Progress In Electromagnetics Research, PIER 61, 291–322, 2006

SPARSE FACTORIZATION OF THE TM$_z$ IMPEDANCE MATRIX IN AN OVERLAPPED LOCALIZING BASIS

R. J. Adams and A. Zhu

Electrical & Computer Engineering
University of Kentucky
Lexington, KY, USA

F. X. Canning

Simply Sparse Technologies, Inc.
Morgantown, WV, USA

Abstract—It has been observed that localized solution modes provide sparse factored representations of the discrete integral equations encountered in the simulation of electromagnetic phenomena at low frequencies. This paper extends these results by incorporating overlapped localizing modes. For TM$_z$ scattering from a rectangular array of perfectly conducting obstacles, it is observed that the complexity scaling of the resulting factorization is significantly reduced relative to previously reported results. The memory complexity of the resulting factored representation scales approximately as $O(N)$ for electrically small arrays. Limitations and possible extensions of these results are discussed.

1. INTRODUCTION

Numerical solutions of surface integral equation formulations of time-harmonic electromagnetic interactions with perfect electric conductors (PECs) require solving linear systems of the form [1]

$$E^i + ZJ = 0, \tag{1}$$

where Z is the N-by-N impedance matrix, the vector J contains the coefficients of the basis functions used to represent the electric currents on the conductor, and the vector E^i is determined by samples of an impressed electric field. The purpose of this paper is to summarize and demonstrate a sparse direct solution strategy for discretizations of (1) obtained for low-frequency, TM_z scattering applications. These results were initially presented as [2]. (In this paper, the phrase "low-frequency" is used to refer to a situation in which the maximum linear dimension of a scattering configuration is small relative to the wavelength of the harmonic excitation.)

The low-frequency factorization and solution strategy discussed below relies on previous work providing sparse implementations of Z^{-1} using so-called local-global solution (LOGOS) modes [3–7]. A LOGOS mode is a solution (J) of the global boundary condition (1) that is localized to order-ε within a smaller part of the global simulation domain. These modes can be divided into radiating and nonradiating types. Because we restrict our attention in this paper to low-frequency applications of (1), it is only necessary to incorporate the nonradiating modes in order to determine $O(N)$ factored representations of Z.

In [3–7], a nonradiating LOGOS mode is defined as a solution to (1) in which both the surface current and the associated scattered fields are localized to the same small spatial region. In the following we generalize previous definitions of nonradiating LOGOS modes to include solution modes in which the currents are localized to overlapping spatial regions. However, as in [3–7], the scattered fields remain localized to non-overlapping spatial regions. Such modes are hereinafter referred to as overlapped localizing LOGOS (OLL) modes. The use of such modes in determining $O(N)$ sparse factored representations of the impedance matrix was originally reported in [2].

The remainder of this paper is organized as follows. Section 2 outlines the components of the proposed solution strategy. Section 3 summarizes the multilevel quad-tree grouping strategy and some related notation, and Section 4 recapitulates the multilevel simply sparse method (MLSSM), which provides a sparse representation of Z. Sections 5 to 8 define the overlapped LOGOS modes, the associated sparse factorization strategy of the MLSSM representation of Z, and the resulting sparse direct solution strategy for (1). Section 9 contains representative numerical examples. The paper is summarized and areas for additional work are identified in Section 10.

2. OUTLINE OF SPARSE SOLUTION STRATEGY

Given an arbitrary set of linearly independent excitation vectors, E^i, and a sparse representation of Z, we are interested in efficiently determining a sparse factored representation of Z, accurate to order-ε, which can be used to rapidly determine the corresponding solution

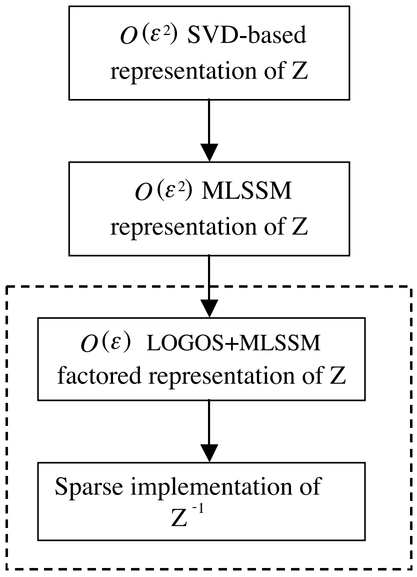
$O(\varepsilon^2)$ SVD-based representation of Z
$O(\varepsilon^2)$ MLSSM representation of Z
$O(\varepsilon)$ LOGOS+MLSSM factored representation of Z
Sparse implementation of Z^{-1}

Figure 1. Proposed direct solution strategy. This paper discusses the algorithms associated with the blocks inside the dashed box.

vectors, $J = -Z^{-1}E^i$.

Figure 1 illustrates the structure of the solution strategy considered in this paper. The procedure begins by using a fast scheme to determine a sparse, $O(N \log N)$, SVD-based representation of Z from sparse samples of the underlying free-space TM_z kernel. The procedure used to determine the SVD-based representation from sparse samples of Z is essentially similar to algorithms previously reported elsewhere [8] and will not be discussed further here. As indicated in the figure, the relative RMS error between the compressed form and the actual impedance matrix is required to be $O(\varepsilon^2)$, where ε is the desired RMS error in the boundary condition (1).

The sparse, SVD-based representation is subsequently transformed into the $O(N)$ sparse MLSSM (multilevel simply sparse method) representation discussed in [9] and summarized below. The details of the procedure used to determine the $O(\varepsilon^2)$ MLSSM representation of Z from the SVD-based representation will be communicated separately.

This paper focuses on the remaining two blocks of the solution procedure indicated in Figure 1. In particular, the following discussion assumes the availability of an $O(\varepsilon^2)$ MLSSM representation of Z and proceeds to manipulate this into a factored form amenable to fast sparse implementations of Z^{-1} via overlapped, localizing LOGOS modes.

3. SPATIAL GROUPS

The LOGOS factorization algorithm for the MLSSM representation of Z relies on a multilevel spatial decomposition of the underlying geometry. This is accomplished in the following examples using a multilevel quad-tree. The resulting spatial groupings are similar to those used to implement the multilevel fast multipole algorithm in two dimensions [10]. As demonstrated elsewhere, alternate multilevel organizations of target points may provide additional efficiency gains in some instances [6]. This possibility is not considered here.

The number of levels in the quad-tree will be denoted by L. The individual levels are indexed by l, $l = 1, \ldots, L$. The level $l = 1$ is the root level of the tree. The root level consists of a single group containing all spatial samples. The number of nonempty groups at the lth level of the tree is $M(l)$, and the number of spatial samples in each group is approximately $m(l) = N/M(l)$. The total number of levels, L, is chosen so that the smallest spatial group in each branch of the multilevel tree contains approximately ten points.

At a given level (level-l) of the tree, those groups having boundaries which touch the box bounding the ith group are referred to as the level-l near-neighbors of the ith group. (The ith group is also considered to be a near-neighbor to itself.) The remaining groups at level-l are referred to as distant (or non near-neighbor) groups.

In the following discussion it will at times be convenient to indicate the ith group at level-l using the notation $i(l)$. Similarly, the notation $G_{i(l)}$ will be used to indicate the columns of a given matrix G associated with sources in the ith level-l group. The notation $\{i(l), n\}$ will be used to indicate the set of all level-l groups which are near-neighbors of group $i(l)$. Thus, the matrix $G_{\{i(l),n\}}$ indicates the columns of G associated with source degrees of freedom located in groups which are near-neighbors of group $i(l)$.

4. MLSSM REPRESENTATION OF Z

The multilevel simply sparse method (MLSSM) provides a compressed representation of the impedance matrix [9]. The complexity of the MLSSM representation of Z is $O(N)$ for electrically small configurations (this is demonstrated by the numerical examples of Section 9). The MLSSM also provides a convenient representation of the LOGOS-based projections of Z indicated by (8) and (9) below.

The structure of the MLSSM is indicated by the following multilevel recursion:

$$Z_m = \hat{Z}_m + U_m^H Z_{m-1} V_m. \tag{2}$$

If the MLSSM recursion indicated by (2) is used to provide a multilevel compressed representation of Z in (1), the original impedance matrix is recovered from (2) when $m = L$ (i.e., $Z = Z_L$).

In (2), $\hat{Z}_m$ is the sparse matrix containing all near-neighbor interactions at level-m of the quad-tree which were not represented at a finer level of the tree. The matrices U_m and V_m are non-rectangular, orthonormal, block diagonal matrices which effectively compress interactions between non-touching groups at level-m of the quad-tree. This compression is accomplished by using a singular value decomposition to determine the minimum number of DoF (degrees of freedom) required to represent interactions between separated sources and observers [9].

The MLSSM representation (2) will be used in the following to compress both the impedance matrix, Z, and level-l projections of the impedance matrix (cf. (8) and (9)). In all such instances, (2) is valid for $m = 3, \ldots, l$. When $m = 2$, equation (2) reduces to

$$Z_2 = \hat{Z}_2. \tag{3}$$

The matrices U_2 and V_2 are not defined in this case because all interactions at level-2 are between touching (i.e., near-neighbor) groups. For the same reason, the recursion indicated by (2) and (3) is not continued to level-1. The original impedance matrix is recovered from (2) when $m = l = L$:

$$Z = Z_L. \tag{4}$$

The MLSSM discussed above and in [9] is a multilevel modification of the simply sparse method (SSM) previously discussed by Canning and Rogovin [11].

The LOGOS-based factorization procedure discussed below assumes the availability of an $O(\varepsilon^2)$ L-level MLSSM representation of Z. Given this representation, the remainder of the paper illustrates the determination of a sparse, $O(\varepsilon)$ factored representation of Z via overlapped, localizing LOGOS modes.

5. OVERLAPPED LOCALIZING LOGOS MODES

Previously reported LOGOS-based sparse factorization algorithms for Z at low frequencies utilize nonoverlapped, localizing LOGOS modes

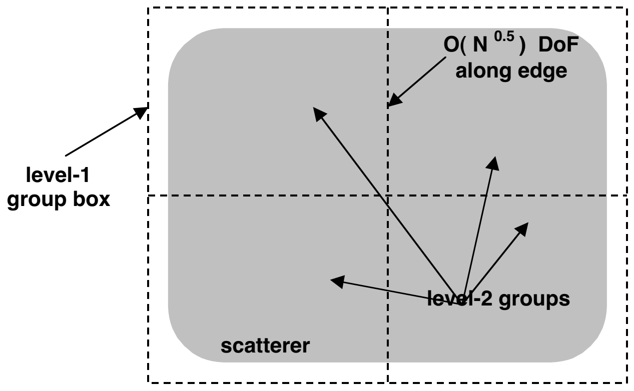

Figure 2. Two-level quad-tree decomposition of a general scatterer in two dimensions.

[3–7]. These modes were obtained by analyzing the impedance matrix to determine surface current modes satisfying (1) for which both the current, J, and the scattered field, ZJ, are localized (within $O(\varepsilon)$) to identical spatial groups. Due to the reliance on standard, nonoverlapping, multilevel spatial decompositions (quad- and oct-trees), the computational efficiencies of these algorithms are limited to approximately $O(N^{1.5})$ in two dimensions and $O(N^2)$ in three dimensions.

These computational efficiencies are a consequence of constraining LOGOS modes to nonoverlapping spatial groups. For example, consider the two-level quad-tree decomposition of a general two-dimensional target illustrated in Figure 2. Assuming a total of N unknowns and a uniform discretization of the shaded region indicated in the figure, it follows that there are $O(\sqrt{N})$ degrees of freedom along any line which crosses through the center of the scatterer. Two such boundaries are introduced at level-2 of the quad-tree decomposition of the target. Any DoF which might be localized in a small neighborhood which intersects one of the level-2 boundaries cannot be captured until the root level of the quad-tree is reached. Since there are about $O(\sqrt{N})$ such DoF (for large N at low frequency), it follows that the computational complexity of previously reported LOGOS factorization algorithms are bounded from below by $O(N^{1.5})$ for general targets in 2-D. A similar argument demonstrates that the complexity of previously reported 3-D LOGOS factorization algorithms is bounded from below by $O(N^2)$.

These complexities can be significantly reduced by expanding Z

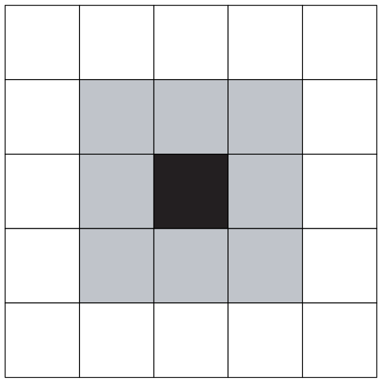

Figure 3. Illustration of overlapped LOGOS mode definition. Rectangles represent different groups at a given level of the quad tree decomposition. The black shaded block represents the "self" group, and the gray shaded blocks represent the self group's near-neighbors. For the indicated self group: an overlapped, localizing LOGOS (OLL) mode is defined as a current (J) that is generally nonzero in all the shaded blocks (self and near-neighbor), and which produces a scattered field, ZJ, that is nonzero only within the self group (to $O(\varepsilon)$).

in a basis of *overlapped* localizing LOGOS (OLL) modes. An OLL mode is defined by a source (J) which is generally confined to a larger spatial region than the scattered field (ZJ) [2]. The specific OLL mode definition used in this paper is provided in Figure 3. As indicated in the figure, at level-l, an overlapped localizing LOGOS mode associated with group $i(l)$ is defined as a current (J) which is nonzero in the set of near-neighbor groups, $\{i(l), n\}$, and produces a scattered field, ZJ, which is nonzero only within the self group, $i(l)$.

The LOGOS-based factorization algorithm described below uses a multilevel basis of these OLL modes to obtain a sparse factorization of the MLSSM representation of Z.

6. LOGOS FACTORIZATION OF MLSSM REPRESENTATION OF Z

The LOGOS-based factorization strategy summarized in this section consists of combining the Schur decomposition strategy reported in [7] with: (1) the MLSSM representation summarized above and in [9], and (2) the OLL modes defined in the previous section. We begin the discussion by recapitulating the basic features of the LOGOS-based Schur factorization algorithm.

6.1. LOGOS-based Schur Factorization

The LOGOS-based Schur factorization algorithm is reported in [7]. The matrix

$$\Lambda_l = \left[\Lambda_l^{(L)}, \Lambda_l^{(N)}\right] \tag{5}$$

indicates the decomposition of the columns of the level-l LOGOS transformation (Λ_l) into current modes that generate scattered fields which are ($\Lambda_l^{(L)}$) and are not ($\Lambda_l^{(N)}$) localized within a single level-l group. A similar segregation of the columns of the associated projection matrix, P_l, is also possible (we use P_l to indicate the projection operator in this paper; the same operator was denoted V_l in [7]):

$$P_l = \left[P_l^{(L)}, P_l^{(N)}\right]. \tag{6}$$

As discussed in [7], the block diagonal matrix P_l is determined by the (localized) scattered fields, $Z\Lambda_l^{(L)}$. Notice in (5) and (6) that the superscript (L) is used to indicate portions of the operators Λ_l and P_l associated with LOGOS modes that generate scattered fields localized to a single level-l group. The superscript (N) is used to indicate quantities with degrees of freedom (DoF) that are not localized to a single level-l group.

With these definitions, the level-l impedance matrix can be represented as

$$Z_{l+1}^{(NN)} = \left(P_l^H\right)^{-1} P_l^H Z_{l+1}^{(NN)} \Lambda_l \Lambda_l^{-1} \approx \left(P_l^H\right)^{-1} \begin{bmatrix} I & Z_l^{(LN)} \\ 0 & Z_l^{(NN)} \end{bmatrix} \Lambda_l^{-1}, \tag{7}$$

where the approximation is accurate to $O(\varepsilon)$ [7]. (Observe that the level-l transformations Λ_l and P_l are applied to the level-($l+1$) operator $Z_{l+1}^{(NN)}$.) When $l = L$ we recover the impedance matrix: $Z = Z_{l+1}^{(NN)}$.

The submatrices $Z_l^{(LN)}$ and $Z_l^{(NN)}$ in (7) are defined in terms of the components of Λ_l and P_l,

$$Z_l^{(LN)} = \left(P_l^{(L)}\right)^H Z_{l+1}^{(NN)} \Lambda_l^{(N)}, \tag{8}$$

$$Z_l^{(NN)} = \left(P_l^{(N)}\right)^H Z_{l+1}^{(NN)} \Lambda_l^{(N)}. \tag{9}$$

The submatrix $Z_l^{(LN)}$ indicates the simultaneous projection of (i) the domain of $Z_{l+1}^{(NN)}$ onto non-localizing level-l LOGOS modes, $\Lambda_l^{(N)}$, and (ii) the range of $Z_{l+1}^{(NN)}$ onto the conjugate (cf. [7]) of the scattered

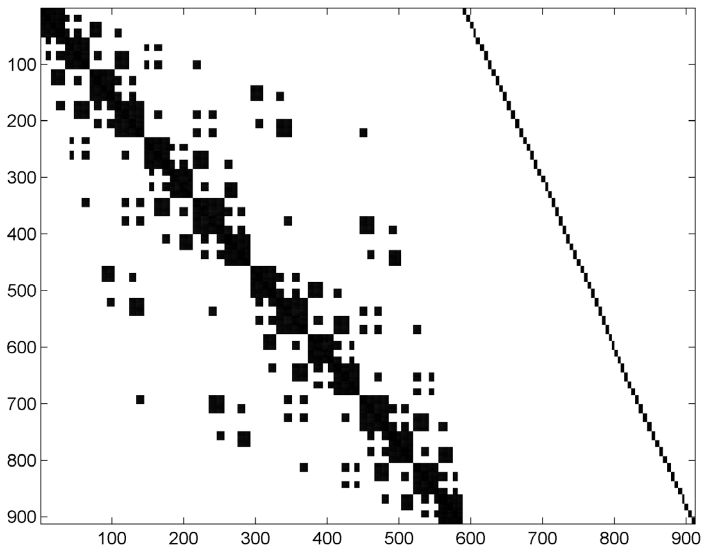

Figure 4. Illustration of the general structure of overlapped LOGOS transformation matrix, Λ_l. The structure of the decomposition (5) is evident in the figure, with the nondiagonal, localizing submatrix $\Lambda_l^{(L)}$ appearing on the left, and the block diagonal, nonlocalizing submatrix $\Lambda_l^{(N)}$ on the right.

fields $Z_{l+1}^{(NN)}\Lambda_l^{(L)}$. The matrix $Z_l^{(NN)}$ is similarly interpreted. The dimensions of the submatrices appearing on the right side of (7) are determined by the number of local (L) and nonlocal (N) LOGOS modes determined at level-l, which in turn depends on the properties of the underlying scattering problem.

In this paper we are interested in using a Schur decomposition with the same structure used in [7] (indicated by (5) through (9) above), but which is based on the overlapped localizing LOGOS (OLL) modes discussed in the previous section. The primary difference introduced by this modification is in the structure of the submatrix $\Lambda_l^{(L)}$ of Λ_l. The matrices Λ_l and P_l used with the Schur factorization algorithm reported in [7] were both permuted block diagonal matrices for all l (see illustrations in [7]). Due to the present use of the overlapping LOGOS modes, the Λ_l obtained in the algorithm reported in this paper are no longer block diagonal. Figure 4 illustrates the typical structure of the Λ_l associated with the algorithm reported in this paper. It is evident from the figure that the structure of the $\Lambda_l^{(L)}$ component of Λ_l is no longer block diagonal. However, the portion of Λ_l associated with modes that are not localized to level-l (i.e., $\Lambda_l^{(N)}$) remains block diagonal. Similarly, due to the manner in which the overlapped localizing LOGOS modes have been defined (see Section 5), the projection operators P_l remain block diagonal.

Hence, apart from the modification of Λ_l resulting from our use of overlapped localizing LOGOS modes to define $\Lambda_l^{(L)}$, the structure of the multilevel Schur factorization algorithm reported in [7] remains unchanged. That algorithm consists of recursively applying the decomposition strategy indicated by (7) to the submatrices $Z_l^{(NN)}$ obtained at each level of the recursion. In this way, a factored representation of Z is determined which is defined by the matrices Λ_l, $P_l, Z_l^{(LN)}$ and the matrix $Z_3^{(NN)}$ obtained at level-3, at which level the multilevel recursion stops. (The multilevel recursion was continued to level-2 in [7]. Due to our use of overlapped modes in the present paper, the recursion is herein terminated at level-3.)

In order to render this overlapped factorization algorithm more computationally efficient than previously reported results, two requirements must be met:

1. It is necessary to define a fast algorithm capable of determining the overlapped localizing modes (used to define Λ_l and P_l at each level) from the $Z_l^{(NN)}$.
2. A sparse storage format is required for the $Z_l^{(LN)}$ and the intermediate $Z_l^{(NN)}$ operators. (As indicated above, although only $Z_3^{(NN)}$ is required in the final factored representation for Z, the $Z_l^{(NN)}$ must be temporarily stored at each level of the recursion and used to define the subsequent transformation matrices Λ_{l-1} and P_{l-1}.)

Both of these requirements are satisfied below using the MLSSM representation. The use of the MLSSM to store $Z_l^{(LN)}$ and $Z_l^{(NN)}$ (requirement 2) is discussed in Section 6.2. The resulting MLSSM representations can subsequently be used to compute the OLL modes at each level (requirement 1). The algorithm used to accomplish this is described in Section 6.3.

Finally, we observe that alternate versions of the LOGOS-based factorization strategy discussed in this paper can be obtained by using other methods capable of providing sparse representations of the impedance matrix and its projections. One such possibility is indicated in Section 10.

6.2. Sparse Representations of $Z_l^{(LN)}$ and $Z_l^{(NN)}$

The MLSSM provides a convenient format to store the impedance matrix, Z, and its projections, $Z_l^{(LN)}$ and $Z_l^{(NN)}$, at each level of the multilevel tree. To illustrate, consider the level-l projection of $Z_{l+1}^{(NN)}$ indicated by (7). Assume that the matrix $Z_{l+1}^{(NN)}$ on the left of this equation has an l-level MLSSM representation with the form indicated by (2):

$$Z_m = \hat{Z}_m + U_m^H Z_{m-1} V_m, \tag{10}$$

where $3 \leq m \leq l$. In this case, $Z_{l+1}^{(NN)}$ is recovered from (10) when $m = l : Z_{l+1}^{(NN)} = Z_l$.

As discussed above, the matrices Λ_l and P_l are (permuted) block diagonal, except for the columns of Λ_l which correspond to overlapped level-l localizing modes (i.e., the submatrix $\Lambda_l^{(L)}$). Furthermore, the diagonal blocks of $\Lambda_l^{(N)}$ and P_l are coincident with the diagonal blocks of U_l and V_l obtained from the MLSSM representation (10) when $m = l$. Hence, the block diagonal portions of Λ_l and P_l can be rapidly applied to the MLSSM representation of $Z_{l+1}^{(NN)}$ by applying them only to the exterior U_l and V_l matrices and the sparse near-neighbor operator $\hat{Z}_l$ obtained from (10) when $m = l$.

In this way, the MLSSM representation of $Z_l^{(NN)}$ in (7) is obtained from the MLSSM representation of $Z_{l+1}^{(NN)}$ indicated by (10) as

$$\begin{aligned} Z_l^{(NN)} &= \left(P_l^{(N)}\right)^H Z_{l+1}^{(NN)} \Lambda_l^{(N)} \\ &= \left(P_l^{(N)}\right)^H \left(\hat{Z}_l + U_l^H Z_{l-1} V_l\right) \Lambda_l^{(N)}, \\ &= \hat{Z}_l^{(NN)} + \left(U_l^{(N)}\right)^H Z_{l-1} V_l^{(N)} \end{aligned} \tag{11}$$

where $\hat{Z}_l^{(NN)} = \left(P_l^{(N)}\right)^H \hat{Z}_l \Lambda_l^{(N)}$, $U_l^{(N)} = U_l P_l^{(N)}$, $V_l^{(N)} = V_l \Lambda_l^{(N)}$ and Z_{l-1} is unaffected by the projections. The MLSSM representation of $Z_l^{(LN)}$ is similarly obtained.

At this point it is useful to make two observations. First, notice that the non block-diagonal portion of Λ_l ($\Lambda_l^{(L)}$, cf. Figure 4) is not used in obtaining the matrices $Z_l^{(LN)}$ and $Z_l^{(NN)}$ required by the factorization (7). (The effect of $\Lambda_l^{(L)}$ is to generate the submatrices I and 0 in (7).) This is essential to maintaining the basic, nested structure of the MLSSM representation at each level of the recursive factorization indicated by (7).

Secondly, we observe that the multilevel factorization algorithm of [7] applies the LOGOS factorization indicated by (7) to the $Z_l^{(NN)}$ obtained at successively coarser levels of the multilevel tree. For example, the next step in the multilevel LOGOS-based Schur factorization of (7) will consist of determining Λ_{l-1} and P_{l-1} from, and applying them to, $Z_l^{(NN)}$. Because Λ_{l-1} and P_{l-1} must be defined on groups at level-$(l-1)$, it is necessary to collapse the l-level MLSSM representation of $Z_l^{(NN)}$ in (7) to an $(l-1)$-level MLSSM representation prior to determining Λ_{l-1} and P_{l-1}. This collapse of the MLSSM representation is efficiently accomplished by expanding Z_{l-1} in the last line of (11) to obtain

$$\begin{aligned} Z_l^{(NN)} &= \hat{Z}_l^{(NN)} + \left(U_l^{(N)}\right)^H \hat{Z}_{l-1} V_l^{(N)} \\ &\quad + \left(U_l^{(N)}\right)^H (U_{l-1})^H Z_{l-2} V_{l-1} V_l^{(N)} \\ &= \hat{Z}_{l-1}^{(NN)} + \left(U_{l-1}^{(N)}\right)^H Z_{l-2} V_{l-1}^{(N)} \end{aligned} \tag{12}$$

where

$$\hat{Z}_{l-1}^{(NN)} = \hat{Z}_l^{(NN)} + \left(U_l^{(N)}\right)^H \hat{Z}_{l-1} V_l^{(N)}, \tag{13}$$

$$U_{l-1}^{(N)} = U_{l-1} U_l^{(N)}, \tag{14}$$

$$V_{l-1}^{(N)} = V_{l-1} V_l^{(N)}. \tag{15}$$

Due to the nested nature of the multilevel quad-tree used to define the MLSSM representation, the matrices $U_{l-1}^{(N)}$ and $V_{l-1}^{(N)}$ of (14) and (15) are block diagonal, with each diagonal block corresponding to a single level-$(l-1)$ group. Similarly, $\hat{Z}_{l-1}^{(NN)}$ is a sparse matrix whose nonzero entries correspond only to interactions between near-neighbors at level $(l-1)$. The remaining component of $Z_l^{(NN)}$ in (12), Z_{l-2}, is unchanged from its original definition in (10). Consequently, equation (12) is an $(l-1)$ level MLSSM representation of $Z_l^{(NN)}$ (with the minor caveat that $U_{l-1}^{(N)}$ and $V_{l-1}^{(N)}$ are no longer orthonormal transformations).

The fact that the Schur decomposition (7) yields submatrices $Z_l^{(NN)}$ that can be collapsed to an MLSSM representation at the next coarser (i.e., parent) level is significant. It implies that if we are given an algorithm for computing OLL modes from the MLSSM representation of a general matrix, this algorithm can be applied repeatedly to the blocks $Z_l^{(NN)}$ resulting from the decomposition (7) at different levels of the quad-tree. Due to the upper triangular nature of (7), this in turn yields a factored representation of the impedance matrix.

6.3. Efficient Determination of Overlapped Localizing Modes

Given the LOGOS transformation matrices, Λ_l, associated with each of the $Z_l^{(NN)}$, the LOGOS-based Schur decomposition algorithm outlined above provides an efficient method of determining a sparse, factored representation of the impedance matrix. In this section we describe an efficient algorithm for determining the OLL modes (and thus the Λ_l) from the MLSSM representation of the impedance matrix and its projections.

To motivate this algorithm, Section 6.3.1 summarizes the original procedure [3, 5] used to determine localizing LOGOS modes from the full columns of the impedance matrix. In so doing, we also make the minor extension of using the associated LOGOS mode calculation algorithm to find the OLL modes described above. (References [3, 5] only consider the determination of non overlapping modes from full columns of the impedance matrix.) Section 6.3.2 demonstrates the existence of an efficient alternative to the procedure used to find LOGOS modes in Section 6.3.1. The specific OLL mode calculation algorithm used in the numerical examples of Section 9 is finally outlined in Sections 6.3.3 and 6.3.4.

6.3.1. Original Algorithm

Consider the determination of overlapped localizing LOGOS modes associated with group i at level-l, $i(l)$, from $Z_{l+1}^{(NN)}$. (Recall that, as indicated by (7), level-l LOGOS modes are used to project the level-$(l+1)$ matrix $Z_{l+1}^{(NN)}$.) Let the submatrix $Z_{\{i(l),n\}}^{(NN)}$ indicate the (full) columns of $Z_{l+1}^{(NN)}$ associated with sources located inside group $i(l)$ and its near-neighbors. The SVD-based LOGOS mode calculation procedure described in [3, 5] can be straightforwardly applied to $Z_{\{i(l),n\}}^{(NN)}$ in order to determine excitation modes spanning group $i(l)$ and its neighbors which generate scattered fields localized (to $O(\varepsilon)$) within group $i(l)$. To summarize this procedure, let

$$Z_{\{i(l),n\}}^{(NN)} = usv^H \tag{16}$$

indicate the SVD of $Z^{(NN)}_{\{i(l),n\}}$, and let $u_{i(l)}$ indicate rows of the unitary matrix u associated with observers located in group $i(l)$. Because $u_{i(l)}$ is a submatrix of a unitary matrix, an additional SVD of $u_{i(l)}$ suffices to identify transformations of the original matrix, $Z^{(NN)}_{\{i(l),n\}}$, which generate scattered fields localized only to group $i(l)$ [3, 5].

The primary computational limitation of this algorithm for determining OLL modes arises from the need to compute the SVD indicated by (16). For example when $l = L$, $Z^{(NN)}_{\{i(l),n\}}$ has N rows, and $O(N)$ such decompositions are required (since the number of level-L groups is $O(N)$). These considerations indicate that computing the overlapped LOGOS modes at level-L will require $O(N^2)$ operations using the procedure reported in [3, 5]. The remainder of Section 6.3 summarizes a more efficient procedure which can be used to determine the overlapped localizing LOGOS modes at all levels ($l = 3, \dots, L$). However, in order to simplify the discussion, we only explicitly consider the efficient determination of OLL modes at the finest level ($l = L$) of the tree, which involves the original impedance matrix, $Z^{(NN)}_{l+1} = Z$. Extension of the results to coarser levels ($l < L$) is straightforward, as indicated in Section 6.3.5.

6.3.2. Motivation of an Efficient Alternative

To motivate a computationally efficient procedure for determining the required OLL modes, consider decomposing the columns of Z associated with group $i(L)$ and its near-neighbors as

$$Z_{\{i(L),n\}} = \begin{bmatrix} Z^{(n)}_{\{i(L),n\}} \\ Z^{(f)}_{\{i(L),n\}} \end{bmatrix}, \tag{17}$$

where $Z^{(n)}_{\{i(L),n\}}$ indicates those rows of $Z_{\{i(L),n\}}$ associated with all observers contained by level-L groups that are near-neighbors of at least one of the groups in the set $\{i(L), n\}$. The matrix $Z^{(f)}_{\{i(L),n\}}$ contains the remaining rows of $Z_{\{i(L),n\}}$, which are associated with observers located in groups that are separated from group $i(L)$ and all of its near-neighbors by at least one level-L group.

Recall that, for sufficiently long wavelengths, only $O(1)$ DoF (degrees of freedom) are required to represent the fields radiated by sources located inside group $i(L)$ and its near-neighbors to separated observers within a given tolerance [12]. From this fact it follows that the number of modes that must be retained in the SVD of $Z^{(f)}_{\{i(L),n\}}$ to maintain an $O(\varepsilon^2)$ representation is similarly $O(1)$ (i.e., the number of required modes is independent of the number of sources contained by group $i(L)$ and its near-neighbors):

$$Z^{(f)}_{\{i(L),n\}} = u^{(f)} s^{(f)} (v^{(f)})^H + O(\varepsilon^2), \tag{18}$$

with $s^{(f)}$ an $O(1) \times O(1)$ matrix. Inserted in (17), this provides

$$Z_{\{i(L),n\}} = \begin{bmatrix} I & 0 \\ 0 & u^{(f)} \end{bmatrix} \begin{bmatrix} Z^{(n)}_{\{i(L),n\}} \\ s^{(f)}(v^{(f)})^H \end{bmatrix} + O(\varepsilon^2). \tag{19}$$

Because the leading matrix in (19) is unitary, the desired OLL modes can be determined by applying the original SVD-based calculation of [3, 5] (indicated above by (16) and the associated discussion) to a reduced submatrix, $Z^{(r)}_{\{i(L),n\}}$, instead of the original submatrix $Z_{\{i(L),n\}}$ where

$$Z^{(r)}_{\{i(L),n\}} = \begin{bmatrix} Z^{(n)}_{\{i(L),n\}} \\ s^{(f)}(v^{(f)})^H \end{bmatrix} = \begin{bmatrix} Z^{(n)}_{\{i(L),n\}} \\ \tilde{Z}^{(f)}_{\{i(L),n\}} \end{bmatrix}, \tag{20}$$

and we have used

$$\tilde{Z}^{(f)}_{\{i(L),n\}} = s^{(f)}(v^{(f)})^H. \tag{21}$$

The advantage of applying the original, SVD-based analysis of [3, 5] to (20) rather than to $Z_{\{i(L),n\}}$ is that the former has dimension $O(1) \times O(1)$, whereas the dimension of the latter is $N \times O(1)$.

Having thus established that it is possible to determine overlapped LOGOS modes via an SVD-based analysis of the reduced matrix $Z^{(r)}_{\{i(L),n\}}$, we continue by specifying a computationally efficient procedure for determining the reduced representation (20) from a sparse representation of Z. In particular, as suggested by Figure 1, we outline an efficient algorithm for determining $Z^{(r)}_{\{i(L),n\}}$ (and subsequently Λ_L) from the MLSSM representation of Z. The discussion of this algorithm is contained in the following two subsections. Section 6.3.3 summarizes an algorithm with an $O(N)$ memory complexity for determining $Z^{(r)}_{\{i(L),n\}}$ from the MLSSM representation of Z. Section 6.3.4 discusses a modified version of this algorithm with significantly improved CPU performance.

6.3.3. Computing $Z^{(r)}_{\{i(L),n\}}$ from MLSSM representation of Z

As indicated by (20), $Z^{(r)}_{\{i(L),n\}}$ is determined by the submatrices $Z^{(n)}_{\{i(L),n\}}$ and $\tilde{Z}^{(f)}_{\{i(L),n\}}$. Because $Z^{(n)}_{\{i(L),n\}}$ is $O(1) \times O(1)$, in the numerical examples reported below this part of $Z^{(r)}_{\{i(L),n\}}$ is simply computed by directly expanding the relevant portion of the MLSSM representation (2) for $Z = Z^{(NN)}_{L+1}$. Computing $\tilde{Z}^{(f)}_{\{i(L),n\}}$ is somewhat more complicated, and the pseudo-code for the algorithm described in this section is indicated in Figure 5.

```
1) Initialize $\tilde{Z}^{(f)}_{\{i(L),n\}}$ to an empty matrix.
2) Loop over all levels, proceeding from coarse to fine:
   for $l = 3 : L$
   a) Initialize $A_{l,i(L)}$ to an empty matrix.
   b) Loop over all level-$l$ observer groups, $j(l)$, which
      are both not near-neighbors of the level-$l$ parents
      of $\{i(L), n\}$, and which have not been previously
      included at a coarser level.
      i) For each such group, extract the matrix $A_{j(l),i(L)}$
         of (22) from the MLSSM representation of Z and
         append to existing $A_{l,i(L)}$:

         $$A_{l,i(L)} = \begin{bmatrix} A_{l,i(L)} \\ A_{j(l),i(L)} \end{bmatrix}$$

      - End loop over far-field observer groups.
   c) Use SVD to compress $A_{l,i(L)}$ matrix, maintaining a
      $O(\varepsilon^2)$ representation.
   d) Append compressed $A_{l,i(L)}$ to $\tilde{Z}^{(f)}_{\{i(L),n\}}$:

      $$\tilde{Z}^{(f)}_{\{i(L),n\}} = \begin{bmatrix} \tilde{Z}^{(f)}_{\{i(L),n\}} \\ A_{l,i(L)} \end{bmatrix}$$

   - End loop over coarse levels ( $l$ )
3) Use SVD to compress $\tilde{Z}^{(f)}_{\{i(L),n\}}$, maintaining $O(\varepsilon^2)$ accuracy.
```

Figure 5. Pseudo-code for the algorithm used to compute the submatrix $\tilde{Z}^{(f)}_{\{i(L),n\}}$ of (20) for a given level-L source group, $i(L)$. The notation $\{i(L), n\}$ is used to denote the set of level-L groups which are near-neighbors of group $i(L)$. This set includes the self-group, $i(L)$.

The basic idea behind the algorithm is to build up $\tilde{Z}^{(f)}_{\{i(L),n\}}$ for a given level-L group, $i(L)$, by determining the contributions to this matrix due to non near-neighbor interactions occurring at the coarsest possible levels of the quad tree. This is indicated by the loop, $l = 3 : L$, specified as item 2 in Figure 5. For a given set of source groups, $\{i(L), n\}$, at each level, l, the algorithm subsequently loops (item 2b) over all observer groups, $j(l)$, which satisfy the following two conditions:

1. The observer group $j(l)$ is not a near-neighbor to any of the level-l parents of any of the groups contained in the set $\{i(L), n\}$.
2. The corresponding interactions between the source groups, $\{i(L), n\}$, and the observer group, $j(l)$, have not previously been incorporated at a coarser level.

For each level-l group satisfying these two conditions, the matrix $A_{j(l),i(L)}$, which defines the DoF radiated from the source region to the observer group $j(l)$, is extracted from the MLSSM representation of Z and appended to matrix $A_{l,i(L)}$ (item 2bi). The latter matrix ($A_{l,i(L)}$) is used to accumulate the $A_{j(l),i(L)}$ for all $j(l)$.

The matrix $A_{j(l),i(L)}$ is represented in terms of the MLSSM representation of Z as the nonzero part of

$$A_{j(l),i(L)} = I_{j(l)} \left(U_l^H \; Z_{l-1} \; V_l \; T_l \right) I_{\{i(L),n\}}, \tag{22}$$

where the matrices T_l are defined by the MLSSM source transformations:

$$T_{l-1} = V_l T_l, \tag{23}$$

with T_L an $N \times N$ identity matrix, $T_L = I_{N\times N}$. In (22), $I_{\{i(L),n\}}$ is a matrix with ones on its diagonal for columns associated with sources located in groups $\{i(L), n\}$. All other entries of $I_{\{i(L),n\}}$ are zero. $I_{j(l)}$ is similarly defined, having ones in the diagonal entries associated with observers located in group j at level-l. These exterior matrices are used in (22) to indicate extraction of the appropriate rows and columns from the MLSSM representation of the level-l DoF radiated from groups $\{i(L), n\}$ to distant observers in $j(l)$. In the numerical examples reported in Section 9, the extraction indicated by these operators is implemented by extracting only those elements of the MLSSM representation of Z which provide a nonzero contribution to $A_{j(l),i(L)}$.

After completing the loop over all of the level-l observer groups, and before proceeding to incorporate interactions defined at the next finer level of the tree, the accumulated matrix $A_{l,i(L)}$ is compressed (item 2c of Figure 5) using an SVD. This is accomplished by first computing the $O(\varepsilon^2)$ representation of the existing $A_{l,i(L)}$ matrix ($A_{l,i(L)} = usv^H + O(\varepsilon^2)$) followed by the replacement

$$A_{l,i(L)} \Leftarrow sv^H. \tag{24}$$

As indicated in item 2d of Figure 5, the compressed form of $A_{l,i(L)}$ is subsequently appended to $\tilde{Z}^{(f)}_{\{i(L),n\}}$, and the foregoing procedure is repeated at the next finer level of the quad-tree.

Finally, after completing the loop over the levels, an $O(\varepsilon^2)$ SVD representation of the matrix $\tilde{Z}^{(f)}_{\{i(L),n\}}$ is computed (cf. (18)), followed by the replacement (item 3)

$$\tilde{Z}^{(f)}_{\{i(L),n\}} \Leftarrow s^{(f)}(v^{(f)})^H. \tag{25}$$

At this point, $\tilde{Z}^{(f)}_{\{i(L),n\}}$ provides a mapping from sources in groups $\{i(L), n\}$ to the DoF necessary to represent these sources to distant observers located in all groups at levels 3 to L which are not near-neighbors to any of the source groups, $\{i(L), n\}$. Finally, by augmenting the matrix $\tilde{Z}^{(f)}_{\{i(L),n\}}$ with the near-neighbor interaction matrix, $Z^{(n)}_{\{i(L),n\}}$, we obtain the desired, $O(1) \times O(1)$, reduced matrix of (20), which can subsequently be used to determine the overlapped localizing LOGOS modes associated with group $i(L)$.

Because it relies on the sparse MLSSM representation of Z, the algorithm just described for $Z^{(r)}_{\{i(L),n\}}$ has an $O(N)$ memory complexity at low frequency. However, the CPU cost of the algorithm is unnecessarily high due to a large number of repeated computations performed in computing the $\tilde{Z}^{(f)}_{\{i(L),n\}}$ matrices. These costs can be reduced by computing and storing the data required to form the $A_{l,i(L)}$ in Figure 5. This results in a small increase in the total memory requirement.

6.3.4. A CPU Efficient Algorithm

To more efficiently determine the $\tilde{Z}^{(f)}_{\{i(L),n\}}$ matrices from the MLSSM representation of Z, it is useful to represent $A_{j(l),i(L)}$ of (22) as the product of two matrices,

$$A_{j(l),i(L)} = X_{j(l),P(l,i(L))} T_{P(l,i(L)),i(L)}, \tag{26}$$

where

$$X_{j(l),P(l,i(L))} = I_{j(l)} U_l^H Z_{l-1} V_l I_{P(l,i(L))}, \tag{27}$$

$$T_{P(l,i(L)),i(L)} = I_{P(l,i(L))} T_l I_{\{i(L),n\}}, \tag{28}$$

and T_l was defined in (23).

In (27) and (28), the symbol $P(l, i(L))$ is used to indicate all groups which are level-l parents of at least one group in the set $\{i(L), n\}$. $I_{P(l,i(L))}$ is a matrix with ones on its diagonal for columns associated with these level-l groups. All other entries of $I_{P(l,i(L))}$ are zero. Thus, the matrix $T_{P(l,i(L)),i(L)}$ transforms the DoF in groups $\{i(L), n\}$ to the associated nonzero level-l DoF required for interactions with distant groups. The matrix $X_{j(l),P(l,i(L))}$ contains all level-l interactions between observer group $j(l)$ and the level-l source groups which are parents of at least one group in the set $\{i(L), n\}$.

Using the definitions indicated in (26) to (28), Figure 6 summarizes a more CPU efficient version of the algorithm reported in Figure 5. The computational advantages of the improved algorithm are consequences of two modifications:

1. The matrices $T_{P(l,i(L)),i(L)}$ are filled (item 1b in Figure 6) and applied (item 2b) only once for each different source group.
2. The coarse level interaction matrices, $X_{l,P(l,i(L))}$, are only recomputed (item 2a) if they have not been previously stored (item 2aiv) subsequent to calculation for a different source group.

The CPU advantages of the first modification are evident. For a given $i(L)$, the algorithm in Figure 5 effectively requires re-computing $T_{P(l,i(L)),i(L)}$ for each different $j(l)$, whereas the modified algorithm in Figure 6 only requires that $T_{P(l,i(L)),i(L)}$ be computed once for each $i(L)$. Furthermore, the additional memory costs introduced by this modification are negligible. It is never necessary to store more than $O(\log N)$ of the $T_{P(l,i(L)),i(L)}$ at a given time in the calculation, and the $T_{P(l,i(L)),i(L)}$ are $O(1) \times O(1)$ at low frequency.

The savings obtained by the second modification listed above are more significant than those obtained from the first. However, these larger CPU savings are obtained at the expense of an increase (approximately ten percent) in the overall memory required by the algorithm.

The CPU savings provided by the second modification follow from the fact that many of the $X_{l,P(l,i(L))}$ are identical to one another, both at a given level (level-L in the present case), and across multiple levels. These redundancies are a simple consequence of the fact that, for $l < L$, multiple sets of level-L groups (i.e., the $\{i(L), n\}$) have the same set of

```
1) Initialization:
   a) Initialize $\tilde{\mathrm{Z}}^{(\mathrm{f})}_{\{i(L),\mathrm{n}\}}$ to an empty matrix
   b) Compute $\mathrm{T}_{\mathrm{P}(l,i(L)),i(L)}$ for $3 \le l \le L$.
2) Loop over all levels, proceeding from coarse to fine:
   for $l = 3 : L$
   a) If $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$ has not yet been computed:
      i) Initialize $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$ to an empty matrix.
      ii) Loop over all level-$l$ observer groups, $j(l)$,
         which are both not near-neighbors of the level-$l$
         parents of $\{i(L),\mathrm{n}\}$, and which have not been
         previously included at a coarser level.
         (1) For each such group, extract the matrix
               $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$ of (27)from the MLSSM representation
             of $\mathrm{Z}$ and append to existing $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$:
             $$\mathrm{X}_{l,\mathrm{P}(l,i(L))} = \begin{bmatrix} \mathrm{X}_{l,\mathrm{P}(l,i(L))} \\ \mathrm{X}_{j(l),\mathrm{P}(l,i(L))} \end{bmatrix}$$
         - End loop over far-field observer groups.
      iii) Use SVD to compress $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$ matrix, maintaining
           $O(\varepsilon^2)$ representation.
      iv) Store compressed $\mathrm{X}_{l,\mathrm{P}(l,i(L))}$ for re-use.
   b) Append compressed $\mathrm{A}_{l,i(L)}$ to $\tilde{\mathrm{Z}}^{(\mathrm{f})}_{\{i(L),\mathrm{n}\}}$ using (26):
      $$\tilde{\mathrm{Z}}^{(\mathrm{f})}_{\{i(L),\mathrm{n}\}} = \begin{bmatrix} \tilde{\mathrm{Z}}^{(\mathrm{f})}_{\{i(L),\mathrm{n}\}} \\ \mathrm{X}_{l,\mathrm{P}(l,i(L))}\mathrm{T}_{\mathrm{P}(l,i(L)),i(L)} \end{bmatrix}$$
   - End loop over coarse levels ( $l$ )
3) Use SVD to compress $\tilde{\mathrm{Z}}^{(\mathrm{f})}_{\{i(L),\mathrm{n}\}}$, maintaining $O(\varepsilon^2)$ accuracy.
```

Figure 6. A more CPU efficient version of the algorithm summarized in Figure 5.

level-l parents. This fact implies that many of the $X_{l,P(l,i(L))}$ required by (26) can be stored and reused to construct the $\tilde{Z}^{(f)}_{\{i(L),n\}}$. In contrast, the algorithm of Figure 5 effectively requires computing each of the $X_{l,P(l,i(L))}$ from scratch (i.e., via the MLSSM recursion indicated in (22) and the loop over the $j(l)$ in Figure 5) for each $\tilde{Z}^{(f)}_{\{i(L),n\}}$.

The additional memory required to store all of the $X_{l,P(l,i(L))}$

in the algorithm of Figure 6 is a relatively small, though not insignificant, fraction of the total memory. For all of the numerical examples reported in Section 9, the total memory required to store the $X_{l,P(l,i(L))}$ is less than ten percent of the total memory required by the factorization algorithm. At low frequencies, this memory requirement was observed to scale approximately as $O(N)$ for large N.

Once each $\tilde{Z}^{(f)}_{\{i(L),n\}}$ has been computed via the algorithm indicated in Figure 6, the reduced matrix $Z^{(r)}_{\{i(L),n\}}$ of (20) can be formed by augmenting $\tilde{Z}^{(f)}_{\{i(L),n\}}$ with $Z^{(n)}_{\{i(L),n\}}$. As indicated at the beginning of the previous subsection, the cost of the latter operation is $O(1)$ for each $i(L)$. Finally, given $Z^{(r)}_{\{i(L),n\}}$, the SVD-based analysis of [3, 5] can be used to determine the overlapped LOGOS modes associated with group $i(L)$ in $O(1)$ operations.

6.3.5. Multilevel LOGOS Mode Calculation

The previous subsections outline the algorithm used in the following numerical examples to determine the $Z^{(r)}_{\{i(L),n\}}$. Once the reduced matrices have been thus determined, the procedure used to calculate the corresponding elements of Λ_L and P_L from the $Z^{(r)}_{\{i(L),n\}}$ is essentially unchanged from that described in [3, 5].

The foregoing discussion for efficiently determining the reduced matrices, and thereby Λ_L and P_L, has been specialized to level-L for convenience and notational clarity. The modifications required to determine Λ_L and P_L from $Z^{(NN)}_{l+1}$ for $l < L$ are straightforward and will not be explicitly indicated (the required changes amount to changing subscript labels, etc.).

7. SUMMARY OF MULTILEVEL FACTORIZATION ALGORITHM

Figure 7 summarizes the discussion of previous sections by indicating the flow of the multilevel LOGOS-based factorization procedure for the MLSSM representation of the impedance matrix. Given an input MLSSM representation of Z, the factorization algorithm indicated in Figure 7 recursively finds the overlapping LOGOS modes for the $Z^{(NN)}_{l+1}$ submatrices using the algorithm discussed in Section 6. These modes are used to form the sparse transformation (Λ_l) and projection (P_l) matrices, which are subsequently used to project $Z^{(NN)}_{l+1}$

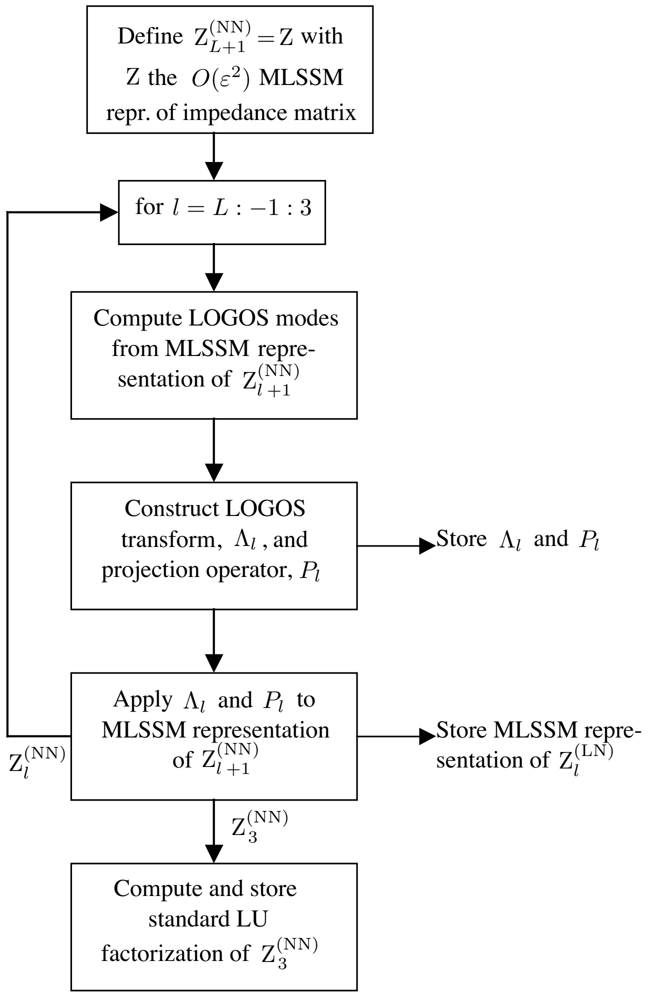
Define $Z_{L+1}^{(NN)} = Z$ with Z the $O(\varepsilon^2)$ MLSSM repr. of impedance matrix
for $l = L : -1 : 3$
Compute LOGOS modes from MLSSM representation of $Z_{l+1}^{(NN)}$
Construct LOGOS transform, Λ_l, and projection operator, P_l
Store Λ_l and P_l
Apply Λ_l and P_l to MLSSM representation of $Z_{l+1}^{(NN)}$
$Z_l^{(NN)}$
Store MLSSM representation of $Z_l^{(LN)}$
$Z_3^{(NN)}$
Compute and store standard LU factorization of $Z_3^{(NN)}$

Figure 7. LOGOS-based factorization of MLSSM representation of Z.

into the submatrices $Z_l^{(NN)}$ and $Z_l^{(LN)}$ of (7) using the MLSSM-based procedure discussed in Section 6.2. The submatrices $Z_l^{(LN)}$ are stored for use in the back substitution procedure discussed in Section 8, and the foregoing procedure is repeated for $Z_l^{(NN)}$ at the next coarser level of the quad tree. Finally, when the primary loop has finished, the remaining matrix, $Z_3^{(NN)}$, is inverted using a standard LU decomposition.

7.1. Complexity Estimate

It has been observed (via the numerical implementation reported in Section 9) that the CPU cost of the algorithm proposed in Figure 7 is dominated at low frequencies by the computation of overlapped modes at level-L of the quad tree. Therefore, we estimate the overall computational complexity of the proposed factorization algorithm by estimating the cost to determine the level-L OLL modes.

The cost to compute the OLL modes at level-L can be decomposed into three components. These components and their estimated complexities follow.

1. Cost to build all of the $X_{l,P(l,i(L))}$.

   We begin by recalling that the $X_{l,P(l,i(L))}$ define the interactions between the level-l parents of sources in the set $\{i(L), n\}$ and non near-neighbor level-l observers that could not be represented at a coarser level. Due to the redundancies indicated in Section 6.3.4, the number of unique $X_{l,P(l,i(L))}$ to be calculated scales as $O(N)$. Furthermore, our assumption of low frequency applications implies that each $X_{l,P(l,i(L))}$ is $O(1) \times O(1)$. Together, these facts imply that the total cost to *store* the $X_{l,P(l,i(L))}$ will scale as $O(N)$ (as indicated in Section 6.3.4, this is consistent with our observations of the numerical implementation reported in Section 9). However, the CPU cost to *compute* any one of the $X_{l,P(l,i(L))}$ is bounded by $O(\log N)$ (and not $O(1)$). For example, when $l \approx L$, calculation of a single $X_{l,P(l,i(L))}$ via equation (27) can require up to $O(\log N)$ operations due to the need to traverse the multilevel tree. Combining this estimate for the cost to compute a single $X_{l,P(l,i(L))}$ with the fact that there are $O(N)$ such matrices to be computed yields an upper complexity bound for this operation of $O(N \log N)$.

2. Cost to compress the $\tilde{Z}^{(f)}_{\{i(L),n\}}$.

   As indicated in item 2b of Figure 6, each of the $\tilde{Z}^{(f)}_{\{i(L),n\}}$ is obtained by accumulating the $A_{l,i(L)}$. Since (i) there are $O(\log N)$ levels, and (ii) each of the $A_{l,i(L)}$ is $O(1)$-by-$O(1)$, it follows that the SVD-based compression required for each $\tilde{Z}^{(f)}_{\{i(L),n\}}$ has a cost of $O(\log N)$. Because there are $O(N)$ groups at level-L, the total cost for this operation is $O(N \log N)$.

3. Cost to compute LOGOS modes from the $Z^{(r)}_{\{i(L),n\}}$.

   Because each of the $Z^{(r)}_{\{i(L),n\}}$ is $O(1) \times O(1)$, the cost to determine the level-L overlapped localizing LOGOS (OLL) modes from the $Z^{(r)}_{\{i(L),n\}}$ for a single group, $i(L)$, is $O(1)$. Since there are $O(N)$ such groups, the total CPU cost required to determine all level-L OLL modes given the $Z^{(r)}_{\{i(L),n\}}$ is $O(N)$.

Together these estimates indicate that the CPU cost to determine the level-L OLL modes from the MLSSM representation of Z is $O(N \log N)$. Because (i) the time to determine LOGOS modes always dominates the total CPU time, and (ii) the time to determine LOGOS modes for each projection of the impedance matrix, $Z^{(NN)}_{l+1}$, decreases rapidly with l, we conclude that this estimate (i.e., $O(N \log N)$) for the cost to determine the level-L OLL modes also bounds the total CPU complexity of the proposed factorization algorithm. This (non rigorous) estimate is evaluated in Section 9 through a comparison with numerical data.

8. SPARSE IMPLEMENTATION OF Z^{-1}

The LOGOS-based factored representation of the impedance matrix yields the following sparse algorithm for implementing $J = -Z^{-1}E^i$.

1. For $l = L : -1 : 3$, use the projection matrices, P_l, to compute:

$$\begin{bmatrix} E^{(L)}_l \\ E^{(N)}_l \end{bmatrix} = \left[P^{(L)}_l, P^{(N)}_l\right]^H E^{(N)}_{l+1}, \tag{29}$$

where $E^{(N)}_{L+1} \equiv -E^i$. Due to its recursive form, the level-l quantities on the left side of (29) can be stored in the memory space initially used to store $E^{(N)}_{l+1}$ on the right side of the equation. The indicated recursion terminates when $l = 3$, by which point the memory initially used to store the incident vector is replaced by the transformed quantity $\tilde{E}$,

$$E^i \Leftarrow \tilde{E} = \left[E^{(L)}_L, E^{(L)}_{L-1}, \ldots, E^{(L)}_3, E^{(N)}_3\right]^T. \tag{30}$$

The elements of $\tilde{E}$ are the projections of the incident field onto the fields scattered by the overlapped, localizing modes at levels L through 3. The final portion of $\tilde{E}$, $E^{(N)}_3$, represents the projection of the incident field onto the scattered fields associated with modes which could not be localized at levels $l \geq 3$.

2. The back substitution procedure continues with the calculation of the quantity
$$J_3^{(N)} = \left(Z_3^{(NN)}\right)^{-1} E_3^{(N)}, \tag{31}$$
where $E_3^{(N)}$ is extracted from $\tilde{E}$ in (30). The inverse operation in (31) is performed using a standard LU decomposition.
3. Given (30) and (31), the entire solution vector is obtained via back substitution. For $l = 3 : L$:
$$J_{l+1}^{(N)} = \Lambda_l \begin{bmatrix} E_l^{(N)} - Z_l^{(LN)} J_l^{(N)} \\ J_l^{(N)} \end{bmatrix}, \tag{32}$$
4. The desired solution vector is $J = -Z^{-1}E^i \approx J_{L+1}^{(N)}$.

The solution procedure indicated by (29) through (32) is used in the following numerical examples.

9. NUMERICAL EXAMPLES

In the following numerical examples, MATLAB implementations of the sparse factorization and solution algorithms summarized above are applied to the EFIE formulation of TM$_z$ scattering from the square arrays of dihedral cylinders indicated by Figure 8. In particular, configurations consisting of 8×8, 16×16, 32×32 and 64×64 dihedral elements are considered. Table 1 lists the associated numbers of unknowns and quad-tree levels used for each array size.

Table 2 lists the number of nonzero (nnz) complex double precision numbers required to store the LOGOS factored impedance matrix for several element sizes (L) and two different values of the inter-element spacing: $\Delta = 0.4L$ and $\Delta = 2.4L$. Table 3 reports the wall-clock times required to perform the standard LU factorization and the LOGOS factorization of Z. The LU factorization was performed using MATLAB's built-in "lu" function, which relies on tuned LAPACK libraries. The LOGOS factorization times were obtained from a straightforward MATLAB implementation of the factorization algorithm discussed above. It is likely that the latter MATLAB timings could be significantly reduced by further improving our implementation. This was not done, however, because the primary purpose of the TM$_z$ implementation reported here is to evaluate the asymptotic complexities of the proposed factorization strategy. More efficient implementations will be developed for three-dimensional electromagnetic applications.

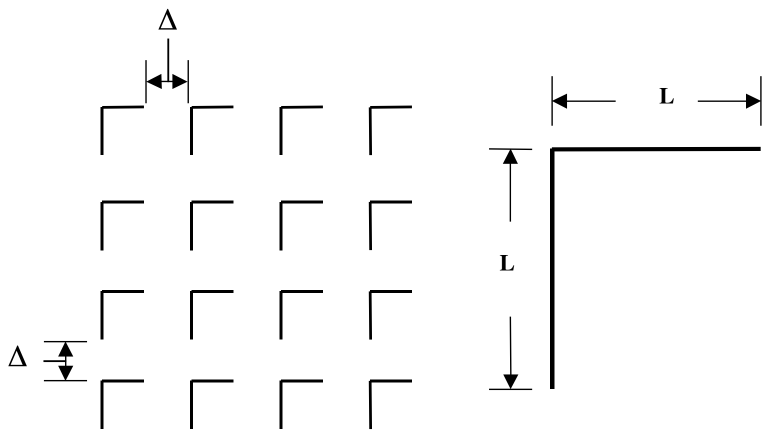

Figure 8. (*Left*) 4-by-4 array of dihedral PEC cylinders. (*Right*) Single element. The numbers of array elements in x, y-directions are N_x and N_y. The spacing between array elements is Δ. The thickness of each array element is zero. In all cases, each individual array element is discretized using 10 facets. The side length of each dihedral element is indicated by the letter "L."

Referring to Table 2, we observe that the memory complexity of the LOGOS/MLSSM factored representation of Z scales nearly linearly with N for electrically small arrays and large values of N. For example, when $L = 0.001\lambda$, the complexity of the factored representation increases by factors of 4.21 ($\Delta = 0.4L$) and 4.15 ($\Delta = 2.4L$) when N is increased by a factor of 4 from $N = 10240$ to $N = 40960$. The electrical dimensions of the associated 64×64 element arrays are approximately $0.09\lambda \times 0.09\lambda$ ($\Delta = 0.4L$) and $0.22\lambda \times 0.22\lambda$ ($\Delta = 2.4L$).

Comparing these results to those listed in Table 2 for larger values of L, it is clear that the complexity of the factored LOGOS representation scales more rapidly with N as the electrical size of the array increases. For example, when $L = 0.1\lambda$ the complexity of the LOGOS representation increases by factors of 4.6 ($\Delta = 0.4L$) and 4.5 ($\Delta = 2.4L$) when the array size increases from 32×32 to 64×64 elements. In these cases, the electrical dimensions of the associated 64×64 element arrays are approximately $9\lambda \times 9\lambda$ ($\Delta = 0.4L$) and $22\lambda \times 22\lambda$ ($\Delta = 2.4L$). These results suggest that, although the complexity of the factored representation increases with electrical size, the algorithm still provides a relatively efficient representation for targets which are several wavelengths in each linear dimension.

The factorization times reported in Table 3 indicate that the CPU

Table 1. Number of unknowns and number of quad tree levels for various square array dimensions used in numerical examples.

array dim.	N	levels
8×8	640	4
16×16	2560	5
32×32	10240	6
64×64	40960	7

Table 2. Complexity of factored representation ($\varepsilon = 0.001$).

array dimension	L/λ	nnz $\Delta = 0.4L$	nnz $\Delta = 2.4L$
8×8	0.001	1.738×10^5	1.553×10^5
16×16	0.001	1.002×10^6	8.741×10^5
32×32	0.001	4.945×10^6	4.400×10^6
64×64	0.001	2.081×10^7	1.825×10^7
8×8	0.01	1.809×10^5	1.616×10^5
16×16	0.01	1.086×10^6	9.522×10^5
32×32	0.01	5.439×10^6	4.899×10^6
64×64	0.01	2.368×10^7	2.228×10^7
8×8	0.1	2.068×10^5	2.015×10^5
16×16	0.1	1.324×10^6	1.495×10^5
32×32	0.1	7.038×10^6	9.216×10^6
64×64	0.1	3.239×10^7	4.908×10^7

time of the LOGOS factorization scales somewhat more rapidly than the memory complexity. Furthermore, the LOGOS factorization time is slower than the LU factorization until the problem size exceeds about 10^4 unknowns. The latter observation may in part be a result of our use of a non-optimized MATLAB implementation of the LOGOS algorithm. It is possible that the LOGOS factorization times reported in Table 3 could be reduced by optimizing the MATLAB code. This possibility will be more fully explored for the 3-D implementation of the factorization algorithm.

Table 3. LU and LOGOS factorization times in seconds. LOGOS timings obtained with $\Delta = 2.4L$ and $\varepsilon = 0.001$. LU timings reported for $N > 2560$ obtained by extrapolating from the $N = 2560$ time assuming at an N^3 scaling rate. All timing results obtained using a PC with a 3 GHz Pentium 4 processor.

array dim.	L/λ	LU	LOGOS
8×8	0.001	0.30	8.2
16×16	0.001	13.7	70.4
32×32	0.001	875	461.9
64×64	0.001	56000	2530
8×8	0.01	0.30	8.45
16×16	0.01	13.7	66.4
32×32	0.01	875	475.6
64×64	0.01	56000	2839
8×8	0.1	0.30	11.2
16×16	0.1	13.7	98.1
32×32	0.1	875	886
64×64	0.1	56000	6370

As discussed in Section 7, it is expected that the computational complexity of the proposed factorization strategy will scale as $O(N \log N)$ for electrically small targets when N is large. For $L = 0.001\lambda$, the data in Table 3 indicate that the factorization time scales by a factor of approximately 6.56 when the array size increases by a factor of four, from 16×16 to 32×32 elements. The factorization time increases by the smaller factor of 5.48 when the array size is further increased from 32×32 to 64×64 elements. These results indicate that our implementation of the sparse LOGOS factorization has yet to clearly demonstrate the anticipated CPU rate of $O(N \log N)$ for problems involving up to 40960 unknowns (the factor of 5.48 corresponds to a rate of approximately $N^{1.23}$ in passing from 10240 to 40960 unknowns). It will be necessary to consider larger problems to determine if the complexity estimate of $O(N \log N)$ is accurate. This point will be further pursued in the 3-D application of the factorization algorithm.

Table 4. Time (in seconds) to solve $J = -Z^{-1}E^{inc}$ for 100 randomly generated incident field vectors using standard LU and sparse LOGOS factorizations of Z when $\Delta = 2.4L$. LU timings reported for $N > 2560$ obtained by extrapolating from the $N = 2560$ time assuming an N^2 scaling rate. All computations done on a 3 GHz Pentium 4 processor.

array dim.	L/λ	LU	LOGOS
8×8	0.001	0.16	0.30
16×16	0.001	2.46	2.89
32×32	0.001	39.4	18.6
64×64	0.001	630	86.3
8×8	0.01	0.16	0.34
16×16	0.01	2.46	4.02
32×32	0.01	39.4	20.8
64×64	0.01	630	102
8×8	0.1	0.16	0.39
16×16	0.1	2.46	5.86
32×32	0.1	39.4	38.23
64×64	0.1	630	251

As observed above for the memory complexity, the computational complexity of the proposed LOGOS factorization algorithm is also expected to increase with the electrical size of the array. This is indeed the case for the configurations reported in Table 3. For a given array dimension, the time to perform the factorization increases as L is increased from $L = 0.001\lambda$ to $L = 0.1\lambda$.

Table 4 shows the wall-clock time required to solve (1) using the solution procedure summarized in Section 8. In all cases, the time to solve (1) using the LOGOS representation is less than the time required using the standard LU factored form.

Finally, Table 5 displays the relative RMS error in the LOGOS factored representation of Z and the original, full impedance matrix for the cases considered in Table 2 and Table 3. In each case, the observed error in the boundary condition (1) is within the requested tolerance of $\varepsilon = 0.001$.

Table 5. Relative RMS error in factored representation of Z for the dihedral arrays considered in Tables 1 and 2. The requested tolerance was $\varepsilon = 0.001$.

array dimension	L/λ	RMS error $\Delta = 0.4L$	RMS error $\Delta = 2.4L$
8×8	0.001	2×10^{-4}	2×10^{-4}
16×16	0.001	4×10^{-4}	5×10^{-4}
32×32	0.001	6×10^{-4}	5×10^{-4}
64×64	0.001	3×10^{-4}	6×10^{-4}
8×8	0.01	2×10^{-4}	2×10^{-4}
16×16	0.01	3×10^{-4}	1×10^{-4}
32×32	0.01	3×10^{-4}	4×10^{-4}
64×64	0.01	6×10^{-4}	6×10^{-4}
8×8	0.1	2×10^{-4}	2×10^{-4}
16×16	0.1	5×10^{-4}	6×10^{-4}
32×32	0.1	3×10^{-4}	7×10^{-4}
64×64	0.1	6×10^{-4}	7×10^{-4}

10. SUMMARY AND DISCUSSION

A sparse factorization strategy for the MLSSM representation of the impedance matrix based on the expansion of the impedance matrix in a basis of overlapped, localizing LOGOS modes has been summarized, and representative numerical examples have been reported. The numerical data indicate that the complexity of the resulting, factored representation of the impedance matrix scales approximately as $O(N)$ at low frequencies. The complexity of the factored representation is observed to be somewhat larger for targets which are several wavelengths in dimension.

The CPU complexity of the reported factorization strategy has been estimated to scale asymptotically as $O(N \log N)$ at low frequencies. However, the numerical examples exhibit a computational complexity which scales of approximately as $O(N^{1.23})$ in passing from 10240 to 40960 unknowns. It is expected that this rate will continue to decrease toward the expected asymptotic rate of $O(N \log N)$ when larger problems are considered. This will be the subject of future investigations.

Several improvements of the LOGOS-based algorithms discussed above are possible. For example, the factorization algorithm reported here relies on an MLSSM representation of Z and its projections. It is expected that similarly efficient factorization strategies that rely on alternate sparse representations of Z are also possible. For example, it seems likely that the algorithms reported above could be modified to use only SVD-based representations instead of the MLSSM representation. However, even if this is correct, the specific advantages provided by such alternatives remain to be determined.

As indicated in Figure 1 and Figure 7, the factorization algorithm reported in this paper relies on an $O(\varepsilon^2)$ representation of the impedance matrix and its projections. This was found to be necessary in order to determine all LOGOS modes localized to $O(\varepsilon)$ within a given group at a given level. However, once these modes have been determined and used to project the impedance matrix (cf. (7)), it is no longer necessary to maintain an $O(\varepsilon^2)$ tolerance for $Z_l^{(LN)}$, since no additional LOGOS modes will be extracted from this matrix. This suggests that additional memory savings might be possible over those reported in Table 2 by storing only $O(\varepsilon)$ representations of the $Z_l^{(LN)}$. This conjecture has been corroborated by preliminary numerical examples and will be pursued further elsewhere.

At low frequencies, the largest computational cost of the factorization algorithm described here is associated with finding the overlapped localizing LOGOS modes at the finest level of the quad-tree. It may be possible to reduce this cost by modifying the proposed algorithm to simultaneously incorporate both overlapped and non-overlapped modes. Such a modification might also provide additional memory savings.

Multiple additional improvements of the direct solution algorithms reported here might also be useful. Some of these are being pursued and will be reported if appropriate. The 3-D implementation of the LOGOS-based factorization and solution algorithms reported here is also in progress.

ACKNOWLEDGMENT

This effort was supported by the Office of Naval Research under contract N00014-04-1-0485 and by the Center for Computational Sciences at the University of Kentucky.

REFERENCES

1. Peterson, A. F., S. L. Ray, and R. Mittra, *Computational Methods for Electromagnetics*, IEEE Press, New York, 1998.
2. Adams, R. J., G. Wang, and F. X. Canning, "Efficient inversion of the impedance matrix in an overlapping, localizing basis," *USNC-URSI National Radio Science Meeting*, Boulder, CO, January 2006.
3. Adams, R. J., F. X. Canning, and A. Zhu, "Sparse representations of integral equations in a localizing basis," *Microwave and Optical Technology Letters*, Vol. 47, 236–240, 2005.
4. Adams, R. J., F. X. Canning, and A. Zhu, "Fast solution of integral equations in a localizing basis," *IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting*, Washington, DC, July 2005.
5. Adams, R. J., A. Zhu, and F. X. Canning, "Efficient solution of integral equations in a localizing basis," *Journal of Electromagnetic Waves and Applications*, Vol. 19, 1583–1594, 2005.
6. Zhu, A., R. J. Adams, F. X. Canning, and S. D. Gedney, "Sparse solution of an integral equation formulation of scattering from open PEC targets," *Microwave and Optical Technology Letters*, Vol. 48, 476–480, 2006.
7. Zhu, A., R. J. Adams, F. X. Canning, and S. D. Gedney, "Schur factorization of the impedance matrix in a localizing basis," *Journal of Electromagnetic Waves and Applications*, Vol. 20, 351–362, 2006.
8. Bebendorf, M., "Approximation of boundary element matrices," *Numerische Mathematik*, Vol. 86, 565–589, 2000.
9. Zhu, A., R. J. Adams, and F. X. Canning, "Multilevel simply sparse method for scattering by PEC," *IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting*, Vol. 4A, 427–430, Washington, DC, July 2005.
10. Chew, W. C., J. M. Jin, E. Michielssen, and J. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*, Artech House, Boston, 2001.
11. Canning, F. X. and K. Rogovin, "A universal matrix solver for integral-equation-based problems," *IEEE Antennas and Propagation Magazine*, Vol. 45, 19–26, 2003.
12. Bucci, O., "On the degrees of freedom of scattered fields," *IEEE Transactions on Antennas and Propagation*, Vol. 37, 1989.