algebra library for distributed memory computers-design issues and performance," *Comput. Phys. Commun.*, vol. 97, no. 1/2, pp. 1–15, 1996.

[47] P. G. Martinsson, G. Quintanaorti, N. Heavner, and R. V. D. Geijn, "Householder qr factorization with randomization for column pivoting (HQRRP). FLAME working note #78," *Siam J. Scientific Comput.*, vol. 39, no. 2, pp. C96–C115, 2016.

[48] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.

[49] A. C. Woo, H. T. G. Wang, M. J. Schuh, and M. L. Sanders, "Benchmark radar targets for the validation of computational electromagnetics programs," *IEEE Antennas Propag. Mag.*, vol. 35, no. 1, pp. 84–89, Feb. 1993.

[50] W. C. Chew, J. M. Jin, E. Michielssen, and J. Song, *Fast Efficient Algorithms in Computational Electromagnetics*. Boston, MA, USA: Artech House, 2001.