ERROR CONTROL AND EFFICIENT MEMORY MANAGEMENT FOR SPARSE INTEGRAL EQUATION SOLVERS BASED ON LOCAL-GLOBAL SOLUTION MODES

Jun-shik Choi
University of Kentucky, junshikchoi@gmail.com

Recommended Citation
Choi, Jun-shik, "ERROR CONTROL AND EFFICIENT MEMORY MANAGEMENT FOR SPARSE INTEGRAL EQUATION SOLVERS BASED ON LOCAL-GLOBAL SOLUTION MODES" (2014). Theses and Dissertations--Electrical and Computer Engineering. 64.
https://uknowledge.uky.edu/ece_etds/64
STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student’s advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student’s thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Jun-shik Choi, Student
Dr. Robert J. Adams, Major Professor
Dr. Cai-Cheng Lu, Director of Graduate Studies
ERROR CONTROL AND EFFICIENT MEMORY MANAGEMENT FOR SPARSE INTEGRAL EQUATION SOLVERS BASED ON LOCAL-GLOBAL SOLUTION MODES

DISSERTATION

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the College of Engineering at the University of Kentucky

By
Jun-shik Choi
Lexington, Kentucky

Director: Dr. Robert J. Adams, Associate Professor of Electrical and Computer Engineering Lexington, Kentucky

2014
Copyright © Jun-shik Choi 2014
ABSTRACT OF DISSERTATION

ERROR CONTROL AND EFFICIENT MEMORY MANAGEMENT FOR SPARSE INTEGRAL EQUATION SOLVERS BASED ON LOCAL-GLOBAL SOLUTION MODES

This dissertation presents and analyzes two new algorithms for sparse direct solution methods based on the use of local-global solution (LOGOS) modes. One of the new algorithms is a rigorous error control strategy for LOGOS-based matrix factorizations that utilize overlapped, localizing modes (OL-LOGOS) on a shifted grid. The use of OL-LOGOS modes is critical to obtaining asymptotically efficient factorizations from LOGOS-based methods. Unfortunately, the approach also introduces a non-orthogonal basis function structure. This can cause errors to accumulate across levels of a multilevel implementation, which has previously posed a barrier to rigorous error control for the OL-LOGOS factorization method. This limitation is overcome, and it is shown that it is possible to efficiently decouple the fundamentally non-orthogonal factorization subspaces in a manner that prevents multilevel error propagation. This renders the OL-LOGOS factorization error controllable in a relative RMS sense. The impact of the new, error-controlled OL-LOGOS factorization algorithm on computational resource utilization is discussed and several numerical examples are presented to illustrate the performance of the improved algorithm relative to previously reported results.

The second algorithmic development considered is the development of efficient out-of-core (OOC) versions of the OL-LOGOS factorization algorithm that allow associated software tools to take advantage of additional resources for memory management. The proposed OOC algorithm incorporates a memory page definition that is tailored to match the flow of the OL-LOGOS factorization procedure. Efficiency of the function of the part is evaluated using a quantitative approach, because the tested massive storage device performances do not follow analytical results. The performance latency and the memory usage of the resulting OOC tools are compared with in-core performance results.

Both the new error control algorithm and the OOC method have been incorporated into previously existing software tools, and the dissertation presents results for real-world simulation problems.
KEYWORDS: local-global solution, error control, out of core
ERROR CONTROL AND EFFICIENT MEMORY MANAGEMENT FOR SPARSE INTEGRAL EQUATION SOLVERS BASED ON LOCAL-GLOBAL SOLUTION MODES

By

Jun-shik Choi

______________________________
Robert J. Adams
Director of Dissertation

______________________________
Cai-Cheng Lu
Director of Graduate Studies

Dec 15, 2014
ACKNOWLEDGMENTS

As the time to complete the study with a PhD degree, I become aware that there are many people around me to support and help. I am so pleased to share this moment with the people.

I would like to thank my advisor Professor Robert J. Adams for his guiding me to finish my research during the long period of time. As an enthusiastic researcher, he shows me a respectable role model not only in the past time but also for the future of my career. He always encourages me to reach the goal whenever I have the difficulties academically and personally.

The members of my dissertation committee, Dr. Robert J. Adams, Dr. Cai-Cheng Lu, Dr. John Young, Dr. Suzanne W. Smith and Dr. D. Manivannan, have thankfully allowed their time for my work. I deeply appreciate their contribution. Dr. Stephen Gedney, who left University of Kentucky, also gave the classes to help me to build the CEM background.

I have to thank specially to my friends providing with the ideas and having the discussions for my research. Especially, Dr. Jin Cheng provides the augmented EFIE codes to test my code with the AEIFE system. I also thank Dr. Xin Xu, Dr. Yuan Xu, Dr. Zhiyong Zeng and Ms. Nastaran Hendijani.

Finally, I would also like to give thanks to my parents, wife, Hyoung-eun, Ian and Benjamin for their patience, support, and love.
# Table of Contents

Acknowledgments ........................................................................................................ iii

List of Figures ............................................................................................................... vii

Chapter 1. Introduction ............................................................................................... 1

1.1. Motivation .............................................................................................................. 1

1.2. Linear systems and solution methods ................................................................. 1

1.3. The fast direct solver using Local-Global Solution (LOGOS) modes .............. 4

1.4. The Error Control in the LOGOS factorization ............................................... 6

1.4.1. Overview .......................................................................................................... 6

1.4.2. Basic Idea ......................................................................................................... 7

1.5. The Out Of Core (OOC) Operation ...................................................................... 8

1.5.1. Overview .......................................................................................................... 8

1.5.2. Basic Idea ......................................................................................................... 9

1.6. Objective and scope ............................................................................................ 10

Chapter 2. The LOGOS Factorization ....................................................................... 12

2.1. The Geometry ..................................................................................................... 12

2.2. Determination of Localizing modes .................................................................. 13

2.3. The solution procedure ...................................................................................... 19

2.4. The Source Range in the LOGOS Factorization ............................................. 21

2.4.1. NL-LOGOS ..................................................................................................... 23

2.4.2. OL-LOGOS .................................................................................................... 23

2.4.3. Shifted Grid (SG) LOGOS ............................................................................ 24

2.5. Error Estimations of the LOGOS Factorization .......................................... 25

Chapter 3. The Error Analysis and Control Of the LOGOS Factorization ........... 35

3.1. The RMS error of the system ........................................................................... 35
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2. The inverse of the basis matrix on the non-orthogonal spaces</td>
<td>36</td>
</tr>
<tr>
<td>3.3. Separation of The Error Terms</td>
<td>39</td>
</tr>
<tr>
<td>3.4. The Errors From $\Omega^{(L)}$ Space</td>
<td>42</td>
</tr>
<tr>
<td>3.4.1. Localizing method for the initial basis</td>
<td>43</td>
</tr>
<tr>
<td>3.4.2. Adjusting the number of the localizing modes on $\Omega^{(L)}$</td>
<td>46</td>
</tr>
<tr>
<td>3.4.3. The basis computation components</td>
<td>47</td>
</tr>
<tr>
<td>3.5. The Next Level Tolerance Computation</td>
<td>49</td>
</tr>
<tr>
<td>3.6. Numerical Results</td>
<td>51</td>
</tr>
<tr>
<td>3.6.1. The AEFIE system result</td>
<td>53</td>
</tr>
<tr>
<td>3.6.2. The MFIE system result</td>
<td>54</td>
</tr>
<tr>
<td>3.6.3. The VIE system result</td>
<td>55</td>
</tr>
<tr>
<td>3.6.4. The EFIE system result</td>
<td>56</td>
</tr>
<tr>
<td>3.6.5. Summary of the Numerical Results</td>
<td>57</td>
</tr>
<tr>
<td>Chapter 4. Efficient memory management</td>
<td>78</td>
</tr>
<tr>
<td>4.1. Background</td>
<td>78</td>
</tr>
<tr>
<td>4.2. Memory Hierarchy</td>
<td>79</td>
</tr>
<tr>
<td>4.3. The Hard Disk Drive</td>
<td>81</td>
</tr>
<tr>
<td>4.3.1. Hardware Properties</td>
<td>81</td>
</tr>
<tr>
<td>4.3.2. Data Operation on the Disk</td>
<td>82</td>
</tr>
<tr>
<td>4.4. The Data page structure for the Out-Of-Core (OOC) LOGOS</td>
<td>83</td>
</tr>
<tr>
<td>4.4.1. The OL-LOGOS Factorization with the Matrix Data Page</td>
<td>85</td>
</tr>
<tr>
<td>4.4.2. The data page structure</td>
<td>85</td>
</tr>
<tr>
<td>4.4.3. The data page operating algorithm</td>
<td>88</td>
</tr>
<tr>
<td>4.4.4. The SG-LOGOS with OOC implementation</td>
<td>90</td>
</tr>
<tr>
<td>4.5. Numerical results</td>
<td>91</td>
</tr>
</tbody>
</table>
4.5.1. The SG-LOGOS with OOC for MFIE ................................................................. 92
4.5.2. The SG-LOGOS with OOC for VIE ................................................................. 93
4.5.3. Summary of the Numerical Results ................................................................. 94

Chapter 5. Conclusions .......................................................................................... 117

5.1. Contributions .................................................................................................. 118
5.2. Future Work .................................................................................................... 118

References ............................................................................................................. 120

Vita .......................................................................................................................... 124
List of Figures

Figure 1. The Square Plate in 3 Level Quad-Tree: The relation from the children to the parent at the higher level of the Quad-Tree. Each Parent has four regular children and the children groups “collapse” into one parent group to build the next level group in the localizing process. ................................................................. 27
Figure 2: NL-LOGOS field group (28) and the source group (28) on the tree .......... 28
Figure 3: NL-LOGOS Basis structure. At both the localizing and the non-localizing matrices are block diagonal and the orthogonal to each other. ....................... 29
Figure 4: OL-LOGOS field group (28) and the overlapped source groups on the tree at the Level-4 ........................................................................................................................................ 30
Figure 5: OL-LOGOS field group (28) and the overlapped source groups of the children’s neighbors on the tree at the Level-4 ......................................................... 31
Figure 6: OL-LOGOS Basis Structure. The localizing basis (red) has a coupling structure, while the non-localizing basis (blue) has a block diagonal (non-coupled) structure. The localizing space is not orthogonal to the non-localizing space, causing a coupling between these subspaces when the matrix is inverted........... 32
Figure 7: SG-LOGOS field group (28) and the shifted source groups on the tree ........ 33
Figure 8: SG-LOGOS Basis Structure. On the localizing basis the coupling is partially released due to the shifted grid based overlapping range. The relation between the localizing and the non-localizing spaces, however, is still not orthogonal. Critically, however, the inverse of this matrix is block diagonal; the coupling range is limited....................................................................................................................... 34
Figure 9: the singular value plot from SVD........................................................................ 59
Figure 10: The non-localizing basis computation relation .............................................. 60
Figure 11: The localizing basis computation and update ................................................. 61
Figure 12: AEFIE error result (Heuristic Error Control) .................................................. 62
Figure 13: AEFIE error result (Error Control) .................................................................. 63
Figure 14: AEFIE CPU performance result (Heuristic Error Control) ......................... 64
Figure 15: AEFIE CPU performance result (Error Control) ........................................... 65
Figure 16: MFIE error result (Heuristic Error Control) .................................................. 66
Figure 17: MFIE error result (Error Control) .................................................................. 67
Figure 18: MFIE CPU performance result (Heuristic Error Control) ......................... 68
Figure 19: MFIE CPU performance result (Error Control) ............................................. 69
Figure 20: VIE error result (Heuristic Error Control) ..................................................... 70
Figure 21: VIE error result (Error Control) .................................................................. 71
Figure 22: VIE CPU performance result (Heuristic Error Control) ........................... 72
Figure 23: VIE CPU performance result (Error Control) ............................................... 73
Figure 24: EFIE error result (Heuristic Error Control) ................................................... 74
Figure 25: EFIE error result (Error Control) ................................................................. 75
Figure 26: EFIE CPU performance result (Heuristic Error Control) ......................... 76
Figure 27: EFIE CPU performance result (Error Control) ............................................. 77
Figure 28: the simple memory hierarchy ..................................................................... 96
Figure 29: The logical structure of the magnetic disk .................................................... 97
Figure 30: the data page based on the grandparent level for the SG-LOGOS ............. 98
Figure 31: the data page based on the parent level for the SG-LOGOS ....................... 99
Figure 32: the flowchart for the page change ............................................................... 100
Figure 33: the page removal flowchart ....................................................................... 101
Figure 34: the matrix block data allocation from the data page and the modified block 102
Figure 35: MFIE CPU performance with the tolerance=10e-2 ................................ 103
Figure 36: MFIE Peak Memory with the tolerance = 10e-2 ....................................... 104
Figure 37: MFIE CPU performance with the tolerance=10e-3 .................................. 105
Figure 38: MFIE Peak Memory with the tolerance=10e-3 ......................................... 106
Figure 39: MFIE CPU performance with the tolerance=10e-4 .................................. 107
Figure 40: MFIE Peak Memory with the tolerance=10e-4 ......................................... 108
Figure 41: VIE CPU performance with the tolerance=10e-2 .................................... 109
Figure 42: VIE Peak Memory with the tolerance=10e-2 ........................................... 110
Figure 43: VIE CPU performance with the tolerance=10e-3 .................................... 111
Figure 44: VIE Peak Memory with the tolerance=10e-3 ........................................... 112
Figure 45: VIE CPU performance with the tolerance=10e-4 .................................... 113
Figure 46: VIE Peak Memory with the tolerance=10e-4 ........................................... 114
Figure 47: MFIE Hit Ratio........................................................................................................... 115
Figure 48: VIE Hit Ratio........................................................................................................... 116
Chapter 1. INTRODUCTION

1.1. Motivation

In computational electromagnetic (CEM) applications, direct solution methods using local-global modes (LOGOS) have been shown to reduce computational complexities compared to traditional direct solutions methods (such as LU factorization). This work addresses two issues related to LOGOS-based factorizations. First, the LOGOS factorization is approximate; secondly, the memory costs of the factorization can limit the range of problems that can be addressed. Since the factorization is approximate, it is critical that the error be controllable. Prior to this work, such error control has not been possible when the most efficient forms of the factorization are used. In addition, this dissertation provides an analysis and implementation of efficient external device operation methods for the LOGOS framework.

1.2. Linear systems and solution methods

Linear time-harmonic electromagnetic problems are represented with the boundary value problems with the linear operators ($\mathcal{L}$), the unknown quantity ($\phi$) and the excitation or forcing function ($f$). Either the integral or differential operators can be chosen to be the operator by the governing equation:

$$\mathcal{L}\phi = f$$

(1.1)

In most cases, the equation (1.1) is not solved analytically, so then the numerical methods [1-10] are used to project the operator on the continuous domain onto the discrete domain in the form of:

$$Z\vec{x} = \vec{F}$$

(1.2)

The equation (1.2) is in a form that is solvable with the computer system. $Z$ is the system matrix, generally with the dimension of $N \times N$; the right hand side (RHS), $\vec{F}$, has excitation or forcing vectors; finally, $\vec{x}$ contains the unknown vectors associated with the corresponding to the RHS vectors.

For the electromagnetic problems, the vector Helmholtz equation can plays the role of the linear operator, resulting in the PDE problem. The PDE can be discretized both
directly from the differential operator, and it can be discretized using an integral operator along the boundary with the given excitation at the RHS. The example of the former methodology is the finite element method (FEM) [3, 4, 8, 10], and the locally corrected Nyström method [9] is one of the integral equation methods. FEM discretizes the entire space of the domain, and it leads the relatively larger system matrices than ones from the integral equation methods. The FEM system matrices often have great value of the condition numbers due to the unbounded property of the differential operator; with proper formulation, however, the integral operator may be bounded [11]. Regarding the matrix structure, the localness of FEM formulations produces the sparse structure of the matrix. The sparse structure allows efficient algorithms to exploit time saving linear algebras, like the matrix-vector multiplication and so on. In the case of the integral equations, however, the non-local property of the operators result in the dense system matrix. The complexity of the algebraic works for the dense matrices grows at even larger than the quadratic rate with respect to the number of unknowns, $N$.

The linear system equation can be solved iteratively or with direct methods. The iterative method examples are the conjugate gradient (CG) method, the generalized minimal residual (GMRES) method [12], and Bi-CGSTAB method [13]. These methods computes the series of the vectors converging to the target solution. The direct methods may construct the inverse operator $Z^{-1}$ directly or decompose the system matrix to compute the factors to get the solution. The examples of the factoring methods are like the LU-decomposition, and the Cholesky factorization [14, 15]. The computational cost required to the iterative methods are less than that of the fast direct methods. Computing the matrix inverse and factoring the system matrix demand more complexity for the CPU performance and the memory usage compared to the iterative solvers. Despite the improved computational requirement, the iterative methods have the drawback compared to the fast direct solvers. The most significant problem is the sensitivity to the system condition. The number of the repeated converging loop increases when the ill-conditioning system is involved into the solution to degrade the solution time. The large condition numbers might be caused from the scattering problems near resonant frequencies, the geometry like the percolation object, the improperly modeled tips of the geometry, and the finite element discretization. For the iterative methods, the ill-
conditioning troubles can be resolved with the proper preconditioners for the systems, but the additional computation can be the bottleneck of the overall solution procedure. The fast direct method solution times don’t get slower with the conditioning, and this leads the direct solvers to be preferred in the software products. Besides the robustness problem, the additional computational labor for the multiple RHS with the fixed system matrix is less than the iterative counterparts. Once inverted or factored system matrix can be repeatedly used for the different excitations, while the iterative procedure must start over for each boundary conditions.

In spite of some advantages over iterative solvers, the direct methods still have the CPU and the memory cost problems. The computational complexities can be released by the structured system matrices. The structured matrix allows the fast algorithm to be exploited. The FEM system matrices are sparse and the efficient matrix-vector multiplying algorithm is developed. This routine can be used for the sparse system solution for the iterative methods, which uses the matrix-vector multiplying in the procedures. Similarly, the fast direct solution package like SuperLU [16] and UMFPACK [17] include the efficient routines for the Gaussian Elimination for the sparse matrix to factor the system into the LU decomposition. The traditional LU factorization costs computational labor of $O(N^3)$ for CPU and $O(N^2)$ for memory, and the sparse property reduces those asymptotic costs. The discretized integral equations, however, have an inherently dense system matrix, and the fully filled matrices can’t avoid the traditional computational complexities. Although the system restricts the discretized domain only along the boundaries while the FEM treats the whole space around the objects, the large scale dense matrices increase the computational costs too high to be feasible for either of the solution methods because the sparse solution methods can’t be applied to the dense system matrices.

With the smaller dimension and the well-conditioning system matrix, the integral equation system is attractive solution method despite the dense matrix pattern. To improve the efficiency of the dense matrix computation, the various techniques are developed for the sparse representation of the matrices including panel clustering algorithms [18-20], wavelet methods [21-28], interpolation [29] and fast multipole methods [30-36]. The panel clustering methods and the fast multipole methods sparsify
the integral equation system matrices with the degenerate approximated kernels. The panel clustering methods truncate the expansion from the Taylor representation, and FMM refers the separated sources and targets relations. Both of them are not for the general implementation due to the difficulties of the approximations of the analytic kernel functions. The interpolation methods use the numerical methods rather than the analytic expansions, while the computation labor costs too expensive. These methods are based on the low-rank property of the integral equations, and provide the improved matrix-vector multiplication. Implemented with the methods so far, the integral equation problems with the sparsified system matrices can be efficiently solved iteratively.

The dense matrices can be compressed with the algebraic method proposed by Canning [37-39], too. This method observed that the off-diagonal blocks of the system matrices have low rank property. The approximation of the system matrix is found based on the singular values of the sub matrix of the system matrix. This method allowed the inverse of the matrices, which is fundamental factor of the direct method. With the similar idea of the low-rank property, Börm and Hackbusch et al. [40-45] proposed the hierarchical matrix (H-Matrix), and Chandrasekharan et al. [46-55] developed the hierarchical semiseparable (HSS) matrix. These techniques are closely related to each other, and have the multi-level structure from the hierarchical partitioning of the geometric domain. H-Matrix exploits the low-rank property based on the panel clustering method [18], while HSS matrix is the special case of the fast multipole method representation [54]. The arithmetic versions of the low-rank data-sparse representations, allows the direct solution factors like the nested dissection and Schur complement method from the sparse structured factorization [48, 56, 57].

1.3. The fast direct solver using Local-Global Solution (LOGOS) modes
Adams et al. [58] developed the fast direct solution method relying on the local-global solution (LOGOS) modes. The system matrix in the sparse representation is factored based on the concept of “LOGOS modes.” The different types of LOGOS modes are classified based on the combination of the source fields and the target fields. If the source fields radiate onto only the region in which sources are located, the LOGOS modes are the non-radiating modes; LOGOS modes that radiate outside the source region
are so-called radiating LOGOS modes. The source and the target field regions are
decided based on the geometric information. For low-to-moderate frequency problems,
the non-radiating (aka localizing) LOGOS modes are used for the factorization of this
direct solver. This is because at such electrical sizes, most degrees of freedom do not
radiate over appreciable distances.

The LOGOS solution procedure is multi-level procedure, where the levels have the
hierarchical partitioning of the geometric domain. The lower (finer) levels are divided
into the smaller regions, and the regions in the higher (coarser) levels consists of the
collection of the region groups from the lower levels as their children groups. The non-
radiating modes of the source with respect to the specific regions are used for the
localizing basis computing, while the other modes are transferred up to the parent region
of the target for the next level modes decision. In other words, the radiating modes are
checked if they are still radiating outside of the expanded region. This multi-level process
splits the region into the narrow region and compute the modes in one level, and then the
remaining modes are merged into the larger (parent) region for factoring before passed up
for the even larger region. The support of the source field region can be defined in
various ways, and different types of non-radiating (aka, localizing) LOGOS factorization
algorithms have been developed based on how the source region is constructed at each
level of the multi-level partition of the geometry. The most basic distinction is non-
overlapping versus overlapping source functions. As the name suggests, at a given level
of the partition tree, non-overlapping modes associated with a given spatial region do not
overlap with modes in another region. In contrast, overlapping sources share their
physical regions of support somewhat. Additional details are outlined below, but an
essential result is that the overlapping sources provide both better asymptotic
performance and smaller initial constant of the performance time.

The localizing modes are approximately decided during the factorization based on
the desired factorization error (or tolerance). In the case of the non-overlapping source
regions, the localizing modes in a given spatial group do not interact with the other
localizing modes. This decouples any approximation errors, and permits a rigorous error
analysis of the associated LOGOS factorization algorithm [59]. Unfortunately, such non-
overlapping factorizations are not as computationally efficient as overlapping
factorizations. This means that the error control is lacking for the computationally efficient forms of LOGOS factorization. A primary contribution of this work is to provide such an error control algorithm.

In addition to this analytical issue, this dissertation also examines the practical issue of reducing the memory costs of the LOGOS factorization algorithm by using both in- and out-of-core computer memory storage.

1.4. The Error Control in the LOGOS factorization

1.4.1. Overview

As mentioned in the section 1.3, the LOGOS framework provides the multi-level direct solution method utilizing the approximated non-radiating modes. Through the levels, the more localizing modes found at fine levels of the tree, the smaller is dimension of the next level system matrix. For example, a factorization with a relaxed tolerance causes the fast asymptotical performance. If the tolerance is tightened, the system dimension for the next level increases; in some cases, this effect can cause the asymptotic costs of the algorithm to increase as the tolerance is tightened (this issue is also related to the system formulation).

To estimate the factorization error, it is necessary to begin by defining how the errors will be measured. The errors studied by Xu [59] and in this study measure the factorization error in terms of the relative RMS error in the matrix factorization, because it is the most directly measureable error. Others, such as the solution residual error, provide less direct way since they depend also on the system condition number.

Briefly, the RMS error and the residual errors can be formulated like:

\[
\xi_{\text{RMS}} = \frac{\| Z - \bar{Z} \|_F}{\| Z \|_F}
\]  

(1.3)

and,

\[
\xi_{\text{res}} = \frac{\| Z\bar{Z} - \bar{F} \|_F}{\| \bar{F} \|_F} \leq \| Z\bar{Z}^{-1} - I \|_F
\]

(1.4)

\[
= \| Z\bar{Z}^{-1} - \bar{Z}\bar{Z}^{-1} \|_F \leq \xi_{\text{RMS}} \cdot \text{cond}(\bar{Z}).
\]
Equations (1.3) and (1.4) are the system RMS error and the solution residual errors respectively. The matrix \( \hat{Z} \) indicates the reconstructed system matrix obtained by multiplying all of the LOGOS factorization factors together. As seen, in the case of residual error, the result depends on the condition number of the system matrix. For such reasons, in this study, the rebuilt system RMS error is the main target of the error analysis and error control algorithm. The residual error will be seen as another result from the error control. (The impact of the matrix condition number on error control is a critical factor from a practical standpoint. Fortunately, such issues can often, though not always, be segregated and addressed separately. In this work, the focus is on error control for overlapping LOGOS factorizations of any system matrix, regardless of condition number. The impact of condition number is considered only as a by-product in terms of the resulting residual and solution errors, and the computational complexity of the resulting LOGOS factorization.)

The non-overlapped (NL) LOGOS factorization error analysis and control were studied by Xu [59], where the decoupled and orthogonal structure of the basis enabled the isolation of factorization approximations (i) among different spatial regions at a given level of the multilevel partition, and (ii) between different levels of the tree. At the same level, the error relating to the localizing modes are removed by the decoupled spatial structure. Through the different levels, the error can propagate only through the non-orthogonal relation between the localizing basis space \( (\Omega^{(L)}) \) and the non-localizing basis space \( (\Omega^{(N)}) \). Because such coupling is zero, the NL-LOGOS does not have such multilevel error propagation, and error can be controlled without considering the rebuilding procedure. The more efficient, overlapped (OL) LOGOS has neither advantage, and requires a much more careful error analysis. The purpose of this study is to show how to analyze and control such error propagation through the general factorization structure.

1.4.2. Basic Idea

Out of the two main issues to derive the error control formulation (spatial coupling, and multilevel coupling), the non-orthogonal property between the subspaces \( \Omega^{(L)} \) and \( \Omega^{(N)} \) causes the more serious problem, due to the error interaction among the levels.
However, the coupling is not avoided due to the fundamental structural property between the spaces. It is shown that the error interaction between the levels can be resolved by introducing a factor and using the triangle inequality property of the matrix. The formulation leaves the error term isolated at the next level to compute. Without the separated term, the other terms can be computed at the working level.

The issue from the coupling among the $\Omega^{(L)}$ space introduces the interaction among the errors at the same working level, while the factorization performed for each group are independent from the others. Due to this reason the error interaction should be checked after the factorizations are done all over the coupled groups. This is a non-linear error control problem, and it is solved using an iterative bisection process. If a given truncation point fails to satisfy the tolerance test, then the number of the localizing modes is reduced, and the error control condition is tested again. This procedure repeats until the tolerance is satisfied, and it is guaranteed to always converge.

For the next level, the isolated error term from the separation between $\Omega^{(L)}$ and $\Omega^{(N)}$ is now set as the next tolerance based on the working level factorization result. At the following level, the system matrix is obtained from the non-localized space of the working level. A relation between next level system matrix and the non-localized space on the working level are shown to determine the appropriate localization tolerance to be used at the next level.

1.5. The Out Of Core (OOC) Operation

1.5.1. Overview

For iteration methods, performance relies on the matrix-vector multiply speed during the iterations, and the cost to save the matrix; there are no other significant memory costs. The OL-LOGOS factorization, like other direct methods, demands the data storage space for the factorization results (including the localizing modes and the projection blocks). It is more handy only to use the system memory space for the data, but when the problem size increases, the solution method may more space than the built in system memory. In this case it is not the good idea whenever it happens to expand the system memory, or to give up the solution. So the external storage devices can be the answer to this situation. In the computer memory hierarchy, the normally used system memory is the just below the
cache spaces like L1 and L2 caches. The hierarchy generally means that level goes lower for the larger spaces and the slower access time. To use the larger space like the hard disk, proper strategies should be developed to minimize the performance penalty associated with the larger access latencies. The goal of this study is to build an algorithm for efficient LOGOS factorization in the presence of higher latency external storage.

1.5.2. Basic Idea

In this study, the magnetic storage device is considered as the data saving unit. The magnetic devices have the mechanical parts, and they cannot overcome the speed deficiency compared to the electric parts. The access time of this device is computed from the sum of the arm locating time, disk spin time and the data transfer time from the disk to the main memory unit [60]. To reduce the latency at the disk unit, it is concluded that the number of disk accesses should be minimized as much as reasonably possible.

The algorithm designed below is based on both a device-dependent data size decision and a modified scheduling of the LOGOS groups in the factorization algorithm. In this dissertation, the block matrices group, which is loaded from the disk, is called a “data page.” It is known that reading one large data block from the disk is faster than many small data blocks that combine to be the same total data size (due to the large latency). Therefore, the data page is set to cover certain number of groups based on the geometric property of the domain to be more efficient for both factorization performance and memory savings.

The algorithm scheduling is based on the LOGOS factorization process. Now that the factorization performs over the groups at the same level independently, the order of the groups for the factorization can take any order. So the scheduling means the reordering of the groups based on the same geometric property as the (designed) data pages, rather than the somewhat arbitrary order of the group indices. Such coherence between the data page and the order minimizes the external device access time by improving the page hit ratio. A complication with the data page decision lies in the fact that different kinds of group orderings optimize the factorization and the post-factorization operations. This is addressed by setting the overlapped page is to optimize both constraints as much as possible. To implement this algorithm, the overlapped groups should be considered to be updated on all overlapped pages.
1.6. Objective and scope

The purpose of this work is to develop an error controllable and memory efficient LOGOS based fast direct solution method. The non-orthogonal and coupled basis space is considered as the general LOGOS error analysis. The formulation is to be derived as outlined at 1.4.2. The magnetic disk device is assumed as the massive storage for the out-of-core factorization, and the background of the decisions of the block size and the data page range are shown.

Chapter 2 is about the LOGOS concept and the algorithm of the factorization. The LOGOS is the main framework of the solution method in this research, and this review is important as the background of the error control and the OOC operation. The tree structure is described to explain the multi-level procedure and the geometric domain. As the core component, numerical formulation of determining the localizing modes used in the factorization is introduced in the chapter. The several ways to decide the range of the source groups is discussed to show the different basis space structures (non-overlapped, overlapped neighbors, children’s neighbors, and shifted grid).

Chapter 3 explains the error analysis and control formulation in detail. The error terms in the reconstruction procedure are separated to allow independent error control on $\Omega^{(L)}$ and $\Omega^{(N)}$ spaces. A trial-and-error algorithm is provided to resolve the error interacting problem on the coupled localizing/non-localizing mode spaces. This development introduces potentially computationally expensive operations to the basic LOGOS procedure (like local matrix inverses); this chapter describes how to reduce the CPU costs of such operations in order to minimize the impact on the overall factorization time. The algorithm is tested for several different problems, and it is shown that the algorithm provides reliable error control, including in cases for which the previously developed OL-LOGOS factorization algorithms fail to provide error control. The role of the system condition number on the overall performance is also discussed.

Chapter 4 discusses the OOC algorithm, which focuses on the geometry based LOGOS scheduling and the massive storage operation. For the data page decision, the detailed description about the external storage performance for the large data is provided. The scheduling of the algorithm is treated by considering the relationship between the
data page and the geometry; this includes situations in which groups belonging to the multiple pages arise and the associated algorithmic modifications for handling this case. The OOC performance data are provided for the LOGOS performance and the peak memory change, compared with the in-core data. The different kinds of the data page decision strategies are shown to show the performance differences for the choices.

Chapter 5 concludes the work and suggests related areas for future work.
Chapter 2. THE LOGOS FACTORIZATION

Time harmonic surface and volume integral equation formulations of electromagnetic interaction problems are converted to matrix equations using appropriate subspace projection methods. The LOGOS framework provides a conceptual framework for solving such problems. In the following we consider LOGOS-based factorization methods for the linear systems obtained by discretizing surface and volume integral equations. An essential element of the LOGOS factorization is the use of a multi-level tree (such as quad- and oct-trees) to partition the geometric elements.

2.1. The Geometry

The domain of the linear system is divided in the form of the hierarchical tree structure like the quad-tree and the oct-tree. Figure 1 demonstrates the square plate with the regions based on the 3 level quad-tree. The oct-tree has the 3D property while the vertical relation of the region groups is identical to the quad-tree. The multi-level solution process is through the hierarchy of the tree from the lowest level to the highest level. The lowest level consists of the groups with less number of the discretized fields. As the level goes up, the groups contain other groups from the lower level as their children. Because the lower levels have more divided groups than the higher levels, the lower levels are called the finer levels, whereas the higher levels are referred to as being the coarser levels. At each level the degrees of freedom (or unknowns) are reorganized; localizing LOGOS modes are determined for each group, and the remaining, non-localizing modes are passed to the higher (or parent) level. This continues to the coarsest (root) level, where a traditional direct method (such as dense LU) is performed as the solution method.

The geometric meaning of the localizing LOGOS modes is that the energy of the field generated by a group of such sources radiates, to within some tolerance, only to the same regions of space that contains the sources. The source region may be a single group (non-overlapping case) or a small group of groups (overlapping case) at a given level. The modes from the former source, where typically the same group as the field group, is the NL-LOGOS modes, while the latter source induces the OL-LOGOS modes. Actually
the factorization process is exactly same as each other except that the source support in the two cases differ. However, the spatial segregation of the NL-LOGOS factorization makes its numerical analysis much easier to perform. Rigorous analysis of the OL-LOGOS case has not previously been accomplished.

2.2. Determination of Localizing modes

Let the domain of the simulation be denoted \( S \), which consists of the non-overlapped \( N \) subdomains,

\[
S = S_1 + S_2 + \cdots + S_{N-1} + S_N
\]  

(2.1)

The regional division allows the associated system equation (1.2) to have the form:

\[
\begin{bmatrix}
Z_{11} & Z_{12} & \cdots & Z_{1(N-1)} & Z_{1N} \\
Z_{21} & Z_{22} & \cdots & Z_{2(N-1)} & Z_{2N} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
Z_{(N-1)1} & Z_{(N-1)2} & \cdots & Z_{(N-1)(N-1)} & Z_{(N-1)N} \\
Z_{N1} & Z_{N2} & \cdots & Z_{N(N-1)} & Z_{NN}
\end{bmatrix}
\begin{bmatrix}
x_{1,m} \\
x_{2,m} \\
\vdots \\
x_{(N-1),m} \\
x_{N,m}
\end{bmatrix}
=
\begin{bmatrix}
F_{1,m} \\
F_{2,m} \\
\vdots \\
F_{(N-1),m} \\
F_{N,m}
\end{bmatrix}
\]  

(2.2)

where the subscript \( m \) on \( x_m \) and \( F_m \) means the index of the LOGOS modes. A single LOGOS mode is defined by an excitation and solution pairing \( (F_m, x_m) \). The integers \( (n=1\ldots N) \) on the sub-vector \( x_{n,m} \) indicate the associated source subdomain region of the decomposed system matrix like the pairing of \( (Z_{t,n}, x_{n,m}) \). On top of the non-overlapped subdomains, the overlapped ranges are defined:

\[
OL_1 = S_1 + S_2 \\
OL_2 = S_2 + S_3 \\
\vdots \\
OL_{N-1} = S_{N-1} + S_N \\
OL_N = S_{N-1} + S_N
\]  

(2.3)

Each overlapped range (or group) has two subdomains in it, and one of the regions is shared with other overlapped range. The last overlapping is defined to share the same domains with \( N-1 \) range. (The actual regions of support for overlapping modes will be defined below using the oct-tree; the definitions (2.3) are used here to illustrate the mode computation procedure.)
Consider the determination of LOGOS modes \((F_m, x_m)\) which only have the nonzero support in \(OL_1\) (i.e. \(x_{1,m}, x_{2,m} \neq 0\) and for others \(x_n = 0\)). The local condition associated with these modes is:

\[
\begin{bmatrix}
Z_{11} & Z_{12}
\end{bmatrix}
\begin{bmatrix}
x_{1,m}
ox_{2,m}
\end{bmatrix}
=
\begin{bmatrix}
F_{1,m}
\end{bmatrix}
\]  
(2.4)

The global condition is:

\[
\begin{bmatrix}
Z_{21} & Z_{22} \\
\vdots & \vdots \\
Z_{N1} & Z_{N2}
\end{bmatrix}
\begin{bmatrix}
x_{1,m}
ox_{2,m}
\end{bmatrix}
=
\begin{bmatrix}
F_{2,m} \\
\vdots \\
F_{N,m}
\end{bmatrix}
\]  
(2.5)

Combining (2.4) and (2.5) leads to the local-global condition:

\[
\begin{bmatrix}
Z_{21} & Z_{22} \\
\vdots & \vdots \\
Z_{N1} & Z_{N2}
\end{bmatrix}
\begin{bmatrix}
Z_{11} & Z_{12}
\end{bmatrix}
\begin{bmatrix}
x_{1,m}
ox_{2,m}
\end{bmatrix}
=
\begin{bmatrix}
F_{2,m} \\
\vdots \\
F_{N,m}
\end{bmatrix}
\]  
(2.6)

satisfied by all LOGOS modes. Using (2.6), the LOGOS modes confined to overlapping region 1 up to order- \(\varepsilon\) are computed.

The LOGOS modes in subdomain 1 are computed by setting the right hand side of (2.6) to zero to order \(\varepsilon\),

\[
\begin{bmatrix}
Z_{21} & Z_{22} \\
\vdots & \vdots \\
Z_{N1} & Z_{N2}
\end{bmatrix}
\begin{bmatrix}
Z_{11} & Z_{12}
\end{bmatrix}
\begin{bmatrix}
x_{1,m}
ox_{2,m}
\end{bmatrix}
=
\begin{bmatrix}
0 \\
\vdots \\
0
\end{bmatrix}
\]  
(2.7)

The Logos modes are designated as localizing LOGOS modes because the fields radiating from subdomain 1 to other subdomains are \(\varepsilon\)-order zero.

To find the localizing modes satisfying (2.7), the procedure follows [61]. First, perform a QR decomposition on the matrix block associated with the sources in \(OL_1\),

\[
Z_{OL1} = \begin{bmatrix}
Z_{1,OL1} \\
\vdots \\
Z_{N,OL1}
\end{bmatrix}
= \begin{bmatrix}
Q_{1,OL1} \\
\vdots \\
Q_{N,OL1}
\end{bmatrix}
R_{OL1}
\]  
(2.8)

where \(Q_{1,OL1} \ldots Q_{N,OL1}\) has the same dimension as \(Z_{1,OL1} \ldots Z_{N,OL1}\), and \(R_{OL1}\) is a square
upper triangular matrix. Taking out the submatrix $Q_{1,OL1}$ and performing the singular value decomposition (SVD) lead to:

$$Q_{1,OL1} = u_is_{1,OL1}v_{1,OL1}^H$$

(2.9)

In (2.9), the singular values matrix, $s_i$ plays the role of amplifier of the sources to the radiating field. The maximum singular value of (2.9) should be less or equal to 1, because $Q_{1,OL1}$ is from the orthonormal matrix. Because $s_i$ has the singular values in descending order, the number of the columns of $v_{1,OL1}$, which is the source side unitary matrix, is considered to be the number of the approximate non-radiating field. To get the exactly non-radiating modes, one takes the vectors corresponding to the singular values that are exactly unity in $v_{1,OL1}^H$.

For order-$\varepsilon$ non-radiating modes, the allowable approximation of the non-radiating field is needed in terms of the impact on the targeted error control indicated in (1.3). The tolerance is based on the error estimation of the approximate factorization, and depending on the error estimation and the LOGOS mode, the strategy of taking the localizing number from $s_i$ varies. A detailed discussion of this issue is the primary focus of this research, and is treated in the Chapter 3. Let $N^L$ denote the number of non-radiating source modes. This number divides $v_{1,OL1}$ into blocks $v_{1,OL1}^{(L)}$ and $v_{1,OL1}^{(N)}$, that is $v_{1,OL1} = \begin{bmatrix} v_{1,OL1}^{(L)} & v_{1,OL1}^{(N)} \end{bmatrix}$. $u_1$ and $s_1$ are also be rewritten in the local and non-local combination form like $u_1 = \begin{bmatrix} u_1^{(L)} & u_1^{(N)} \end{bmatrix}$ and $s_1 = \begin{bmatrix} s_1^{(L)} & s_1^{(N)} \end{bmatrix}$. So (2.8) can be represented like:

$$Z_{OL1} = \begin{bmatrix} u_is_1 \\ Q_{2,OL1}v_{OL1}^{(L)} \\ \vdots \\ Q_{N,OL1}v_{OL1}^{(L)} \\ \end{bmatrix} v_{1,OL1}^HR_{OL1} = \begin{bmatrix} u_is_1^{(L)} \\ Q_{2,OL1}v_{1,OL1}^{(L)} \\ \vdots \\ Q_{N,OL1}v_{1,OL1}^{(L)} \\ \end{bmatrix} \begin{bmatrix} u_is_1^{(N)} \\ Q_{2,OL1}v_{1,OL1}^{(N)} \\ \vdots \\ Q_{N,OL1}v_{1,OL1}^{(N)} \\ \end{bmatrix} v_{1,OL1}^HR_{OL1}$$

(2.10)

Therefore,
\[
Z_{OL1}R_{OL1}^{-1} \begin{bmatrix} v_{1,OL1}^{(L)} & v_{1,OL1}^{(N)} \end{bmatrix} = \begin{bmatrix} u_1s_1^{(L)} & u_1s_1^{(N)} \\
Q_{2,OL1}v_{1,OL1}^{(L)} & Q_{2,OL1}v_{1,OL1}^{(N)} \\
\vdots & \vdots \\
Q_{N,OL1}v_{1,OL1}^{(L)} & Q_{N,OL1}v_{1,OL1}^{(N)} \end{bmatrix}
\] (2.11)

Now that \(Q_{1..N,OL1}\) is orthonormal and \(v_{1,OL1}\) is unitary,
\[
\begin{bmatrix} u_1s_1 \\
Q_{2,OL1}v_{1,OL1}^{(L)} \\
\vdots \\
Q_{N,OL1}v_{1,OL1}^{(L)} \end{bmatrix} = \begin{bmatrix} Q_{1,OL1} \\
Q_{2,OL1} \\
\vdots \\
Q_{N,OL1} \end{bmatrix} v_{1,OL1}
\] (2.12)

is also orthonormal due to the unitary property of \(v_{1,OL1}\). In the right hand side matrix, \(Q_{2..N,OL1}v_{1,OL1}^{(L)}\) are all zeros to the order-\(\epsilon\) because \(Q_{OL1}\) is orthonormal matrix, and then \(v_{1,OL1}^{(L)}\) from (overlapping) subdomain 1 must be the null space of the \(v_{2..N,OL1}\), which are the result of the SVD for other subdomains. Therefore,
\[
Q_{2..N,OL1}v_{1,OL1}^{(L)} = 0
\] (2.13)

while,
\[
u_1s_1^{(L)} = \tilde{0}
\] (2.14)

where \(\tilde{0}\) the approximate zero matrix to order-\(\epsilon\), where the non-zero values from the localization. As a result,
\[
Z_{OL1}R_{OL1}^{-1} \begin{bmatrix} v_{1,OL1}^{(L)} & v_{1,OL1}^{(N)} \end{bmatrix} = Z_{OL1} \begin{bmatrix} \Lambda_{1,OL1}^{(L)} & R_{OL1}^{-1}v_{1,OL1}^{(N)} \end{bmatrix} = \begin{bmatrix} u_1s_1^{(L)} & u_1s_1^{(N)} \\
\tilde{0} & Q_{2,OL1}v_{1,OL1}^{(N)} \\
\vdots & \vdots \\
\tilde{0} & Q_{N,OL1}v_{1,OL1}^{(N)} \end{bmatrix}
\] (2.15)

where \(\Lambda_{1,OL1}^{(L)} = R_{OL1}^{-1}v_{1,OL1}^{(L)}\) are the (overlapped) localizing modes that radiate into no other subdomain area than region 1, approximately. Out of the overlapped range 1 of \(\Lambda_{1}^{(L)}\), the orthonormal complements of the group 1 region of \(\Lambda_{1}^{(L)}\) are chosen for the non-localizing basis, \(\Lambda_{1}^{(N)}\), for the subdomain region 1.

The localizing modes for other regions can be found in the same manner as outlined above for overlapping region 1. The localizing modes for the other groups can be found
by replacing 1 in region 1 and OL1 with $n:1\ldots N$. When the localizing process is over at the working level $L$, the basis space is defined like:

$$
\begin{bmatrix}
\Lambda^{(L)}_{OL1(1)} & 0 & \ldots & 0 & 0 \\
\Lambda^{(L)}_{OL1(2)} & \Lambda^{(L)}_{OL2(2)} & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & \Lambda^{(L)}_{OL(N-1)(N-1)} & \Lambda^{(L)}_{OL(N-1)(N-1)} \\
0 & 0 & \ldots & \Lambda^{(L)}_{OLN(N-1)} & \Lambda^{(L)}_{OLN(N-1)}
\end{bmatrix}
\begin{bmatrix}
\Lambda^{(N)}_{1(1)} & 0 & \ldots & 0 & 0 \\
0 & \Lambda^{(N)}_{1(2)} & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & \Lambda^{(N)}_{N-1(N-1)} & 0 \\
0 & 0 & \ldots & 0 & \Lambda^{(N)}_{N(N)}
\end{bmatrix}
$$

(2.16)

where, the number in the last parentheses are the group numbers of the overlapped ranges to specify the source region. The basis matrix of (2.16) is divided by the bar to help distinguish the localizing basis and the non-localizing basis. To compute the localized modes from the system matrix:

$$
\begin{bmatrix}
Z_{11} & Z_{12} & \ldots & Z_{1(N-1)} & Z_{1N} \\
Z_{21} & Z_{22} & \ldots & Z_{2(N-1)} & Z_{2N} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
Z_{(N-1)1} & Z_{(N-1)2} & \ldots & Z_{(N-1)(N-1)} & Z_{(N-1)N} \\
Z_{N1} & Z_{N2} & \ldots & Z_{N(N-1)} & Z_{NN}
\end{bmatrix}
$$

$$
\times
\begin{bmatrix}
\Lambda^{(L)}_{OL1(1)} & 0 & \ldots & 0 & 0 \\
\Lambda^{(L)}_{OL1(2)} & \Lambda^{(L)}_{OL2(2)} & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & \Lambda^{(L)}_{OL(N-1)(N-1)} & \Lambda^{(L)}_{OL(N-1)(N-1)} \\
0 & 0 & \ldots & \Lambda^{(L)}_{OLN(N-1)} & \Lambda^{(L)}_{OLN(N-1)}
\end{bmatrix}
\begin{bmatrix}
\Lambda^{(N)}_{1(1)} & 0 & \ldots & 0 & 0 \\
0 & \Lambda^{(N)}_{1(2)} & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & \Lambda^{(N)}_{N-1(N-1)} & 0 \\
0 & 0 & \ldots & 0 & \Lambda^{(N)}_{N(N)}
\end{bmatrix}
$$

$$=\begin{bmatrix}
\tilde{u}_1s_1^{(L)} & \tilde{\tilde{0}} & \ldots & \tilde{\tilde{0}} & \tilde{\tilde{0}} \\
\tilde{\tilde{0}} & \tilde{u}_2s_2^{(L)} & \ldots & \tilde{\tilde{0}} & \tilde{\tilde{0}} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\tilde{\tilde{0}} & \tilde{\tilde{0}} & \ldots & \tilde{u}_{N-1}s_{N-1}^{(L)} & \tilde{\tilde{0}} \\
\tilde{\tilde{0}} & \tilde{\tilde{0}} & \ldots & \tilde{\tilde{0}} & \tilde{u}_N s_N^{(L)}
\end{bmatrix}
\begin{bmatrix}
Z_{11}^{(N)} & Z_{12}^{(N)} & \ldots & Z_{1(N-1)}^{(N)} & Z_{1N}^{(N)} \\
Z_{21}^{(N)} & Z_{22}^{(N)} & \ldots & Z_{2(N-1)}^{(N)} & Z_{2N}^{(N)} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
Z_{(N-1)1}^{(N)} & Z_{(N-1)2}^{(N)} & \ldots & Z_{(N-1)(N-1)}^{(N)} & Z_{(N-1)N}^{(N)} \\
Z_{N1}^{(N)} & Z_{N2}^{(N)} & \ldots & Z_{N(N-1)}^{(N)} & Z_{NN}^{(N)}
\end{bmatrix}
$$

(2.17)

where, the non-localized modes are located on the right side of the vertical divider of the last matrix. The localized modes are computed in the same way as the case of $Z_{11}^{(L)}$ presented like:

$$Z_{11}^{(L)}=[Z_{11} \ Z_{12}] \Lambda^{(L)}_{OL1} = Z_{11} \cdot \Lambda^{(L)}_{OL1(1)} + Z_{12} \cdot \Lambda^{(L)}_{OL1(2)}$$

(2.18)
In (2.18) the columns of the system matrix elements means the overlapping range, and the range of system matrix is multiplied by the localizing basis through all rows to show the approximate zero matrices for the non-localized region. Because the sources were defined on overlapping spatial regions, the localizing basis matrix also has an overlapped structure. The non-localized modes are built by simply multiplying the system matrix with the non-localizing basis in the form of block diagonal matrix.

Now that the basis function set up is completed for the current (or, working) level, another component of the LOGOS factorization, the projection matrix, needs to be computed from the result of (2.17). For each group, the local projection matrices are

\[ p_{n1..N} = \begin{bmatrix} u^{(L)}_{n1..N} & u^{(N)}_{n1..N} \end{bmatrix} \] (2.19)

The local dimensions on the right hand side of (2.19) are identical to those for the \( \Lambda \) matrix above. The projection matrices are block diagonal (as was \( \Lambda^{(N)} \)):

\[
P = \begin{bmatrix} p^{(L)} & p^{(N)} \end{bmatrix} = \begin{bmatrix} p^{(L)}_1 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & p^{(L)}_2 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \ldots & p^{(L)}_{N-1} & 0 & 0 & 0 & \ldots & p^{(N)}_{N-1} \\
0 & 0 & \ldots & 0 & 0 & p^{(L)}_N & 0 & 0 & \ldots & p^{(N)}_N \end{bmatrix} \quad (2.20)
\]

The completed LOGOS factorization is then computed like:

\[
P^H Z \Lambda = \begin{bmatrix} (P^{(L)})^H & (P^{(N)})^H \end{bmatrix} \begin{bmatrix} Z_{11} & Z_{12} & \ldots & Z_{1(N-1)} & Z_{1N} \\
Z_{21} & Z_{22} & \ldots & Z_{2(N-1)} & Z_{2N} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
Z_{(N-1)1} & Z_{(N-1)2} & \ldots & Z_{(N-1)(N-1)} & Z_{(N-1)N} \\
Z_{N1} & Z_{N2} & \ldots & Z_{N(N-1)} & Z_{NN} \end{bmatrix} \begin{bmatrix} \Lambda^{(L)} \\
\Lambda^{(N)} \end{bmatrix} \] (2.21)

The identity matrix and the zero matrix under the tilde means these are approximations due to the error control of the localization.

Up to now, the notation hasn’t indicated the level of computation. Let the preceding matrices correspond to level \( L \) of the tree. Then, for a multilevel tree, (2.21) can be written using the level index \( L \),

\[
P^H Z \Lambda = \begin{bmatrix} \tilde{I} & Z^{(LN)} \\
\tilde{0} & Z^{(NN)} \end{bmatrix} \]
\[
\hat{Z} = P_L^H Z_L \Lambda_L = \begin{bmatrix}
    \tilde{I} & Z_L^{(LN)} \\
    0 & Z_L^{(NN)}
\end{bmatrix}
\]

(22)

\(Z_L^{(NN)}\) in (22) plays the role as the system matrix at the next (coarser) level, level-\((L-1)\). Before proceeding to the coarser level, the elements of \(Z_L^{(NN)}\) are first combined based on the parent-child relations provided by the tree structure. Once this “collapsing” of the sibling groups is performed, the preceding localization and projection procedure is repeated at level-\((L-1)\) in the same manner as was done at level-\(L\). This continues until the coarsest level of the tree is reached.

### 2.3. The solution procedure

From the factorization results \((P \text{ and } \Lambda)\), the system matrix can be inverted and finally the numerical linear system is solved from the inverse. In this process, the \(\tilde{I}\) and \(\tilde{0}\) of (22) are all considered as normal identity and zero matrices,

\[
\hat{Z} = P_L^H Z_L \Lambda_L = \begin{bmatrix}
    I & Z_L^{(LN)} \\
    0 & Z_L^{(NN)}
\end{bmatrix}
\]

This approximation causes error in the factorization process, but this is a necessary step to maintain the data sparse nature of the factorization; an important goal is to control the errors that such truncations introduce.

To get \((Z_L)^{-1}\), \(P_L^H\) and \(\Lambda_L\) are transferred to the right hand side.

\[
(Z_L)^{-1} = \left(P_L \begin{bmatrix}
    I & Z_L^{(LN)} \\
    0 & Z_L^{(NN)}
\end{bmatrix} \Lambda_L^{-1}\right)^{-1}
\]

(24)

\[
= \Lambda_L \cdot \begin{bmatrix}
    I & Z_L^{(LN)} \\
    0 & Z_L^{(NN)}
\end{bmatrix}^{-1} \cdot P_L^H
\]

As seen in (24), \(P_L^H\) and \(\Lambda_L\) are transferred and inverted, so they keep the form of the factorization without any computation on themselves. To the inverse of the factorization result, applying Schur complement [15]:

\[
\begin{bmatrix}
    A & B \\
    C & D
\end{bmatrix}^{-1} = \begin{bmatrix}
    (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\
    -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1}
\end{bmatrix}
\]

(25)
to (2.23) by replacing $A = I$ and $C = 0$ then \((A - BD^{-1}C)^{-1} = I\) to make the formulation for the inverse of the system matrix:

$$
\begin{bmatrix}
I & Z_L^{(LN)} \\
0 & Z_L^{(NN)}
\end{bmatrix}^{-1} = 
\begin{bmatrix}
I & -Z_L^{(LN)} & (Z_L^{(NN)})^{-1}
0 & (Z_L^{(NN)})^{-1}
\end{bmatrix}
$$

(2.26)

The right hand side of (2.26) can be separated like:

$$
\begin{bmatrix}
I & -Z_L^{(LN)} & (Z_L^{(NN)})^{-1}
0 & (Z_L^{(NN)})^{-1}
\end{bmatrix} =
\begin{bmatrix}
I & 0
0 & I
\end{bmatrix}
\begin{bmatrix}
I & 0
0 & (Z_L^{(NN)})^{-1}
\end{bmatrix}
$$

(2.27)

By combining (2.24) and (2.27), the inverse of the system matrix at the level-$L$ is:

$$
(Z_L^{-1}) = \Lambda_L \cdot 
\begin{bmatrix}
I & -Z_L^{(LN)} \\
0 & I
\end{bmatrix} \cdot 
\begin{bmatrix}
I & 0
0 & (Z_L^{(NN)})^{-1}
\end{bmatrix} \cdot P_L^H
$$

(2.28)

In (2.28), the inverse of the current level, \((Z_L^{-1})\) formulation contains the higher level inverse, \((Z_{L-1}^{-1})\). From the multi-level property of the LOGOS factorization, the coarser level inverses cascade from the stop level, where the factorization stops (usually the coarsest level, level-1) to the finest level of the tree.

Finally, the solution form of the linear system is:

$$
\bar{x} = (Z_L^{-1}) \cdot \bar{F}
$$

$$
= \Lambda_L \cdot 
\begin{bmatrix}
I & -Z_L^{(LN)} \\
0 & I
\end{bmatrix} \cdot 
\begin{bmatrix}
I & 0
0 & (Z_L^{(NN)})^{-1}
\end{bmatrix} \cdot P_L^H \cdot \bar{F}
$$

(2.29)

As mentioned, the formulation uses Schur complement and takes advantage of the simple form by assuming the approximated matrix blocks to order-$\varepsilon$ in (2.22) are exactly the identity and the zero matrices.
2.4. The Source Range in the LOGOS Factorization

In the LOGOS procedure, the factorization determines the number of the localizing modes, which localize the scattered field to a target range from the source group range. Because the LOGOS factorization is a multi-level method, the factorization results at a given level influence the performance at subsequent levels; when few modes are found at one level, more work is done at the next level. Conversely, when the dimension of the system matrix is smaller at the next level, less work is required. The dimensions of the matrix at the next level is determined from the number of non-localized modes at the working level. This suggests that the performance will be optimized when the number of localizing modes is maximized. The number of modes that can be found is influenced by many factors, including the shape of the underlying geometry, the material properties, the matrix condition number, etc. These issues are beyond the control of the factorization algorithm. However, one item within the algorithm’s control is the strategy used to define the spatial support of the localizing modes. (The size of the field localization area is also controllable, but modifying that would change the algebraic structure indicated in (2.29). Changing the support of the source functions, Λ, does not change that structure. As discussed below, it does complicate the error control, however.)

As discussed above, there are two basic categories for localizing LOGOS factorizations, non-overlapping and overlapping, where the “overlap” refers to whether the source functions are spatially segregated or not. In the case of non-overlapping (NL) LOGOS modes, the source range is restricted to the field group itself. Due to the small dimension of the source support for NL-LOGOS modes, the QR decomposition time is less than the other, overlapped localizing (OL) LOGOS modes case. However, the cost of faster QR decomposition times in the individual mode computations is passing more non-localizing modes to coarser levels of the tree. This causes the more dimension for the next level, and accumulatively though the levels, and the overall asymptotic efficiency of the factorization degrades for large problems. On the other hand, the use of OL-LOGOS modes requires more cost for the QR decomposition time for the lower factorization levels. However, in many cases the projected system matrix at coarser levels of the tree is smaller than is obtained using NL-LOGOS modes, yielding improved asymptotic performance with the OL-LOGOS algorithm.
There are several strategies to choose the support of the overlapping source regions in the OL-LOGOS factorizations. In the following, the properties of the NL-LOGOS and the OL-LOGOS algorithms are reviewed. As discussed below, a specific variant of the OL-LOGOS factorization, referred to as the Shifted Grid (SG) LOGOS algorithm, provides the most efficient LOGOS factorization method. Fortunately, this form of the factorization is also amenable to rigorous error control in ways that are not possible with other OL-LOGOS variants.
2.4.1. NL-LOGOS

The NL-LOGOS has only one source group to radiate exclusively to the group itself. As an example, consider group 28 as indicated in Figure 2. An NL-LOGOS modes for this group consists of sources in group 28 that radiate fields only to group 28 (to within a given tolerance); the sources that produce such localized fields are call the NL-LOGOS source modes. This process is applied to all the groups at the working level of the tree. The most expensive component of the factorization, the QR decomposition, which has the complexity of $\frac{2}{3}n^2(3m-n)$ [15], has a size corresponding to the number of degrees of freedom in group 28. Because no other groups are included, this cost is minimized at the current level by not incorporating other source groups. However, the localizing modes found with such an NL-LOGOS algorithm are restricted to modes from the field group itself, which results in fewer modes found (compared to OL-LOGOS algorithm below), causing more work to be passed up to coarser tree levels. Eventually, the number of DOF in each group will become large so that the QR decomposition work will become too large for an efficient asymptotic factorization. In general, the computational complexity of the NL-LOGOS factorization can be as large as $O(n^2)$.

The NL-LOGOS factorization results in a desirable, block sparse matrix structure for all elements of the factorization. As seen in Figure 3, the basis space, $\Lambda$, is block diagonal. Furthermore, the localizing and non-localizing matrix blocks for a given group (e.g., group 28 above) are orthogonal; consequently, $\Lambda_L^{(l)}$ is orthogonal to $\Lambda_L^{(n)}$. This property accommodates the error analysis and control of the LOGOS factorization. This is because the inversion of $\Lambda$ causes no coupling between the localizing and non-localizing subspaces. Not only does this restrict approximation errors to the localizing subspace, but it also prevents the propagation of factorization errors between the levels.

2.4.2. OL-LOGOS

One version of the OL-LOGOS factorization consists in defining the source region as consisting of the neighbor groups of the targeted field group (the localizing region); this is depicted in Figure 4 (OL-Range A and OL-Range B). In this way, the localizing modes are found not only from the field group itself, but also from the neighbor groups,
which are collapsed at the following levels. This source range has the asymptotic advantage of providing \( O(n \log n) \) asymptotic complexity in some cases, which is much better than the asymptotic performance of the NL-LOGOS method of \( \left( O\left(n^2\right)\right) \) [62]. However, the load to the QR decomposition is quite high. This can be addressed by redefining the source region to comprise only the nearly located children groups. The resulting algorithm is still of \( O(n \log n) \), and the costs of the QR decompositions are much lower [62]. The organization of the so-called neighbors-children OL-LOGOS approach is illustrated in the Figure 5.

Although the neighbors-children method improves the performance of the OL-LOGOS algorithm, it has still problems with the basis function structure. Figure 6 shows the overlapped \( \Lambda^L \) structure. The inverse of the basis matrix leads to coupling between all of the basis functions. In terms of the error analysis, this means that the error control cannot be performed locally, but must be done on a global basis for such OL-LOGOS factorizations. Besides the interaction between the groups on the \( \Lambda^L \) space, \( \Lambda^L \) and \( \Lambda^{(N)} \) matrices also have a non-orthogonal relation. This non-orthogonal relation further complicates the error analysis by allowing factorization error to propagation up the factorization tree. Such error propagation through the non-orthogonal spaces are described in Chapter 3.

### 2.4.3. Shifted Grid (SG) LOGOS

To release the global coupling problems of the OL-LOGOS method, a special form of the OL-LOGOS factorization was introduced [63]. The shifted grid (SG-LOGOS) is based on a deferred strategy for covering all seams in the multilevel geometric partitioning tree. At a given level, modes are defined on the original oct-tree as in the NL-LOGOS method. Then, before proceeding to the next level, another step is performed in which DOF are localized to the same field group at the same level but this time using a shifted grid of source groups at the same level (see Figure 7).

There are three advantages with this approach. First, the QR decomposition dimension is reduced compared to the other OL-LOGOS approaches. Second, the global
coupling observed with the other OL-LGOOS methods discussed above is removed in the SG-LOGOS $\Lambda^{(L)}_L$ matrix.

Figure 7 demonstrates the source groups in the shifted grid approach. For localizing to the group 28, the QR decomposition is performed with respect to the SG-Range B. After this, the decomposition is used again for the new siblings in the same shifted grid parent group. The coupling problem is completely removed between the shifted parent groups, and it enables independent error analysis between the range groups. However, the matrices $\Lambda^{(L)}_L$ and $\Lambda^{(N)}_L$ are still non-orthogonal; this means the error can cascade through the levels of factorization. This issue is treated in the Chapter 3, where it is shown how to derive rigorous error control even with the non-orthogonal basis matrices described here.

2.5. Error Estimations of the LOGOS Factorization

It is necessary to evaluate the error from the localization in order to decide the precision of the LOGOS factorization. Errors arise by assuming the $\bar{I}$ and $\bar{0}$ matrices can be replaced by true identity and zero matrices, respectively, during the factorization. The descriptions about the error in the LOGOS factorization is detailed in [59], which weighs the relative RMS error in the system matrix more than others. The tilde is used on the same elements of the linear system equations to represent the elements derived or rebuilt from the factorization. $\tilde{x}$ means the solution from the LOGOS procedure, and $\tilde{Z}$ and $Z^{-1}$ mean the rebuilt system matrix and the inverse of the system matrix from the LOGOS factorization, respectively. The norms for the following evaluations are all Frobenius norms.

The solution error is almost always the most important error quantity in linear system solution. Unfortunately, exact solutions are seldom known except for a limited number of the special cases. More practical error measures include the well-known residual error. The residual error is measured from the difference between $\bar{F}$ and $\tilde{F}$, which is from the original system matrix with $\bar{x}$ replaced by $\tilde{x}$ from the LOGOS solution:
in which, \( \text{cond}(\cdots) \) means the condition number of the matrix in the parentheses. (2.30) shows that \( \xi_{\text{res}} \) depends on the condition number of the rebuilt system matrix. The condition number of the rebuilt system is decided by the original system matrix. So this value is not controlled by the solution methods directly, rather, resulted from the system condition. As a result, this quantity is not proper for evaluating the solution method by itself. So the directly controllable error quantity is the rebuilt, relative RMS error:

\[
\xi_{\text{RMS}} = \frac{\| Z - \tilde{Z} \|_F}{\| Z \|_F},
\]

where, the rebuilt system matrix is computed from the LOGOS factorization result, and it is compared with the system matrix directly. (2.31) demands no other factors from the exact solution results, and it is directly comparable from the given linear system matrix. As indicated by (2.30), it provides a strategy for controlling the residual error as well. As a consequence, \( \xi_{\text{RMS}} \) is used to evaluate the LOGOS factorization. In the following examples, the quantity \( \xi_{\text{res}} \) is also displayed. The error measure (2.31) is used to analyze the localization errors in Chapter 3.
Figure 1. The Square Plate in 3 Level Quad-Tree: The relation from the children to the parent at the higher level of the Quad-Tree. Each Parent has four regular children and the children groups “collapse” into one parent group to build the next level group in the localizing process.
Figure 2: NL-LOGOS field group (28) and the source group (28) on the tree at the Level-4
Figure 3: NL-LOGOS Basis structure. At both the localizing and the non-localizing matrices are block diagonal and the orthogonal to each other.
Figure 4: OL-LOGOS field group (28) and the overlapped source groups on the tree at the Level-4
Figure 5: OL-LOGOS field group (28) and the overlapped source groups of the children’s neighbors on the tree at the Level-4
Figure 6: OL-LOGOS Basis Structure. The localizing basis (red) has a coupling structure, while the non-localizing basis (blue) has a block diagonal (non-coupled) structure. The localizing space is not orthogonal to the non-localizing space, causing a coupling between these subspaces when the matrix is inverted.
Figure 7: SG-LOGOS field group (28) and the shifted source groups on the tree at the Level-4
Figure 8: SG-LOGOS Basis Structure. On the localizing basis the coupling is partially released due to the shifted grid based overlapping range. The relation between the localizing and the non-localizing spaces, however, is still not orthogonal. Critically, however, the inverse of this matrix is block diagonal; the coupling range is limited.
Chapter 3. The Error Analysis and Control of the LOGOS Factorization

The LOGOS factorization error is introduced when making a decision on whether a source mode is localizing or not. This chapter shows how that decision should be made in order to control the error (2.31) in a multilevel SG-LOGOS factorization.

3.1. The RMS Error of the System

In Chapter 2, the system RMS error, which is the relative difference between the given system matrix and the rebuilt system matrix, is mentioned as a directly controllable error. The error formulation is given as (2.31), and let \( \|Z - \tilde{Z}\|_F = \|Z_{\text{err}}\|_F \) to make:

\[
\varepsilon_{\text{RMS}} = \frac{\|Z_{\text{err}}\|_F}{\|Z\|_F} \leq \varepsilon
\]  

(3.1)

where \( \varepsilon \) is a given quantity specifying the desired accuracy of the factorization. Due to the multi-level property of the LOGOS framework, each level has the linear system to be localized. To represent the error at level-L:

\[
\varepsilon_{\text{RMS,L}} = \frac{\|Z_{\text{err, L}}\|_F}{\|Z_L\|_F} \leq \varepsilon_L
\]  

(3.2)

The formulation of the numerator of (3.2) is derived by applying (2.22) and (2.23):

\[
\|Z_{\text{err, L}}\|_F = \|Z_L - \tilde{Z}_L\|_F = \left\| P_L \cdot \tilde{Z}_L \cdot (\Lambda_L)^{-1} - P_L \cdot Z_L \cdot (\Lambda_L)^{-1} \right\|_F = \left\| P_L \cdot (\tilde{Z}_L - Z_L) \cdot (\Lambda_L)^{-1} \right\|_F = \left\| \left( Z_L^{(L)} - Z_L^{(N)} \right) \cdot (\Lambda_L)^{-1} \right\|_F = \left\| Z_{\text{err, L}} \cdot Z_L^{(N)} \cdot (\Lambda_L)^{-1} \right\|_F
\]  

(3.3)

The factorization error occurs only in the localized space of the system, and the non-localized space stays intact from the level-L localization. The error on the non-localized space is assumed from the coarser level (level-(L-1)). The error on the level-L is also
transferred to level-(L+1), until the finest level. The basis function \((\Lambda_L)^{-1}\) can be shown to satisfy:

\[
(\Lambda_L)^{-1} = \begin{bmatrix} \Lambda_L^{(L)} & \Lambda_L^{(N)} \end{bmatrix}^{-1} = \begin{bmatrix} (\Lambda_L)^{-1,(L)} \\ (\Lambda_L)^{-1,(N)} \end{bmatrix}
\]

(3.4)

Let the space spanned by \(\Lambda_L^{(L)}\) and \(\Lambda_L^{(N)}\) be \(\Omega^{(L)}\) and \(\Omega^{(N)}\), respectively. In general, \(\Omega^{(L)}\) and \(\Omega^{(N)}\) are non-orthogonal; this implies that \(\Lambda_L^{(L)}\) and \(\Lambda_L^{(N)}\) are not inverted solely on their own space, due to the interaction of the spaces. If they were orthogonal to each other, then the following relations would hold: \((\Lambda_L)^{-1,(L)} = (\Lambda_L^{(L)})^\dagger\) and \((\Lambda_L)^{-1,(N)} = (\Lambda_L^{(N)})^\dagger\). These relations are used for the NL-LOGOS error analysis and control in [59], and greatly simplify the analysis. Unfortunately, for SG-LOGOS methods this is not possible.

By applying (3.4) to (3.3):

\[
\|Z_{err,L}\|_F = \left\| \begin{bmatrix} Z_{err,L}^{(L)} & Z_{err,L}^{(N)} \end{bmatrix} \begin{bmatrix} (\Lambda_L)^{-1,(L)} \\ (\Lambda_L)^{-1,(N)} \end{bmatrix} \right\|_F
\]

\[
= \left\| Z_{err,L}^{(L)} \cdot (\Lambda_L)^{-1,(L)} + Z_{err,L}^{(N)} \cdot (\Lambda_L)^{-1,(N)} \right\|
\]

(3.5)

If the spaces \(\Omega^{(L)}\) and \(\Omega^{(N)}\) are orthogonal, the \(Z_{err,L}^{(N)}\) term would be zero. When nonzero, \(Z_{err,L}^{(N)}\) has the role only with the non-orthogonal \(\Omega^{(N)}\) space to the localizing space as the source of cascaded error from the other levels.

### 3.2. The inverse of the basis matrix on the non-orthogonal spaces

The analysis of [59] assumes that the localizing and non-localizing spaces, \(\Omega^{(L)}\) and \(\Omega^{(N)}\) are orthogonal; this permits the straightforward development of a rigorous error control procedure for NL-LOGOS factorizations. Unfortunately, overlapped LOGOS factorizations (i.e., OL-LOGOS factorizations, which include the SG-LOGOS factorization) have a non-orthogonal relation between the two modes. The development
of a rigorous error control strategy for these methods requires a precise understanding of the effect that this non-orthogonality has on the factorization approximations.

To check how the two spaces interact with each other while inverted, we consider one of the spaces as the “pivot space.” The projection matrix of the pivot space is computed from the unitary matrix of the space. For example, \( \Lambda_L^{(L)} \) of \( \begin{bmatrix} \Lambda_L^{(L)} & \Lambda_L^{(N)} \end{bmatrix} \) is chosen as the pivot space, and the projection matrix is named \( \omega_L^{(L)} \). Using \( \omega_L^{(L)} \), the \( \Lambda_L^{(N)} \) can be projected onto the span and the null space of the \( \Lambda_L^{(L)} \), denoted by \( \Lambda_{L,L}^{(N)} \) and \( \Lambda_{L,L}^{(N)} \) respectively. The inverse with respect to the pivot space and its associated projection matrices is derived like:

\[
\Lambda_L^{-1} = \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} = \left( \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right] + [0 \ \Lambda_L^{(N)}] \right)^{-1} = \left( \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right] + \omega_L^{(L)} I [0 \ \Lambda_L^{(N)}] \right)^{-1} = \left( \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right] + \omega_L^{(L)} I [0 \ \Lambda_L^{(N)}] \right)^{-1} \]

\[
= \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} - \left( \Lambda_L^{(N)} \Lambda_L^{(N)} \right) \omega_L^{(L)} \left( I - [0 \ \Lambda_L^{(N)}] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} \omega_L^{(L)} \right) \left[ 0 \ \Lambda_L^{(N)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1}
\]

\[
= \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} - \left( \Lambda_L^{(N)} \Lambda_L^{(N)} \right) \omega_L^{(L)} \left( I - [0 \ \Lambda_L^{(N)}] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} \omega_L^{(L)} \right) \left[ 0 \ \Lambda_L^{(N)} \right] \left( \Lambda_L^{(L)} \Lambda_L^{(N)} \right)^{-1}
\]

\[
= \left( \Lambda_L^{(L)} \Lambda_L^{(N)} \right)^{-1} \left[ \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} - \omega_L^{(L)} I \left[ 0 \ \Lambda_L^{(N)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} \omega_L^{(L)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1}
\]

\[
= \left( \Lambda_L^{(L)} \Lambda_L^{(N)} \right)^{-1} \left[ \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} - \omega_L^{(L)} I \left[ 0 \ \Lambda_L^{(N)} \right] \left( \Lambda_L^{(L)} \Lambda_L^{(N)} \right)^{-1} \omega_L^{(L)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1}
\]

\[
= \left( \Lambda_L^{(L)} \Lambda_L^{(N)} \right)^{-1} \left[ \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} - \omega_L^{(L)} I \left[ 0 \ \Lambda_L^{(N)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1} \omega_L^{(L)} \right] \left[ \Lambda_L^{(L)} \Lambda_L^{(N)} \right]^{-1}
\]

(3.6)
Where, on the third line of the derivation, the Sherman-Morrison-Woodbury formulation is applied,

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U \left( C^{-1} - VA^{-1}U \right) VA^{-1}. \quad (3.7)$$

$\Lambda^{-1}_L$ is described in terms of $A = [\Lambda_{L}^{(L)} \ \Lambda_{L}^{(N)}]$, $U = \omega_{L}^{(L)}$, $C = I$ and $V = [0 \ \Lambda_{L}^{(N)}]$ for (3.7). As seen from (3.6), the non-pivot part of the matrix is the pseudo-inverse of the null projected matrix with respect of the pivot matrix. If we (i) switch the pivot matrix and repeat the derivation, and then (ii) combine the counter part of the two results, the final result is:

$$\Lambda^{-1} = \begin{bmatrix}
(\Lambda_{L}^{(L)})^\dagger \left( I - \Lambda_{L}^{(L)} (\Lambda_{L}^{(L)})^\dagger \right) \\
(\Lambda_{L}^{(N)})^\dagger \\
(\Lambda_{L}^{(L)})^\dagger \\
(\Lambda_{L}^{(N)})^\dagger \left( I - \Lambda_{L}^{(L)} (\Lambda_{L}^{(L)})^\dagger \right)
\end{bmatrix} \quad \quad (3.8)$$

(3.8) demonstrates that the inverse of the matrix consists of the non-orthogonal spaces is the pseudo inverses of the null projections of the other spaces.

One particular application of (3.8) occurs when one of the submatrices is unitary. In this case, the projection matrix is computed without extra decomposition, and it reduces the computational load of the regular inverse computation routine, which has a complexity of $4n^3/3$. When the unitary part has bigger dimension than another, the computational efficiency is expected to be improved more, since the companion matrix will have a smaller dimension for the QR decomposition, which is the most expensive component of the computation.

This property is used in the actual coding for the error control to reduce the computation time of the basis inverse. The non-localizing basis is actually the unitary matrix, and the tighter tolerance tends to increase the column dimension of the matrix, increasing the savings provided by this approach.
3.3. Separation of The Error Terms

It has been shown that the non-orthogonal basis spaces, \( \Omega^{(L)} \) and \( \Omega^{(N)} \), lead to coupling issues during the inversion of the basis matrix. This problem causes the interaction of the error terms in (3.5), because these rise at different levels. The approximation errors at level-\( L \) are restricted to the localizing modes; no approximations are made for the non-localizing part of the basis at level-\( L \). Any errors on the \( \Omega^{(N)} \) space is due to approximations made at the next (coarser) level of the factorization. Therefore, if (3.5) is directly used to control the error at level-\( L \), one must anticipate the errors from the coarser levels of the factorization. Because the factorization proceeds from fine to coarse levels, this information is not available.

For this and related reasons, the error terms from \( \Omega^{(L)} \) and \( \Omega^{(N)} \) have to be separated to make the error analysis a causal (or, explicit) procedure:

\[
\frac{\|Z_{err,L}\|_F}{\|Z_L\|_F} = \frac{\|Z_{err,L}^{(L)} (\Lambda_L)^{-1(L)} + Z_{err,L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_L\|_F} \\
\leq \frac{\|Z_{err,L}^{(L)} (\Lambda_L)^{-1(L)}\|_F + \|Z_{err,L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_L\|_F} \\
\leq \varepsilon_L
\] (3.9)

The triangular property of the matrix norm is applied to the numerator the equation. In this case, the error distribution ratio turns out to be another issue. Because the number of the localizing modes varies based on the factors like tolerance and system conditioning, a flexible ratio is needed for a reasonable procedure. To derive the formulation, (3.9) can be expanded:

\[
\frac{\|Z_{err,L}^{(L)} (\Lambda_L)^{-1(L)}\|_F + \|Z_{err,L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_L\|_F} = \frac{\|Z_{err,L}^{(L)} (\Lambda_L)^{-1(L)}\|_F}{\|Z_L\|_F} + \frac{\|Z_{err,L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_L\|_F} \\
\leq \varepsilon_L
\] (3.10)

By defining,
\[
\zeta_L = \frac{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L\|_F},
\]
(3.11)

(3.10) can be written with \( \zeta_L \):
\[
\frac{\|Z_{err,L}^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_{err,L}^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F} \leq \frac{\epsilon_L}{\zeta_L} = \epsilon_{L,updated}
\]
(3.12)

From (3.12), \( ER^{(L)} \) and \( ER^{(N)} \) is defined:
\[
ER^{(L)} = \frac{\|Z_{err,L}^{(L)}(\Lambda_L)^{-1(L)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F}
\]
(3.13)
\[
ER^{(N)} = \frac{\|Z_{err,L}^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}
\]
(3.14)

With (3.13), (3.14) and assuming \( ER^{(L)} \geq ER^{(N)} \), the left hand side of (3.12) is:
\[
\frac{\|Z_{err,L}^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_{err,L}^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F} = \frac{(ER^{(L)} - ER^{(N)}) \cdot \|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + ER^{(N)} \cdot \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}
\]
(3.15)
\[
= ER^{(N)} + (ER^{(L)} - ER^{(N)}) \cdot \frac{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}
\]
\[
= ER^{(L)} - (ER^{(L)} - ER^{(N)}) \cdot \frac{\|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}{\|Z_L^{(L)}(\Lambda_L)^{-1(L)}\|_F + \|Z_L^{(N)}(\Lambda_L)^{-1(N)}\|_F}
\]
The implication of (3.15) is that the left hand side of (3.12) has a value between $ER^{(L)}$ and $ER^{(N)}$. Therefore (3.13) and (3.14) satisfy the condition of (3.12) with the inequality conditions,

$$ER^{(L)} \leq \varepsilon_{L,updated},$$  \hspace{1cm} (3.16)
$$ER^{(N)} \leq \varepsilon_{L,updated}.$$  \hspace{1cm} (3.17)

Because all components of (3.12) except $Z_{err,L}^{(N)}$ are all decided after the localization, the factorization at the level-$L$ now has no future input; this feature is critical to computationally efficient error control.

After $\zeta_L$ is computed to satisfy the localization error condition, $\left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F$, which means the error on the non-localizing modes, can be decided. The errors from $\Omega^{(N)}$ are caused at the following levels, so the error for the non-localizing space is to be used to set up the tolerance of the next level. The procedure continues up to the coarsest level, at which point the LOGOS factorization stops. (A conventional LU factorization is used for the remaining, non-localized DOF at the coarsest level.)

The combined condition of (3.16) and (3.17) is quite conservative because the final RMS error at each level has an error less than the target. This overly conservative error control can cause the computational costs to be higher than desired. To account for this effect, the next level’s tolerance is set a little higher than $\varepsilon_{L,updated}$ so that the combined RMS error is closer to $\varepsilon_{L,updated}$. Eqn. (3.12) components are replaced with results from the factorization to calculate $\left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F$, rather than using (3.17). As a result, (3.16) is used for the localization at the level initially, then the results including $\zeta_L$ are used for the next level’s tolerance setup. The formulation is like this:

$$\varepsilon_{L,NextLevel} = \varepsilon_{L+1},$$

$$= \frac{\left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F}{\left\| Z_{L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F}$$

$$\leq \varepsilon_{L,updated} \left( \left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F \right) + \left\| Z_{L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F - \left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F \right) \leq \varepsilon_{L,updated} \left( \left\| Z_{err,L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F \right) \leq \varepsilon_{L,updated} \left( \left\| Z_{L}^{(N)} \left( \Lambda_L \right)^{-1}(N) \right\|_F \right)$$

(3.18)
A detailed description of the factorization and the derivation of the next level tolerance are explained in the following sections.

### 3.4. The Errors From $\Omega^{(L)}$ Space

By defining $\zeta_L$, the two basis space relation (or, coupling) problem is resolved. Next, we focus on the error from the localizing modes in the SG-LOGOS factorization. In this direction, we recall that the localizing source modes on the shifted grid share support with other groups on the shifted grid (see above). The approximation errors introduced by these shifted-grid overlapping basis functions (on the $\Omega^{(L)}$ space) are coupled with the errors introduced for the other groups that have overlapping support. (This coupling occurs during the inverse of the $\Lambda$ matrix, which is required to reconstruct the factored approximation to the system matrix; the latter is what we seek to control.)

Because the localization computation is performed for each group independently, the error due to the basis coupling result is known only after the end of the localization of all overlapped groups and the actual inverse of the basis is performed to check if (3.16) is satisfied. The quantity $\zeta_L$ is not available until the factorization is completed, because the computation relies on results from the factorization, $\Lambda_L^{(L)}$ and $\Lambda_L^{(N)}$. The actual error check can thus be performed only after the localizing basis has been found. This leads to an error control algorithm that relies on a trial-and-error loop, with the initial $\zeta_L$ value set to 1. After the factorization is performed, $\zeta_L$ and the error on $\Omega^{(L)}$ space can be computed. If the result does not satisfy the error control condition, the number of the localizing modes are reduced to further reduce the errors introduced by the localizing approximation. The procedure is described in the following algorithm.
Algorithm 1: the adjusting loop the error on the localizing space

\begin{align*}
\zeta_L \text{ and } \Lambda_L \text{ Loop} \\
\text{Actual Error } &\leftarrow \text{Localizing (} \Lambda_L \text{ calculated) with } \varepsilon_L \\
\text{Calculate } &\zeta_L \\
\varepsilon_{L,\text{updated}} &\leftarrow \varepsilon_L / \zeta_L \\
\text{while } \{\text{Actual Error } > \varepsilon_{L,\text{updated}} \} \text{ do} \\
\text{Reduce the number of Localizing Modes} \\
\text{Actual Error } &\leftarrow \text{Updated } \Lambda_L \\
\text{Calculate } &\zeta_L \\
\varepsilon_{L,\text{updated}} &\leftarrow \varepsilon_L / \zeta_L \\
\text{end while}
\end{align*}

This procedure can be viewed as an iterative solution to the nonlinear error control problem for the overlapped, localizing basis functions on the shifted grid groups. The factorization error on \( \Omega^L \) is controlled by repeatedly checking the coupling errors on the space and reducing the number of modes until the error control condition is passed. Algorithm 1, however, can cause efficiency problems. When the number of the modes is reduced, it is also a problem to approach the right number to satisfy the pass condition. In addition to that, it performs the basis calculations over and over, and the computational cost is expensive because multiple groups are involved in the case of the OL-LOGOS. The following sections discuss efficient strategies for addressing these issues.

### 3.4.1. Localizing method for the initial basis

The initial localization basis calculation is performed before the error control loop is entered. In this case, an individual group has no consideration for the other overlapped groups’ localization (such consideration occurs later during the RMS computation). Thus, the coupling effect from the inverse of the basis is treated after the error condition check with the basis from the initial localization calculation. The number of the modes is then tuned based on the error evaluation. Two methods of error control for these initial localization approximations have been suggested [59]. They are referred to as a “tight”
error condition and a “relaxed” condition. Both are based on the error control algorithm for the NL-LOGOS factorization, which assumes orthogonal spaces for the localizing and non-localizing bases [59]. Briefly, the “tight” control chooses fewer localizing modes than the “relaxed” method. In the non-coupling case, no additional work is required after the initial basis computation; the more general OL-LOGOS situation considered here requires additional tuning due to the coupling of the localizing functions during the matrix reconstruction process. The right number of the modes is important for the next level system dimension and further the asymptotic performance of the LOGOS solution method.

The tighter formulation defines the modes with conservative condition, and it typically excludes too many localizing modes. In contrast, the relaxed formulation assumes optimal conditioning for the system matrix, and yields the maximum possible number of localizing modes. Due to the conditioning assumption, the initial number of the modes is likely to be excessive. It means that only the less or equal numbers to the initial turn to be the answer, to have the adjustment take the decreasing way from the initial number. Because the latter case has only single direction tuning, it is convenient for the repeated basis computation suggested above for the OL-LOGOS. The detailed formulations follow.

A) The tight formulation

The tighter initial basis computation uses an inequality for the multiplication of square matrices. \((q_L)^H\) and \((Z_L^{(L)})^\dagger\) have unitary rows to be eliminated as the last factor of the norm computation. The derivation of this formulation is like:

\[
\begin{align*}
\left\| Z_{err,L}^{(L)} (A_L)^{-1,L(L)} \right\|_F & = \left\| Z_{err,L}^{(L)} (r_L)^{-1} (q_L)^H \right\|_F \\
\left\| Z_L^{(L)} (A_L)^{-1,L(L)} \right\|_F & = \left\| Z_L^{(L)} (r_L)^{-1} (q_L)^H \right\|_F \\
& = \left\| Z_{err,L}^{(L)} \left( Z_L^{(L)} \right)^\dagger Z_L^{(L)} (r_L)^{-1} \right\|_F \\
& \leq \left\| Z_{err,L}^{(L)} \left( Z_L^{(L)} \right)^\dagger \right\|_F \\
& = \left\| Z_{err,L}^{(L)} \right\|_F \leq \epsilon_L
\end{align*}
\]
where, \( \|Z_{\text{err},L}\|_F = \sqrt{\sum_{i=1}^{n} (1 - s_i^2)} \), and \( s_i \) are the singular values from the localization calculation.

**B) The relaxed formulation**

Instead of using the inequality of the matrix norms as above, the maximum and the minimum singular values of \( (r_L)^{-1} \) are taken to have the inequality condition. The maximum value is from the numerator and the minimum is from the denominator for the condition. The ratio of these values is the condition number of the matrix, so that the computation now depends on the basis condition. To have the more relaxed formulation than (3.19), the condition number is replaced by unity, which means that \( Z_L^{(L)} \) is considered to be a unitary matrix, which implies that \( \|Z_L^{(L)}\|_F \) has a value of \( \sqrt{n} \) (\( n \) is number of columns of \( Z_L^{(L)} \)).

The details of this strategy to derive the RMS error are:

\[
\begin{align*}
\frac{\|Z_{\text{err},L}^{(L)} (A_L)^{-1}\|_F}{\|Z_L^{(L)} (A_L)^{-1}\|_F} &= \frac{\|Z_{\text{err},L}^{(L)} (r_L)^{-1}(q_L)^H\|_F}{\|Z_L^{(L)} (r_L)^{-1}(q_L)^H\|_F} \\
&= \frac{\|Z_{\text{err},L}^{(L)} (r_L)^{-1}\|_F}{\|Z_L^{(L)} (r_L)^{-1}\|_F} \\
&= \frac{\|Z_{\text{err},L}^{(L)} \cdot u_{r^{-1}} \cdot s_{r^{-1}}\|_F}{\|Z_L^{(L)} \cdot u_{r^{-1}} \cdot s_{r^{-1}}\|_F} \leq \frac{\text{MAX}(s_{r^{-1}}) \cdot \|Z_{\text{err},L}^{(L)}\|_F}{\text{MIN}(s_{r^{-1}}) \cdot \|Z_L^{(L)}\|_F} \\
&= \text{cond} \left( (r_L)^{-1} \right) \frac{\|Z_{\text{err},L}^{(L)}\|_F}{\|Z_L^{(L)}\|_F} \leq \varepsilon_L
\end{align*}
\]

And finally, applying the assumed conditioning of the matrix,

\[
\|Z_{\text{err},L}^{(L)}\|_F \leq \varepsilon_L \sqrt{n}
\]  

(3.21)

By comparing (3.21) with (3.19), it is clear the relaxed condition can find more localizing modes than the “tight” error control formulation.
3.4.2. Adjusting the number of the localizing modes on $\Omega^{(L)}$

To optimize the efficiency of the algorithm used to find the number of localizing modes satisfying the error condition, the number of times that the trial-and-error basis computation loop is executed should be reduced. One option for this loop is to decrease the number of localizing modes by one when the error does not pass the control condition. This method, however, makes the loop repeat as many times as the difference of the modes between the initial modes and the most satisfying modes.

To decrease the number of loops, a bisectional algorithm is used. The simplest way is to take half of the modes until the error condition is satisfied. Once the condition is passed, the most satisfying number is found in the range of the last number and the previous number of modes. The binary search algorithm uses the divide-and-conquer strategy, it has the computational complexity of $O(\log n)$ normally. This method can be applied because the singular values are in sorted order from the SVD decomposition.

However, the condition number of the matrix has the significant effect for the inverse of the matrix, and the condition number can be decided from the minimum and the maximum singular values. It means that when the smaller singular value is included in the localizing modes, the error is more amplified than the case of the value of 1 or closer to one. Therefore, the bisectional method is applied to find the next point with the weighted values. The point is approached with the moment with the number of the modes times corresponding singular value difference from 1.00. It is like the formulation of the centroid of the weighted singular values. The smaller singular values have more effect on the condition number of the matrix, and the larger difference of the singular value can enough influence on the inverse. Then the decreasing rate is less in the case of the larger singular value difference is included. In the case of increasing the number, the singular values closer to 1 enlarge the number faster. The searching speed is expected equal to the non-weighted method for the situation when a matrix having unity condition number is involved. In other cases, from Figure 9, the formulation of the next location is computed like:
\[
P_{\text{next}} = \frac{\sum_{i=0}^{N} i \cdot (1 - s_i)}{\sum_{i=0}^{N} (1 - s_i)}
\] (3.22)

In the case of the increase, assuming the bottom limit of the number is \(N_B\) and the top limit is \(N_T\):

\[
P_{\text{next}} = \frac{\sum_{i=N_B}^{N_T} i \cdot (1 - s_i)}{\sum_{i=N_B}^{N_T} (1 - s_i)} = \frac{\sum_{i=0}^{N_T} i \cdot (1 - s_i) - \sum_{i=0}^{N_B} i \cdot (1 - s_i)}{\sum_{i=0}^{N_T} (1 - s_i) - \sum_{i=0}^{N_B} (1 - s_i)}
\] (3.23)

### 3.4.3. The basis computation components

For tuning the number of the localizing modes, the most contributing singular value needs to be found out of those groups. The singular values of each group sharing the source range are collected and sorted to apply the bisectional searching method explained just before. The combined singular value list is sorted with the Quick Sort method to get the arranged list. Besides the singular value, the accumulated singular value difference and the moment values also need to be stored along the sorted list to improve the next point computation.

The basis matrix computation includes the QR decomposition for the null space of the localizing field group as the non-localizing matrix, and the error condition must be compared with the norm of the rebuilt system matrix which demands the inverse of the basis matrix. Performing QR and the inverse are the most expensive operations; repeating those components results in performance degradation. So, reducing the number of such operations and accelerating them leads the method to a computationally efficient algorithm. The detailed description of these components follows.

#### A) The basis inverse routine

To check the pass condition, (3.13), and to compute \(\zeta_L\), the inverse of the basis matrix need to be computed. The regular inverse formulation demands a significant computation time with complexity of \(O\left(\frac{4n^3}{3}\right)\). The error correction procedure repeats
until the condition is satisfied, the time needs to be saved for the solution method performance. The OL-LOGOS methods have multiple groups involved in the coupling by the source range, so in the extreme case, the whole basis matrix must be inverted. To avoid this, we limit our consideration to the SG-LOGOS, which is one of the OL-LOGOS method and allows the overlapping groups to share the QR decomposition. The basis matrices have the similar structure of Figure 8. All the groups in the source range are localized exclusively from each other, and this property enables the matrix to have the block diagonal structure, even though the coupling and the non-orthogonal relation between $\Omega^{(L)}$ and $\Omega^{(N)}$ still remains. Owing to the SG-LOGOS basis structure, the OL-LOGOS problems can be solved separately among the groups sharing the shifted parent group.

In the case of the oct-tree based geometry, one block of the basis matrix has normally 8 field groups to treat, and it remains a nontrivial computational load for the mathematic routines for the factorization. For each block, $\Lambda^{(L)}_L$ has the coupling structure, while $\Lambda^{(N)}_L$ is the unitary structure. To take advantage of this property, the second row of (3.8) is used. The pseudo-inverse of the unitary matrix $\Lambda^{(N)}_L$, $(\Lambda^{(N)}_L)^\dagger = (\Lambda^{(N)}_L)^H$, and also the projection matrix on the $\Omega^{(N)}$, $\omega^{(N)}_L = (\Lambda^{(N)}_L)^H \cdot \Lambda^{(N)}_L$. From these, $\Lambda^{(L)}_{L\parallel} = \omega^{(N)}_L \cdot \Lambda^{(L)}_L$ and $\Lambda^{(L)}_{L\perp} = (I - \omega^{(N)}_L) \cdot \Lambda^{(L)}_L$. Then $(\Lambda^{(L)}_{L\perp})^\dagger$ remains to be computed with the QR decomposition. From the decomposition result have the reduced R dimension to take inverse, and Q for the Hermitian as the pseudo-inverse. This method is used to improve the inverse computation speed by using the unitary property of the non-localizing basis. In the case of the unitary space has bigger dimension, it is expected to become more effective. In the LOGOS factorization, when the tolerance of the localizing modes is tightened, the non-localizing space portion in the basis matrix increases. The procedure reduces the size of the LU factorization and it leads to the less divisions involved. The division computation needs more CPU cycle than multiplication case.

\textbf{B) The non-localizing basis computation}
The non-localizing basis is computed from the localizing basis. Out of the overlapping source range, the source, which is corresponding to the field group, is taken and perform the QR decomposition. Then the null space portion of the Q matrix is chosen as the non-localizing basis with respect to the field group. During the error correction loop, repeating the QR decomposition for all groups in the source range seems inefficient. So the fact that the initial basis overestimates the number of localizing modes is taken advantage of. $\Lambda^{(L)}_L$ is computed with $R^{-1} \cdot v^{(L)}$ where $R^{-1}$ is from the QR decomposition of the system matrix over the source range, and this matrix has no variation for error control. $v^{(L)}$, however, depends on the number of the localizing modes, and the number is largest at the loop starting point. $v$ is computed from the SVD, and up to localizing number of the column from the column 1 is taken to compute $\Lambda^{(L)}_L$. So the corrected $\Lambda^{(L)}_L$ has less column dimension of the initial condition. The dimension relation of the computation is illustrated in the Figure 11. The identically colored part of the $\Lambda^{(L),\text{init}}_L$ and $\Lambda^{(L),\text{adj}}_L$ means that they have the same matrix entries over those region. This means that the localizing basis does not need to be computed repeatedly if the initial basis is stored. The updated basis just takes the reduced part from the initial basis matrix.

As a result, the QR result for the $\Lambda^{(N)}_L$ does not change with the reduced dimension of $\Lambda^{(L)}_L$. The Q matrix is unitary, and once the first localizing number of columns have the right null space over the remaining parts of the Q matrix. So the decomposition performs only once and in the loop the computation only includes the sub-sectioning from the initial basis matrix and the Q matrix. This coding demands the memory space to keep the initially computed data, while the CPU cost can be decreased compared with an approach that re-computes the QR decompositions repeatedly.

### 3.5. The Next Level Tolerance Computation

The localization error control formulation and coding at level-$L$ are reviewed above. These are performed on the $\Omega^{(L)}$ space, and the $\Omega^{(N)}$ space projection of the system matrix does not cause any error at the level. The system RMS error computation is completed when the errors from the $\Omega^{(L)}$ space and from $\Omega^{(N)}$ are combined. Due to the
non-orthogonal relation between those spaces, the errors interact with each other. The error from the $\Omega^{(N)}$ space is the result of the factorization at the next level, level-$(L-1)$. The formulation (3.18) describes that relation between those errors. Once the factorization error is decided, the tolerance for the next level depends on the value to have the combined error at the level. In this section, we study how the next level factorization error propagates through the non-localizing space; from this analysis, it is determined how the next level tolerance is applied to the factorization at level-$(L-1)$.

The system matrix at the next level is just $Z_{L}^{(NN)}$ of (2.22). There is no error due to $Z_{L}^{(LN)}$, which stays intact (no localization approximations are made for this submatrix). Therefore,

$$Z_{err,L}^{(N)} = \begin{bmatrix} P_{L}^{(L)} \quad P_{L}^{(N)} \end{bmatrix} \begin{bmatrix} Z_{err,L}^{(LN)} \\ Z_{err,L}^{(NN)} \end{bmatrix},$$

(3.24)

because $Z_{err,L}^{(LN)}$ has all zero entries.

The next-level error formulation with the tolerance is set as the level-$L$ like:

$$\frac{\|Z_{err,L}^{(NN)}\|}{\|Z_{L}^{(NN)}\|} = \frac{\|Z_{err,L}^{(L)}\|}{\|Z_{L-1}^{(L)}\|^2} \leq \varepsilon_{L-1}$$

(3.25)

Using (3.18), (3.24) and (3.25), from equation (3.14), the following relation is derived:

$$\frac{\|Z_{err,L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_{L}^{(N)} (\Lambda_L)^{-1(N)}\|_F} = \frac{\|P_{L}^{(N)} \cdot Z_{err,L}^{(NN)} \cdot (\Lambda_L)^{-1(N)}\|_F}{\|Z_{L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}$$

$$\leq \frac{\|Z_{err,L}^{(NN)}\|_F}{\|Z_{L}^{(NN)}\|_F} \cdot \frac{\|\Lambda^{-1(N)}\|_F}{\|Z_{L}^{(NN)}\|_F} \cdot \frac{\|Z_{L}^{(N)} (\Lambda_L)^{-1(N)}\|_F}{\|Z_{L}^{(NN)}\|_F}$$

(3.26)

$$\leq \varepsilon_{L, NextLevel}$$

In (3.26), the unitary matrix $P_{L}^{(N)}$ is deleted from the norm, and $Z_{L}^{(NN)} = 1$ is put into the denominator to match the formulation (3.25). To simplify (3.26), let the other factors, $\rho_L$ and $\kappa_L$ be defined.
\[
\rho_L = \frac{\left\| Z_L^{(N)} \left( \Lambda_L \right)^{-1,(N)} \right\|_F}{\left\| Z_L^{(NN)} \right\|_F}
\]  (3.27)

\[
\kappa_L = \left\| \Lambda_L^{-1,(N)} \right\|_F
\]  (3.28)

From (3.26), transferring \( \rho_L \) and \( \kappa_L \) to the right hand side of the inequality relation:

\[
\left\| Z_{err,L}^{(NN)} \right\|_F \leq \frac{\rho_L}{\kappa_L} \epsilon_{L,\text{NextLevel}}
\]  (3.29)

Finally, by (3.25) and (3.29), the allowed RMS on the non-localizing space, \( \Omega^{(N)} \) is related to the next level system matrix tolerance like below.

\[
\epsilon_{L-1} = \frac{\rho_L}{\kappa_L} \epsilon_{L,\text{NextLevel}}
\]  (3.30)

If the relation (3.30) is applied to the NL-LOGOS case, the error control algorithm that results is identical to that proposed in [59]. The difference for the NL-LOGOS algorithm is the non-localizing part of the basis is orthonormal, so \( \left( \Lambda_L \right)^{-1,(N)} \) is just \( \Lambda_L^{L} \), which is canceled out from the first line of the derivation in (3.26) from both the denominator and the numerator. This leads \( \kappa_L \) to be zero in (3.28), and it is concluded that the \( \rho \) from [59] and (3.27) have the same value. For the OL-LOGOS (and SG-LOGOS), \( \kappa_L \) cannot be deleted owing to the fact that the \( \Lambda_L^{(L)} \) and \( \Lambda_L^{(N)} \) are not orthogonal to each other.

### 3.6. Numerical Results

The error control formulation is tested on several CEM systems. The augmented electric field integral equation (AEFIE) and the magnetic field integral equation (MFIE) scattering problem on the PEC sphere object as the scatterer with the radius of 0.44 \( \lambda \) at the frequency of 133MHz. The volume integral equation (VIE) problem is applied to the dielectric sphere shell of the relative permittivity \( (\varepsilon_r) \) of 4, the thickness of the shell varying to have the same number of cells through the shell as the resolution of the cell increases. As the last problem, the error control algorithm is tested on the electric field integral equation (EFIE). It is well known that the EFIE is an ill-conditioned system; it is
expected that the LOGOS factorization procedure without rigorous error control will result in significant RMS error.

The main goal of this research is the error control in the LOGOS framework. In the test cases, the priority is to control the RMS matrix error, keeping it less than the given (i.e., input) tolerance. As described above, the rigorous error control algorithm requires additional computations. Consequently, it is expected that the new algorithm will be slower than the previously reported SG-LOGOS factorization algorithm. However, it is also expected that the amount of extra computation should be less for well-conditioned systems. For these reasons, the following numerical examples are performed with the proposed, rigorous error control algorithms both “on” and “off” in order to evaluate the differences in both the error performance and the computational time.

The LOGOS framework has showed that the RMS error is mostly less than the tolerance in the case of the stable system. However, the low tolerance is given, sometimes the error goes beyond the target error. It is known that the lower tolerance includes the localizing modes which correspond to the larger singular difference from 1.0. In the cases the lower singular values are involved in the rebuilt the linear system, they have the errors amplified when the relaxed error control formulation of [59] is applied directly to the OL-LOGOS. It is to be observed what the lower tolerance case result in when the error control is applied. It is known that the EFIE system has the large condition number and so AEFIE formulation is developed to have the improved system condition. The EFIE systems have relatively higher rebuilt error than the stable systems even with the tighter tolerance, and it is expected that the EFIE test shows how the performance changes when the error control routine plays more dominant role in the LOGOS solution procedure.

The stable systems with the lower condition number like the MFIE and the VIE of this research are used to investigate how the error control works with these linear systems. In the tests, it is expected to show how well-conditioned systems respond when the error control code is implemented. Due to the error check routine in the LOGOS procedure, the overall CPU time level may increase the cost of the rigorously controlled rebuilt error. Because the asymptotic performance is an important performance target of the LOGOS
solution procedure, the variations of the performing rate are compared with the results from the LOGOS without the error control routine.

The OL-LOGOS strategy considered is the SG-LOGOS, because it represents the OL-LOGOS performance property exactly [63], the decoupled structure of the localizing basis enable the error control be performed over the shifted parent block (see Figure 8) to reduce the number of the interaction groups to release the localizing modes check load. In addition to those reasons, the SG-LOGOS has the least computational overheads than the regular neighbor range OL-LOGOS by reducing dimension of the submatrix requiring QR decomposition.

The main part of the control method is the trial-and-error loop. The loop contains the highest latency routines. This means that repeating the loop increases the role of the routine in the performance curve. Even though tolerance constraints are satisfied, the solution method should not increase the asymptotic complexity too much. Well-conditioned systems have the smallest variation in the singular values, and that leads to faster adjustment to reach the exact number of modes.

### 3.6.1. The AEFIE system result

The AEFIE has a controlled RMS error even when the error control function is deactivated. Figure 12 shows the error result of this case. Only the tolerance of $10^{-2}$ case does not make the expected result beyond the tolerance of the solution.

The error control function is activated to control even the case of $10^{-2}$, which causes the erroneous result when deactivated. The Figure 13 shows that the rebuilt RMS error is controlled for all given tolerances. The RMS error of the $10^{-2}$ tolerance reduces to less than the allowable system error. The other tolerances remain around the no error control results.

Figure 14 is the plot of CPU to show the asymptotical performance. It has the performance rate of $O(n \log n)$ which is the expected OL-LOGOS performance. The expanded source range for the localizing improves the performance compared to the case with the single group of the source range, which is advantageous for QR decomposition cost.
Because error control formulation decreases the error amount with the tolerance of $10^{-2}$, and therefore, as seen in the Figure 15, the performance rate becomes slower ($O(N^{1/3})$) than the case of no error control.

### 3.6.2. The MFIE system result

As seen in Figure 16, without the error control formulation activated, the rebuilt error looks controlled. This is because the MFIE system is very stable, having a fast asymptotic rate ($O(N^{0.9})$) in Figure 18. The well-conditioned system results in an efficient solution method, and the errors are controlled for all the tolerance given, even at the lowest tolerance, $10^{-2}$, unlike the AEFIE. However, the system solution is not equipped with a rigorous error control, and it is not possible to conclude that other test cases would all have acceptable performance given any value of the tolerance or other geometries, etc.

The results of the factorization using the rigorous error control SG-LOGOS factorization are shown in Figure 17, where it is seen that all error levels decrease from the error plot of the no-rigorous-error-control system. In these cases, the results are well beneath the tolerance line in the plot. The difference between the actual RMS error and the tolerance given for the solution method, is caused by the inequalities used in deriving the error control algorithm. The inequalities cause the error control to overestimate the norm relations.

As mentioned in the no error control error analysis, the systems have very efficient asymptotic curve in Figure 18. However, the controlled errors have the extra tightened error amount than the tolerance, the result plots run along the more conservative condition curve. That means the performance rate goes slower (about $O(N \log N)$) in Figure 19 than in Figure 18.

For the well-conditioned MFIE system, the RMS error does not exceed the specified tolerances in the test case in this study. The provided error control formulation caused a tighter condition in finding the localizing modes and elevates the performance slope compared to the no-rigorous-control method.
3.6.3. The VIE system result

The VIE system is tested with the dielectric sphere shell of the radius set to 1m, the low relative permittivity value \( (\varepsilon_r) \), 4, and the 1Hz testing frequency. The test condition provides a stable linear system matrix like the MFIE case. In this test, the lowest tolerance is set to \( 10^{-1} \), with which very low singular value is to be included in the localizing modes. Even with this small value of the tolerance, the stable system shows a rebuild error under the tolerance. As seen from Figure 20, the RMS error values are very close to the given tolerances from \( 10^{-1} \) to \( 10^{-4} \), even if the solution residual errors are still less than the tolerance. As mentioned above, the residual error is not directly controlled; it depends on the product of the RMS error and the system condition number.

Figure 21 shows the results of the error controlled factorization of the same system as Figure 20. Compared to the MFIE cases, which also have low condition numbers, the RMS error drop due to the error control system is not significant for the tolerance from \( 10^{-2} \) to \( 10^{-4} \). But the result of the tolerance of \( 10^{-1} \) case has a large RMS error drop. This occurs because the singular values for this generous tolerance play a significant role in the loop exit condition. Even including the result of the loose tolerance, however, the overall performance rate does not change owing to the results with the tolerance between \( 10^{-2} \) and \( 10^{-4} \). Even the CPU asymptotic rate with the tolerance of \( 10^{-1} \) does not slow down significantly.

The VIE system solution results shows that this stable system has the same performance rates for the tighter tolerances, while the loose target error has the extra error drop to result in the higher, yet still acceptable, rate. The error results and the asymptotic results with the error control formulation for the VIE illustrate that the proposed error control algorithm can provide important improvements on previous results, while still providing a computationally efficient algorithm. It also demonstrates the importance of well-conditioned system matrices to the SG-LOGOS factorization.

The error results so far show that well-conditioned systems often have RMS errors under the given tolerance, but the lower tolerance cases may cause the error to exceed acceptable levels. Although the test cases show that the errors keep less than the tolerance even without error control in many cases, the non-orthogonal space error propagation is
not treated rigorously. The error control formulation sometimes tighten the tolerance, due to the inequality properties used in the derivation, which makes the CPU performance plots follow along the curve expected to be the result of the lower tolerance. This means that the asymptotic performances are sometimes slower than the no error control case, while the lower rebuild error drops keep the performance rate. When the RMS error is close to the no error control case, this indicates that an almost identical number of localizing modes are found and results in a similar performance rate.

3.6.4. The EFIE system result

Unlike the results so far, Figure 24 for the EFIE shows uncontrolled RMS errors for most of the cases considered. The frequency and mesh instabilities of the EFIE make the conditioning quite poor (large condition number) [64]. The dramatic drop of the singular values (in the descending ordered list) causes the factorization to find fewer localizing modes, and relatively smaller singular values can be included in the determination of the localizing modes. These cause the rebuild error to increase especially by the larger singular value difference from 1.0.

To get the rebuild RMS error from the solution method, the error control formulation is activated. Because the results from deactivated factorization procedure have a clearly higher error value for several cases, the RMS error drops are expected to be quite large compared to results of the other tests. The corrected results with the formulation are plotted in Figure 25 to show that even the most relaxed tolerance \( \varepsilon = 10^{-1} \) condition is satisfied. The plots on Figure 25 have relatively stable traces compared to the ones on the Figure 24. The trouble in the testing these cases is the number of the localizing modes. The more localizing modes the LOGOS finds, the less system matrix at the coarse level is built; at the stop level, the system matrix is solved with the traditional way. At the last level of the procedure, the linear system dimensions are expected to have the reduced order of dimension for the traditional methods can have proper behavior. But in these error corrected EFIE systems, the localizing modes are not found enough for the final level solution method can be performed. The dimension of the system at the final level is so large to fail to be solved. So the plots have no error results at around 250K of DOF cases.
Figure 26 has the plots of the CPU performance rate of uncontrolled error solution results. Compared to the AEFIE results, the EFIE CPU behaviors have increased rates. The AEFIE systems are developed to improve the condition number of the EFIE, and this results in the asymptotic differences. Even though the AEFIE is based on the EFIE, the improved system condition means that more localizing modes can be found at finer levels for the AEFIE, yielding a lower computational complexity.

The error corrected CPU behavior is plotted on Figure 27. As mentioned in the controlled error results, the localizing modes to satisfy the tolerances decrease eventually. It is coupled with more repetitions of the control loop, leading to plots that have the much higher asymptotic rate than the normal OL-LOGOS results. The ill-conditioned system error control drops the RMS errors from the uncontrolled results, and in the process, the control formulation loop repeats more than the stable linear system cases. Despite the high behavior rate, the RMS errors of the solution is controlled.

3.6.5. Summary of the Numerical Results

The numerical results show several system factorization results both with and without error control formulation. The AEIE, MFIE, and VIE are the well-conditioned system results. The EFIE is an example of an ill-conditioned linear system.

Even without error control, the SG-LOGOS method provides controlled RMS errors. With the error control algorithm, the actual RMS errors drop further. This effect results in an increased asymptotic CPU performance rate, but the increasing amount is not too significant to hurt efficiency of the OL-LOGOS. Most of all, even among the low condition number systems, some test cases fail to satisfy the tolerance condition, and the proposed error control formulation provides a rigorously controlled RMS error result.

The EFIE system, which is poorly condition, mostly failed to provide acceptable rebuild errors when the error control formulation was deactivated. With the control loop activated, the error dropped to restrict the factorization error to the tolerance. The error results are all controlled, even the $10^{-1}$ tolerance is satisfied, and the controlled results cost the CPU behavior drops due to the decreased number of the localizing modes.

For the tested systems, the target rebuilt RMS errors are all satisfied when the proposed error control formulation is activated. The amount of change in the CPU time
was observed to depend on the conditioning of the underlying matrix equation. For well-conditioned linear systems, the change in CPU time was not large; however, for a poorly conditioned system, the increase in the CPU time was significant.
Figure 9: the singular value plot from SVD
Figure 10: The non-localizing basis computation relation
Figure 11: The localizing basis computation and update
Figure 12: AEFIE error result (Heuristic Error Control)
Figure 13: AEFIE error result (Error Control)
Figure 14: AEFIE CPU performance result (Heuristic Error Control)
Figure 15: AEFIE CPU performance result (Error Control)
Figure 16: MFIE error result (Heuristic Error Control)
Figure 17: MFIE error result (Error Control)
Figure 18: MFIE CPU performance result (Heuristic Error Control)
Figure 19: MFIE CPU performance result (Error Control)
Figure 20: VIE error result (Heuristic Error Control)
Figure 21: VIE error result (Error Control)
Figure 22: VIE CPU performance result (Heuristic Error Control)
Figure 23: VIE CPU performance result (Error Control)
Figure 24: EFIE error result (Heuristic Error Control)
Figure 25: EFIE error result (Error Control)
Figure 26: EFIE CPU performance result (Heuristic Error Control)
Figure 27: EFIE CPU performance result (Error Control)
Chapter 4. Efficient Memory Management

Unlike iterative methods, direct solution methods perform factorizations to solve the linear system. The factorizations roughly demand the memory space with the scales of $O(N^{1.5})$ depending on which method is used. The extra memory usages may restrict the maximum dimension of the linear system, which can be solved, due to the memory shortage on the system. The CEM problems often include the large size linear system problems and it leads the increasing demands for the memory space coupled with the advanced modern computer technology. The actual computer systems still have the restricted amount of the memory space, which stores the operating system (OS), the program codes, the data and etc. This study is about the efficient memory operation using mass storage devices (esp. hard drives), which can hold large amounts of data, with slow latency being the primary drawback as compared to RAM. The following sections explain how the algorithms of the LOGOS factorization can be reorganized to overcome the negative aspect of the external storage.

4.1. Background

If the linear system has the dense structured matrix, one of the basic linear system solution methods is the LU factorization method of the system matrix, and take the inverse from the results such that

$$Z = LU$$

$$Z^{-1} = U^{-1} \cdot L^{-1}$$

The $U$ is the upper triangular matrix and the $L$ is the lower triangular matrix. The factorization can be stored in a matrix with the identical dimension of the system matrix, $Z$, sharing the empty space in the matrix of each other. As the number of the unknowns increases with respect of $N$, then the memory demand for the factorization also follow the complexity of $O(N^2)$. The sparse matrix allows more efficient algorithm referred as the sparse LU factorization method. The sparse pattern structures are the core factors for the efficiency of the algorithm. The system matrices with the sparse patterns are the
character of the numerically projected problem with the finite element methods (FEM). The programming libraries for the sparse matrix problems are released for the public uses. The examples are SuperLU [16] and UMFPACK [17]. These solution methods have reduced memory complexity than the basic LU factorization, and it becomes about $O(N^{1.5})$. The improved complexity is still considered excessive, and demands of the system memory increase at the rate. The LOGOS framework is developed by Adams et. al. [58] as the fast direct solver based on the LOGOS modes. The LOGOS procedure is described in the Chapter 2 in detail. The factorization of the LOGOS method also causes the memory demands as other direct methods. If the required memory space is more than the available memory capacity, then there can be trouble in completing the computational work.

The LOGOS framework is designed to use the main memory system like other computational libraries. The memory components are built with the random access memory (RAM) on the computers. As the claims for the computational resources grow, the modern computer systems are equipped with the large capacity memories. Nevertheless, the large size numerical solutions ask even larger memory spaces. This requisition can cause out-of-memory trouble during program run time. To resolve this trouble, it is desirable to incorporate other memory resources. External memory algorithms, or out-of-core (OOC), are algorithms that are designed to treat computer data which is too large to fit into the main memory at one time [65]. In this dissertation, the OOC operation of the LOGOS solution is studied. At first, the memory hierarchy of the computer systems is demonstrated. Next, the basic factors of the memories are reviewed, and then the LOGOS strategy for the OOC is built based on the property of the device.

4.2. Memory Hierarchy

The memory hierarchy provides an economical solution to the claim for the unlimited amounts of fast memory [60]. The hierarchy is developed to build the faster and cheaper (spacious) memory system. Because the faster device is more expensive, the hierarchy consists of the several levels in it. Each levels are faster, smaller and more expensive than the next levels. The contents in each level can be the copy of the subset of the lower levels. The fastest memory, the register in the CPU, is located at the top level of
the structure, while the external device is at the bottom level. The external storage device is connected to the computer system through the IO bus. Each level has a different capacity and speed due to the memory device performance properties and interfaces. The memory system built in the hierarchy is operational based on the principle of locality, which supposes that the more recently used data is more likely to be used again. The principle makes the faster memory device (cache) keep the data expecting the more chance to be referred on coming computation [66]. By loading the more expected data in the faster device from the slower, more spacious memory enables the most economic memory hierarchy. Figure 28 illustrates a simple memory hierarchy structure with five levels.

Like other application programs, LOGOS applications are loaded into the main memory and operates on data in the main memory. The main memory is built with the dynamic RAM (DRAM) and connected to the CPU via the memory bus. The memory is located at the level of the devices between the caches and the external storage. Because the unlimited demands for the memory space for the data referred by the programs, the data operation technique might be claimed to treat the data sets which does not fit into the main memory. We have to look to the external storage device, which is generally equipped with the magnetic devices of the mass capacity for the data and programs. Even though the components far from containing the unlimited capacity, they are still much more spacious than the main memory. As known from the Figure 28, the lower level of the hierarchy is very slow device for the performance and the IO bus.

In this research, to support the main memory system, the most popular mass storage device, the HDD, is chosen, and the operation on the component is to be described for the LOGOS factorization. For the asymptotically efficient LOGOS solution method, the effect from the algorithm implemented with the component which may change the performance property significantly should be investigated to improve the system performance property. The disk drives have mechanical parts that impose certain device properties. The background of the HDD is reviewed to design the operation of the LOGOS solution method appropriately.
4.3. The Hard Disk Drive

4.3.1. Hardware Properties

Unlike other memory components located at the higher level in the memory hierarchy, the HDD has the mechanical components, which is inevitably slower than the electric components. The speed of the SDRAM (DDR3), which is recently popular for the PCs, are measured with a time unit of nano-seconds \(10^{-9} \text{ sec.}\), while the magnetic drives have a latency on the milli-seconds \(10^{-3} \text{ sec.}\) scale. Although there exist various HDD devices and main memory components with different speed properties, the measuring units show the rough outline of the performance scales between the device groups.

Three factors to decide the access time of the HDD to the data are the seek time, the rotational latency and the data transfer time. Figure 29 shows the logical structure on a single plate of the disk. The arrow in Figure 29(a) over the concentric circles points into the direction of track number increasing. In the case of the multi-platter HDD, the collection of the same track on the disk presents the cylinder, and the tracks on the identical disks are accessed at the same time. The disk in Figure 29(b) is divided into 8 slices and each section on the track is the sector of the disk. The data on the disk is accessed by the head assembly attached on the end of the arm moving along the radius of the concentric circles. The time for the arm to reach at the track which contains the data is the seek time. The data containing sector is reached by the disk rotation, and the time to take to the sector with the data is the rotational latency. The time from the data from the disk to the target memory becomes the data transfer rate. These three factors are combined to represent the total data access time from the HDD. The outline of the disk latency computation follows the description in \[66\]. With the time factors of the magnetic drive, the total access time can be outlined like:

\[
T_{\text{Total}} = T_{\text{seek}} + T_{\text{rotation}} + T_{\text{data transfer}}
\]  

(4.3)

Another thing which should be considered is that the mechanical parts depend on the moving distance from the current head location to the place where the data exist on the disk surface. The electric memory components have uniform access latency to any
address of the data. As a result, the access time to the magnetic drive may vary based on the travel time of the moving parts. The seek time described in the specification of the HDD product measures the relocation time for the head assembly located at the track 0 to the track at opposite end of the disk. By considering the all possible seek distance among the tracks, the average seek time of the head is supposed to be the one third of the seek time of the HDD product [67]. The rotational latency is described with the unit of the rotation per minutes (RPM), and the reciprocal value multiplied by the rotations is the rotational latency. Like the case of the relocation of between tracks, the average rotation time is assumed as the half of the full rotation time [66]. Besides the description of the moving parts latency, there are other detailed factors like the acceleration time of the action of the parts due to their mechanical characters, but they are not considered here for the simple outline.

4.3.2. Data Operation on the Disk

The logical HDD operations in the program do not represent the data access on the device at the physical level. Because the overhead of the mechanical device control is significant, the observation of the disk I/O at the device level may help the latency of the program performing the external I/O storage.

The data on the disk is stored in the form of the file. An operating system (OS) file manager treats the collection of the single unit of the data set to reduce the overhead. The bundled data groups are the contiguous records, and it can improve the efficiency of the drive latency by decreased locating distance of the head unit. For example, the cluster consists of the number of the sectors of the disk. The larger size of the data block may waste the capacity of the disk if the block has the fixed size, and the data does not fit in the block exactly. The variable size of the block leads to avoiding the sector fragmentation problem, but it can increase the track fragmentation. For the drive operations with large capacity, the operating system treats the data in the form of the multiple records. [68]

The data I/O with the file on the disk is operated in two kinds of way. The random access file and the sequential file inputs and outputs. The random files allow the data input and output at any point of the file, while the sequential files treat the data as the
stream input and output. The former type is suited to the case for the small pieces of records I/O like the database system. The latter one is for the large size data in one file, and the similar case of the streaming is the multimedia service through the internet. If the sequential file size is large, the writing procedure tries to keep each record being contiguous as possible. The random access file also depends on the head travel distance, but if the moving parts latencies are assumed to be the average seek time and the average rotational latency, then the performance computing includes the seek time and the rotational time for every record sets. As the result, the same total amount of the data amount is read/written with the overhead for the moving parts as many as the number of the data fractions. The sequential files, however, contain the data located consecutively from the previous record, to reduce the overhead by the head assembly movements. [67, 68] In the case that the random access data size is transferred in a large size of the block, for the data block alone the sequential effect can be expected. So the performance of the random file, it is important to keep the size of the random data pieces as larger as possible to have the contiguous allocation result from the HDD. [67] The performance results for the cases of the random access file and the sequential file are presented in [67, 68].

For the fast HDD I/O with the LOGOS solution method, it is known that the larger data set per an I/O transfer as described above. Based on the logical and physical properties of the file I/O, the strategy for the LOGOS to use the magnetic drive as the external mass storage is explained in the following section.

4.4. The Data page structure for the Out-Of-Core (OOC) LOGOS

The core component of the LOGOS solution method is the localizing procedure into the target field from the source group range. The solution procedure is performed based on the multi-level structure, and at a level, the localized field group is independent from other group process. Until the end of the factorization at the level, the localization at a group does not have any impact from the results from other groups. Even after the system is factored, the post-processing for the next level can be performed independently for each group. This property of the procedure leads to the idea that the whole data set does not have to be loaded into the main memory. So it can be the spatial efficient in the main memory to remove the current unused data and to load the data needed by the undergoing
process. The discarded data from the main memory is written into the more spacious storage, the HDD.

This kind of the data storage operation imitates the cache memory operation to have the more hit ratio for the faster memory I/O. Generally, the cache operation takes advantage of the principle of locality. As explained in [66], there are two types of the locality, the temporal locality and the spatial locality. The locality in time is the recently referred data is more likely to be referred soon, and the locality in space is the data located nearby is more likely to be referred in short time. The cache memory is filled with the data from the lower level, main memory in a larger data block which contains the address the program tries to read or to write. The larger block means the spatial locality, because it includes the data in the neighbor address. The data loading the small, expensive and fast memory component, cache, expects the temporal locality. The similar effect of the main-cache memory can be applied to the relation between the main memory and the HDD.

Unlike the cache operation for the general applications, the LOGOS procedure can predict the schedule of the groups by the specified solution algorithm. The order of the factorization for groups is even free to be changed before the groups processed. So we can apply the spatial locality to hit all the localization target groups in the data block, called the data page of the matrix. Once a specific data page is loaded for the groups to be localized in the page, the data stay in the main memory until whole factorization process is completed. At the same level, the page once used is loaded again during the post factorization for the next level to be projected on to the space that consists of the neighbor groups outside of the page. The complete procedure is done with the minimum number of the page loading, and it can be considered that the temporal locality is satisfied with the data page based on the factoring schedule.

It has been reviewed that the LOGOS algorithm provides the memory operation component with the exact order to be processed. Then while a page data for a matrix exists in the main memory, it is possible to hit all data for the localization and any other additional page information is required during the factoring. It means that no other factorization needs the information in the page loaded. The matrix data in a page is used only once in the localization, and during the post factorization. From the reason that the
only the group data is claimed in a page procedure, the memory operation does not need to load the multiple pages of a matrix data at once. By the scheduling the order of the localization, the hit rate of the locality of both the time and the space keeps higher not to need any more reloading for the page until the next post factorization or the next level procedure. So only single data page of a matrix is operational from the main memory for the computation, and the fact helps to maintain the smaller memory space for the computation.

Now the memory-storage relation and the operation background is reviewed, the algorithm for scheduling strategy description follows. The procedure explains the specific LOGOS algorithm, the SG-LOGOS to apply the procedure based detail of the coding for the data management from a matrix.

4.4.1. The OL-LOGOS Factorization with the Matrix Data Page

The OL-LOGOS solution method has the improved performance rate with respect of the asymptotic curve along the number of the unknowns. However, it costs a lot to use the magnetic drives, then the latency of the external device can dominate the performance time of the programs, because the access time to the data on the HDD is significantly longer than the memory I/O latency. The OL-LOGOS performance can become slower with the implementation of the magnetic drive as the sub-storage device. This research is about the efficient algorithm minimizing the overhead of the external storage access time.

The properties of the HDD are mentioned before, and it is important to set up the strategy to use the performance details in the OL-LOGOS solution procedure. To extract the fastest access from the external storage, the size of the data block has to be large. Even though it would have the fast access that the data block contains the large amount of the matrix data, the block sized must be reasonable. If the data capacity of the block is too large, the matrix is mostly filled from the data, and the main memory demand increases from the matrices with the data block.

4.4.2. The data page structure

The OL-LOGOS solution consists of the factorization and the projection onto the localized space which have the process with respect to the independent groups to which
the localization is performed. Because each group can be factored without considering any result from other groups, the collection of the field groups can be set arbitrary. So the groups contained in a data block on the disk can be categorized into a data page of the matrix. The system matrix has the block sparse format with the compressed sparse column (CSC) format. The structure is built as the column major data block access, and the columns of the matrix include the matrix block along the row. Rather than to choose any individual matrix block, it is more efficient to take whole rows of a column to be decided as the member of the page. So the data page consists of the columns of the block sparse matrix (BSC) corresponding to the selected groups. A column of a BSC means the source group of the field in the system matrix. If the data page range is set for the general case of the OL-LOGOS, the overlapping sources are all coupled through the level. If any of the source range is not in the page, the data block on the disk should be loaded to change the previous page existing in the main memory. If the missing range contains multiple groups in the separate page, then all missing data should be read from the multiple files. The page should be set to contain large amount of the data, and although the data is stored in the contiguous location on the disk, the transfer rate itself can be the waste of the time because the block may include the column data which is not required at the moment. The unused data read into the memory must be discarded from the memory must be called for the other time it is needed, and it causes the multiple load with respect to a single page. So all coupled property of the system matrix makes the storage manage algorithm very complex for the efficient operation.

Out of the OL-LOGOS, the SG-LOGOS has the decoupled source range up to some level. The separated structure can accommodate the simple data page setup for the OL-LOGOS factorization. If the shifted groups are all in one data page, the field groups that coincide with the source groups only require the block matrix in the page. The localization only claims the data from the single data page, and the single page operation for any of the field groups leads the overall storage operation. The single access to a data page restrict the number of the data block loading from the HDD which has the slow I/O speed. By choosing the SG-LOGOS as the source range strategy of the OL-LOGOS, the number of the reading and writing times can be minimized to reduce the disk access latency. Considering 4.3, the bottom line of the efficient data page structure with the
external storage operation has two core issues to be achieved. The first one is that the source groups under the shifted parent should be in one data page for the factorization to be performed in the single page. This can reduce the I/O time by keeping the data in the data from multiple loading. The next issue is to make page size larger up to the size which does not demand the memory capacity too much. The larger data block leads the contiguous location of the data on the storage to reduce the latency from the mechanical parts of the storage. The first one is the access time saving solution at the logical level of the program as the data structure to be treated by the algorithm, while the bigger data page is going to be effective on the hardware level. By combining both of the solutions for the SG-LOGOS with the HDD storage support, the groups under the multiple shifted parent group in the lower level is collected in one data page, which is treated as a data block on the surface of the magnetic drive.

To get the most efficient result from the data structure plan for the SG-LOGOS solution method, the algorithm of the factorization is modified. Because of the independent localization of each group, the factorization is free from the group order to be referred as the target field. Following the data page structure, the groups under a shifted parent is in one group. Including the groups under the multiple shifted parents the data block on the disk have the big enough size. So the order of finding the localizing modes is rearranged to treat all the members of the group in the page. The SG-LOGOS does not require any source data outside of the page to find the LOGOS modes of the groups under the page. So far the number of the shifted parents to become the member of the same data page still remains to be decided. Considering the memory operation and the size of the data of a page, the shifted groups referring the parent group of the parent groups. This means that the groups under the same grand parent groups are chosen. The range of the groups is shifted to cover all the shifted sibling groups. In the Figure 30, the sky blue region is the groups sharing the grandparent group at two lower level. The groups covered by the pink square are the shifted page from the grandparent level. Up to the grandparent level, the groups in the shifted range are in one data page, and this is the efficient data page for the factorization to restrict the disk I/O time. For the one level higher than the stop level, which does not have the tree structure at two level coarser
level, it just takes the parent level as the data page like the Figure 31, in which the blue and the pink area has the same meaning as the grandparent level case.

Now the data page is set for finding the localizing modes from the system matrix. However, the shifted range is the efficiency for the factorization only. The system matrix for the next level is computed by projecting the localized field onto the projection matrix. The procedure is not based on the shifted parents, but on the regular parents for which the children groups (all siblings of each other) are unified to build the parent group. In the Figure 30 and Figure 31, there are the regions which is only in the sky blue region, to be excluded from the shifted parent region. So the post factorization process needs the complex algorithm to operate the page I/O efficiently for the parent level projections. The multiple data page reading or writing may hurt the asymptotic efficiency of the OL-LOGOS even though the best performance can be achieved from that kind of the operations. So, to make the algorithm simple and efficient for the performance time, it is decided to include the only blue region is also included into the page. This kind of the data page structure allow the only one access for both the factorization and the projection of the localized system matrix.

By putting the all groups sharing the grandparent and the shifted grandparent into one data page, the number of the page referred for the fields about which the localizing modes to be found, and the parent group built from the factorization becomes unique. The coincidence of the groups from different kind of the grandparent causes the issue with the data page structure. From the Figure 30 and Figure 31, the groups under the pink square only, which share the shifted parent with the ones covered by the sky blue part as well, belong to the other data pages.

4.4.3. The data page operating algorithm

It has been studied how to implement the data structure of the page for the efficient operation by the program. With the data page, it is required that the algorithm to manage the data structure between the main memory and the HDD.

The LOGOS solution method claims the matrix block from the block sparse matrix in the code. The LOGOS library get the pointer to the matrix block for the matrix operation and then release it whether it is modified or not. So the block matrix requesting
and the releasing subroutines are good point for the code managing the data page for the external storage support. The data claim is designed to be executed in the main memory, and the data loading managing code from the storage to the memory is put in the matrix claim subroutine. The data loading procedure checks if the requested matrix block exists in the memory. If the request is not hit from the block sparse matrix, the data page exchange is performed. To make it sure that the modified or the generated matrix data is saved, the current page status is tested. If it has no changed matrix block, the page is simply removed from the memory and the new page is loaded. However, in the case that any of the matrix blocks got changed during the matrix operations, the page should be saved, and then deleted from the memory. The Figure 32 describes the algorithm of the outline of the matrix block request and release procedure in the flowchart form. The page exchange contains the saving and the deleting process, which need to have the more detailed illustration. The data page dumping procedure is explained below.

It is not effective way for the program running time that the other pages exclude the groups in other group and load the page for the information corresponding to the groups, because the way make the file I/O increase to the multiple times for the data pages. So when it comes to the overlapped groups referred, if the groups are modified or generated at a data page, then the dumping process check whether it is included in other page. If they are, the matrix data will stay in the main memory until the other page can save the modified information of the matrix. The other page is not modified, then the matrix data is to be removed, because it means that the page already stored the matrix. To achieve this, when the matrix block is modified, the regular and the shifted pages are all marked as the modified, and the dumping routine switch the flag when the corresponding page is saved. The detailed algorithm of the dumping page is illustrated in the Figure 33.

Once the data page is loaded into the main memory, the page contains the integer information table bearing the location of the data array of the page, and block matrix block dimension records. And a real type of array the data of the matrices belong to. If the individual matrices requests the data copied into their data area, the subroutine latency is going to be longer. So the flag variable in the logical type in the matrix blocks, and the data in each matrices point out the location of the combined data block. This way does not ask any memory copy and it saves the loading time of the individual block of
matrices. In the Figure 34, the allocatable variables in the matrices point the data in the unified data block. In the FORTRAN syntax, the one dimensional array can be converted to be referred by the two dimensional pointer which is defined in the matrix structure of the platform program. The matrix (N-1) is the only one modified from the program, and only it has the flag off meaning the modified block, and points the new data block rather than the data from the page. The modified matrices are treated in the dumping procedure as explained a paragraph ago.

The implementation of the data page structure for the OOC OL-LOGOS is set to have a single access for the factorization and the post-factorization with allowing the overlapping groups in the multiple pages, and the operation is explained. The algorithm is ready to perform at the fastest rate involving the external drives. In the algorithm, it is still not resolved that the groups in the multiple data pages stay until the overlapping data page is called or updated. Because the groups included in the multiple pages are outside groups of the data page. When the grandparent groups are the sharing group, the number of the groups in multiple pages formulation is like:

\[
2D: \frac{4^2 - 2^2}{4^2} = \frac{3}{4} \tag{4.4}
\]

\[
3D: \frac{4^3 - 2^3}{4^1} = \frac{7}{8} \tag{4.5}
\]

As seen from (4.4) and (4.5), in the case that the pages sharing the overlapped group is not updated as soon as possible, especially the 3D geometry is applied, the non-trivial amount of the memory space can be allocated by the matrix data of the groups pertained into the multiple pages. If the parent group is the sharing group at the working level, the rates goes worse than the examples shown above.

**4.4.4. The SG-LOGOS with OOC implementation**

With the data page structure and the algorithm, the SG-LOGOS performs in the OOC function activated for the block sparse matrix. The group list in the data page are defined based on the groups at a level where the factorization executes. The levels have the list of the page groups sharing the grandparent or the parent level relation as the way explained. If the grandparent groups are chosen as the working group page list, the
previous level of the stop level does not have the sharing group. So in the case, the page group is used to reduce the sharing level by 1. For the stop level, there is no lower level than that, the group can remain as the unique page. However, in some tight tolerance solution cases of the OL-LOGOS, the system matrix in the stop level to be solved in the traditional way could have the large amount of the data. Therefore, the groups are divided into the multiple groups. In the testing program, it is used to have 4 groups for the stop level page group list.

The page group lists at each level are defined to rearrange the factorization order for the efficient OOC function. The factorization loop is performed in the order of the groups which are under same data pages. At first the groups belonging to the first data page are processed, and the groups in the second page follows. This loop is going until the last data page is completed with the factorization. The group numbers given to the group when the oct-tree is generated is not the order of the factorization any more for this. This scheduling of the factorization order is optimized to use the data page with the highest hit rate for the factorization and the projection of the localized system.

4.5. Numerical results

As the test case of the SG-LOGOS with OOC, the magnetic field integral equation (MFIE) and the volume integral equation (VIE) are chosen. As in the error control part, the MFIE has the PEC scattering sphere as the geometry with the dimension of $\frac{4}{9} \lambda$ and 133.33 MHz frequency planewave is applied. The VIE has the relative permeability is also chosen to be $\varepsilon_r = 4$, and 1 Hz frequency plane wave is applied. The object is the dielectric sphere shell with the empty center sphere.

To compare the OOC performances, the SG-LOGOS operated in the main memory, the grandparents sharing page group list and the parents sharing page group list are all tested. The two kinds of the OOC data page strategies are tested to compare the effect of the block size for the CPU performance and the memory efficiency from both cases. The in-core (IC) performance results are provided to see how the OOC results are different from the IC, which has the OOC mode off. The tolerances of the tests are from $10^{-2}$ to $10^{-4}$. Along the tolerances, the CPU and the memory performances are compared. The
effect of the disk access time to the performances is presented by the data hit ratio, where
the less hit ratio expects the more frequent HDD I/O time. The memory data missing
happens at the point the data page is replaced at the end of the schedule of the data page.
During the scheduled factorization for the data page, all the data requests to the memory
are expected to be hit.

4.5.1. The SG-LOGOS with OOC for MFIE

For the MFIE test, the surface of the scatterer is discretized based on the oct-tree. The
surface is the 2D object, and the groups in the tree has the 2D relation with their
neighbor groups. The property of the groups in the tree is expected to let the less
overlapped groups stay in the main memory than the 3D relation.

The Figure 35, Figure 37 and Figure 39 represent the CPU performance with the
tolerances $10^{-2}$, $10^{-3}$ and $10^{-4}$ respectively. Even though the performance time level is
elevated due to the disk access time, the asymptotical rates of the grandparent group (GG)
based OOC have the close to the performance rate results of the IC case. Compared to the
GG-OOC, the parent group (PG) based OOC has the slower rate than the IC case and the
GG-OOC results. The rate change can be explained by the memory hit ratio which is
shown in the Figure 47. The GG-OOC keeps the hit ratio higher because the number of
the data page is less than that of the PG-OOC. For the same amount of the data, the larger
block size like the GG-OOC reduces the data page change time, and it leads to the less
number of the disk access. From the hit ratio plot, the number of the disk access time has
the influence on the CPU performance rate. The larger page capacity can keep the ratio of
the IC performance, while the time level is inevitably lifted from the non-OOC
performance.

The Figure 36, Figure 38 and Figure 40 are the plot for the memory performance
with the tolerances $10^{-2}$, $10^{-3}$ and $10^{-4}$ respectively. The plots show the level of the peak
memory while the SG-LOGOS is executing. Both the GG-OOC and the PG-OOC have
the less peak memory occupation status along the tolerances than the IC memory result.
This proves that the sub-storage operation reduces the main memory demand. By
comparing the two OOC cases, the memory claim rate is faster with the GG-OOC. The
overlapped group effect has the bigger influence with the PG-OOC which has more
groups belong to the multiple data pages. The GP-OOC memory performance shows the smaller memory capacity level, and the improved rated along the number of the unknowns in the plots.

As the result of the tests of the MFIE, the GG-OOC with the larger data page size has the as fast asymptotical rate as the IC CPU time, and the more improved memory demand rate than IC memory performances. The PG-OOC which operates the smaller size of the data page has the less memory claim than the IC, but still more memory demand than the GP-OOC results. The CPU rate of the PG-OOC shows the increased number of the magnetic drive slows down the asymptotic performance.

4.5.2. The SG-LOGOS with OOC for VIE

The VIE test cases has the dielectric object with the relative permittivity of 4 with the frequency 1Hz. The stable linear systems are different from the MFIE cases in the geometric relation among the neighbor groups. The object is discretized in the 3D structure based on the oct-tree. With the 3D neighbor connections, more overlapped edge groups are expected, and the tests shows how the effect makes the different results than the MFIE test cases.

The CPU performances are shown in the Figure 41, Figure 43 and Figure 45. Each of the plots represents the time rate with the tolerances $10^{-2}$, $10^{-3}$ and $10^{-4}$ respectively. Comparing the performing rates of both OOCs, the GG-OOC has the faster rates than the case of the PG-OOC. The Figure 48 represents the hit ratio of the VIE case, and also shows the PG-OOC has the more disk access time due to the missed memory reference times. With the same reason as the MFIE cases, the more frequent disk access of the PG-OOC make the rate curves steeper than the results of GG-OOC. It can be seen that the larger data page size still has the effect on the VIE performances with the performances. The GG-OOC time rate has also the increased level of the curve than the IC curve results. The asymptotic rates, however, can be considered close to the IC performance rates. The minimized disk access time makes the efficient CPU rates with all the tolerances, while the PG-OOC results have the slower asymptotic rate than the GG-OOC and the IC cases.

The Figure 42, Figure 44 and Figure 46 shows the memory efficiency curve with the same tolerances of the CPU performances. Both of the OOCs have the lower memory
capacity levels than the case of the IC memory occupations does. In the memory performances, the overlapped groups have the effect on the rates of the curve like the MIFE cases. The PG-OOC that expects more edge groups to be overlapped by the multiple data pages has the drawback in performing the efficient memory than the GG-OOC case. The GP-OOC has the less efficient memory performance curves through the plots than both the IC curves and the GG-OOC rates.

4.5.3. Summary of the Numerical Results

For the OOC operation of the SG-LOGOS, the MFIE and the VIE linear systems are tested with three tolerances. The performances of the CPU and the memory operations plots are shown as the results of the linear system solution methods.

From all the CPU performance plots, the effect of the disk access time can dominate the solution asymptotic rates. Even though the disk access time, which is measured with the different digit (ms) from the SDRAM case ($\mu$s), lifts up the CPU performance time level, the reduced number of the disk access times allow the closer asymptotic rate to the non-OOC curve in both of the cases of the MFIE and the VIE.

Like that the disk access time is important factor for the CPU performance with the OOC function, the number of the overlapped groups has the significant effect on the memory operations. The GG-OOC, which has the less number of the groups included in the multiple of the data pages shows a more efficient rate than the PG-OOC. Both of the OOCs have lower memory occupations for the same linear systems than the memory status of the IC method for the MFIE and the VIE. Due to the number of the overlapped groups between the 2D and 3D cases, the VIE memory performance curves of both OOCs are elevated closer to the IC than the case of the MFIE tests.

The data page structure is designed for the SG-LOGOS to be more efficient when the groups share the groups in the lower level, like the grandparent level and the parent level, and when the data page size is larger. Because the shifted structure of the page groups, the overlapped groups can increase the memory demands. The GG-OOC satisfies the conditions better than the PG-OOC, and the performances of both the CPU and the memory usage are more improved than the PG-OOC. Comparing to the IC results, the
memory performances have a relatively efficient rate, while the CPU times are asymptotically close to the IC speeds.
Figure 28: the simple memory hierarchy
Figure 29: The logical structure of the magnetic disk
(a) the arrow in the track direction
(b) the arrow in the section direction
Figure 30: the data page based on the grandparent level for the SG-LOGOS
Figure 31: the data page based on the parent level for the SG-LOGOS
To refer a block matrix in the block sparse

To get the matrix pointer from the block sparse

Mem. Hit?

Yes

The matrix block is ready to be referred

No

To find the file for the data page

To remove the page in the memory

File exists?

Yes

To load the page from the file

No

To create a file

To get the matrix pointer from the block sparse

Figure 32: the flowchart for the page change
Figure 33: the page removal flowchart
Figure 34: the matrix block data allocation from the data page and the modified block
Figure 35: MFIE CPU performance with the tolerance=10e-2
Figure 36: MFIE Peak Memory with the tolerance = 10e-2
Figure 37: MFIE CPU performance with the tolerance=10e-3
Figure 38: MFIE Peak Memory with the tolerance=10e-3
Figure 39: MFIE CPU performance with the tolerance=10e-4
Figure 40: MFIE Peak Memory with the tolerance=10e-4
Figure 41: VIE CPU performance with the tolerance=10e-2
Figure 42: VIE Peak Memory with the tolerance=10e-2
Figure 43: VIE CPU performance with the tolerance=10e-3
Figure 44: VIE Peak Memory with the tolerance=10e-3
Figure 45: VIE CPU performance with the tolerance=10e-4
Figure 46: VIE Peak Memory with the tolerance=10e-4
Figure 47: MFIE Hit Ratio
Figure 48: VIE Hit Ratio
Chapter 5. CONCLUSIONS

Two significant improvements to the SG-LOGOS algorithm have been presented. A rigorous error control procedure has been derived, implemented, and tested. Similarly, an out-of-core implementation of the SG-LOGOS algorithm has been developed and tested to improve the memory performance of the SG-LOGOS algorithm. Both of these developments can potentially slow the SG-LOGOS performance since they use additional resources to gain something (improved error control, and improved memory performance, respectively). However, it has been observed that the new implementations provide similar asymptotic performance as compared to the original SG-LOGOS algorithm for appropriately formulated surface and volume integral equations.

A rigorous error control method has been developed for the SG-LOGOS factorization, which is characterized by non-orthogonal (local and non-local) subspaces. The new formulation provides reliable error control even in cases for which previous versions of the SG-LOGOS factorization failed to control the relative RMS error. This is because the SG-LOGOS factorization without error control has the problem in the influence of the smaller singular values. The error control method developed here has provided reliable results for linear systems with both large and small condition numbers. For ill-conditioned systems, the error is controlled, but at the expense of significantly increased CPU time. For more well-conditioned linear systems, the additional costs of the error-control procedure are significantly less and have little impact on the asymptotic performance of the SG-LOGOS factorization algorithm. This occurs because fewer cycles through the mode truncation loop are required. These results emphasize the importance of using well-conditioned formulations with SG-LOGOS solvers.

The out-of-core operation data page structure used herein is designed to reduce the data access time in the hardware side and the number of disk I/O operations on the software side. The latter is achieved by rescheduling the SG-LOGOS factorization order so as to extract the best result from the data page structure. The performance of the OOC algorithm with reduced access time is closer to the asymptotic performance to the non-OOC SG-LOGOS. The memory demand from the OOC is less than the IC cases, and the memory claims increase at a lower rate than the IC results.
5.1. Contributions

The following items are the non-trivial contributions to the SG-LOGOS solution method included in this dissertation.

- A rigorous error control method based on the SG-LOGOS has been developed. The method has been tested for several different surface and volume integral equations, including examples that range from poorly conditioned to well-conditioned. Importantly, the error control algorithm developed herein does not increase the asymptotic computational complexities of the SG-LOGOS factorization algorithm for well-conditioned system matrices.

- The SG-LOGOS algorithm has been re-formulated (re-organized) for use with data storage support from the HDD. The data structure has been designed to be efficient based on the hardware side and the software side. To use the data structure efficiently, the factorization order is rescheduled to improve the hit ratio of the data staying in the main memory. Numerical examples have been provided to demonstrate the efficiency of the proposed techniques.

5.2. Future Work

- Additional testing of the proposed algorithms is possible for a variety of integral equation formulations. For example, it would be interesting to consider the performance of the proposed algorithms for volume integral equations with high material contrast values. Similarly, performance for dielectric surface integral equations remains to be verified.

- The test cases are all based on the integral equation method systems in this dissertations. The methods, however, can be applied to other systems that can be solved with the LOGOS factorization like the finite element method.

- A unique feature of the LOGOS solution framework is that it provides an opportunity for module solution methods via solution locality. It may be
possible to extend the error control and OOC strategies developed herein to similarly treat such module analysis and design problems.
REFERENCES


Vita

Jun-shik Choi

Education

- M.S. in Electrical and Computer Engineering, University of Wisconsin – Madison, 2002-2005
- B.S. in Electrical and Computer Engineering, University of Kentucky, 1999-2002
- B.S. in Civil Engineering, Yonsei University, Seoul, Republic of Korea, 1991-1995

Publications