

MULTIPLE-PRECISION MLFMA FOR EFFICIENT AND ACCURATE SOLUTIONS OF BROADBAND ELECTROMAGNETIC PROBLEMS

A DISSERTATION SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

By
Mert Kalfa
August 2020

MULTIPLE-PRECISION MLFMA FOR EFFICIENT AND ACCU-
RATE SOLUTIONS OF BROADBAND ELECTROMAGNETIC
PROBLEMS

By Mert Kalfa

August 2020

We certify that we have read this dissertation and that in our opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Vakur B. Ertürk (Advisor)

Özgür Ergül (Co-Advisor)

Sencer Koç

Orhan Arıkan

Ergin Atalar

Lale Alatan

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

MULTIPLE-PRECISION MLFMA FOR EFFICIENT AND ACCURATE SOLUTIONS OF BROADBAND ELECTROMAGNETIC PROBLEMS

Mert Kalfa

Ph.D. in Electrical and Electronics Engineering

Advisor: Vakur B. Ertürk

Co-Advisor: Özgür Ergül

August 2020

The multilevel fast multipole algorithm (MLFMA) is a popular full-wave electromagnetic solver that enables the solution of electrically large problems with an extremely large number of unknowns. As with all computational electromagnetics solvers, active research is ongoing to extend the limitations of MLFMA for larger problems with finer geometrical details. For electrically small structures MLFMA suffers from the low-frequency breakdown, while more efficient schemes are required for electrically larger problems.

We propose and demonstrate an elegant solution to the aforementioned problems by introducing a multiple-precision arithmetic (MPA) framework to the inherent hierarchical tree structure of MLFMA, dubbed the multiple-precision multilevel fast multipole algorithm (MP-MLFMA). With the introduction of MP-MLFMA we show that a distinct machine precision can be assigned to each level of the tree structure of MLFMA, which enables accurate and efficient solutions of problems with deep tree-structures over arbitrarily large frequency bandwidths. To determine the required machine precisions for a given tree structure, as well as the number of harmonics required for an accurate error control of the translation operator of MP-MLFMA, we introduce and validate a novel error control scheme with accurate design curves that is valid at all frequencies, for the first time in the literature. Combined with the proposed error control scheme, we present the capabilities of MP-MLFMA over a wide range of broadband and deep tree-structure scattering problems. We also illustrate the true potential efficiency of MP-MLFMA, with a simple MPA framework implementation on a single-precision processor. With the hardware-defined implementation, we show the super-linear speed-up potential of MP-MLFMA for low-precisions.

Keywords: multilevel fast multipole algorithm, error analysis, multiple-precision arithmetic.

ÖZET

GENİŞBANT ELEKTROMANYETİK PROBLEMLERİN VERİMLİ VE İSABETLİ ÇÖZÜMÜ İÇİN ÇOKLU-HASSASİYETLİ ÇOK SEVİYELİ HIZLI ÇOKKUTUP YÖNTEMİ

Mert Kalfa

Elektrik-Elektronik Mühendisliği, Doktora

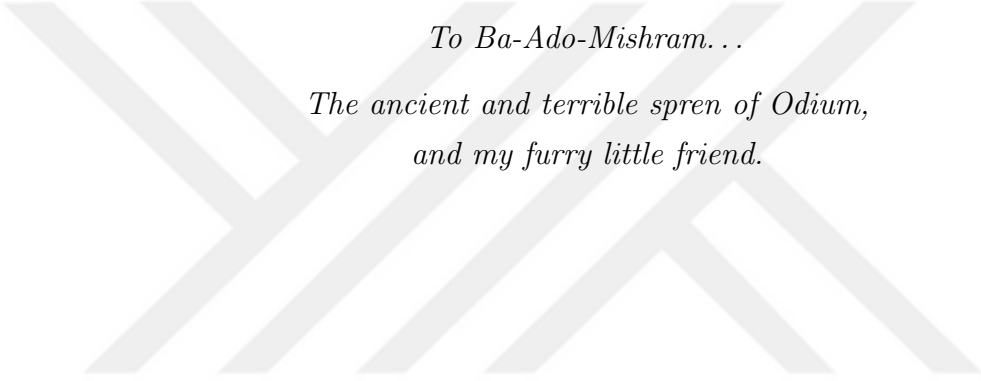
Tez Danışmanı: Vakur B. Ertürk

İkinci Tez Danışmanı: Özgür Ergül

Ağustos 2020

Çok seviyeli hızlı çokkutup yöntemi (ÇSHÇY), elektriksel boyutu yüksek ve yüksek sayıda bilinmeyen içeren elektromanyetik problemlerin çözümünde kullanılan popüler bir tam-dalga çözümleyicisidir. Diğer hesaplamalı elektromanyetik çözümleyicilerde olduğu gibi ÇSHÇY'nin daha büyük ve daha ince detaylı problemlere genişletilmesi konusunda araştırmalar devam etmektedir. Bu araştırmalar elektriksel boyutu düşük problemlerde düşük-frekans bozulmasının giderilmesine, büyük problemlerde ise verimin artırılmasına odaklanmaktadır.

Yukarıda listelenen problemlerin her ikisini de adresleyen ve ÇSHÇY'nin ağaç yapısına entegre bir çoklu-hassasiyetli aritmetik (ÇA) yapısı önerilerek doğrulanmıştır. Önerilen çoklu-hassasiyetli ÇSHÇY (ÇH-ÇSHÇY) ile ağaç yapısının her seviyesinde farklı bir makine hassasiyeti tanımlanarak derin ağaç yapısında problemlerin istenen frekans bant genişliklerinde isabetli ve verimli çözümü mümkün olmaktadır. Verilen bir ağaç yapısı için gerekli makine hassasiyetlerinin ve kaydırma operatörünün isabetli hata kontrolü için gerekli harmonik sayılarının belirlenmesi için, özgün bir hata kontrol yöntemi geliştirilmiş ve doğrulanmıştır. Önerilen hata kontrol yöntemi, literatürdeki mevcut çalışmalardan farklı olarak tüm frekanslarda isabetli sonuç vermektedir. Önerilen hata kontrol yöntemi ile birlikte ÇH-ÇSHÇY'nin kabiliyetleri, çok sayıda genişbant ve derin ağaç yapılı saçılım problemlerinde gösterilmiştir. ÇH-ÇSHÇY'nin verimlilik potansiyelini gösterebilmek için tekli-hassasiyetli bir işlemcide ÇA yapısı donanım seviyesinde uygulanmış, ÇH-ÇSHÇY'nin özellikle düşük hassasiyetli problemlerde doğrusal-üstü hızlanma potansiyeli gösterilmiştir. *Anahtar sözcükler:* çok-seviyeli hızlı çokkutup algoritması, hata analizi, çoklu-hassasiyetli aritmetik.



To Ba-Ado-Mishram...

*The ancient and terrible spren of Odium,
and my furry little friend.*

Acknowledgement

This dissertation is the culmination of four years of hard work, during undoubtedly the most tumultuous period of my personal life. Although it was very hard—and surprisingly rewarding—I have pulled through my ordeals physically and mentally with the help of my friends and family. I consider this dissertation to be the crowning achievement of my personal growth in the last four years, and it would never see the light of the day—at least in its best possible form—without the help and support of a lot of people. So, thanks are in order.

First and foremost, I would like to express my sincerest gratitude to my advisor Prof. Vakur B. Ertürk for his valuable friendship and incredible support throughout almost all of my adult life. I thank him for his invaluable comments and advice on my work, which elevated this dissertation to its best possible version that you are reading now. I would also like to extend my gratitude to my co-advisor Assoc. Prof. Özgür Ergül, who is second to none on MLFMA research, and who always gave great feedback that kept this work on track.

My sincere thanks also goes to the Thesis Monitoring Committee members Prof. Sencer Koç, Prof. Ayhan Altıntaş, and Prof. Orhan Arıkan for their extremely helpful insights and comments throughout my Ph.D. studies. I am grateful for the additional examination committee members Prof. Ergin Atalar, and Prof. Lale Alatan for showing interest in my studies and accepting to read and review my dissertation.

I would like to thank ASELSAN Inc., my employer for the last decade, for allowing me to conduct my research, and giving me permission to use the available workstations for some of the simulations you will see in this work. I would also like to thank Halil Topözlü from ASELSAN, for his help on the assembly coding of an x86 processor.

Last but not least, I am eternally grateful for my lovely family. I know most people consider their family to be the best; but mine really are, so there! I would like to express my best gratitude to the *Sister of Aes Sedai* Merve, the *Mother of Dragons* Emine, and the *Stormfather* Ahmet for their endless support and encouragement when I most needed.

Life before death, strength before weakness, journey before destination...

Contents

1	Introduction	1
2	Multilevel Fast Multipole Algorithm	7
2.1	Introduction to MLFMA	7
2.2	Limitations of MLFMA in Fixed-Precision Computers	12
2.3	Proof-of-Concept for a Multiple-Precision Implementation of MLFMA	16
3	Error Control of MLFMA	23
3.1	Previous State-of-the-Art on Error Control	24
3.2	Proposed Error Control Scheme	25
3.2.1	Estimating the Optimum Truncation Number	25
3.2.2	Estimating the Optimum Machine Precision	28
3.2.3	Validation of the Proposed Error Control Scheme	31
3.3	Simplified Expressions for Universal Error Control of MLFMA . .	38
4	Multiple-Precision Multilevel Fast Multipole Algorithm (MP- MLFMA)	44
4.1	MP-MLFMA Notation	45
4.2	Proposed Implementation of MP-MLFMA	47
4.2.1	Error Control of MP-MLFMA	47
4.2.2	Aggregation	49
4.2.3	Translation	49
4.2.4	Disaggregation	50
4.3	Discussions on a Software-Defined MP-MLFMA	50
4.4	Discussions on a Hardware-Defined MP-MLFMA	56

4.5	Hardware-Defined MPA for Addition & Multiplication	57
5	Numerical Results	60
5.1	Error Analysis of MLFMA	60
5.1.1	Design Curves for Error Control of MLMFA	61
5.1.2	Validation of the Proposed Design Curves	65
5.2	Validation of MP-MLFMA	68
5.2.1	Preliminary MP-MLFMA Analyses	68
5.2.2	Broadband MP-MLFMA Solutions	71
5.2.3	MP-MLFMA Solutions for Deep Tree-Structures	76
6	Conclusion	80
A	Floating-Point Arithmetic Standard	89
B	Asymptotic Expressions for Bessel and Hankel Functions	92
B.1	Large Argument Approximations	93
B.2	Small Argument Approximations	94
B.3	Large Order Approximations	94
C	Discussion on Legendre Polynomials	95

List of Figures

2.1	Recursive division of the tree structure (Source: Ergül [5, 52]) . .	9
2.2	One-box-buffer Scheme (Source: Ergül [5, 52])	10
2.3	Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda/512$	13
2.4	Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda/128$	14
2.5	Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda/4$	15
2.6	Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda$	15
2.7	Implemented MVM Scenario	18
2.8	Relative Errors for Electrically Small Boxes for $a_1 = \lambda/128$	18
2.9	Relative Errors for Electrically Small Boxes for $a_1 = \lambda/512$	19
2.10	Relative Errors for Electrically Large Boxes: $(N, a_1) = (5, \lambda/4)$, with translation @ 2λ	20
2.11	Relative Errors for Electrically Large Boxes: $(N, a_1) = (3, 2\lambda)$, with translation @ 8λ	21
2.12	Single-Level translation model for the robustness analysis.	22
2.13	Robustness analysis showing the relative error with respect to the free-space Green's function over 50 Monte Carlo iterations.	22
3.1	Critical points for the shift vectors shown on source and observa- tion boxes in a one-box-buffer scheme. Corners and edge centers are shown with filled circles and face centers are shown as circles.	26

3.2	Relative error estimate ($\hat{\epsilon}$) for $a = \lambda/32$, $\epsilon_d = 10^{-3}$, $\vec{w} = [0 \ 2a \ 0]^T$, $\vec{v} = \vec{v}_1 + \vec{v}_2 = [a \ a \ a]^T$	28
3.3	Numerical integration weights of uniform and Gauss-Legendre sampling for different truncation numbers τ	30
3.4	Relative errors with respect to free-space Green's function when Table 3.1 is used in an MPA environment for $\vec{w} = [0 \ 2a \ 0]^T$. Dashed lines represent the desired relative error thresholds.	32
3.5	Relative errors with respect to free-space Green's function when Table 3.2 is used in an MPA environment for $\vec{w} = [3a \ 3a \ 3a]^T$. Dashed lines represent the desired relative error thresholds.	33
3.6	Comparison of the truncation numbers found by the proposed scheme to [12–14] and [27] for $\epsilon_d = 10^{-2}$ and $\vec{w} = [0 \ 2a \ 0]^T$	34
3.7	Comparison of the machine precisions found by the proposed scheme to double precision and [27] for $\epsilon_d = 10^{-2}$ and $\vec{w} = [0 \ 2a \ 0]^T$	35
3.8	Truncation numbers obtained via (3.13) for $a = \lambda/2048$	40
3.9	Relative error of truncation numbers obtained via EBF [12–14], (3.26), and (3.28) for different box sizes and translation distances.	41
3.10	RMSE of truncation numbers obtained via EBF [12–14], (3.26), and (3.28) for different box sizes and translation distances.	42
3.11	Relative error of truncation numbers obtained via EBF [12–14], (3.27), and (3.31) for different box sizes for a one-box-buffer scheme.	43
3.12	RMSE of truncation numbers obtained via EBF [12–14], (3.27), and (3.31) for different box sizes for a one-box-buffer scheme.	43
4.1	An example of the recursive division into cubic boxes at $N = 5$. (Source: Ergül [52])	46
4.2	Memory usage (in bytes) for a single matrix element at different machine precisions, using [54].	51
4.3	Computation time (in miliseconds) for a 1000×1000 MVM at different machine precisions averaged across 100 trials, using [54].	52
4.4	Comparison of CPU-times, allocated memory, and achieved rela- tive errors for varying box sizes when the truncation numbers in Table 3.1 is used for $\epsilon_d = 10^{-2}$ with double precision and MPA (averaged over 10 runs).	54

4.5	Comparison of CPU-times, allocated memory, and achieved relative errors for varying box sizes when the truncation numbers in Table 3.1 is used for $\epsilon_d = 10^{-5}$ with double precision and MPA (averaged over 10 runs).	55
4.6	CPU-times, allocated memory, and achieved relative errors for $a = \lambda/2048$, $\epsilon_d = 10^{-5}$, $\tau_{opt} = 66$, using MPA at different machine precisions (averaged over 10 runs).	55
4.7	Parallel addition architecture for low-precision operations. W, R corresponds to write and read operations, respectively.	58
4.8	Parallel multiplication architecture for low-precision operations. W, R corresponds to write and read operations, respectively. . . .	59
4.9	Average addition/multiplication times in a hardware-level MPA implementation on an x86 platform	59
5.1	Minimum achievable errors versus the box size in a one-box-buffer scheme (i.e., $w = \vec{w} = 2a$).	62
5.2	Minimum box size versus the machine precision to achieve different desired error thresholds for $w = \vec{w} = 2a$, i.e., the one-box-buffer scheme.	63
5.3	Minimum box size versus the machine precision for different translation vectors to achieve a desired error of $\epsilon_{min} = 10^{-3}$	64
5.4	Scattering from a PEC sphere of $\lambda/4$ diameter with a $1V/m$ plane-wave at single-precision. Azimuth angles of 0° and 360° correspond to backscattering, whereas 180° corresponds to forward scattering. Results for $\epsilon = 0.01$ is repeated in the inset to illustrate the difference in scale.	66
5.5	Scattering from a PEC sphere of $\lambda/4$ diameter with a $1V/m$ plane-wave at double-precision. Azimuth angles of 0° and 360° correspond to backscattering, whereas 180° corresponds to forward scattering. Results for $\epsilon = 0.01$ is repeated in the inset to illustrate the difference in scale.	67
5.6	MP-MLFMA vs. Mie Series Solution for PEC Sphere @2000 MHz, $\epsilon_d = 0.33$. Angles of 0° and 180° correspond to back-scattering and forward-scattering directions, respectively.	70

5.7	MP-MLFMA vs. MoM Solution for PEC Rod @300 MHz, with $\epsilon_d = 0.33$. Angles of 0° and 180° correspond to back-scattering and forward-scattering directions, respectively.	70
5.8	Recursive division of a PEC Sphere with triangular mesh edge lengths of $l_{RWG} = 25mm$, into an MP-MLFMA tree structure for $N = 3$	72
5.9	Far-zone co-polarized E-fields at $f = 15.63$ MHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$	73
5.10	Far-zone co-polarized E-fields at $f = 250$ MHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$	74
5.11	Far-zone co-polarized E-fields at $f = 4$ GHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$	75
5.12	ARPE of Far-zone E_θ for all the cases listed in Table 5.1.	75
5.13	Sphere array geometry for the demonstration of deep tree-structure capabilities of MP-MLFMA. Spheres are placed along the z -axis, with a z -polarized plane-wave impinging from x -direction.	76
5.14	Far-zone E_θ fields obtained by MP-MLFMA for Case-1 at $f = 125$ MHz with $D = 0.6m$, $K = 1$, and $N = 3$, compared to the Mie series and DP-MLFMA.	77
5.15	Far-zone E_θ fields obtained by MP-MLFMA for Case-2 at $f = 125$ MHz with $D = 0.6m$, $K = 2$, $d = 1.8m$, and $N = 5$, compared to the MoM solution.	78
5.16	Far-zone E_θ fields obtained by MP-MLFMA for Case-3 at $f = 125$ MHz with $D = 0.6m$, $K = 4$, $d = 1.4m$, and $N = 6$, compared to the MoM solution.	79
5.17	Far-zone E_θ fields obtained by MP-MLFMA for Case-4 at $f = 125$ MHz with $D = 0.6m$, $K = 10$, $d = 1m$, and $N = 7$, compared to the MoM solution.	79
A.1	Floating-point number representation defined in IEEE-754 standard [28].	90

C.1	The value of Legendre polynomials for various orders.	96
-----	---	----



List of Tables

3.1	Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for Various Desired Relative Error Thresholds (ϵ_d) and Box Sizes (a) when $\vec{w} = [0 \ 2a \ 0]^T$	36
3.2	Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for Various Desired Relative Error Thresholds (ϵ_d) and Box Sizes (a) when $\vec{w} = [3a \ 3a \ 3a]^T$	37
4.1	Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for the CPU-time & Memory Test	53
5.1	Scenarios analyzed to illustrate the broadband capabilities of MP-MLFMA	71
5.2	Deep tree-structure scenarios under analysis for $f = 125$ MHz and $\epsilon_d = 0.01$	77
A.1	Definitions and properties of some popular precision levels in IEEE-754 [28]	91

Chapter 1

Introduction

The Fast Multipole Method (FMM) [1] has been named as one of the top-ten algorithms of the 20th century by the Society of Industrial and Applied Mathematics [2]. The Multilevel Fast Multipole Algorithm (MLFMA) [3–5], which is a hierarchical extension of FMM, is able to achieve $O(N \log N)$ complexity for N unknowns, enabling the solution of extremely large electromagnetic problems compared to $O(N^2)$ for a Krylov-subspace iterative algorithm [6] applied on full matrices. This increase in efficiency is due to the ability to compute the interactions between basis and testing functions in a group by group manner, which is made possible by the Gegenbauer’s addition theorem and the diagonalization of the translation operator [1, 3].

As with other popular computational electromagnetics solvers like Finite Element Method (FEM) [7] and Finite-Difference-Time-Domain (FDTD) [8–10], contemporary research on MLFMA focuses on pushing its capabilities on two frontiers: namely, improving its capability to accurately model smaller details for a given problem, and the ability to solve larger and multi-scale deep tree-structure problems with increased efficiency.

The research on pushing the capabilities of MLFMA towards electrically small problems mainly focuses on the inherent low-frequency problem [11], which is

caused by the diagonalized form of the free-space Green's function formulation used in MLFMA. The drawback of the diagonalized form is that it includes an infinite summation over spherical harmonics involving Hankel functions, which become numerically unstable as the truncation number (order of Hankel functions) increases due to limited machine precision of the computing platform. The numerical stability problem of the translation operator is the main culprit behind the low-frequency breakdown problem, which makes selecting the truncation number a non-trivial and important part of the error control of MLFMA.

There have been several classical papers about the error control of MLFMA, and most of them focus on the translation operator and its truncation. In [12–14], the excess bandwidth formula (EBF) is used to determine the truncation numbers. Although widely used in majority of MLFMA implementations, there are two main limitations of EBF: First, the formula is derived using the large argument approximation of the Bessel and Hankel functions [15] of the translation operator, which makes it invalid for small box sizes (i.e., low-frequency problems). Second, the numerical stability of the Hankel function given the available machine precision is not taken into account. The second shortcoming is partly addressed in [16], where the accuracy lost due to the overflow of the Hankel function is incorporated into the formulation. However, the resulting error control scheme is still only limited to electrically large boxes.

In the case of electrically small boxes, there are many different studies available in the literature to treat the low-frequency breakdown that fall into mainly two categories. One popular approach is to use the multipoles explicitly [17–22], while another is to deform the angular integration so that the evanescent waves are taken into account for sub-wavelength interactions [23–26]. Methods in both categories require the solver to be implemented from the ground up, while increasing complexity due to alternative expansion formulations of the Green's function. A simple alternative to the treatment of the low-frequency problem is proposed in [27] where multiple-precision arithmetic (MPA) is used to handle overflowing summations when necessary. Since the widely-used EBF is not valid for low-frequency problems, authors in [27] determined the optimum truncation numbers and machine precision levels empirically by extensive numerical simulations. As

shown in [27], simply increasing the number of decimal digits of precision leads to accurate solutions for smaller box sizes, effectively pushing the low-frequency limit downward.

The use of MPA in computational analysis warrants further discussion. In a digital computer, each number is stored and processed with a certain amount of bits during a numerical computation. The number of bits ultimately determines the precision of the actual number to be held in the memory and the processor. This allocation can be implemented at the hardware level (e.g. 32-bit or 64-bit CPU/GPU/RAM architectures) or can be defined through software to achieve an arbitrary level of precision. Currently double-precision (64-bits, with ~ 16 decimal digits of resolution) is default in most CPU based computers, whereas single-precision (32-bits, ~ 7 decimal digits) is the most common architecture in the state-of-the-art GPUs [28]. At the time of writing this dissertation, the literature on the use of MPA for computational electromagnetic problems is scarce, but not non-existent. Most of the research involves MPA-enabled iterative matrix equation solutions [29–34] and high-precision integer arithmetics for FDTD simulations [35–37].

On the other side of the coin, the research on improving the efficiency of MLFMA for electrically large and multi-scale problems with deep tree-structures mainly focuses on robust partitioning of the tree structure and efficient parallelization. Researchers have implemented non-uniform clustering [38, 39], hierarchical skeletonization [40, 41], and incomplete tree structures [42] to handle multi-scale problems that include electrically large and small box sizes simultaneously. However, they require ground-up implementations of MLFMA both for the tree structuring and the low-frequency treatments explained before [24, 25]. On the other hand, parallelization research focuses on the strong and weak scalability of MLFMA for large problems. In strong scalability, speed-up ratios for different number of nodes are observed for a fixed problem size. In a weak scalability analysis, the problem size *per node* is fixed, which indicates the ability to solve larger problems with large number of computing nodes. MLFMA, being a complex algorithm, does not allow simple partitioning schemes that scale properly because of the load-balancing issues and inevitable communication requirement between the

nodes. In recent years, advanced partitioning schemes that distribute either field points or boxes to different nodes were developed with improved load-balancing, near-optimal communication schemes [43–45]. Later, a hierarchical partitioning scheme that distributes both the field points and the boxes adaptively across different levels is proposed, where parallelization efficiencies of more than 80% are reported for 64-nodes [46]. More recently, the efficiency of the parallelization scheme in [46] is increased even further by partitioning along both θ and ϕ axes and showing that this partitioning scheme exhibits weak scalability [47].

A hierarchical multiple-precision framework offers an elegant solution to address both the low-frequency problem and the efficiency of multi-scale problems with deep tree-structures listed above. For starters, in low-frequency problems one does not need to increase the precision for every level within MLFMA. Only smaller boxes require relatively higher precision levels. Moreover, MLFMA is a very good candidate for MPA implementations even for high-frequency problems due to its hierarchical tree structure. Intuitively, during the aggregation stage MLFMA should require less and less precision as the radiated fields become more and more compounded. Hence, an optimal strategy can be defining specific precision levels for each level throughout all stages of MLFMA.

The use of low-precision arithmetics for large boxes can greatly improve performance and efficiency. If a certain arithmetic operation does not require the particular default precision of the computing platform, the computation can be greatly accelerated by using a lower precision. For example, if a common 64-bit CPU architecture is used for a 32-bit (single precision) addition operation, a single 64-bit adder can be used to perform two simultaneous 32-bit additions, while utilizing the extremely fast L1 and L2 caches much more efficiently, leading to a super-linear speed-up. Therefore, for an algorithm that may require different levels of precision at different points in its course –for which MLFMA is a perfect example– an optimal implementation intuitively *must* utilize an MPA framework.

Although MPA offers the best of both worlds in terms of accuracy and efficiency, an accurate error control method that is valid for all frequencies while also estimating the necessary machine precision level for each level of MLFMA is

still not available in the current literature.

To summarize, MLFMA is an extremely popular full-wave solver for electrically large problems; however, efficiency for multi-scale problems with deep tree-structures and the low-frequency problem are active research topics that need addressing. Moreover, the current error control methods for MLFMA do not work for small-boxes while not taking into account the machine precision of the computing platform. To address all the shortcomings listed above, we propose a hierarchical multiple-precision MLFMA formulation for the accurate and efficient analysis of extremely large problems at all frequencies.

The novel Multiple-Precision Multilevel Fast Multipole Algorithm (MP-MLFMA) proposed in this dissertation allows a hierarchical implementation of different machine precision levels at different partition levels of an MLFMA tree structure, leading to an optimal accuracy and efficiency for each level individually. Specifically, for large box sizes, the machine precision can be reduced below the default precision for faster solutions with reduced memory, without accuracy problems. Moreover, the machine precision can be increased above the default precision for small boxes (or low-frequencies) to alleviate the low-frequency breakdown of the translation operator, thereby allowing accurate solutions for all frequencies. To determine the required truncation numbers and machine precision levels, we also introduce and demonstrate a novel error control scheme for MLFMA that is valid at all frequencies. The proposed error control scheme provides the optimum truncation numbers given the box size and the desired relative error threshold at all frequencies for the first time in the literature, while yielding compatible results with EBF at high frequencies. Additionally, the proposed scheme provides the required machine precision levels for each case, results of which are used as a precursor to the efficient and robust implementation of the novel MP-MLFMA solver. With the novel universal error control scheme proposed in this dissertation, error analyses and design curves for efficient MLFMA interpretations at a wide range of frequencies and machine precision levels are also provided for the first time in the literature, enabling users with a rigorous and intuitive guideline on how to implement MLFMA on both fixed- and variable-precision platforms, such as CPUs, GPUs, FPGAs, etc., of the state-of-the-art

technology.

The rest of this dissertation is organized as follows: In Chapter 2, an introduction to MLFMA and a discussion of its limits for a fixed-precision computer are provided. In Chapter 3, the proposed novel error control scheme for MP-MLFMA is rigorously derived and explained in a detailed way, together with simplified expressions for easy implementation. Chapter 4 presents the details of MP-MLFMA while focusing on both software and hardware implementations. Numerical results for the error analysis of MLFMA and the scattering results with MP-MLFMA are presented in Chapter 5. Concluding remarks are given in Chapter 6. In Appendix A, the floating-point arithmetics defined by the IEEE-754 standard is briefly explained. Appendix B provides the asymptotic expressions for Bessel and Hankel functions, which are helpful in the error analysis of MLFMA. Finally in Appendix C, a discussion on the Legendre polynomials and some simplified expressions are provided, which are important for accurate error control of MLFMA especially for low-frequency problems.

Note that an $e^{-i\omega t}$ time convention, where $\omega = 2\pi f$ and f is the operating frequency, is assumed and suppressed throughout this dissertation.

Chapter 2

Multilevel Fast Multipole Algorithm

The proposed novel multiple-precision framework in this dissertation is built upon an MLFMA implementation on a Method of Moments (MoM) solver. To illustrate the motivation behind this proposal, we introduce the basics of MLFMA, show the limitations of MLFMA under fixed-precision computers, and demonstrate a proof-of-concept for a multiple-precision implementation using numerical and statistical analyses in this chapter.

2.1 Introduction to MLFMA

A great range of electrical engineering problems including but not limited to antenna design (radiation) and stealth platform design (scattering) require full-wave electromagnetic (EM) solutions. Although Maxwell's equations can be used to describe any EM phenomena, their rigorous solutions only exist for canonical shapes such as plates, cylinders or spheres. For complex geometries, numerical methods are implemented to solve the Maxwell's equations. Of the numerical methods, integral equation (IE) based solution methods are preferred as they

inherently include the radiation condition (i.e., they do not require external radiation boundaries such as the perfectly-matched-layer [10]) and only require discretization along the discontinuities of the medium [48, 49]. This is due to the fact that differential solvers try to solve the whole field whereas the IE formulations solve for the equivalent sources [49]. On the other hand, compared to the sparse matrices formed by other popular frequency-domain solvers like FEM [7], IE-based solvers result in a fully populated (dense) matrix, which makes the method more resource demanding for the same number of unknowns. IE-based matrix equations require $\mathcal{O}(N^3)$ operations for direct inversion using LU decomposition. Using iterative techniques, the required matrix-vector-multiplication (MVM) operations can be reduced to $\mathcal{O}(N^2)$ per iteration; however, such a solution will only be valid for a single excitation (right-hand-side of the matrix equation) as the inverse of the matrix is never explicitly calculated. To reduce the computational cost, FMM [1, 50] and a logical hierarchical extension of FMM, MLFMA [4, 14, 51] is introduced. Both FMM and MLFMA exploit the simpler interaction between far-away elements during the matrix fill, and greatly reduce MVM computation times during the iterative solution process as the computation cost is reduced from $\mathcal{O}(N^2)$ (for a Krylov-subspace algorithm [6]) to $\mathcal{O}(N \log N)$. In the rest of this section, the basic steps of an MLFMA implementation are summarized. Note that some of the figures and equations in this section are taken from [5, 52], courtesy of Dr. Ergül, one of the supervisors in this dissertation.

MLFMA starts with constructing a tree by dividing the structure under analysis recursively into cubic boxes and subboxes as shown in Fig. 2.1. Then, using the one-box-buffer scheme as shown in Fig. 2.2, MVM operation can be separated into near-field and far-field components as follows:

$$\sum_{n=1}^N \overline{\overline{Z}}[m, n] \overline{\gamma}[n] = \sum_{n=1}^N \overline{\overline{Z}}_{NF}[m, n] \overline{\gamma}[n] + \sum_{n=1}^N \overline{\overline{Z}}_{FF}[m, n] \overline{\gamma}[n], \quad (2.1)$$

where $\overline{\overline{Z}}$ is the impedance matrix and $\overline{\gamma}$ is the solution of the unknown coefficient vector in the current iteration of an iterative matrix equation solver. Computation of the near-field interactions is performed as in the standard MoM procedure [48], while for far-field interactions the whole MVM –rather than each interaction– is computed in three separable stages of aggregation, translation and disaggregation,

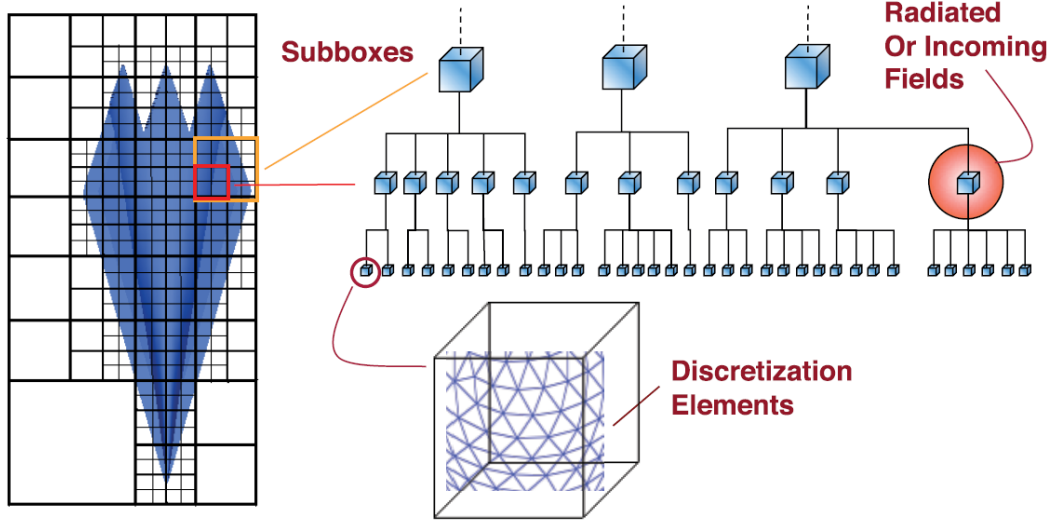


Figure 2.1: Recursive division of the tree structure (Source: Ergül [5, 52])

represented by the vectors $\vec{v}_1$, $\vec{w}$, and $\vec{v}_2$, respectively. The separation of the aforementioned stages is possible due to the Gegenbauer's addition theorem and diagonalization of the Green's function [1]. The Gegenbauer's addition theorem expands the free-space Green's function in terms of spherical harmonics as

$$\frac{\exp(ik|\vec{w} + \vec{v}|)}{4\pi|\vec{w} + \vec{v}|} = \frac{ik}{4\pi} \sum_{t=0}^{\infty} (-1)^t (2t+1) j_t(kv) h_t^{(1)}(kw) P_t(\hat{w} \cdot \hat{v}), \quad (2.2)$$

where j_t and $h_t^{(1)}$ are the spherical Bessel and Hankel functions of the first kind, respectively. In (2.2), P_t is the Legendre polynomial of order t , while $w = |\vec{w}|$ and $v = |\vec{v}| = |\vec{v}_1 + \vec{v}_2|$ represent the translation and shift vectors, respectively. Note that (2.2) is only valid when $w > v$. Representing the spherical waves as integrals over the plane-wave spectrum, the diagonal form of the Green's function is obtained as

$$\frac{\exp(ik|\vec{w} + \vec{v}_1 + \vec{v}_2|)}{4\pi|\vec{w} + \vec{v}_1 + \vec{v}_2|} \approx \frac{ik}{(4\pi)^2} \int d^2\hat{k} \beta(\vec{k}, \vec{v}_1) \alpha_{\tau}(\vec{k}, \vec{w}) \beta(\vec{k}, \vec{v}_2), \quad (2.3)$$

where

$$\beta(\vec{k}, \vec{v}) = \exp(i\vec{k} \cdot \vec{v}), \quad (2.4)$$

$$\alpha_{\tau}(\vec{k}, \vec{w}) = \sum_{t=0}^{\tau} (i)^t (2t+1) h_t^{(1)}(kw) P_t(\hat{k} \cdot \hat{w}), \quad (2.5)$$

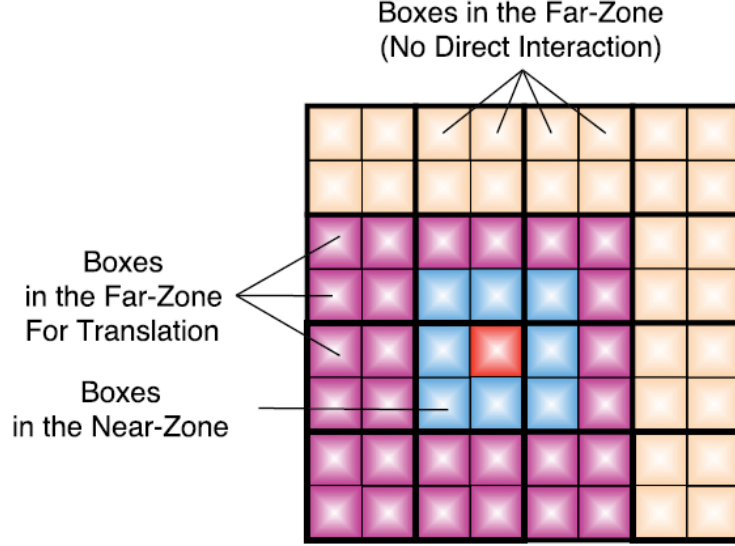


Figure 2.2: One-box-buffer Scheme (Source: Ergül [5, 52])

are the shift and the translation operators, respectively. In (2.5), τ is the truncation number which directly affects the accuracy of the translation operator and ultimately the accuracy of MLFMA. The truncation number also determines the number of sampling points along θ and ϕ axes of the spherical coordinate system [5, 52] for the angular integration in (2.3).

The truncation number (τ) is defined for each level typically using EBF [12]. At this point, it should be emphasized that although EBF is the standard practice for MLFMA users, it is only valid for large translation distances and fails at low-frequencies or small box sizes. Hence, an important contribution of this work will be the development of an alternative error control scheme that is valid for all frequencies. Detailed derivations and discussions on the proposed novel error control scheme is given in Chapter 3.

In the aggregation stage of MLFMA, the radiated fields for each box at every level are computed recursively from the leaf-level (lowest level) to the mother-level

(highest level encompassing the whole structure) as

$$\overline{S}_C(\vec{k}, \vec{r}_C) = \begin{cases} \sum_{\vec{b}_n \in C} \overline{\gamma}[n] \beta(\vec{k}, \vec{r}_C - \vec{r}_n) \overline{S}_n(\vec{k}, \vec{r}_n), & \text{leaf level} \\ \sum_{C' \in C} \beta(\vec{k}, \vec{r}_C - \vec{r}_{C'}) \overline{S}_{C'}(\vec{k}, \vec{r}_{C'}), & \text{otherwise,} \end{cases} \quad (2.6)$$

where subscripts C and n denote the box of interest, and the basis index, respectively. $\overline{S}_n$ refers to the radiation pattern of the n^{th} basis function, while the set C' refers to the child boxes of box C . After the computation of aggregated fields using (2.6), translations are computed for boxes with available far-zone interactions (denoted as $F\{C\}$), given by

$$\overline{G}_C(\vec{k}, \vec{r}_C) = \sum_{C' \in F\{C\}} \alpha_\tau(\vec{k}, \vec{r}_C - \vec{r}_{C'}) \overline{S}_{C'}(\vec{k}, \vec{r}_{C'}). \quad (2.7)$$

Translated fields $\overline{G}_C(\vec{k}, \vec{r}_C)$ are distributed to lower level boxes by disaggregation, which is performed from top to bottom to obtain the disaggregated result, $\overline{G}_{C'}^+(\vec{k}, \vec{r}_{C'})$ as

$$\overline{G}_{C'}^+(\vec{k}, \vec{r}_{C'}) = \overline{G}_{C'}(\vec{k}, \vec{r}_{C'}) + \beta(\vec{k}, \vec{r}_{C'} - \vec{r}_C) \overline{G}_C^+(\vec{k}, \vec{r}_C) \quad C' \in C. \quad (2.8)$$

Finally, MVM is computed by multiplying with the testing function radiation patterns $\overline{F}_m(\vec{k}, \vec{r}_m)$ for $m = 1, \dots, N$, and integrating over the radiation sphere, given by

$$\sum_{n=1}^N \overline{\overline{Z}}_{FF}[m, n] \overline{\gamma}[n] = \left(\frac{ik}{4\pi} \right)^2 \int d^2\hat{k} \overline{F}_m(\vec{k}, \vec{r}_m) \cdot \beta(\vec{k}, \vec{r}_m - \vec{r}_C) \overline{G}_C^+(\vec{k}, \vec{r}_C). \quad (2.9)$$

The resulting MVM is then used across the iterations of an iterative matrix equation solver such as the generalized minimal residual method (GMRES) [53].

2.2 Limitations of MLFMA in Fixed-Precision Computers

The reduced computational complexity offered by MLFMA is thanks to its use of diagonalized form of the free space Green’s function, as given in (2.3)–(2.5). With the diagonalized form, two new sources of error is introduced: namely the truncated infinite summation, and the overflow of the Hankel function in (2.5). For large orders and small arguments, the spherical Hankel function increases exponentially, which may lead to the overflow of the registers for a fixed-precision computer. This is actually the well-known low-frequency breakdown problem of MLFMA [11]. Even when the registers do not overflow, the standard floating-point arithmetic computers experience reduced resolution for very large numbers [28]. See Appendix A for a brief introduction to the floating-point arithmetic standard, and Appendix B for a detailed discussion on the Bessel and Hankel functions, and their asymptotical forms.

To take into account the reduced resolution and accuracy caused by the Hankel function, EBF is modified in [16], which will be explained in detail in Chapter 3. Note that even in its modified form, EBF is still not valid for small box sizes and low-frequencies, meaning the low-frequency breakdown still needs addressing. To the best of our knowledge, the only publication prior to our work that deals with the low-frequency breakdown in the context of multiple-precision arithmetic is given in [27]. In [27], authors show that by simply increasing the native precision of the platform, the onset of the low-frequency breakdown can be pushed forward to higher truncation numbers, allowing accurate solutions even at extremely small box sizes.

To illustrate the effect of machine precision on the accuracy of the diagonalized Green’s function, we have computed (2.3) for the following set of parameters:

- Box sizes: $a \in \{\lambda/512, \lambda/128, \lambda/4, \lambda\}$
- Translation vector: $w = |\vec{w}| = |[0 \ 2a \ 0]^T| = 2a$, representing the lowest

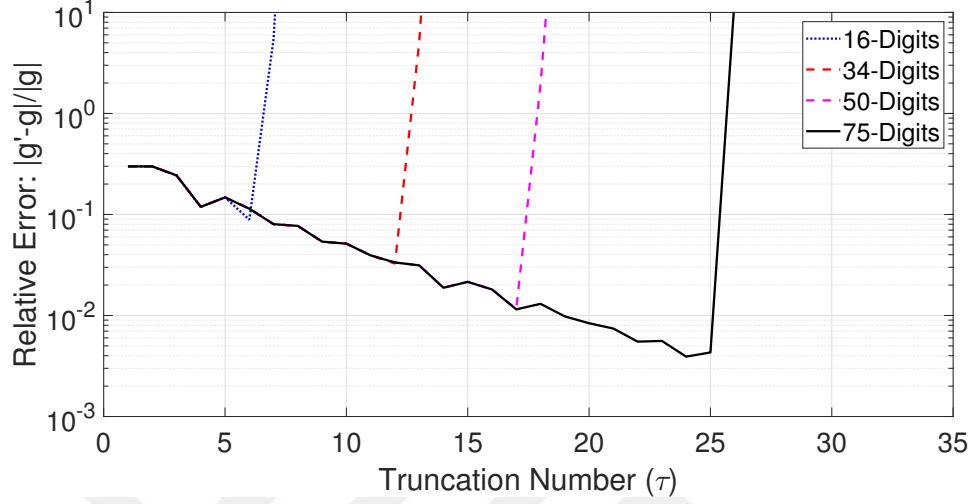


Figure 2.3: Relative errors of the diagonalized form of Green’s function across different truncation numbers for $a = \lambda/512$

value with the one-box-buffer scheme, with T denoting the matrix transpose

- Shift vector: $v = |\vec{v}_1 + \vec{v}_2| = |[a \ a \ a]^T| = a\sqrt{3}$, representing the largest possible value
- Machine Precision (in decimal digits): $MP \in \{16, 34, 50, 75\}$

Note that for the largest errors, the translation vector is chosen as its smallest value, while the shift vector is selected as its largest possible value within a one-box-buffer scheme. Also, we have used a third-party software toolbox [54] to emulate a multiple-precision framework within MLFMA. The relative errors with respect to the free space Green’s function are computed for the above parameters and are shown in Figs. 2.3–2.6.

As seen in Figs. 2.3–2.6, the relative errors monotonically decrease with respect to the truncation number until the low-frequency breakdown occurs. Increasing the machine precision delays the onset of the breakdown; hence, allowing accurate solution for smaller box sizes and larger truncation numbers. Another important observation from Figs. 2.3–2.6 is that the breakdown problem is not actually limited to low-frequencies: even a relatively large box at $a = \lambda$ starts experiencing the breakdown for large truncation numbers as shown in Fig. 2.6. Hence, an

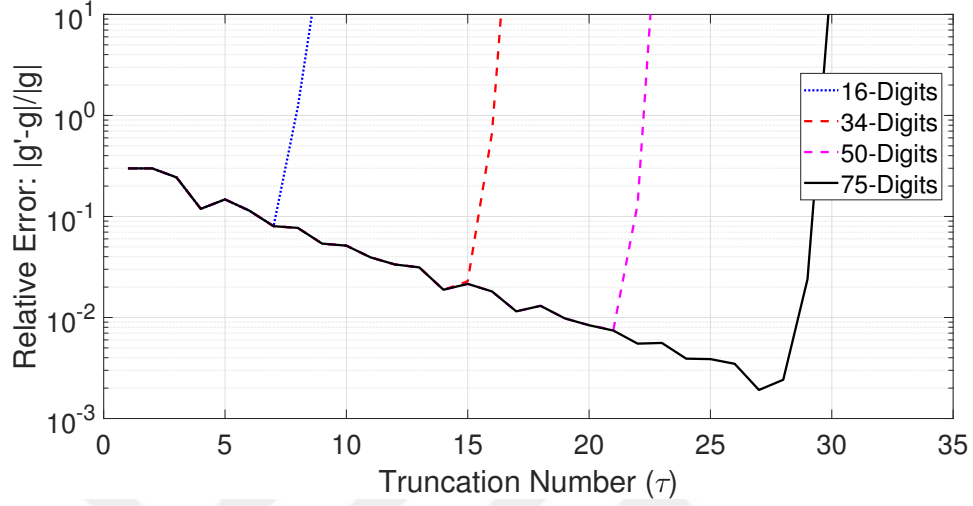


Figure 2.4: Relative errors of the diagonalized form of Green’s function across different truncation numbers for $a = \lambda/128$

important conclusion is that the “low-frequency breakdown” is actually a *large-order* problem, with the onset of the problem occurring at larger truncation numbers for large boxes or high-frequencies.

In the next section, the dependency of the accuracy of MLFMA on the machine precision is analyzed statistically in detail.

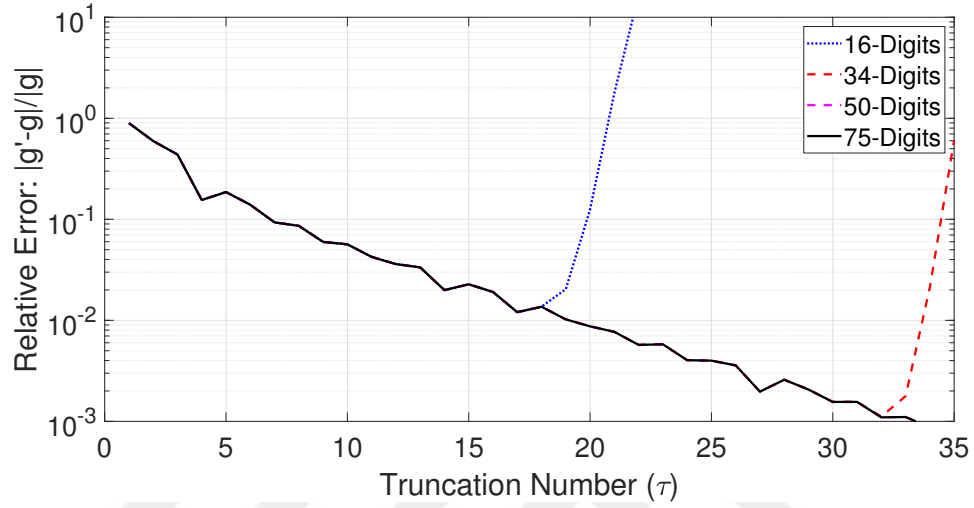


Figure 2.5: Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda/4$

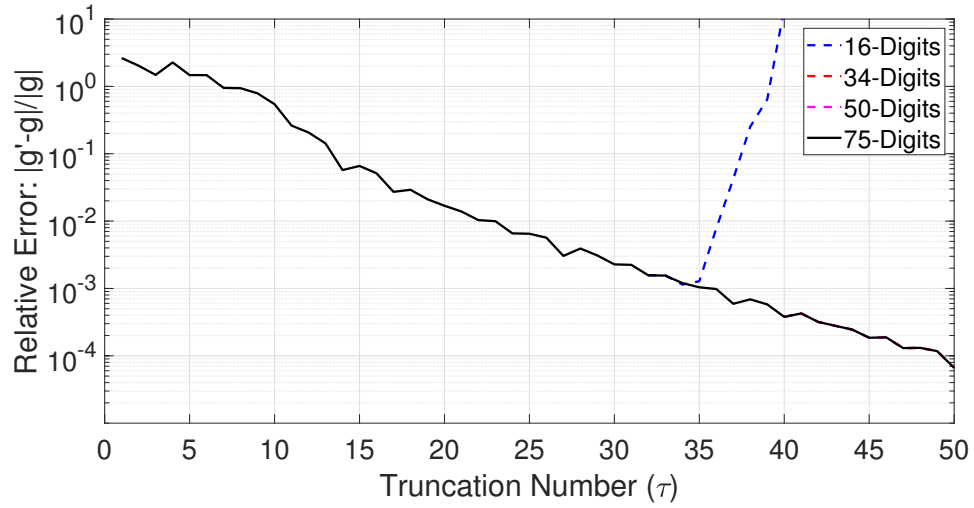


Figure 2.6: Relative errors of the diagonalized form of Green's function across different truncation numbers for $a = \lambda$

2.3 Proof-of-Concept for a Multiple-Precision Implementation of MLFMA

To test the feasibility of implementing an MPA framework within MLFMA, we have constructed a flexible MATLAB code where the far-zone interactions between the basis functions in an N -Level box and a single testing function in another N -Level box are computed. The unknown coefficients for the basis and testing points are selected randomly from a circular complex Gaussian distribution to observe the robustness against different sets of excitations which can represent different problem geometries. For each of the N levels a separate precision can be defined. Together with a *base precision* that is defined for all coefficients such as the speed of light, π etc., $N + 1$ distinct precision levels are allowed. For simplicity, all basis and testing functions are assumed to be scalar point sources with isotropic radiation patterns.

For all simulations presented in this section, we define the error as the relative error with respect to the analytical MVM obtained from the Green's Function. The relative error is defined as

$$\epsilon = \frac{|MVM - MVM_{GF}|}{|MVM_{GF}|}, \quad (2.10)$$

where the analytical MVM_{GF} is given by

$$MVM_{GF} = \frac{ik}{4\pi} \sum_{n=1}^{\#ofbases} \bar{\gamma}[n] \frac{\exp(ikR_n)}{R_n}, \quad (2.11)$$

where R_n is the Euclidean distance from the n^{th} basis point to the testing point.

Using the above described testbench, we initially show that different box sizes require different machine precision levels for similar relative errors, especially for electrically small ($a \ll \lambda$) boxes where the low-frequency breakdown is prevalent. Therefore, simulations with the following parameters are conducted:

- Number of levels: $N = 3$ (MVM between two 3-level boxes, actual problem has at least $N + 1 = 4$ levels)

- Leaf-Level Box Size: $a = \{\lambda/512, \lambda/128\}$
- Number of harmonics: $\tau = \{15, 20, 25\}$ (constant across N -levels)
- Basis point separation: $d = a/4$ (distance between uniformly spaced basis points in all directions)
- Translation vector: $\vec{w} = [0 \ 2a \ 0]^T$, i.e., the one-box-buffer scheme
- Shift Vector at disaggregation: $\vec{v}_2 = [0.5a \ 0.5a \ 0.5a]^T$ (towards the corner of the “test box”)
- Excitation Distribution: $|\bar{\gamma}_n| \sim N(1, 0.1^2)$, $\angle \bar{\gamma}_n \sim U[0, 2\pi)$ (complex circular Gaussian)

Note that for electrically small boxes, EBF [12] cannot be used, as it is only valid for large boxes. Therefore we use the tabulated results given in [27] for selecting the number of harmonics. In Chapter 3, a novel and accurate error control scheme is presented to find the required number of harmonics and machine precision levels.

The scenario listed above is illustrated in Fig. 2.7. For each case given in the above list, we present 3 analyses: one in which the precision of all levels are varied together (*SurveyLevel: all*), one in which only the leaf-level precision is varied (*SurveyLevel: leaf*), and one where only the translation level’s precision is varied (*SurveyLevel: tx*). When the precision of a particular level is varied, all other levels are kept at maximum precision, which is selected empirically as 70 decimal digits.

The relative error graphs obtained for leaf-level box sized $a = \lambda/128$ and $a = \lambda/512$ are given in Figs. 2.8–2.9.

As seen in Figs. 2.8–2.9, the low-frequency breakdown of the MLFMA is mitigated for large enough machine precision levels (typically larger than double precision, ~ 16 decimal digits). Also, the relative errors are decreasing with increasing τ as expected. Moreover, a higher τ leads to a higher required machine

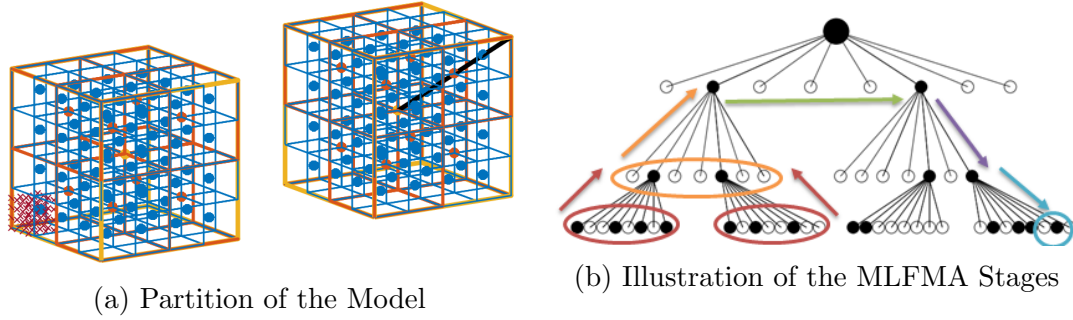


Figure 2.7: Implemented MVM Scenario

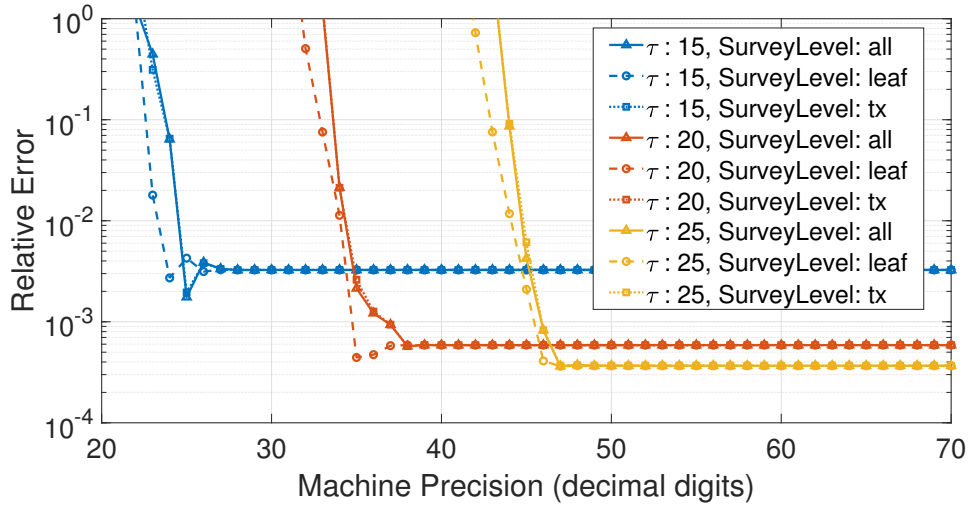


Figure 2.8: Relative Errors for Electrically Small Boxes for $a_1 = \lambda/128$

precision which can be seen in the onset of the breakdown points. Another important observation is that the precision of the translation level dominates the overall required precision for a given box size. Note that reducing the precision a little bit further is possible in aggregation stage (see *SurveyLevel: leaf*); however, it may not be worth the computational effort to change the precision across the stages. Figs. 2.8–2.9 clearly demonstrate that in a typical MLFMA problem with large number of levels, each translation level and the corresponding lower levels can be computed via different machine precision levels.

For electrically large boxes where the translation level box-size is larger than λ , we present new simulations with the following parameters:

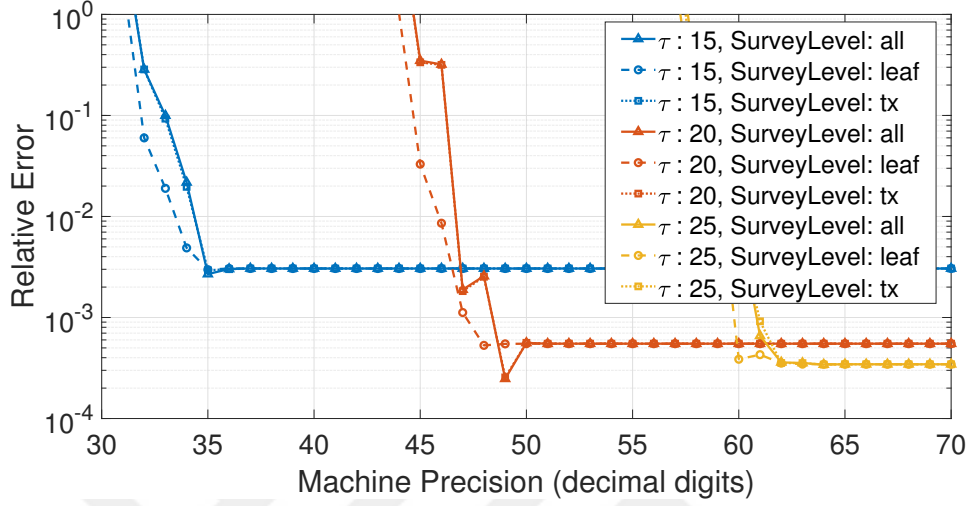


Figure 2.9: Relative Errors for Electrically Small Boxes for $a_1 = \lambda/512$

- $(N, a) = \{(5, \lambda/4), (3, 2\lambda)\}$ (translation box size: 2λ and 8λ)
- Number of harmonics: using EBF [12] with decimal digits of accuracy $d_0 = 3$, see Section 3.1 for explicit formulas.
- Basis separation: $d = \lambda/16$ (distance between uniform basis points in all directions)
- Translation vector: $\vec{w} = [0 \ 2a \ 0]^T$, i.e., the one-box-buffer scheme
- Shift Vector: $\vec{v}_2 = [0.5a \ 0.5a \ 0.5a]^T$ (towards the corner of the “test box”)
- Excitation Distribution: $|\bar{a}_n| \sim N(1, 0.1^2), \angle \bar{a}_n \sim U[0, 2\pi)$ (complex circular Gaussian)

The relative error graphs obtained for the above parameters are given in Figs. 2.10–2.11.

Figs. 2.10–2.11 illustrate another advantage of a multiple-precision implementation of MLFMA, namely the big reduction in the required machine precision for larger boxes. Considering the standard double precision corresponds to ~ 16 decimal digits of accuracy, we show that the machine precision can at least be halved

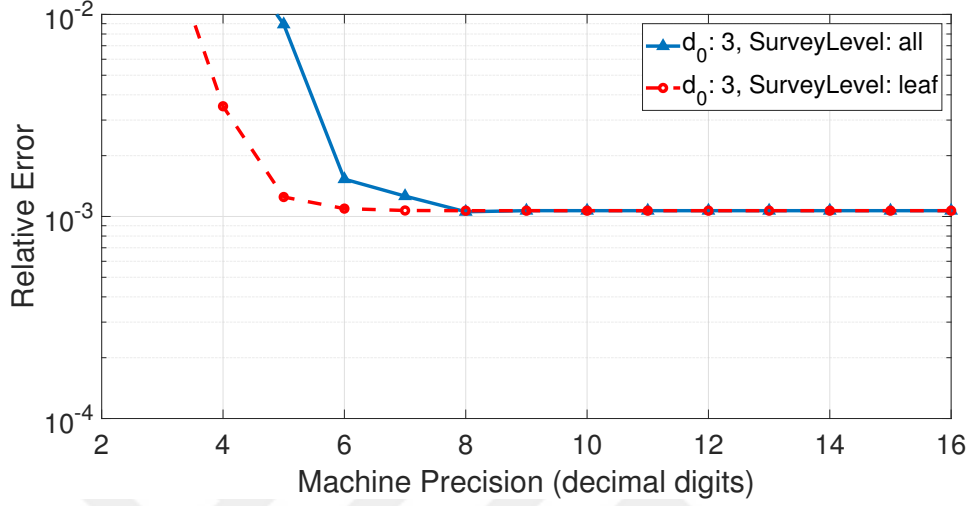


Figure 2.10: Relative Errors for Electrically Large Boxes: $(N, a_1) = (5, \lambda/4)$, with translation @ 2λ

for larger box sizes. Additionally, the further decrease of the machine precision in aggregation stages can be observed more clearly in the large box cases.

Finally we present our findings on the robustness of the results we present above across random excitations (possibly representing different types of problems). For that we analyze a single-level translation problem where the box size is $a = \lambda/4$ and excitation distribution is the same as the previous cases. For different number of harmonics and a Monte Carlo iteration count of 50, the problem geometry is shown in Fig. 2.12 and the results showing the 95% confidence intervals are given in Fig. 2.13.

As Fig. 2.13 illustrates, different realizations of the random excitation distribution do not alter the obtained error graphs significantly with the given 95% confidence limits.

To summarize, we reach the following conclusions in this section:

- There exist *optimal machine precisions* for different levels and stages within the MLFMA.
- The low-frequency breakdown can be elegantly treated simply by increasing

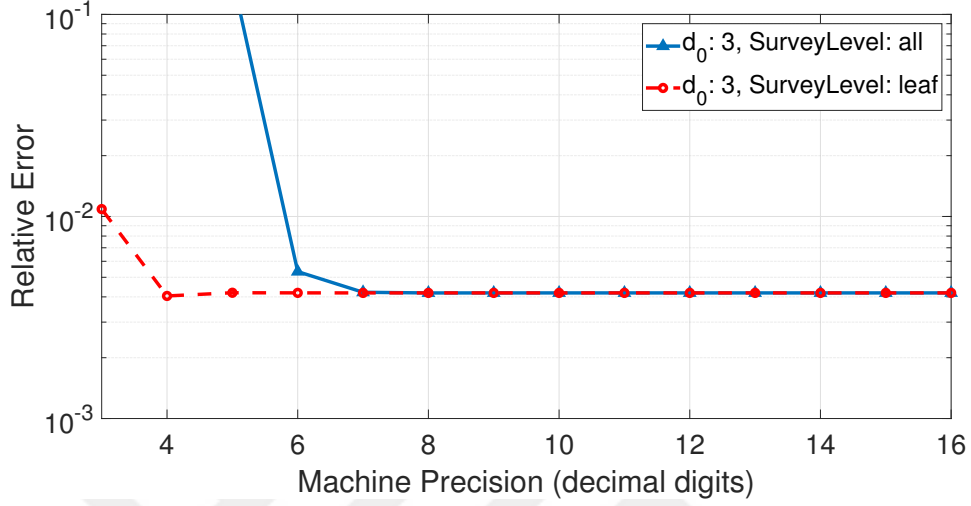


Figure 2.11: Relative Errors for Electrically Large Boxes: $(N, a_1) = (3, 2\lambda)$, with translation @ 8λ

the machine precision when necessary.

- For electrically large boxes (or at high frequencies) the efficiency of MLFMA can be increased by reducing the machine precision below the default double (or even single) precision.
- Using random excitations and Monte Carlo simulations, we observe that the obtained results are robust across random changes in the excitation functions.

Although this section demonstrates the feasibility of the proposed method, a key piece for a successful implementation is so far missing: an accurate error control scheme for various box sizes. Therefore in the next chapter, we address this problem and propose a novel error control scheme that is valid for all frequencies and can take into account the machine precision of the computing platform.

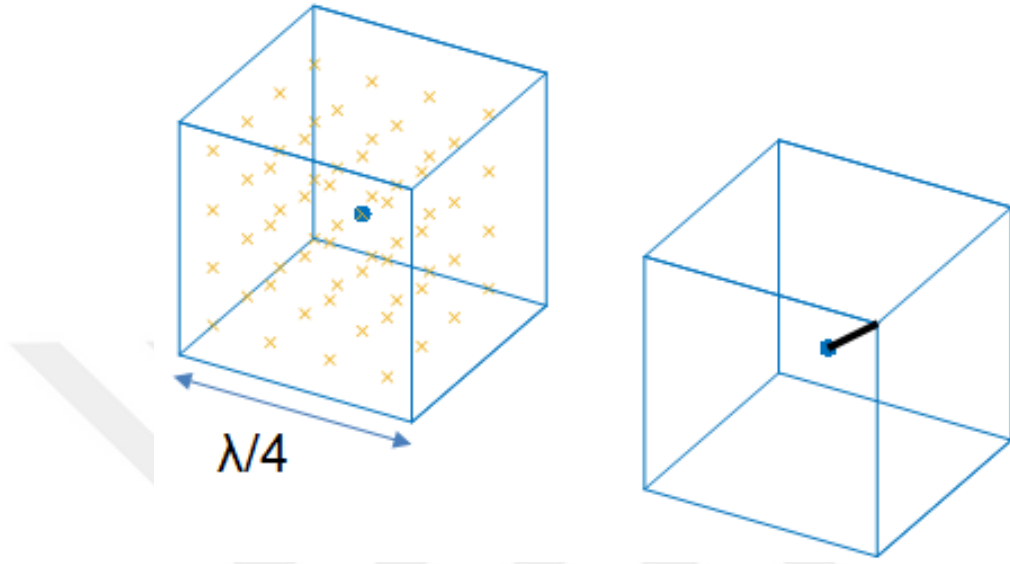


Figure 2.12: Single-Level translation model for the robustness analysis.

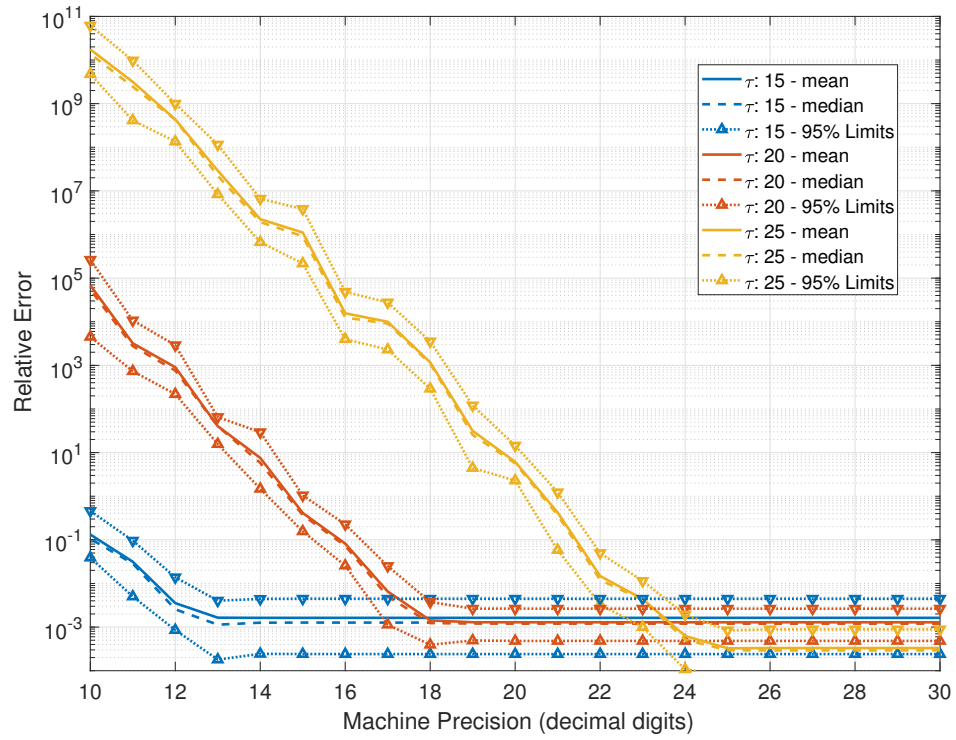


Figure 2.13: Robustness analysis showing the relative error with respect to the free-space Green's function over 50 Monte Carlo iterations.

Chapter 3

Error Control of MLFMA

MP-MLFMA scheme proposed in this dissertation offers accurate solutions at all frequencies, enabling efficient solutions of electrically large deep tree-structure problems in an elegant fashion. The accuracy of MLFMA depends heavily on the selection of the truncation number (τ) of the truncated infinite summation over spherical harmonics in (2.3)–(2.5), as well as the finite machine precision of the computing platform. However, the previous state-of-the-art on error control of MLFMA is only valid for high-frequencies and large box sizes. Therefore, in this chapter we propose a novel error control scheme that can take into account the machine precision, while offering accurate control across all frequencies and box sizes. In the following sections; the existing literature on the error control of MLFMA is briefly summarized, the proposed error control scheme is rigorously derived, and some simplified formulations for easy implementation of the proposed scheme are presented.

3.1 Previous State-of-the-Art on Error Control

With a truncation number of τ in the infinite summation in (2.2), the relative error ($\hat{\epsilon}$) with respect to the free-space Green's function can be found from

$$\hat{\epsilon} = kR \left| \sum_{t=\tau+1}^{\infty} (-1)^t (2t+1) j_t(kv) h_t^{(1)}(kw) P_t(\hat{w} \cdot \hat{v}) \right|, \quad (3.1)$$

where $R = |\vec{w} + \vec{v}|$ is the total radiation distance. Assuming that the leading term of (3.1) is the dominant error contributor, the relative error can be approximated as

$$\hat{\epsilon} \approx kR(2\tau+3) \left| j_{\tau+1}(kv) h_{\tau+1}^{(1)}(kw) P_{\tau+1}(\hat{w} \cdot \hat{v}) \right|. \quad (3.2)$$

Using the large-argument asymptotic expressions for the Bessel and Hankel functions on (3.2), and assuming $|P_{\tau+1}(\hat{w} \cdot \hat{v})| = 1$, EBF [12–14] can be derived to determine the truncation number given the desired error as

$$\tau \approx ka\sqrt{3} + 2.18(d_0)^{2/3}(ka)^{1/3}, \quad (3.3)$$

where a is the box edge length and d_0 is the desired decimal digits of accuracy which is related to the desired relative error threshold (ϵ_d) as

$$d_0 \triangleq -\log_{10}(\epsilon_d). \quad (3.4)$$

Further discussion on the asymptotic expressions for the Bessel/Hankel functions are given in Appendix B, while the validity of the Legendre polynomial is discussed in Appendix C.

EBF given in (3.3) is extremely popular among MLFMA users because of its simplicity and accuracy for large box sizes. However, due to its dependence on the large-argument approximations of the spherical functions in (3.2), EBF becomes invalid for small box sizes or low-frequencies. Moreover, the actual relative error of (2.2)–(2.5) with respect to the free-space Green's function may exceed the desired threshold (ϵ_d) due to resolution problems in the evaluation of the spherical Hankel function on a floating-point computer with limited machine precision. This behavior of the Hankel function is partly addressed in [16], where

the decimal digits of accuracy that is lost due to the spherical Hankel function is modeled as

$$d_1 = \left[\frac{\tau - 2ka}{1.8(2ka)^{1/3}} \right]^{1.5}, \quad (3.5)$$

which can be used to estimate the effective decimal digits of accuracy (d_{eff}) given by

$$d_{\text{eff}} = d_0 - d_1. \quad (3.6)$$

Note that the accuracy estimate of (3.6) is still only valid for electrically large boxes since it is based on the large argument approximation of Hankel functions.

3.2 Proposed Error Control Scheme

The proposed novel error control scheme can be used to determine the truncation numbers as well as the machine precision levels required for a given problem and the desired error thresholds. In the following, we present the estimation of the truncation numbers and the machine precision levels separately.

3.2.1 Estimating the Optimum Truncation Number

Unlike the previous works on error control [12–14] that use the large argument approximations of the Bessel and Hankel functions in (3.2), we use the *large order* approximations given in (B.12)–(B.13) to obtain valid expressions for all box sizes. Substituting (B.12)–(B.13) into the spherical Bessel and Hankel functions in (3.2), we obtain

$$j_{\tau+1}(kv) = \sqrt{\frac{\pi}{2kv}} J_{\tau+1.5}(kv) \approx \frac{0.5\psi_j}{\sqrt{(\tau+1.5)kv \tanh \gamma_j}}, \quad (3.7)$$

$$h_{\tau+1}^{(1)}(kw) = \sqrt{\frac{\pi}{2kw}} H_{\tau+1.5}^{(1)}(kw) \approx \frac{0.5\psi_h - i\psi_h^{-1}}{\sqrt{(\tau+1.5)kw \tanh \gamma_h}}, \quad (3.8)$$

where ψ_j and ψ_h are defined as

$$\psi_j \triangleq \exp [(\tau+1.5)(\tanh \gamma_j - \gamma_j)], \quad (3.9)$$

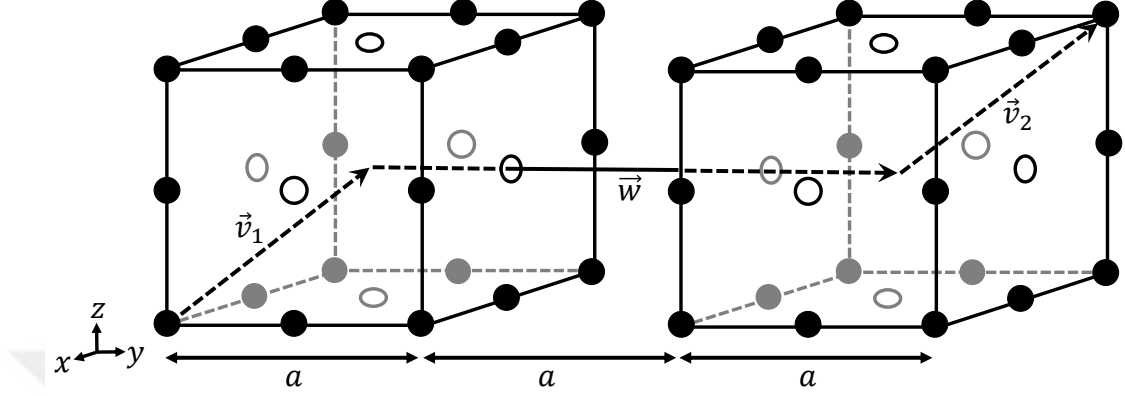


Figure 3.1: Critical points for the shift vectors shown on source and observation boxes in a one-box-buffer scheme. Corners and edge centers are shown with filled circles and face centers are shown as circles.

$$\psi_h \triangleq \exp [(\tau + 1.5)(\tanh \gamma_h - \gamma_h)], \quad (3.10)$$

with

$$\gamma_j = \operatorname{sech}^{-1} \left(\frac{kv}{\tau + 1.5} \right), \quad (3.11)$$

$$\gamma_h = \operatorname{sech}^{-1} \left(\frac{k w}{\tau + 1.5} \right). \quad (3.12)$$

Substituting (3.7) and (3.8) into (3.2), we obtain

$$\hat{\epsilon} \approx \frac{R}{\sqrt{wv}} \left| \frac{P_{\tau+1}(\hat{w} \cdot \hat{v}) \psi_j (0.5\psi_h - i\psi_h^{-1})}{\sqrt{\tanh \gamma_j \tanh \gamma_h}} \right|, \quad (3.13)$$

which can be used to estimate the relative error for all possible translation and shift vectors.

To find the maximum error given a box size (a), the translation distance is taken as its minimum value (i.e., $w = 2a$ for a one-box-buffer scheme) and the total shift distance is taken as its maximum value ($v = |[a \ a \ a]^T| = a\sqrt{3}$), respectively, as shown in Fig. 3.1.

Note that the error control schemes used in [12–14] assume $|P_{\tau+1}(\hat{w} \cdot \hat{v})| = 1$ as the worst case when estimating the relative error. However, the values of the argument where the absolute value of the Legendre polynomial reaches unity ($\hat{w} \cdot \hat{v} = \pm 1$) are not necessarily the points where the largest errors will occur;

e.g., $\hat{w} \cdot \hat{v} = 1/\sqrt{3}$ for the worst case illustrated in Fig. 2.2. Moreover, the root-mean-square (RMS) value of Legendre polynomials over the range $\hat{w} \cdot \hat{v} \in [-1, 1]$ can be calculated as

$$\sqrt{\int_{-1}^1 P_t^2(z) dz} = \sqrt{\frac{2}{2t+1}}, \quad (3.14)$$

which can be derived using the Rodrigues' formula [15, eq. (8.6.18)], and integration by parts, as discussed in Appendix C. Equation (3.14) shows that assuming $|P_{\tau+1}(\hat{w} \cdot \hat{v})| = 1$ in (3.13) leads to overestimation of the truncation numbers, especially for larger box sizes and/or smaller desired relative errors. Moreover, even slightly overestimated truncation numbers cause a much higher required machine precision, especially for low frequencies because of the increased numerical instability of the Hankel functions for small arguments. As a result, the Legendre polynomial is kept intact in (3.13). More detailed discussions on the behavior of Legendre polynomials are provided in Appendix C.

To the best of our knowledge, (3.13) cannot be solved for τ analytically, due to the inclusion of the Legendre polynomial. Therefore the number of harmonics for a given box size (a) and a desired relative error threshold (ϵ_d) must be found numerically. Moreover, due to the oscillatory nature of Legendre polynomials, there are more than one solution for a given relative error threshold. This behavior is illustrated in Fig. 3.2, in which the relative error estimate as a function of the truncation number is given for an example scenario. Therefore, to provide an accurate upper-bound for the relative error, we define a set of feasible truncation numbers as

$$\tilde{\tau}(a, \epsilon_d) \triangleq \left\{ \tau \in \mathbb{Z}^+ \mid \hat{\epsilon}(a, \tau) < \epsilon_d, \hat{\epsilon}(a, \tau - 1) > \epsilon_d \right\}. \quad (3.15)$$

Then, the optimum truncation number (τ_{opt}) can simply be found as the maximum value of the feasible set as

$$\tau_{opt}(a, \epsilon_d) \triangleq \max(\tilde{\tau}(a, \epsilon_d)). \quad (3.16)$$

From a practical standpoint, (3.15) and (3.16) correspond to finding the zero-crossings of $\epsilon_d - \hat{\epsilon}$, and then selecting the largest τ for which the estimated relative error is below the desired relative error. This approach has a computational

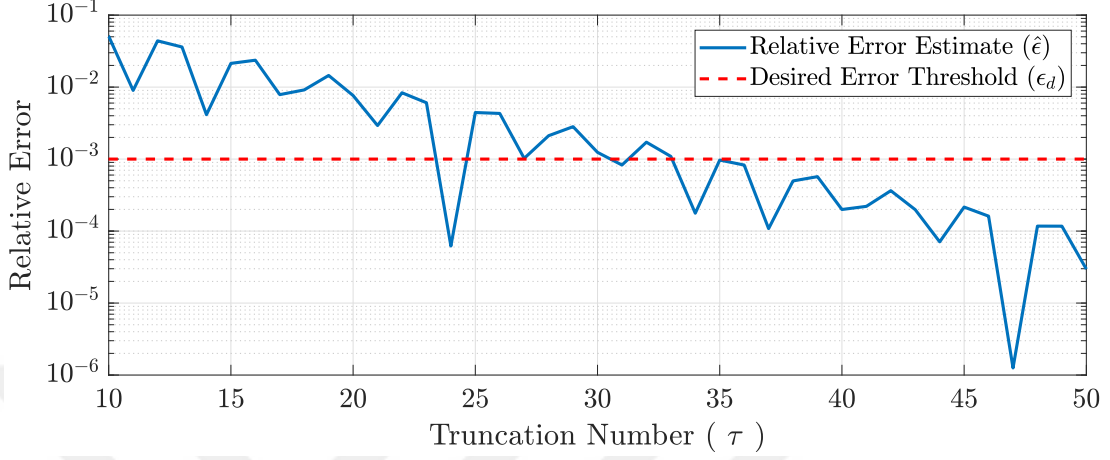


Figure 3.2: Relative error estimate ($\hat{\epsilon}$) for $a = \lambda/32$, $\epsilon_d = 10^{-3}$, $\vec{w} = [0 \ 2a \ 0]^T$, $\vec{v} = \vec{v}_1 + \vec{v}_2 = [a \ a \ a]^T$.

complexity of $O(\tau_{opt})$ and ensures the actual relative error of the translation operator stays below the specified level.

3.2.2 Estimating the Optimum Machine Precision

After finding the optimum number of harmonics (τ_{opt}), the far-zone interactions between boxes are computed using the diagonal form of the Green's function in (2.3). The machine precision must be able to handle each of the individual elementary functions in (2.3), as well as all of the intermediate combinations (i.e., products, summations, integrations) before the final result. Note that the required machine precision is highly dependent on the implementation (order of computation, canceling terms, etc.). An important assumption on the implementation is that the frequency scaling that comes from multiplication by k in (2.3) is performed after the computation is finished, i.e., the first k term in (2.3) is replaced by 2π while estimating the required machine precision.

To represent the worst case as described above in terms of the required machine precision when computing (2.3), we define

$$G^{MP} \triangleq \frac{2\pi N_{\theta\phi} \Delta\theta \Delta\phi}{(4\pi)^2} (\tau + 1)(2\tau + 1) h_{\tau}^{(1)}(kw) P_{\tau}^{MP}. \quad (3.17)$$

In (3.17), P_τ^{MP} is defined to be the value of the Legendre polynomial $P_\tau(\hat{k} \cdot \hat{w})$ that requires the highest machine precision (i.e., its minimum value other than zero) as

$$P_\tau^{MP} \triangleq \left\{ \min_{\hat{k} \in \mathbb{K}^3} \left(\left| P_\tau(\hat{k} \cdot \hat{w}) \right| \right) \mid P_\tau(\hat{k} \cdot \hat{w}) \neq 0 \right\}, \quad (3.18)$$

where $\mathbb{K}^3$ is the set of unit vectors $\hat{k}$ which are defined by the angular sampling in the numerical evaluation of (2.3). The steps taken to obtain (3.17) from (2.3) are as follows: First, the unit amplitude shift operators $\beta(\vec{k}, \vec{v})$ are omitted. Second, assuming all terms in the truncated summation in (2.3) coherently adds up as the worst case, the summation is replaced with a multiplication by the number of terms, i.e., $(\tau + 1)$. Third, assuming the integrand coherently adds up in (2.3), the integral is replaced with a multiplication by $N_{\theta\phi} \Delta\theta \Delta\phi$, where $\Delta\theta$ and $\Delta\phi$ are the grid sizes along θ and ϕ axes, respectively, and $N_{\theta\phi}$ is the total number of angular samples. Note that we use Gauss-Legendre sampling along θ -axis as given in [5] when computing (2.3). However, while constructing (3.17) we assume uniform sampling, which yields

$$\Delta\theta = \Delta\phi = \frac{\pi}{\tau + 1}, \quad (3.19)$$

$$N_{\theta\phi} = 2(\tau + 1)^2. \quad (3.20)$$

Note that uniform integration weights follow the sample mean of Gauss-Legendre weights with a multiplicative factor of $\pi/2$ (i.e., half of the extent of θ -axis), as illustrated in Fig. 3.3. Therefore, uniform integration weights offer a simpler and computationally tractable alternative when computing (3.17).

The expression given in (3.17) includes multiplicative terms both greater and smaller than one, which we define as overflow-critical (G_+^{MP}) and underflow-critical terms (G_-^{MP}), respectively, as

$$G_+^{MP} = 2\pi N_{\theta\phi}(\tau + 1)(2\tau + 1) \max \left(0.5|\psi_h|, |\psi_h^{-1}| \right), \quad (3.21)$$

$$G_-^{MP} = \frac{\Delta\theta \Delta\phi}{(4\pi)^2} \frac{P_\tau^{MP}}{\sqrt{(\tau + 1.5)kw \tanh \gamma_h}}. \quad (3.22)$$

Note the spherical Hankel function in (3.17) is replaced by its large order approximation in (3.8), and only the dominating term of the numerator of (3.8) is taken

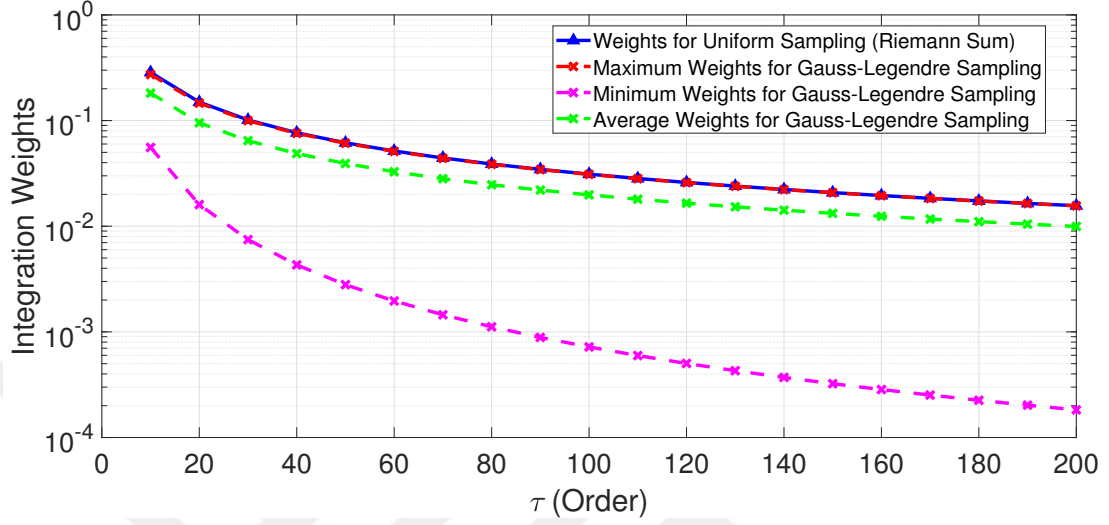


Figure 3.3: Numerical integration weights of uniform and Gauss-Legendre sampling for different truncation numbers τ .

into account ($|\psi_h| \gg 1$ for large boxes and vice versa). In the worst case, the machine precision must be high enough to handle both G_+^{MP} and G_-^{MP} separately. Therefore, the required decimal digits of machine precision for computing (3.17) can be found as

$$\text{MP}_G = \max \left(\log_{10}(G_+^{MP}), -\log_{10}(G_-^{MP}) \right). \quad (3.23)$$

When determining the optimum machine precision, we must also take into account the actual expected amplitude of the Green's function and the desired relative error threshold as

$$\text{MP}_\epsilon \triangleq -\log_{10}(\epsilon_d) + \log_{10}(4\pi R_{max}) + 1, \quad (3.24)$$

where $\epsilon_d \in (0, 1)$ and $4\pi R_{max}$ is the denominator of the free-space Green's function with R_{max} as the maximum value of R for the given box size (for the one-box-buffer scheme, $\vec{v} = [a \ a \ a]^T$ and $R_{max} = a\sqrt{11}$). The $+1$ term in (3.24) is added empirically for safety. Finally, the optimum machine precision (MP_{opt}) for computing (2.3) can be found as

$$\text{MP}_{opt} = \lceil \max(\text{MP}_G, \text{MP}_\epsilon) \rceil. \quad (3.25)$$

To summarize, given a box size (a) and a desired relative error threshold (ϵ_d), (3.13)–(3.16) can be used to find the optimum truncation number (τ_{opt}), while

(3.21)–(3.25) can be used to find the optimum decimal digits of machine precision (MP_{opt}). Note that, (3.13)–(3.16) and (3.21)–(3.25) can also be used to infer the achievable relative errors and the corresponding truncation numbers given the available machine precision.

3.2.3 Validation of the Proposed Error Control Scheme

The proposed error control scheme was implemented in MATLAB, where the MPA environment was constructed using a commercially available toolbox [54]. In order to validate the proposed error control scheme, the following scenarios were investigated:

- Box Size (a): 64λ to $\lambda/2048$ in base-2 logarithmic steps
- Desired Relative Error: $\epsilon_d \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$
- Translation vectors ($\vec{w}$): $[0 \ 2a \ 0]^T$ for the minimum translation distance along the y-axis (see Fig. 3.1) and $[3a \ 3a \ 3a]^T$ for the maximum translation distance for a one-box-buffer scheme (see Fig. 2.2)
- Shift Vectors ($\vec{v} = \vec{v}_1 + \vec{v}_2$): from corners, edge centers, face centers of the source box to those of the observation box (shown in Fig. 3.1)

For each scenario listed above, we calculated the optimum truncation numbers (τ_{opt}) and the optimum decimal digits of machine precision (MP_{opt}) using (3.13)–(3.16) and (3.21)–(3.25), which are reported in Table 3.1 for $\vec{w} = [0 \ 2a \ 0]^T$, and in Table 3.2 for $\vec{w} = [3a \ 3a \ 3a]^T$. Note that for each entry in Tables 3.1 and 3.2, we investigated all critical shift vectors and report the largest τ_{opt} and MP_{opt} pairs.

Using Tables 3.1 and 3.2, the actual relative errors with respect to the free-space Green’s function are given in Figs. 3.4 and 3.5. As demonstrated in Figs. 3.4 and 3.5, the truncation numbers and the machine precisions obtained from the proposed scheme keep the relative errors close to or below the desired levels for

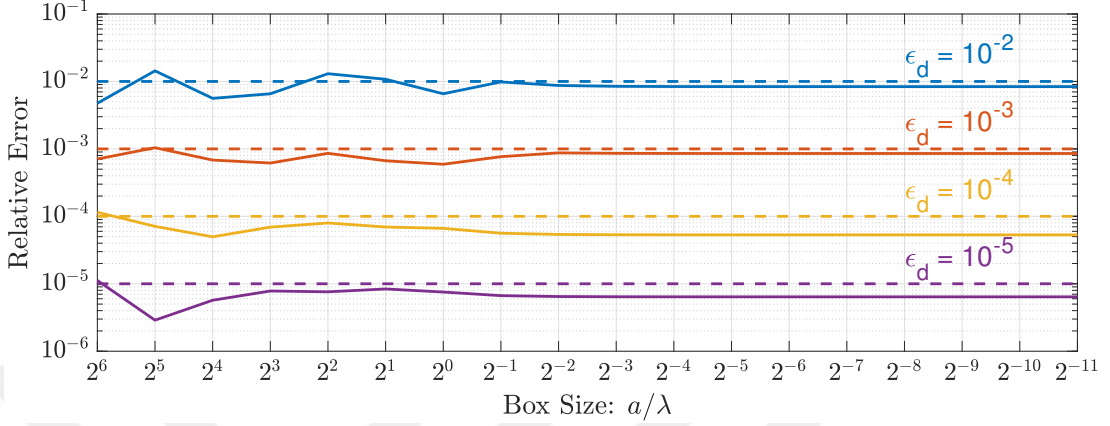


Figure 3.4: Relative errors with respect to free-space Green’s function when Table 3.1 is used in an MPA environment for $\vec{w} = [0 \ 2a \ 0]^T$. Dashed lines represent the desired relative error thresholds.

both large and small boxes. Note that some small oscillations in the actual errors can be observed as the box size increases due to the large order approximations given in (B.12) and (B.13). The large order approximation becomes slightly more erroneous as the arguments of the Bessel and Hankel functions gets closer to their order for very large arguments. However, the error due to the large order approximation for asymptotically large boxes is always bounded, which can be shown analytically by comparing the large order and large argument approximations of the Bessel and Hankel functions (see Appendix B). Therefore, the proposed error control scheme can be used for arbitrarily large box sizes.

An interesting observation for both Table 3.1 and Table 3.2 is that τ_{opt} values for a given error threshold become constant for electrically small boxes. This behavior is also observed in the harmonics of the Green’s function when the multipole expansion is explicitly used as in [21]. Therefore, truncation numbers only depend on the desired relative error thresholds for electrically small boxes. Motivated by this idea, an even simpler error control scheme for small boxes and low-frequency problems will be provided in Section 3.3.

Another important observation from Tables 3.1–3.2 is that there is always a minimum value of MP_{opt} from where the value increases for both increasing and decreasing box sizes. For electrically small boxes, the spherical Hankel function

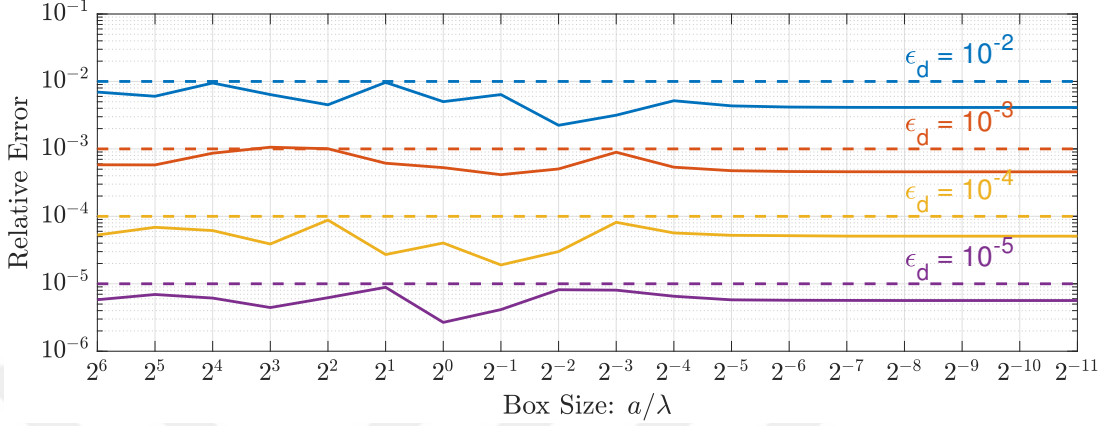


Figure 3.5: Relative errors with respect to free-space Green’s function when Table 3.2 is used in an MPA environment for $\vec{w} = [3a \ 3a \ 3a]^T$. Dashed lines represent the desired relative error thresholds.

in (2.3) dominates every other term and gets larger as the box size decreases asymptotically. For electrically large boxes, the terms due to angular sampling and numerical integration given in (3.19) and (3.20) dominate, which then causes an increase in MP_{opt} as the box size increases asymptotically.

When we compare Table 3.1 and Table 3.2 we observe that as the translation distance (w) increases, the τ_{opt} and MP_{opt} pairs decrease significantly for small boxes, while increasing slightly for large boxes. The behavior for small boxes is again due to the spherical Hankel function dominating the other terms in (2.3). For very small arguments (i.e., electrically small boxes), a relatively small increase in the translation distance from Table 3.1 to Table 3.2 causes a reduction of many orders of magnitude in the value of the spherical Hankel function, leading to dramatic reductions in the estimates of both τ_{opt} and MP_{opt} . On the other hand, the slight increase of MP_{opt} for the larger boxes is due to the first term in the right hand side of (3.13) increasing for larger translation distances.

The τ_{opt} and MP_{opt} pairs obtained with our proposed scheme also agree well with the previous studies found in the literature. Fig. 3.6 compares the proposed scheme with the well-known EBF [12–14] when $\epsilon_d = 10^{-2}$ and $\vec{w} = [0 \ 2a \ 0]^T$. Since EBF is not valid for small boxes, truncation numbers found via numerical simulations in [27] are also shown in Fig. 3.6. The proposed scheme agrees very

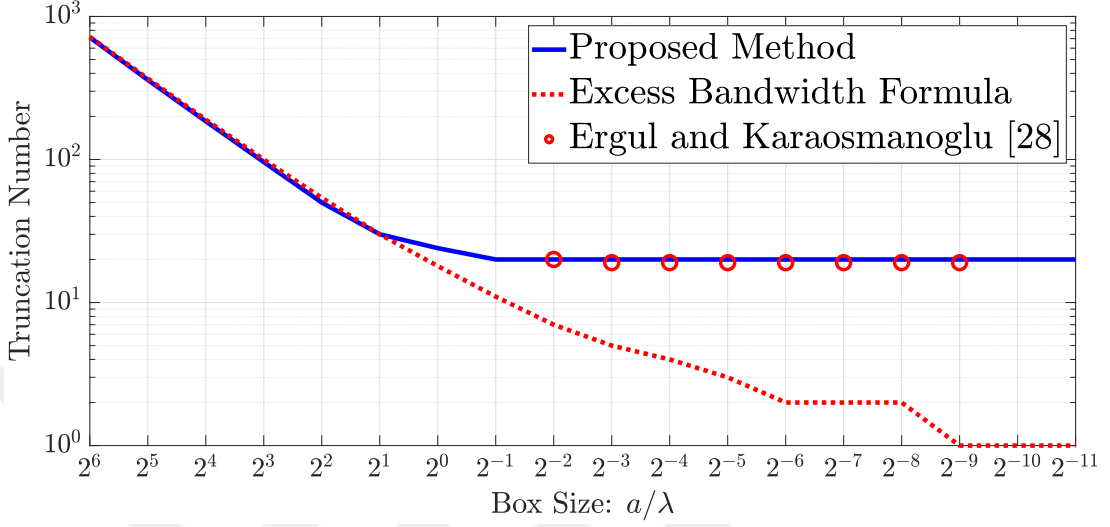


Figure 3.6: Comparison of the truncation numbers found by the proposed scheme to [12–14] and [27] for $\epsilon_d = 10^{-2}$ and $\vec{w} = [0 \ 2a \ 0]^T$.

well with EBF for electrically large boxes while being valid for electrically small box sizes. A similar comparison is given in Fig. 3.7, where the optimum machine precisions given in Table 3.1 for $\epsilon_d = 10^{-2}$ and machine precisions found numerically in [27] are shown. The proposed scheme estimates slightly larger MP_{opt} values while still following the trend found in [27]. This is expected since our method assumes the worst case in terms of the implementation for the optimum machine precision, leading to MP_{opt} estimates that are greater than or equal to experimental values.

The novel error control scheme presented in this section is published by our team in [55].

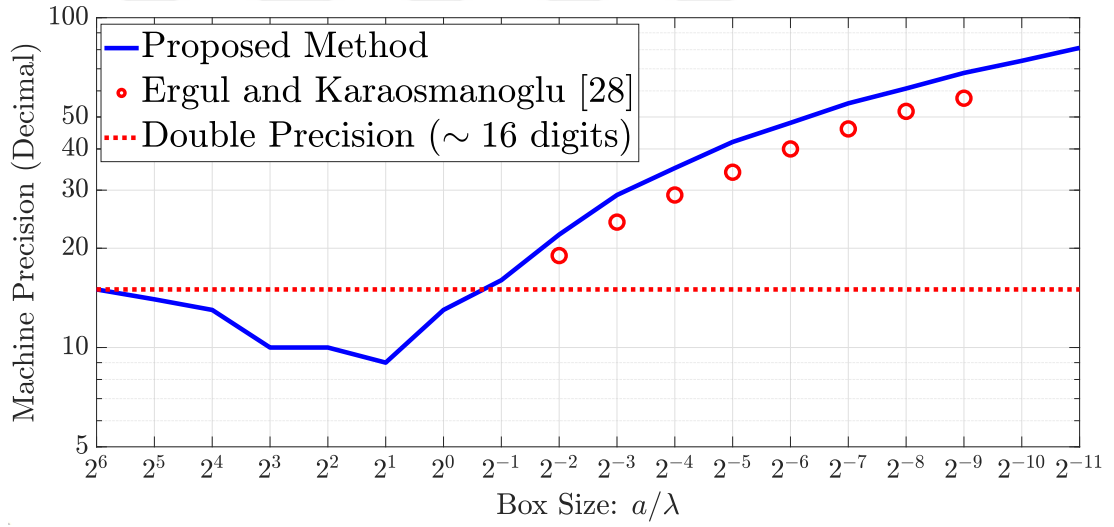


Figure 3.7: Comparison of the machine precisions found by the proposed scheme to double precision and [27] for $\epsilon_d = 10^{-2}$ and $\vec{w} = [0 \ 2a \ 0]^T$.

Table 3.1: Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for Various Desired Relative Error Thresholds (ϵ_d) and Box Sizes (a) when $\vec{w} = [0 \ 2a \ 0]^T$

Error Limit→	10^{-2}		10^{-3}		10^{-4}		10^{-5}	
Box Size ↓	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}
64λ	714	15	721	15	728	15	734	15
32λ	360	14	368	14	375	14	382	14
16λ	185	13	191	13	197	13	201	13
8λ	96	10	102	10	106	11	115	13
4λ	50	10	59	11	69	14	80	19
2λ	30	9	43	15	56	23	69	32
λ	24	13	37	22	50	34	66	50
$\lambda/2$	20	16	36	32	50	49	66	70
$\lambda/4$	20	22	34	41	50	64	66	90
$\lambda/8$	20	29	34	51	50	80	66	110
$\lambda/16$	20	35	34	62	50	95	66	131
$\lambda/32$	20	42	34	73	50	111	66	151
$\lambda/64$	20	48	34	83	50	126	66	171
$\lambda/128$	20	55	34	94	50	142	66	192
$\lambda/256$	20	61	34	105	50	157	66	212
$\lambda/512$	20	68	34	116	50	173	66	232
$\lambda/1024$	20	74	34	126	50	189	66	253
$\lambda/2048$	20	81	34	137	50	204	66	273

Table 3.2: Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for Various Desired Relative Error Thresholds (ϵ_d) and Box Sizes (a) when $\vec{w} = [3a \ 3a \ 3a]^T$

Error Limit→	10^{-2}		10^{-3}		10^{-4}		10^{-5}	
Box Size ↓	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}
64λ	724	15	732	15	739	15	745	15
32λ	370	14	376	14	381	14	386	14
16λ	190	14	195	14	200	14	204	14
8λ	100	11	103	11	108	13	111	13
4λ	54	10	56	10	59	10	62	10
2λ	29	8	32	9	35	9	36	10
λ	17	8	19	8	21	9	23	10
$\lambda/2$	10	7	12	8	14	9	15	10
$\lambda/4$	7	7	8	8	10	9	11	10
$\lambda/8$	5	7	6	8	8	9	10	10
$\lambda/16$	4	7	6	9	8	11	10	13
$\lambda/32$	4	8	6	11	8	14	10	17
$\lambda/64$	4	10	6	13	8	16	10	20
$\lambda/128$	4	11	6	15	8	19	10	24
$\lambda/256$	4	13	6	18	8	22	10	27
$\lambda/512$	4	15	6	20	8	25	10	31
$\lambda/1024$	4	17	6	22	8	28	10	34
$\lambda/2048$	4	18	6	25	8	31	10	38

3.3 Simplified Expressions for Universal Error Control of MLFMA

Truncation numbers computed with (3.13) for a given desired error threshold agree well with EBF for large box sizes and high-frequency problems. However, for small box sizes and low-frequency problems EBF fails as it relies on the large-argument approximation of Bessel and Hankel functions. On the other hand, the requirement for a multiple-precision framework and a lack of closed-form solution complicates the use of (3.13) for most MLFMA users. Therefore, in this section we derive simplified closed-form expressions that do not require a multiple-precision environment for small boxes and low-frequency problems, for the first time in the literature. Together with widely used EBF at high-frequencies as their counterpart, the novel closed-form expressions allow easy error control of MLFMA for all frequencies and translation distances.

To simplify (3.2) for low-frequency problems where $kw \ll 1$ and $, we use the small-argument approximations of the Bessel and Hankel functions given in (B.10)–(B.11). For the simplification of Legendre polynomials, we use the RMS value of Legendre polynomials given in (3.14). Substituting the simplified forms of Bessel/ Hankel functions and Legendre polynomials into (3.2), we obtain the approximate truncation error for low-frequency problems as$

$$\hat{\epsilon} \approx \frac{R}{\sqrt{wv}} \left(\frac{v}{w}\right)^{\tau+1.5} \sqrt{\frac{2}{2\tau+3}}. \quad (3.26)$$

Note that the closed-form expression in (3.26) can be used to estimate the truncation error for different translation and shift distances. However, the common approach in practice is to select the highest possible truncation number for the worst case in terms of error, which corresponds to the one-box-buffer scheme. Therefore, we can further simplify (3.26) by assuming a one-box-buffer scheme and substituting $\vec{w} = [0 \ 2a \ 0]^T$ and $\vec{v} = [a \ a \ a]^T$, resulting in

$$\hat{\epsilon} \approx 1.782 \left(\frac{\sqrt{3}}{2}\right)^{\tau+1.5} \sqrt{\frac{2}{2\tau+3}}, \quad (3.27)$$

where the direction of $\vec{w}$ in a one-box-buffer scheme does not affect the result. The closed-form expression in (3.27) can be used as a complement to EBF for

small boxes to estimate the worst-case errors in a one-box-buffer scheme. On the other hand, the closed-form expression in (3.26) can be used if different truncation numbers are to be used for different translation distances. An important insight for (3.26) and (3.27) is that they do not depend on the box sizes. This behavior is consistent with the low-frequency harmonics of the Green’s function when the multipole expansion is explicitly used [21], as discussed before.

The expressions given in (3.26) and (3.27) still have room for improvement. Specifically, they still require numerical implementations to search for truncation numbers for a given desired error threshold, and may suffer from inaccuracies due to the RMS approximation of Legendre polynomials. Therefore, we provide yet another alternative for the low-frequency scenarios in the form of curve-fitted functions of (3.13), which we consider to be the gold standard among all other alternatives presented in this work.

To generate the data for curve-fitting, we have computed truncation numbers using (3.13) for the following range of parameters:

- desired error threshold: $\epsilon_d \in [3.3\text{e-}1, 1\text{e-}5]$,
- box sizes: $a \in [\lambda/2048, \lambda]$,
- translation vectors $\vec{w}$: all possible vectors within a parent box with symmetries removed (17 unique cases),
- shift vectors $\vec{v}$: from corners, edge centers, face centers of the source box to those of the observation box (62 unique cases).

After the required truncation numbers for the above scenarios are computed, we assume that the truncation error only depends on the translation distance normalized to the box size, i.e., w/a , consistent with our findings of (3.26) and (3.27). As an example, computed truncation numbers for a box size of $\lambda/2048$ for some w/a values are illustrated in Figure 3.8.

A clear observation from Fig. 3.8 is that the relation between d_0 and τ can be closely represented by an affine transformation, where the slope and offset

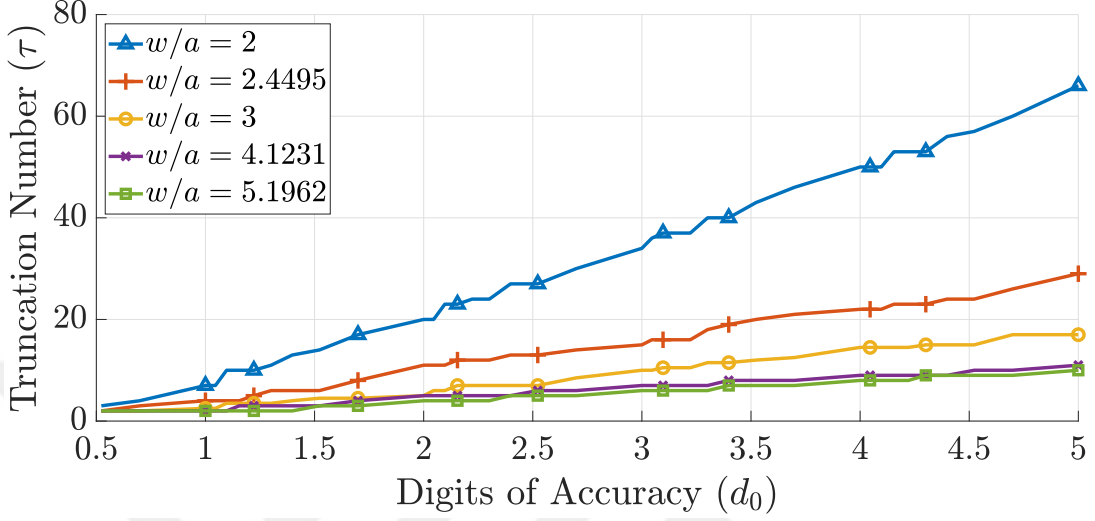


Figure 3.8: Truncation numbers obtained via (3.13) for $a = \lambda/2048$.

parameters of the transformation being functions of w/a as

$$\tau \approx f_1(w/a) d_0 + f_2(w/a). \quad (3.28)$$

Then the slope, $f_1(\cdot)$, and the offset, $f_2(\cdot)$, of the affine transformation that depend on w/a are approximated by double exponential functions that are empirically found to be the best fit as

$$f_1(x) = 3.073e4 \times e^{-4.064x} + 10.34 \times e^{-0.3562x}, \quad (3.29)$$

$$f_2(x) = -28.63 \times e^{-1.071x} - 1.168e8 \times e^{-8.620x}. \quad (3.30)$$

Note that the closed-form expressions given in (3.28)–(3.30) can be used as an alternative to (3.26), if different truncation numbers are to be used for different translation distances within the same parent box of MLFMA. Similar to the derivation of (3.27), we can simplify (3.28) for the maximum error of the one-box-buffer scheme with $w = 2a$, to obtain

$$\tau \approx 14.14 d_0 - 7.17, \quad (3.31)$$

which is an extremely simple closed-form expression to estimate the required truncation numbers for small boxes and low-frequency problems. Note that (3.31) will only yield values larger than unity for $d_0 > 0.58$ or $\epsilon_d < 0.26$ due to its affine nature, which should be useful for almost every scenario.

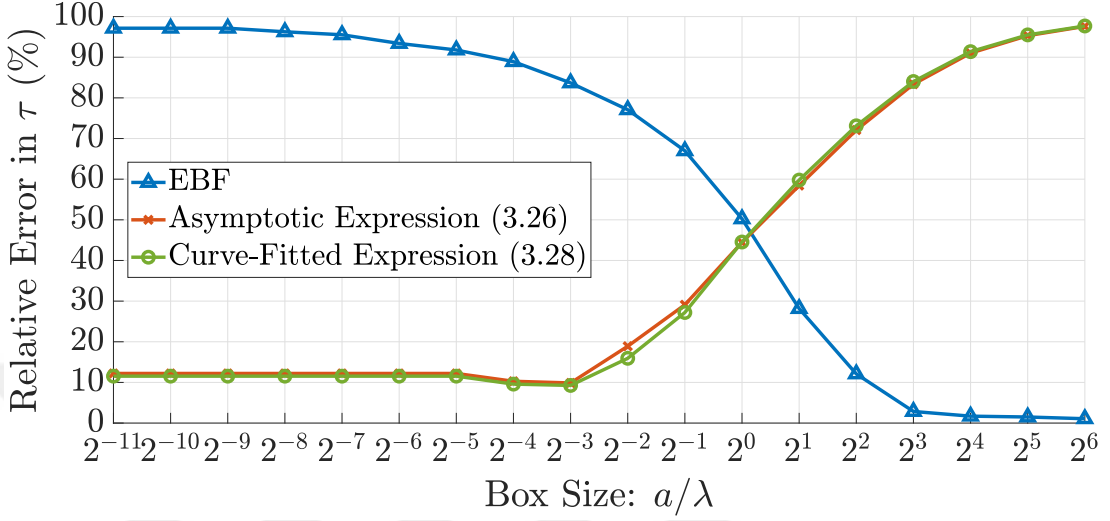


Figure 3.9: Relative error of truncation numbers obtained via EBF [12–14], (3.26), and (3.28) for different box sizes and translation distances.

To illustrate the accuracy of the closed-form expressions as well as EBF given in (3.3), we analyzed the average relative error and RMS error (RMSE) of the truncation numbers over the same ϵ_d , $\vec{w}$, $\vec{v}$ values we have listed previously, with an even greater range of box sizes that include electrically large boxes. The errors for the truncation numbers are shown in Figs. 3.9–3.10 for the variable-translation case with (3.26) and (3.28), and in Figs. 3.11–3.12 for the one-box-buffer scheme using (3.27) and (3.31). For all cases, RMSE values for EBF is calculated for only $w = 2a$, as EBF is not able to handle different translation distances within the same parent box of MLFMA.

As seen in Figs. 3.9–3.12, EBF achieves relatively low errors for large boxes and the proposed closed-form expressions (3.26)–(3.27) and (3.28)–(3.31) perform well for small boxes. Note that although the proposed expressions for low-frequencies exhibit a slightly large relative error, they perform much better in terms of RMSE. This is due to truncation numbers being much smaller at low-frequencies compared to high-frequencies. For example, even an RMSE of 1.2 for $a = \lambda/2048$ has led to a relative error of 11% in Figs. 3.9–3.10. The converse is true for EBF at high-frequencies where the truncation numbers are very large, leading to relatively high RMSE values although their relative errors are very low. Another important observation from Figs. 3.9–3.12 is that the curve-fitted expressions

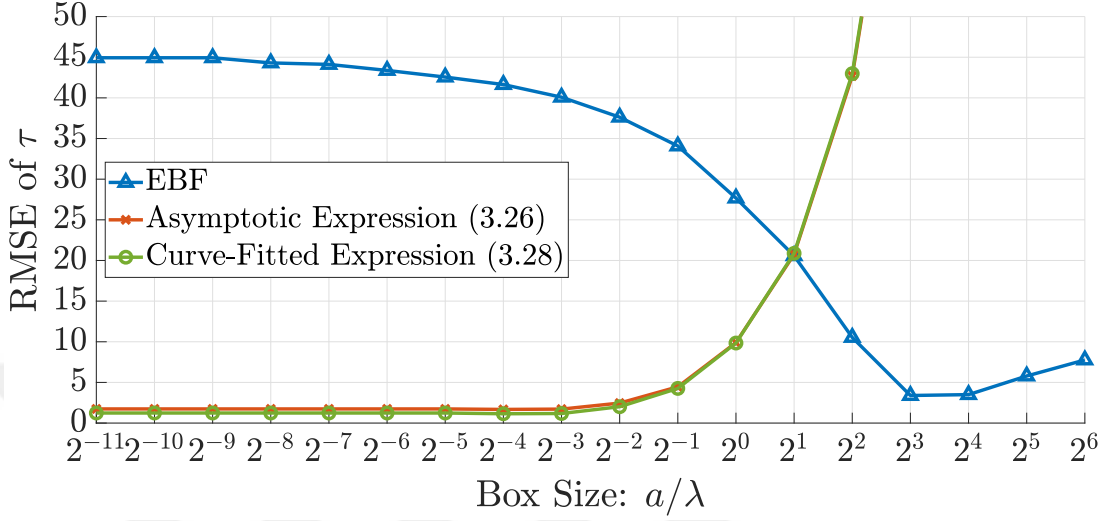


Figure 3.10: RMSE of truncation numbers obtained via EBF [12–14], (3.26), and (3.28) for different box sizes and translation distances.

outperforms the asymptotic expressions, especially for the one-box-buffer scheme in Figs. 3.11–3.12, as they are based on the accurate error control expression in (3.13). As the truncation numbers are selected for the one-box-buffer scheme in practice, Figs. 3.11–3.12 show the power of the extremely simple expression we have provided in (3.31).

As seen in Figs. 3.11–3.12, it is best to use the proposed closed-form expressions for relatively small box sizes, i.e., $a < 2.8\lambda$, whereas EBF should be used exclusively for $a > 2.8\lambda$, where $a = 2.8\lambda$ is approximately the point where the errors of EBF and small-box expressions coincide. Also, the errors are relatively larger in the region $a \in [\lambda, 4\lambda]$ for the combined usage of small-box expressions and EBF. Therefore, users should be aware that these expressions are relatively erroneous in this particular region. We would like to emphasize again that the error control formulation of (3.13) is valid for all box sizes, and should be used if a multiple-precision framework could be implemented by the users. Since a multiple-precision framework may not be available for most users, we provide detailed design curves that can be used to estimate the achievable errors of MLFMA for commonly available fixed-precision computers in Section 5.1.

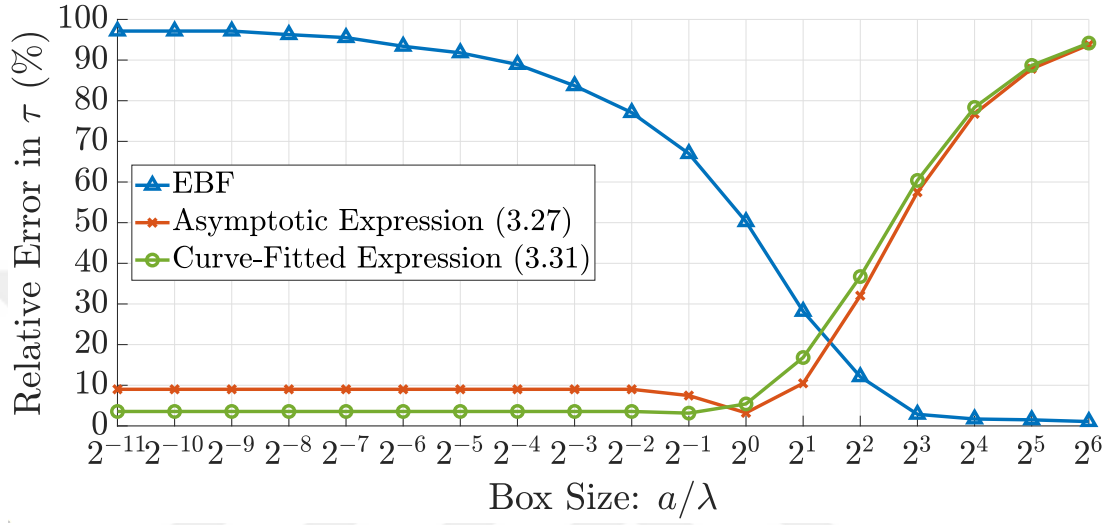


Figure 3.11: Relative error of truncation numbers obtained via EBF [12–14], (3.27), and (3.31) for different box sizes for a one-box-buffer scheme.

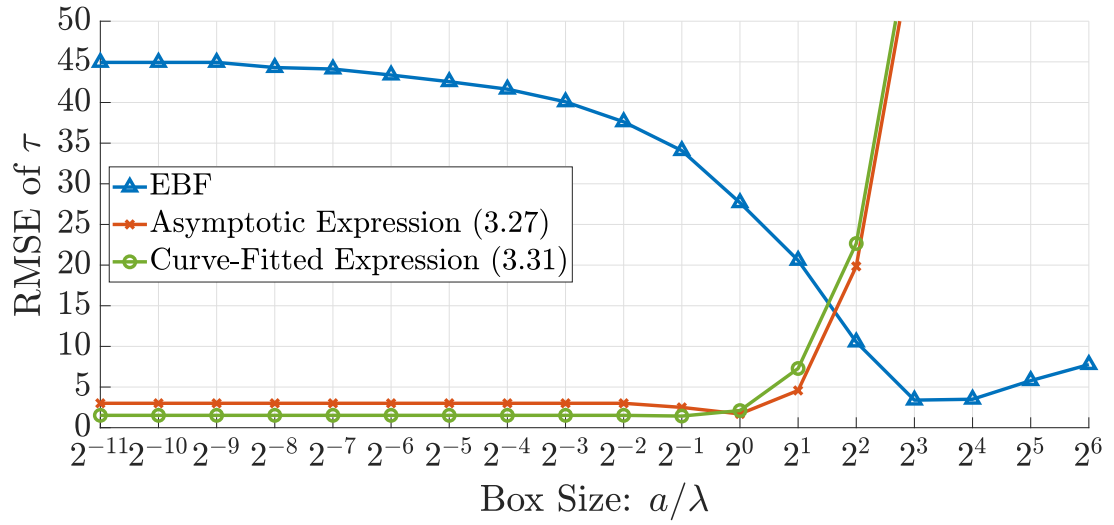


Figure 3.12: RMSE of truncation numbers obtained via EBF [12–14], (3.27), and (3.31) for different box sizes for a one-box-buffer scheme.

Chapter 4

Multiple-Precision Multilevel Fast Multipole Algorithm (MP-MLFMA)

We now introduce MP-MLFMA to enable efficient and accurate broadband solutions of multi-scale electromagnetic problems with deep tree-structures. With MP-MLFMA, problems with relatively large boxes and translation distances can be handled efficiently at low-precision, while electrically small boxes can be handled accurately without any low-frequency breakdown at high-precision. Note that while defining MP-MLFMA, the integral equation on which the algorithm is implemented is intentionally omitted in both Chapter 2 and in this chapter. Depending on the type of problem, both surface and volume integral formulations with a pertinent definition of basis and testing functions are compatible with MP-MLFMA, essentially changing only the matrix equation structure and the far-field radiation patterns of the basis functions $\overline{S}_n(\vec{k}, \vec{r}_n)$ in (2.6) and the testing functions $\overline{F}_m(\vec{k}, \vec{r}_m)$ in (2.9). Therefore, we focus only on the implementation of MP-MLFMA in this chapter, regardless of the specific integral equation formulation used to solve any particular problem.

In the following sections, we first introduce a new notation for MP-MLFMA

that also takes into account the variable machine precision of at each level, before we present the implementation steps of MP-MLFMA in detail. Then we discuss the implications of a software- or hardware-based MP-MLFMA, to present an outlook into the future implementations as well as the true potential of MP-MLFMA.

4.1 MP-MLFMA Notation

Given a problem geometry that is discretized into basis and testing functions, MP-MLFMA starts with the recursive division of the full extent of the geometry into cubic boxes as shown in Fig. 4.1.

The number of levels is denoted by N , where $N \geq 3$ to allow far-zone interaction computations with the diagonal form of the Green's function (2.3), as shown in Fig. 2.2. The level index is defined as L , where $L = 1$ and $L = N$ correspond to the mother level (largest box) and the leaf level (smallest box), respectively. For the L 'th level in MP-MLFMA, we define the set of boxes at that particular level as

$$\mathbb{B}_L = \{B_L | B_L \text{ is a box at Level-}L\}, \quad (4.1)$$

where B_L is an individual box at Level- L , with an edge length of a_L . To define the parent-child relations across levels, we define the children set of B_L , as

$$\check{\mathbb{B}}_L = \begin{cases} \{B_{L+1} | B_{L+1} \text{ is inside } B_L\}, & \text{if } 1 \leq L < N, \\ \{\bar{b}_n, \bar{t}_m | \bar{b}_n, \bar{t}_m \text{ is inside } B_L\}, & \text{if } L = N, \end{cases} \quad (4.2)$$

where $\bar{b}_n$ represents the basis functions discretizing the whole geometry, with n as the basis index. Testing functions are represented by $\bar{t}_m$ with m as the test function index. Similarly, we define the parent box of B_L , $\bar{b}_n$, or $\bar{t}_m$ as

$$\hat{B}_L = \begin{cases} \{B_{L-1} | B_{L-1} \in \mathbb{B}_{L-1}, \text{ and } B_L \in \check{\mathbb{B}}_{L-1}\}, & \text{if } 1 \leq L < N, \\ \{B_{L-1} | B_{L-1} \in \mathbb{B}_{L-1}, \text{ and } \bar{b}_n, \bar{t}_m \in \check{\mathbb{B}}_{L-1}\}, & \text{if } L = N. \end{cases} \quad (4.3)$$

The position vectors for the center points of each box, basis, and testing function are defined as $\vec{r}_{B_L}$, $\vec{r}_n$, and $\vec{r}_m$, respectively. Depending on the integral equation

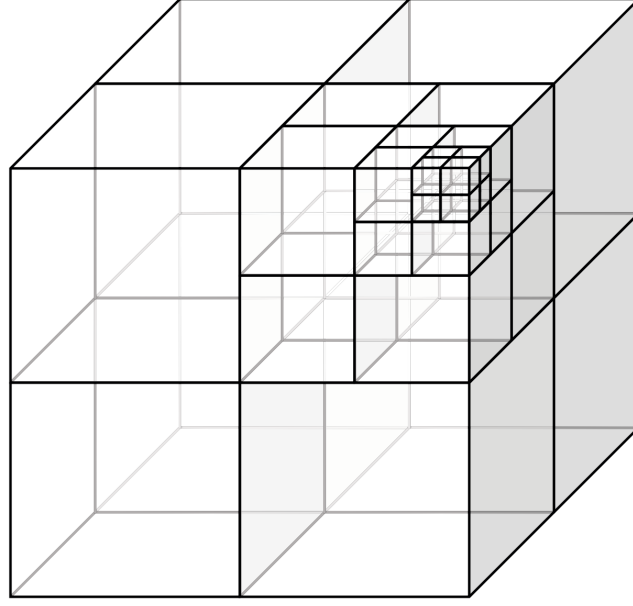


Figure 4.1: An example of the recursive division into cubic boxes at $N = 5$. (Source: Ergül [52])

applied while defining the basis and testing functions, $\overline{S}_n$ and $\overline{F}_m$ correspond to the far-field radiation patterns of the basis and the testing functions, respectively. To define the far-field boxes for a particular box at the same level, we use the set $\mathbb{F}_{B_L}$ to represent the one-box-buffer scheme as

$$\mathbb{F}_{B_L} = \left\{ B_{L'} \mid |\vec{r}_{B_L} - \vec{r}_{B_{L'}}| \geq 2a_L, \left| \vec{r}_{\hat{B}_L} - \vec{r}_{\hat{B}_{L'}} \right| < 2a_{L-1} \right\}. \quad (4.4)$$

Note that the far-field boxes in (4.4) are defined such that the parent boxes are in the near-field of each other, according to the one-box-buffer scheme.

At the start of each iteration for the iterative matrix solution algorithm, we define $\overline{\gamma}[n]$ as the next coefficient vector for the basis functions, which will presumably represent the correct solution after the iterative solution converges.

Using the error control scheme defined in Section 3.2 through equations (3.13) and (3.25), we define truncation numbers and required machine precisions for each level, represented by τ_L and MP_L , respectively. Note that, although we present simplified expressions in Section 3.3 that can be used accurately on a fixed-precision computer, the novel error control scheme defined in Section 3.2 is

still considered to be the gold standard; and should be used if an MPA framework is available. With the introduction of different machine precisions at different levels, the mathematical expressions we use must represent the machine precision at which the expressions are computed. We define the following notation to represent a function $f(\cdot)$ computed at a machine precision of MP_L as

$$f(\cdot)^{@MP_L}. \quad (4.5)$$

Note that the superscript implies that the function is to be computed at MP_L , and all input arguments are assumed to be converted to the MP_L precision before computation of $f(\cdot)$. When there are multiple distinct machine precision levels within the same expression, such as

$$q(\cdot)^{@MP_k} = h\left(f(\cdot)^{@MP_l}, g(\cdot)^{@MP_m}\right), \quad (4.6)$$

it is assumed that $f(\cdot)$ and $g(\cdot)$ are computed at their specified precision, then converted to MP_k before computation of the final result through the function $h(\cdot)$. Note that, the left-hand-side (LHS) of any multiple-precision equation like (4.6) must specifically denote the resulting machine precision to prevent any ambiguity.

Finally, we note that the MP-MLFMA is implemented inside a broader MoM algorithm that is assumed to be running at the native precision of the computing platform (e.g., 64-bits or 16 decimal digits for double-precision). To differentiate the operations implemented at the native precision of the computer, we use the superscript $@NP$ instead of the $@MP_L$ superscripts. Detailed discussion on the standard floating-point precision definitions are provided in Appendix A.

4.2 Proposed Implementation of MP-MLFMA

4.2.1 Error Control of MP-MLFMA

MP-MLFMA starts with the recursive division of the problem geometry into a tree structure as shown in Fig. 4.1. Given the tree structure and the desired

relative error for the free-space Green's function (also the relative error for the translation operator), we use the error control scheme given in Chapter 3 to estimate the required truncation numbers and the corresponding decimal digits of machine precision, as given in **Algorithm 1**.

Algorithm 1: *Error Control of MP-MLFMA*

INPUT:

D : Maximum model dimension along all Cartesian axes

N : Number of levels in MLFMA ($N \geq 3$)

ϵ_d : Desired relative error threshold for free-space Green's function

OUTPUT:

$\bar{\tau} \in \mathbb{Z}^{(N-2) \times 1}$: List of number of harmonics for each level for $N \geq 3$.

$\overline{\mathbf{MP}} \in \mathbb{Z}^{(N-2) \times 1}$: List of decimal machine precisions for each level for $N \geq 3$

1. Initialize $\bar{\tau} = \emptyset$, $\overline{\mathbf{MP}} = \emptyset$
 2. Find the box sizes for each level: $\mathbf{a} = [a_1, a_2, \dots, a_N]$, $a_1 = D$, $a_{L+1} = a_L/2$.
 3. **for** $L = 3, \dots, N$
 4. Set translation and shift vectors for max. error:
 $\vec{w} = [0, 2a_L, 0]^T$ and $\vec{v} = [a_L, a_L, a_L]^T$
 5. Use (3.13) with a_L and ϵ_d to find relative errors for a range of harmonics
 6. Use (3.15)-(3.16) to find required number of harmonics: τ_L
 7. Concatenate to the list of harmonics: $\bar{\tau} = \bar{\tau} \cup \{\tau_L\}$
 8. Use (3.21)-(3.25) to estimate the required machine precision: MP_L
 9. Concatenate to the list of machine precisions: $\overline{\mathbf{MP}} = \overline{\mathbf{MP}} \cup \{MP_L\}$
 10. **endfor**
 11. **return** $\bar{\tau}$ and $\overline{\mathbf{MP}}$
-
-

Using the outputs of **Algorithm 1**, we can now define the aggregation, translation, and disaggregation stages of MP-MLFMA at distinct machine precisions across levels.

4.2.2 Aggregation

In the aggregation stage of MP-MLFMA, given the next set of coefficients $\bar{\gamma}[n]$ for the basis functions, the radiated fields for each box B_L at levels $L = 3, \dots, N$ are computed recursively starting from the leaf-level ($L = N$) as

$$\bar{S}_{B_L}(\vec{k}, \vec{r}_{B_L})^{\textcircled{MP}_L} = \begin{cases} \sum_{\vec{b}_n \in \mathbb{B}_N} \bar{\gamma}[n]^{\textcircled{NP}} \beta(\vec{k}, \vec{r}_{B_L} - \vec{r}_n)^{\textcircled{MP}_L} \bar{S}_n(\vec{k}, \vec{r}_n)^{\textcircled{MP}_L}, & L = N, \\ \sum_{B_{L+1} \in \mathbb{B}_L} \beta(\vec{k}, \vec{r}_{B_L} - \vec{r}_{B_{L+1}})^{\textcircled{MP}_L} \bar{S}_{B_{L+1}}(\vec{k}, \vec{r}_{B_{L+1}})^{\textcircled{MP}_{L+1}}, & 3 \leq L < N. \end{cases} \quad (4.7)$$

Note that in (4.7), the next excitation vector $\bar{\gamma}[n]$ is first held at native precision (NP), then converted into the corresponding machine precision at level $L = N$. For all other MP-MLFMA levels, aggregation function is implemented at a machine precision of MP_{L+1} , then the final result is converted to MP_L .

4.2.3 Translation

After the computation of aggregated fields using (4.7), translations are computed for boxes B_L with available far-zone interactions with boxes $B_L' \in \mathbb{F}_{B_L}$, at levels $L = 3, \dots, N$, given by

$$\bar{G}_{B_L}(\vec{k}, \vec{r}_{B_L})^{\textcircled{MP}_L} = \sum_{B_L' \in \mathbb{F}_{B_L}} \alpha_{\tau_L}(\vec{k}, \vec{r}_{B_L} - \vec{r}_{B_L'})^{\textcircled{MP}_L} \bar{S}_{B_L'}(\vec{k}, \vec{r}_{B_L'})^{\textcircled{MP}_L}, \quad (4.8)$$

where $\alpha_{\tau_L}(\cdot)$ is the translation operator with a truncation number of τ_L , as defined in (2.5). Note that, since the translation operation is applied across boxes within the same level, all operations are performed at the same machine precision, i.e., MP_L at level L .

4.2.4 Disaggregation

Translated fields $\overline{G}_{B_L}(\vec{k}, \vec{r}_{B_L})$ are distributed to lower level boxes recursively from top to bottom, to obtain the disaggregated result, $\overline{G}_{B_{L+1}}^+(\vec{k}, \vec{r}_{B_{L+1}})$ as

$$\begin{aligned} \overline{G}_{B_{L+1}}^+(\vec{k}, \vec{r}_{B_{L+1}})^{\textcircled{MP_{L+1}}} &= \overline{G}_{B_{L+1}}(\vec{k}, \vec{r}_{B_{L+1}})^{\textcircled{MP_{L+1}}} \\ &+ \beta(\vec{k}, \vec{r}_{B_{L+1}} - \vec{r}_{B_L})^{\textcircled{MP_L}} \overline{G}_{B_L}^+(\vec{k}, \vec{r}_{B_L})^{\textcircled{MP_L}}, \text{ if } B_{L+1} \in \check{\mathbb{B}}_L. \end{aligned} \quad (4.9)$$

Since the disaggregation operation is performed across levels, translated fields at the higher level are converted to the lower level's MP_{L+1} precision before computation of the final disaggregated fields.

After the aggregation, translation, and disaggregation stages are completed, the MVM component for far-zone interactions is computed by multiplying the disaggregated fields at the leaf-level with the testing function radiation patterns $\overline{F}_m(\vec{k}, \vec{r}_m)$ and integrating over the radiation sphere, given by

$$\begin{aligned} \left(\sum_{n=1}^N \overline{\overline{Z}}_{FF}[m, n] \overline{\gamma}[n] \right)^{\textcircled{NP}} &= \left(\frac{ik}{4\pi} \right)^2 \times \\ &\int d^2\hat{k} \overline{F}_m(\vec{k}, \vec{r}_m)^{\textcircled{MP_N}} \cdot \beta(\vec{k}, \vec{r}_m - \vec{r}_{B_N})^{\textcircled{MP_N}} \overline{G}_{B_N}^+(\vec{k}, \vec{r}_{B_N})^{\textcircled{MP_N}}, \text{ if } \vec{r}_m \in \check{\mathbb{B}}_N. \end{aligned} \quad (4.10)$$

Note that the final MVM result is converted to the native precision of the computing platform after combination of the disaggregated fields. The resulting MVM is then used across the iterations of an iterative matrix equation solver such as GMRES [53], similar to the implementation of the fixed-precision MLFMA.

4.3 Discussions on a Software-Defined MP-MLFMA

The MPA toolbox [54] implements the variable-precision framework in MATLAB by using a *soft* implementation. This means that the native double-precision

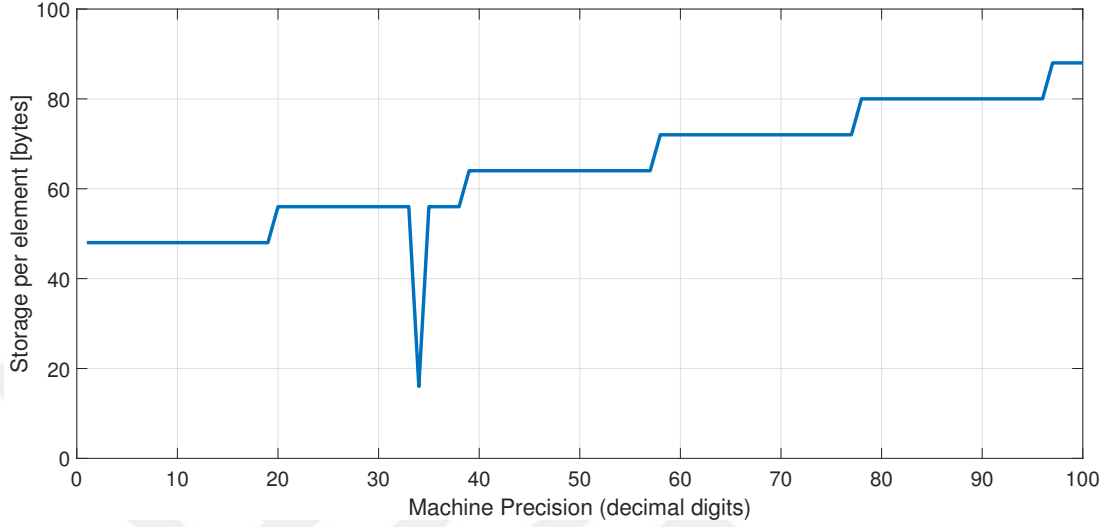


Figure 4.2: Memory usage (in bytes) for a single matrix element at different machine precisions, using [54].

(64-bits) architecture of most of the contemporary CPUs is kept intact while the variable precision is handled at the software level. Although soft implementations are always required for arbitrarily large machine precision levels (arithmetic logic units - ALUs cannot handle arbitrarily large numbers), lower machine precision levels should be implemented at a hardware level for actual efficiency. The toolbox [54] uses the full ALU registers for just a single addition/multiplication operation whether the numbers to be operated are double-precision or less. To illustrate this, we have performed MVM operations on 1000×1000 matrices with randomly generated elements on an Intel Core i7-7500 CPU at 2.70GHz, and reported the average solution times and memory usage per matrix element over 100 trials. The results for the memory usage and computation times are given in Fig. 4.2 and Fig. 4.3, respectively.

As seen in Fig. 4.2, the toolbox uses the same memory allocation per element until after the double-precision (16 decimal digits), and increases the allocation in a staircase manner for increased precision levels. The reduced memory at 34-digits is claimed to be a special dedicated implementation for quadruple-precision arithmetics in [54], and is not relevant to this discussion. The same behavior is observed in Fig. 4.3, where the MVM computation times follow a similar trend to the memory usage, with low-precision levels not reducing the computation times

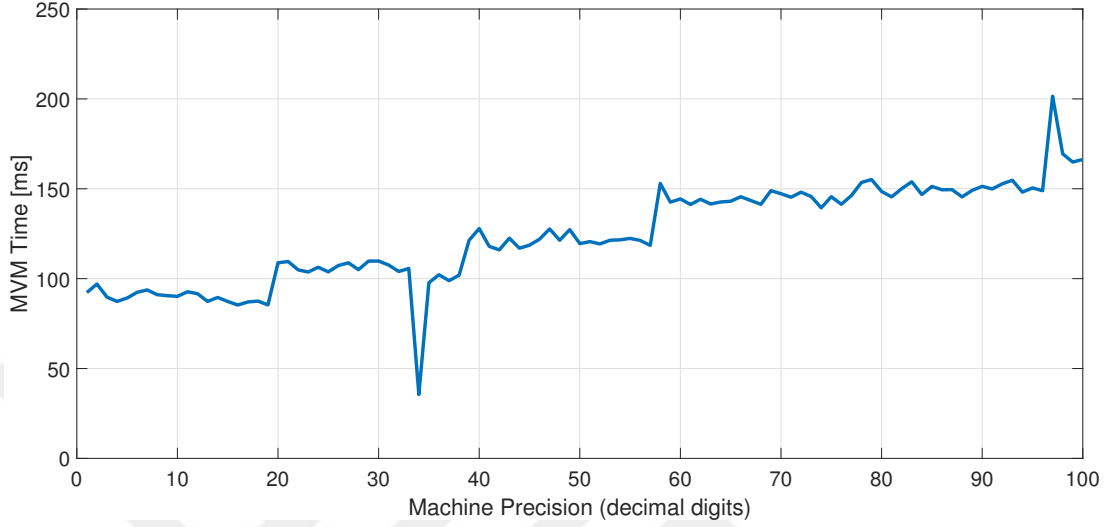


Figure 4.3: Computation time (in milliseconds) for a 1000×1000 MVM at different machine precisions averaged across 100 trials, using [54].

below the native double-precision.

To provide specific performance metrics for MLFMA, we analyzed the CPU-times and the required random access memory (RAM) for the computation of the diagonal form of the Green’s function in (2.3), which inherently includes the setup, aggregation, translation, and disaggregation stages. These simulations were performed on a workstation with 24-core Xeon® E5-2650 processor.

To illustrate the computational overhead introduced by the MPA, we analyzed the CPU-times and allocated memories for the translation operation at varying box sizes in a one-box-buffer scheme. To have a fair comparison with standard double precision, we computed the diagonal form of the Green’s functions using the same truncation numbers obtained from Table 3.1 for both double precision and MPA. The truncation numbers and the required machine precision levels for the analyzed cases are reported again in Table 4.1, for clarity. Note that MPA computations were performed in the corresponding machine precision given in Table 4.1. In Fig. 4.4 and Fig. 4.5, we present the computing requirements (blue for CPU-time, green for memory) and the achieved relative errors (red) for $\epsilon_d = 10^{-2}$, and $\epsilon_d = 10^{-5}$, respectively.

Table 4.1: Optimum Truncation Numbers (τ_{opt}) and Machine Precisions (MP_{opt}) for the CPU-time & Memory Test

Error Limit→	10^{-2}		10^{-5}	
Box Size ↓	τ_{opt}	MP_{opt}	τ_{opt}	MP_{opt}
16λ	185	13	201	13
8λ	96	10	115	13
4λ	50	10	80	19
2λ	30	9	69	32
λ	24	13	66	50
$\lambda/2$	20	16	66	70
$\lambda/4$	20	22	66	90
$\lambda/8$	20	29	66	110
$\lambda/16$	20	35	66	131
$\lambda/32$	20	42	66	151
$\lambda/64$	20	48	66	171
$\lambda/128$	20	55	66	192
$\lambda/256$	20	61	66	212
$\lambda/512$	20	68	66	232
$\lambda/1024$	20	74	66	253
$\lambda/2048$	20	81	66	273

An expected observation from Figs. 4.4–4.5 is that the achieved errors for $\epsilon_d = 10^{-2}$ and $\epsilon_d = 10^{-5}$ *blow up* once the box sizes are smaller than 0.5λ and 4λ , respectively. Another important observation from Figs. 4.4–4.5 is that there is a relatively constant overhead introduced by the MPA toolbox even for double or lower precisions, consistent with our findings in Figs. 4.2–4.3. Moreover, the CPU and RAM requirements increase as the box size increases, since more and more terms need to be included in the summations and integrations in (2.3). For low-frequencies or small boxes, the CPU and RAM requirements are relatively constant and only increases slightly for increasing machine precision (e.g., in Fig. 4.5 the machine precision increases from 13 to 273 as the box size goes from 64λ to $\lambda/2048$).

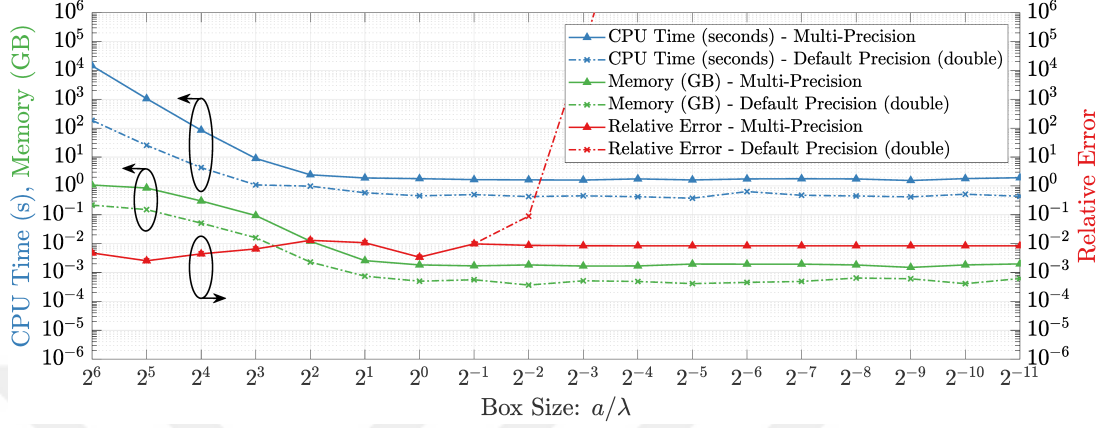


Figure 4.4: Comparison of CPU-times, allocated memory, and achieved relative errors for varying box sizes when the truncation numbers in Table 3.1 is used for $\epsilon_d = 10^{-2}$ with double precision and MPA (averaged over 10 runs).

To isolate the effect of machine precision on the CPU and RAM requirements, we analyzed the same far-zone interaction for a box size of $a = \lambda/2048$ using different machine precision levels with $\epsilon_d = 10^{-5}$. As shown in Fig. 4.6, the increasing trends in both CPU-times and RAM are close to linear, which is expected for an MPA implementation. As the machine precision is increased from 15 to 273 (18.2 times), The CPU-times increased approximately 5.9 times while the RAM increased approximately 3.4 times, both of which are much smaller than the increase in the machine precision. Also note that the staircase behavior previously shown in Fig. 4.2 is also observed in the RAM requirement in Fig. 4.6, again due to the pre-allocation scheme of the third-party toolbox.

In this section, we show that there is a clear overhead in terms of both CPU-times and RAM requirements when MLFMA is implemented within a software-defined MPA framework. However, we note that classical MLFMA with double precision simply does not work at low-frequencies or very small desired errors for even large boxes as shown in Figs. 4.4–4.5. For all cases in which the double-precision works (i.e., where $MP < MP_{opt}$ in Tables 3.1 and 3.2), MLFMA still can be implemented in standard double precision. Moreover, we note that the third-party toolbox implements the MPA framework at the software level, while a hardware implementation would obviously be more efficient and would introduce less overhead, especially at low-precision levels.

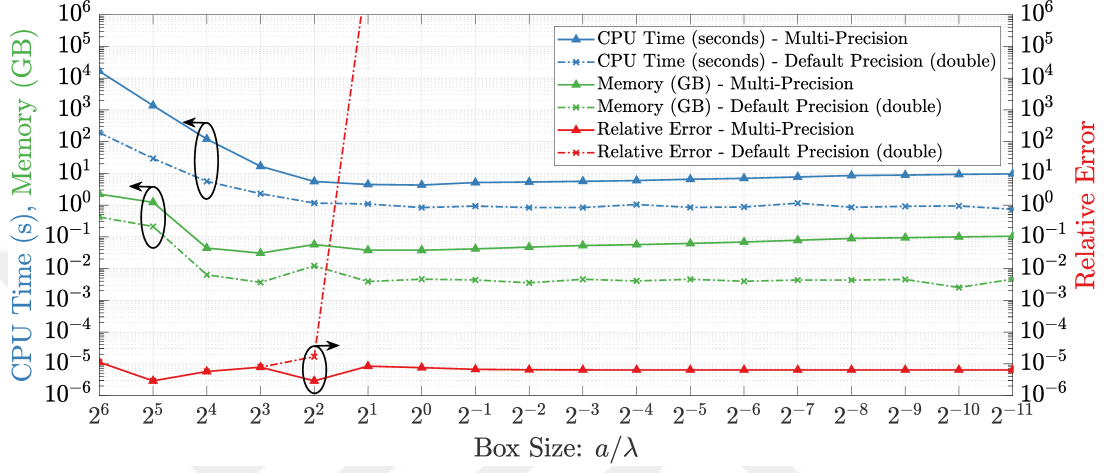


Figure 4.5: Comparison of CPU-times, allocated memory, and achieved relative errors for varying box sizes when the truncation numbers in Table 3.1 is used for $\epsilon_d = 10^{-5}$ with double precision and MPA (averaged over 10 runs).

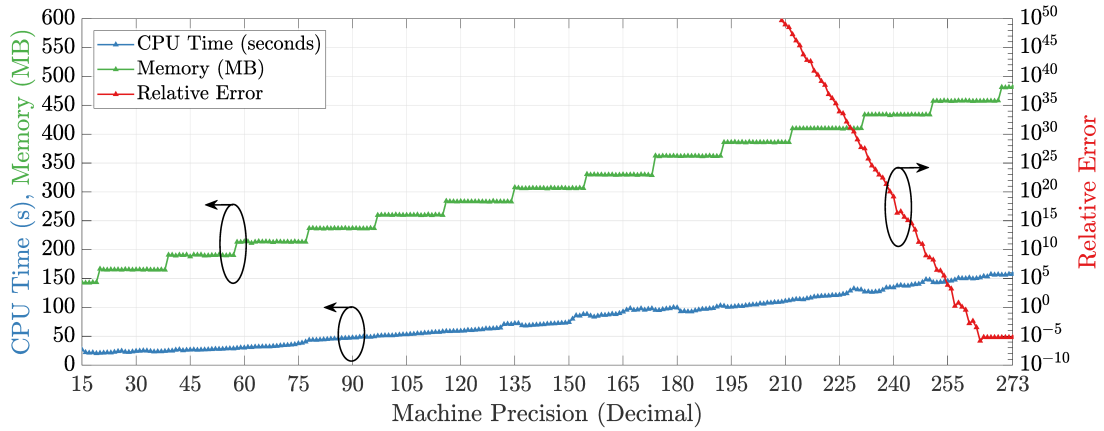


Figure 4.6: CPU-times, allocated memory, and achieved relative errors for $a = \lambda/2048$, $\epsilon_d = 10^{-5}$, $\tau_{opt} = 66$, using MPA at different machine precisions (averaged over 10 runs).

4.4 Discussions on a Hardware-Defined MP-MLFMA

As discussed in Section 4.3, a software-defined MP-MLFMA framework must be implemented if an arbitrarily low error threshold and/or frequency is to be solved, as those problems require arbitrarily large machine precision levels. On the other hand, a main attraction of MP-MLFMA is the potential increase in efficiency for high-frequency problems that may be solved with precisions less than the native precision of the computing platform. Since software-defined MPA frameworks do not offer increased efficiency for low-precision operations, a hardware-defined MPA framework must be employed.

Although low-precision computations for MLFMA problems are a novel idea presented in this dissertation, low-precision computation itself is hardly a new idea. Recent research on the low-precision computation is mostly focused on accelerating the training of machine learning and deep learning convolutional neural networks [56, 57]. The low-precision tensor cores offered by NVIDIA[®] GPUs at 16-bits and 32-bits, combined with the native 64-bits for Tesla[®] architectures allow inherent low-precision computations at the hardware-level without any need for a software-defined MPA framework. There are also application specific integrated circuits (ASICs) in development for variable precision arithmetic computations [58–60], offering extreme leaps in performance at the significantly larger cost of ASIC manufacturing. Another class of low-precision implementation is to perform concurrent low-precision operations in a high-precision processor [61], offering inexpensive solutions with available computing platforms.

An efficient hardware-level implementation on a fixed high-precision processor should address both memory and ALU operations in a smart way. For memory operations, the staircase behavior in Fig. 4.2 should be extended to lower precisions by storing/reading the numbers in a group-by-group manner. For example, a single 64-bit register that can store a double-precision number can also store four half-precision numbers with 8-bits each. This implementation leads to continuation of the piece-wise constant RAM requirement versus the machine

precision towards lower precisions, while preserving the overarching linear trend.

With careful implementation, computation speeds can also be greatly improved. For CPU operations, the ALUs of the computing platform should be modified to allow parallel operations of lower precision numbers simultaneously. In the next section, we present a simple implementation of low-precision MPA architecture on an x86 CPU for addition and multiplication operations.

4.5 Hardware-Defined MPA for Addition & Multiplication

We built a hardware-defined MPA implementation for addition and multiplication operations on an x86 CPU platform with native single-precision ALUs (32-bits). For addition operations in single-precision ALUs, we implemented a simultaneous addition scheme at 4-5-7-9-15-32 bits. For multiplication operations we implemented simultaneous multiplication 2-5-32 bits. Multiplication requires at least double the number of bits for each number; moreover, the cross products occupy some space in the registers as well. The bit allocation schemes for the parallel adder and the parallel multiplier are given in Figs. 4.7–4.8, respectively, to illustrate the limitations of a single 32-bit register.

As shown in Fig. 4.8, possible implementations for low-precision scenarios are severely limited compared to the addition case in Fig. 4.7, where only a single carry bit is required additionally.

Using the parallelization schemes presented in Figs. 4.7–4.8, we generated 80 random number pairs, triplets, etc. to be added/multiplied and re-run the computations 1,000,000 times in a single-core of the x86 CPU. Average addition and multiplication times *per number pair* versus the number of bits are illustrated in Fig. 4.9. Note that the additional memory read-write times arising from parallelization are also included in these plots.

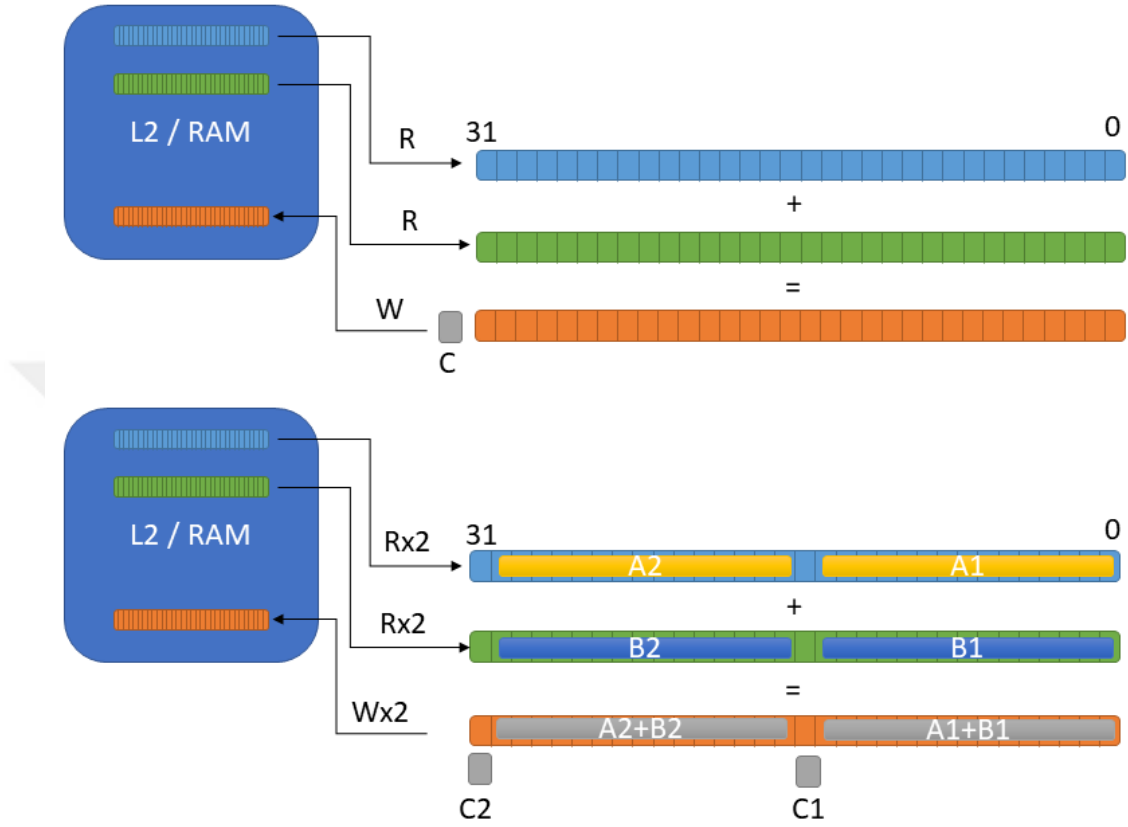


Figure 4.7: Parallel addition architecture for low-precision operations. W, R corresponds to write and read operations, respectively.

As shown in Fig. 4.9, a significant improvement in CPU-times are possible with the hardware-level implementation. Note that both the addition and multiplication operations show a near-linear improvement in computation time with respect to the number of bits of the operands. If combined with a compiler capable of distributing the complex operations of MP-MLFMA into the simple MPA operations defined here, a hardware-level MPA framework offers enormous potential advantages in terms of both computing times and memory requirements.

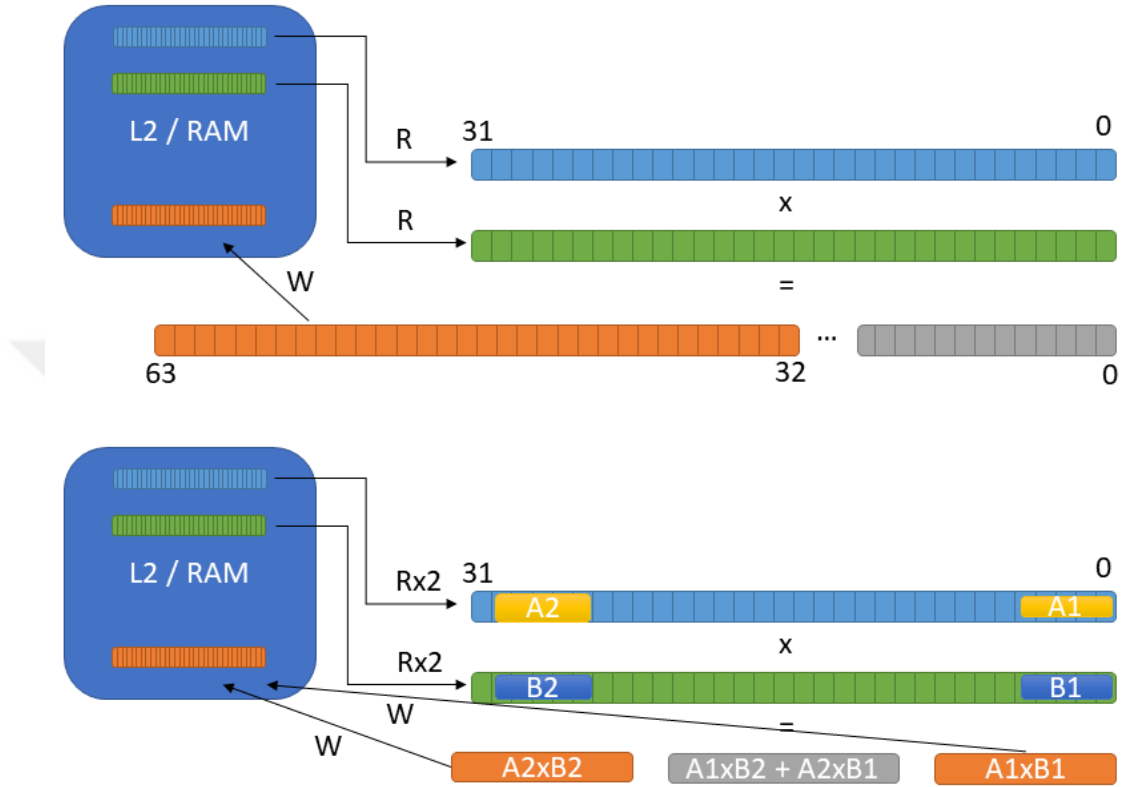


Figure 4.8: Parallel multiplication architecture for low-precision operations. W, R corresponds to write and read operations, respectively.

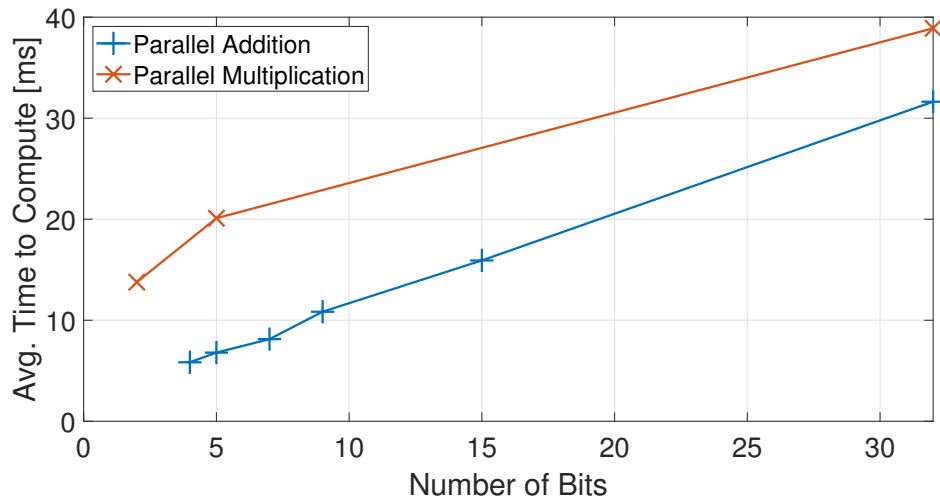


Figure 4.9: Average addition/multiplication times in a hardware-level MPA implementation on an x86 platform

Chapter 5

Numerical Results

In Chapters 3–4, we introduced the novel error control scheme and the MP-MLFMA that enable accurate and efficient analysis of multi-scale deep tree-structure problems over arbitrarily large frequency bandwidths. The novel error control scheme can be used for both MP-MLFMA and the conventional MLFMA at all frequencies, which is not available in the previous state-of-the-art. Combined with the error control scheme in Chapter 3, MP-MLFMA introduced in Chapter 4 can be used to solve extremely large problems at any frequency, offering increased efficiency for high-frequency (electrically large boxes) and accuracy for low-frequency problems (electrically small boxes). In the following sections, we provide design curves for the error control of MLFMA for conventional or MPA implementations, then we validate the MP-MLFMA on a variety of broadband and deep tree-structure problems.

5.1 Error Analysis of MLFMA

The simplified error control scheme with closed-form expressions given in Section 3.3, together with the machine precision estimation method given in Section 3.2 can be used to estimate the minimum achievable errors (denoted as ϵ_{min})

or minimum translation distances for a given fixed machine precision. Note that in this section the machine precision levels are provided as decimal digits of precision in a floating-point standard [28], where single-, double-, and quadruple-precision correspond to approximately 7, 16, and 34 decimal digits of precision, respectively. Detailed discussion on the floating-point standard used here is provided in Appendix A.

Using (3.28)–(3.31) for sub-wavelength box sizes ($a < \lambda$) and (3.13) for all other cases and desired errors larger than $\epsilon_d > 0.26$, we present three design curves to provide a helpful guideline for MLFMA users. Note that although EBF is valid for large boxes, it slightly overestimates the truncation numbers due to simplification of the Legendre polynomials’ exact value as unity [12], while (3.28)–(3.31) do not suffer from such a problem. Therefore, to provide the most accurate design curves possible, we have used (3.13) for box sizes greater than one wavelength. After we introduce the design curves, we will illustrate their validity for a sample scattering problem over the widely used single-precision (32-bits) and double-precision (64-bits) computers.

5.1.1 Design Curves for Error Control of MLMFA

Firstly, we investigate the minimum achievable error versus the box size for various machine precision levels in a one-box-buffer scheme (i.e., $w = 2a$).

Fig. 5.1 shows the achievable error bounds of MLFMA on fixed-precision computing platforms for a wide range of box sizes. For scattering and radiation problems, where only far-zone radiation characteristics are of interest, relatively large errors in the Green’s function can be tolerated. For example, assuming an error bound of $\epsilon = 0.33$, both single- and double-precision platforms are able to handle sub-wavelength box sizes (a minimum of $2^{-4}\lambda$ and $2^{-11}\lambda$, respectively). On the other hand, for problems where accurate current distributions and network parameters are required (e.g., antenna problems), relatively low error bounds for the Green’s function must be enforced. For example, if an error bound of $\epsilon = 0.01$ is selected, both single- and double-precision fail to handle sub-wavelength box

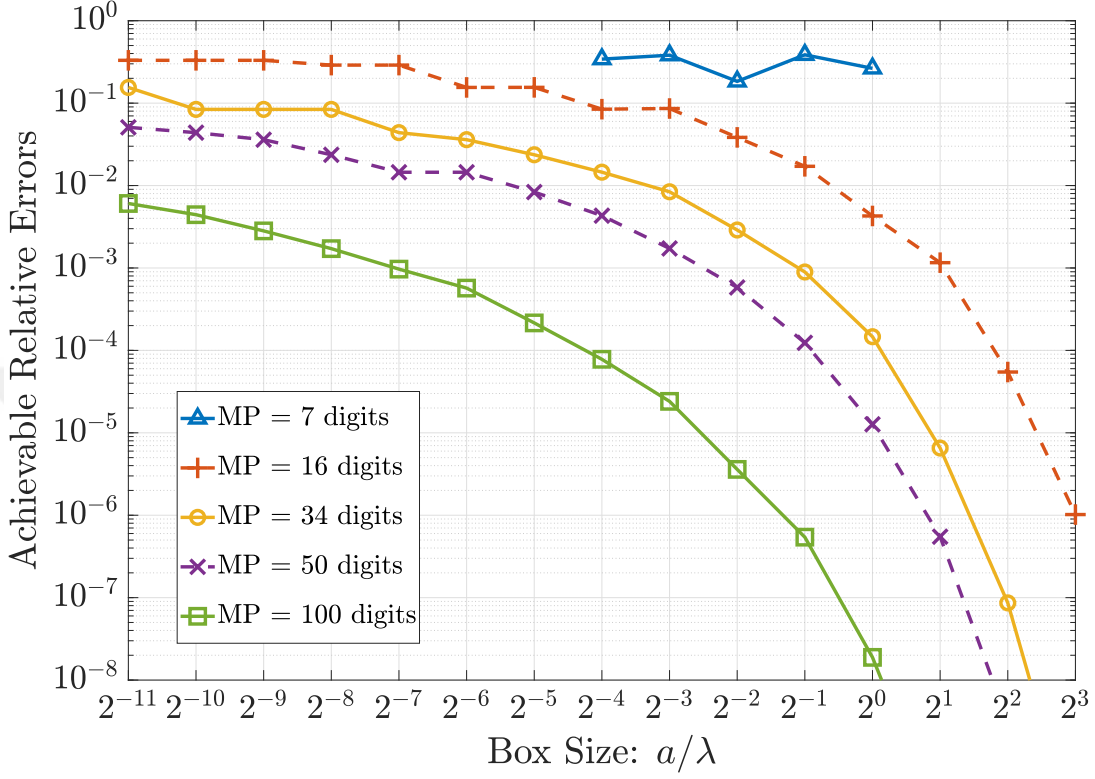


Figure 5.1: Minimum achievable errors versus the box size in a one-box-buffer scheme (i.e., $w = |\vec{w}| = 2a$).

sizes, as seen in Fig. 5.1. As a result, higher precision levels may be required to solve antenna problems with large numbers of MLFMA levels that include sub-wavelength translation distances.

An important observation to note in Fig. 5.1 is that, for the single-precision case (7 decimal digits), the minimum achievable errors are only reported for $a \in [2^{-4}\lambda, \lambda]$, meaning that the error is not bounded outside this region. Specifically, single-precision is not enough to handle the spherical Hankel functions, leading to low-frequency breakdown for box sizes smaller than $2^{-4}\lambda$. On the other hand, for box sizes larger than λ , relatively small numerical integration weights cannot be handled by single-precision as the number of spherical harmonics increases. Nonetheless, single-precision can possibly handle larger boxes (and achieve lower errors) if special care is taken to prevent overflow/underflow during the computation of the translation operator, as explained in Section 3.2.

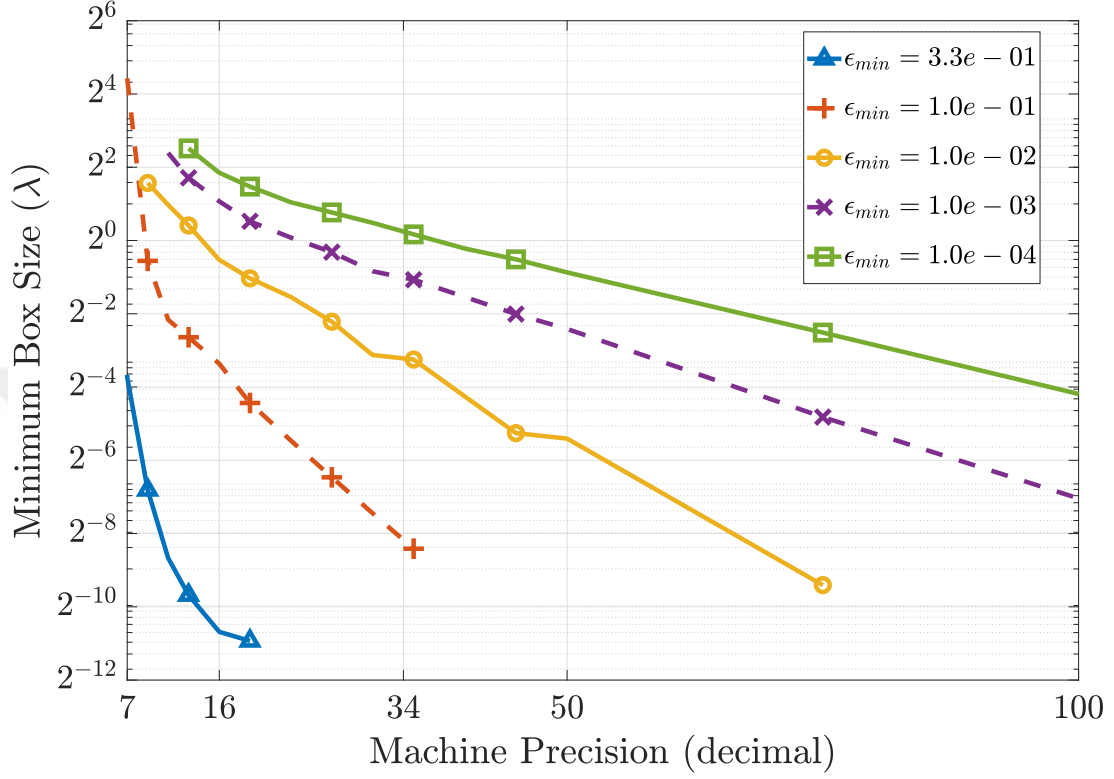


Figure 5.2: Minimum box size versus the machine precision to achieve different desired error thresholds for $w = |\vec{w}| = 2a$, i.e., the one-box-buffer scheme.

Shown in Fig. 5.2, our second design curve illustrates the minimum box size that can be used in MLFMA given a desired relative error threshold.

In addition to demonstrating new information on the achievable error levels in MLFMA, the results in Fig. 5.2 further verifies the conclusions drawn from Fig. 5.1, i.e., single-precision and double-precision fail to accurately handle sub-wavelength boxes for relatively low error levels, and they should only be considered for larger box sizes or problems where large errors of Green's function can be tolerated. Also note that the graphs only show the box sizes included in the analysis (i.e., $a \in [2^{-11}\lambda, 2^4\lambda]$), and the minimum box sizes are expected to decrease further as the machine precision increases.

In our third and final design curve, we illustrate the minimum box size that can be handled for a given translation distance in Fig. 5.3. Different translation distances are selected among possible far-zone translations inside a parent box

of an MLFMA tree structure. As expected, Fig. 5.3 shows that the achievable minimum box size decreases as the magnitude of the translation vector gets larger, since the overflow problem due to the spherical Hankel function in the translation operator becomes less severe as w increases [55]. Thus, lower precision levels can be used for large translation distances even if higher precision levels may be required for the one-box-buffer scheme. Also, similar to Fig. 5.2, the minimum box sizes are expected to decrease when higher machine precision levels are used, though they are limited to the analyzed box sizes in Fig. 5.3.

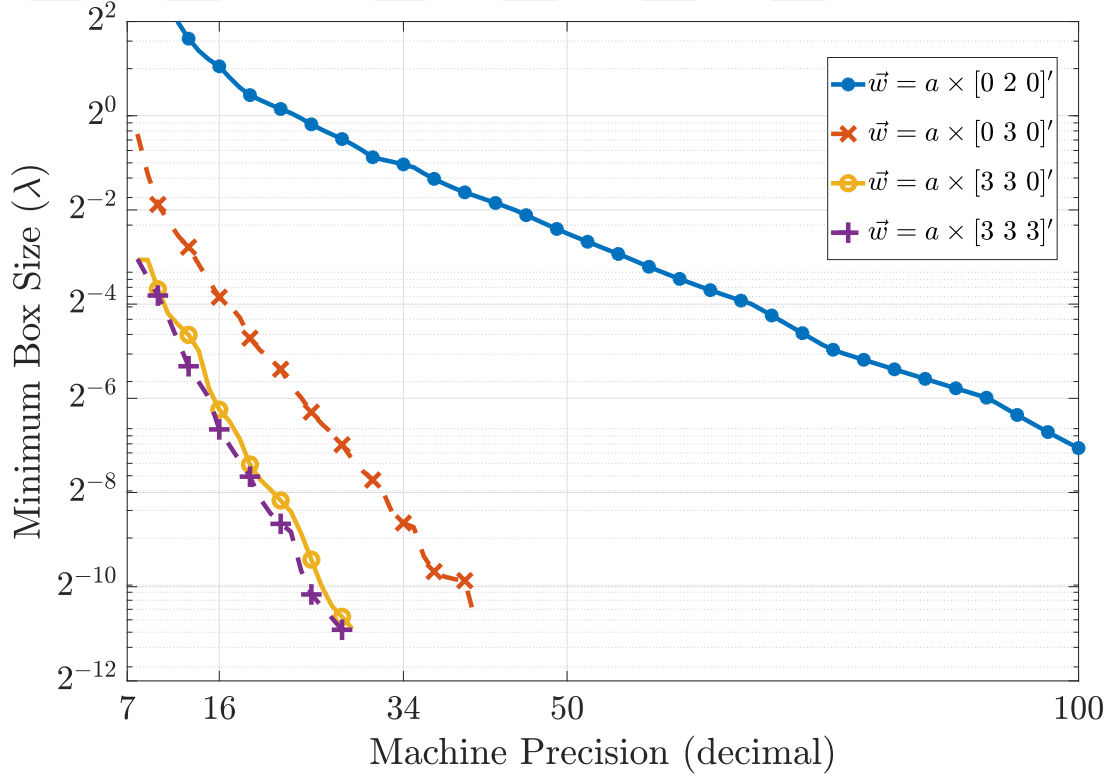


Figure 5.3: Minimum box size versus the machine precision for different translation vectors to achieve a desired error of $\epsilon_{min} = 10^{-3}$.

5.1.2 Validation of the Proposed Design Curves

To illustrate the powerful insights provided by the proposed design curves in Section 5.1.1, a plane-wave scattering problem for a PEC sphere is solved with MLFMA using different desired error thresholds. The PEC sphere used in the analyses has a diameter of 0.6m, and the triangular mesh elements have an edge length of 25mm. Rao-Wilson-Glisson (RWG) functions [62] are used in a Galerkin scheme. The sphere is illuminated by a linearly polarized plane wave with an amplitude of 1V/m. A combined field integral equation (CFIE) formulation is used with a combination parameter of $\alpha = 0.5$. At 250MHz, a 3-level MLFMA tree structure is formed, where the leaf-level box size becomes $\lambda/16$. This is approximately the minimum box size for single- and double-precision floating-point arithmetics at an achievable error of $\epsilon_{min} = 0.33$ and $\epsilon_{min} = 0.10$, respectively, as shown in Fig. 5.2.

The scattering problem described above is solved at three different desired error limits, namely $\epsilon = 0.33$, $\epsilon = 0.10$ and $\epsilon = 0.01$, and at single- and double-precision. Using (3.13) for $\epsilon = 0.33$ and (3.31) for $\epsilon = 0.10$ and $\epsilon = 0.01$, corresponding numbers of spherical harmonics to achieve the desired error thresholds are found to be 3, 7 and 21, respectively. The resulting co-polarized far-zone electric field intensities (scaled to a distance of 1m) and Mie series solutions as the gold standard are presented in Fig. 5.4 and Fig. 5.5.

As predicted by the design curves given in Figs. 5.1–5.3, the scattering results diverge from the Mie series solution below $\epsilon = 0.33$ for single-precision, and below $\epsilon = 0.1$ for double-precision. Note that the solutions for $\epsilon = 0.01$ for both precision levels actually did not converge due to overflow, and their results are repeated in insets of Figs. 5.4–5.5 at a different scale for better visualization. As seen in Fig. 5.2, at least 34 decimal digits of precision (i.e., quadruple-precision) is required for accurate solutions down to $\epsilon_{min} = 0.01$ for a box size of $\lambda/16$. Hence, a solution at quadruple-precision would not present the overflow problem observed in Figs. 5.4–5.5.

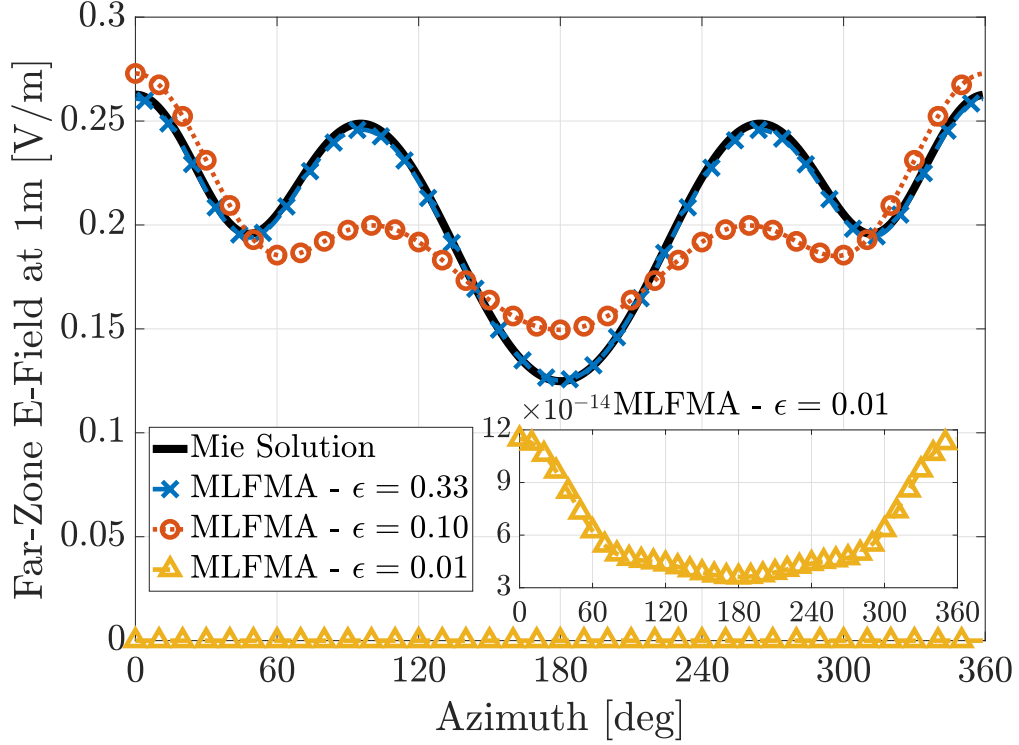


Figure 5.4: Scattering from a PEC sphere of $\lambda/4$ diameter with a $1V/m$ plane-wave at single-precision. Azimuth angles of 0° and 360° correspond to backscattering, whereas 180° corresponds to forward scattering. Results for $\epsilon = 0.01$ is repeated in the inset to illustrate the difference in scale.

The design curves that we present and validate in this section show the limitations of MLFMA for a wide range of box sizes, machine precision levels, and translation distances. These results can be used as a helpful guideline for implementers and users of MLFMA in designing their algorithms on any computing platform and analyzing the results obtained with existing implementations.

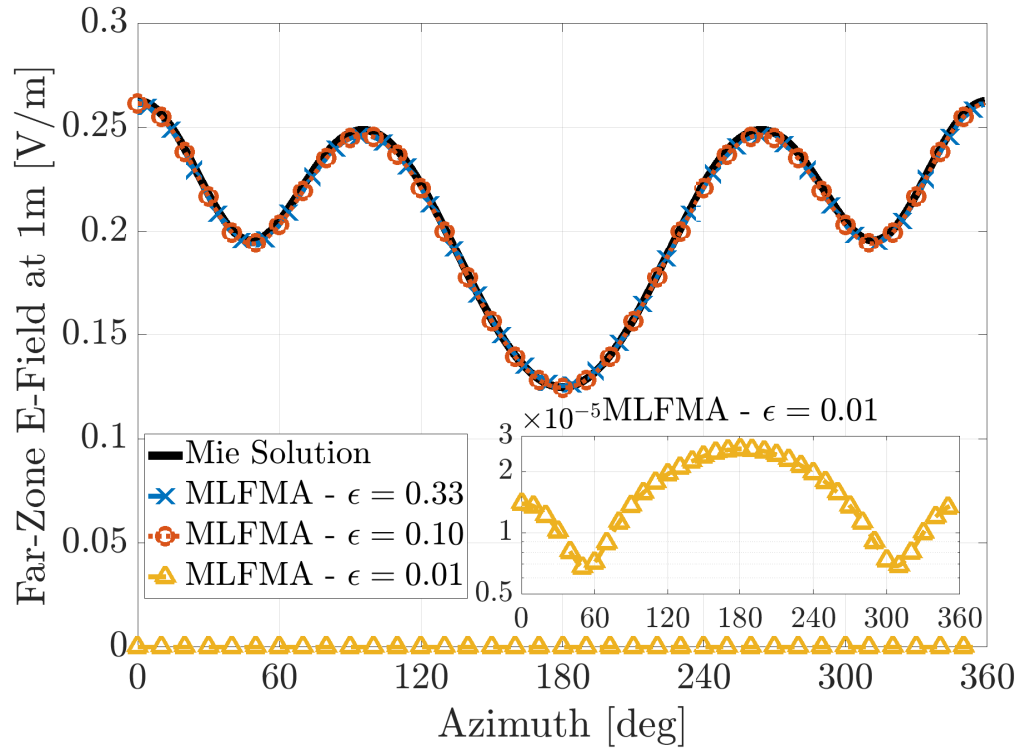


Figure 5.5: Scattering from a PEC sphere of $\lambda/4$ diameter with a $1V/m$ plane-wave at double-precision. Azimuth angles of 0° and 360° correspond to backscattering, whereas 180° corresponds to forward scattering. Results for $\epsilon = 0.01$ is repeated in the inset to illustrate the difference in scale.

5.2 Validation of MP-MLFMA

Introduced in Chapter 4, MP-MLFMA is an elegant and powerful algorithm that addresses open problems of MLFMA at two extremes of the frequency spectrum. Namely, MP-MLFMA alleviates the low-frequency breakdown of MLFMA by increasing the machine precision in an MPA framework, while offering increased efficiency for high-frequency problems by *reducing* the machine precision where applicable, thanks to the novel error control scheme we proposed in Chapter 3.

To validate MP-MLFMA and illustrate its full capabilities, we provide some preliminary analyses, then many scattering results compiled into two-classes in the following subsections: broadband and deep tree-structure problems. For all scenarios of surface scattering solutions, we analyze PEC objects with triangular mesh elements, using RWGs as basis and testing functions in a Galerkin scheme. As the integral equation formulation, a surface CFIE formulation is used with a combination parameter of $\alpha = 0.5$ for all cases.

5.2.1 Preliminary MP-MLFMA Analyses

For preliminary analyses, we implemented **Algorithm 1** given in Chapter 4 to a PEC sphere and a PEC rod, and parameters for both are listed below:

- PEC Sphere
 - Diameter: 0.6m
 - RWG Edge Length: 50mm
 - 1516 RWGs, 508 Nodes, 1012 Triangles
 - $f = 2000$ MHz
 - $N = 3$
 - Box Sizes: $\{a_L\} = \{4\lambda, 2\lambda, 1\lambda\}$
 - $\epsilon_d = 0.33 \rightarrow \tau_3 = 4, \overline{\mathbf{MP}} = \{6, 6, 6\}$

- PEC Rod
 - Rectangular Cylinder with base length: 0.1m
 - height: 5m
 - RWG Edge Length: 50mm
 - 2424 RWGs, 810 Nodes, 1616 Triangles
 - $f = 300$ MHz
 - $N = 5$
 - Box Sizes: $\{a_L\} = \{5\lambda, 2.5\lambda, 1.25\lambda, 0.625\lambda, 0.3125\lambda\}$
 - $\epsilon_d = 0.33 \rightarrow \bar{\tau} = \{14, 7, 4\}, \overline{\mathbf{MP}} = \{10, 9, 7, 6, 6\}$

Note that the parameters are selected so that the required machine precision levels are all below the standard double-precision with 16 decimal digits of precision. Both geometries listed above are illuminated with a z -polarized 1V/m plane-wave impinging from the x -axis. The resulting far-zone co-polarized (E_θ) and cross-polarized (E_ϕ) fields are calculated along the E-plane, which corresponds to the xz -plane with $\phi = 0^\circ$.

With the parameters listed above, the MP-MLFMA results for the sphere and the rod are given in Figs. 5.6 and 5.7. Note that the sphere results are compared to the Mie series in Fig. 5.6, whereas the rod results are compared to the MoM solution in Fig. 5.7. As seen from Figs. 5.6–5.7, both the sphere and rod are accurately analyzed with the hierarchical implementation given in Chapter 4.

In the following subsections, we move on to the detailed demonstrations of the capabilities of MP-MLFMA.

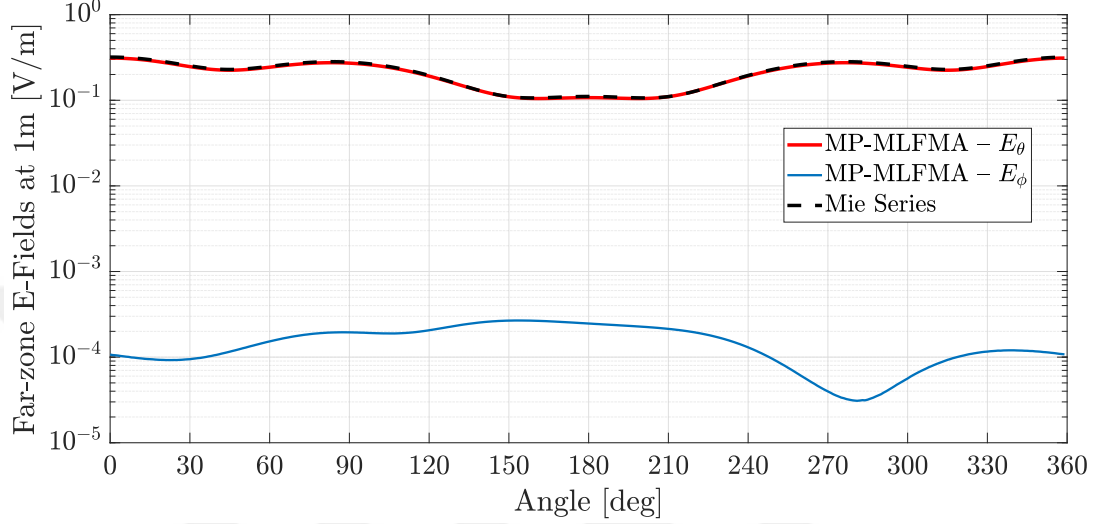


Figure 5.6: MP-MLFMA vs. Mie Series Solution for PEC Sphere @2000 MHz, $\epsilon_d = 0.33$. Angles of 0° and 180° correspond to back-scattering and forward-scattering directions, respectively.

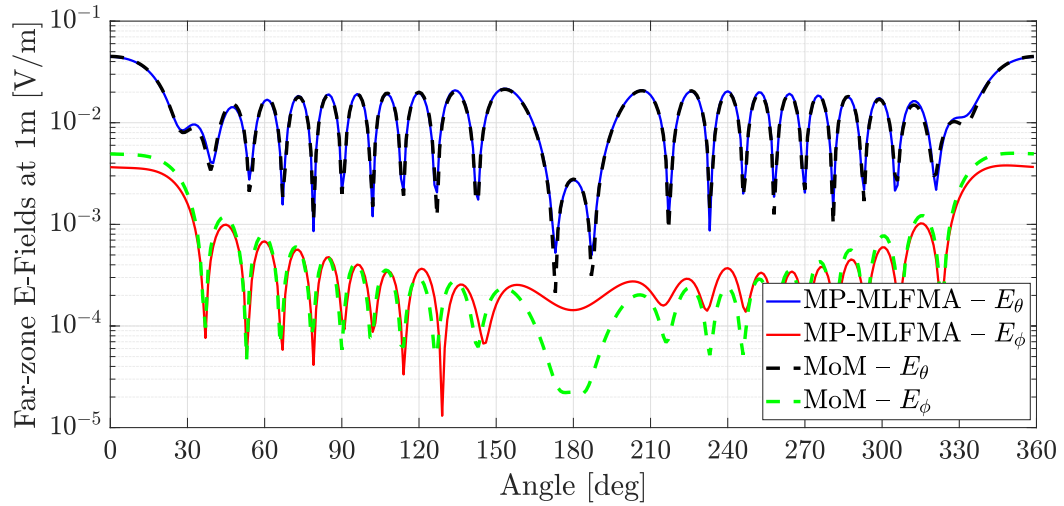


Figure 5.7: MP-MLFMA vs. MoM Solution for PEC Rod @300 MHz, with $\epsilon_d = 0.33$. Angles of 0° and 180° correspond to back-scattering and forward-scattering directions, respectively.

5.2.2 Broadband MP-MLFMA Solutions

To demonstrate the broadband capabilities of MP-MLFMA, we present plane-wave scattering results for a PEC sphere for a wide range of frequencies. For simplicity, an MP-MLFMA tree structure with $N = 3$ levels is formed, which means that the translation operations are performed only at the leaf level, i.e., $L = 3$. The PEC sphere has a diameter of $D = 0.6m$, with triangular mesh elements having edge lengths of $l_{RWG} = 25mm$. The geometry under analysis with its recursive division for $N = 3$ is illustrated in Fig. 5.8.

The PEC sphere shown in Fig. 5.8 is illuminated with a plane-wave impinging from the x -axis, linearly polarized along the z -axis, with an amplitude of 1V/m. The analyzed frequencies, their corresponding box-sizes at $L = 3$, together with the required number of harmonics (τ_3) and machine precisions (MP_3) are provided in Table 5.1, using the error control scheme in Chapter 3.

In Table 5.1, the number of harmonics and the machine precision levels are provided for three different desired error thresholds; namely $\epsilon_d = 0.33$, $\epsilon_d = 0.10$, and $\epsilon_d = 0.01$, over an extremely large frequency bandwidth (a ratio of 256:1). Note that for some of the cases listed in Table 5.1, MP_3 values are larger than 16, corresponding to cases where the low-frequency breakdown occurs for a standard double-precision implementation. Each of the 27 cases listed in Table 5.1 is solved at the double-precision using conventional MLFMA (DP-MLFMA), and

Table 5.1: Scenarios analyzed to illustrate the broadband capabilities of MP-MLFMA

f	$a_3 = 0.15m$	$l_{RWG} = 25mm$	# of Harmonics, τ_3			Machine Precision, MP_3		
			$\epsilon_d = 0.33$	$\epsilon_d = 0.10$	$\epsilon_d = 0.01$	$\epsilon_d = 0.33$	$\epsilon_d = 0.10$	$\epsilon_d = 0.01$
15.63 MHz	$\lambda/128$	$\lambda/768$	3	7	20	11	21	55
31.25 MHz	$\lambda/64$	$\lambda/384$	3	7	20	10	18	48
62.5 MHz	$\lambda/32$	$\lambda/192$	3	7	20	8	16	42
125 MHz	$\lambda/16$	$\lambda/96$	3	7	20	7	13	35
250 MHz	$\lambda/8$	$\lambda/48$	3	7	20	6	11	29
500 MHz	$\lambda/4$	$\lambda/24$	4	7	20	5	8	22
1 GHz	$\lambda/2$	$\lambda/12$	7	10	20	6	8	16
2 GHz	λ	$\lambda/6$	11	14	24	7	7	13
4 GHz	2λ	$\lambda/3$	23	24	34	8	8	9

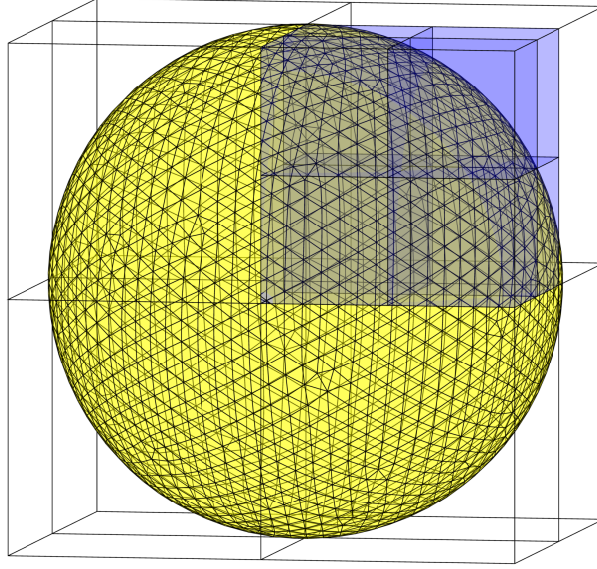


Figure 5.8: Recursive division of a PEC Sphere with triangular mesh edge lengths of $l_{RWG} = 25mm$, into an MP-MLFMA tree structure for $N = 3$.

the proposed MP-MLFMA using the MP_3 values provided in Table 5.1. Co-polarized far-zone electric field intensities along the E-plane is recorded for each case, and compared to the Mie series solution as the gold standard for PEC spheres. Note that for a plane-wave polarized along z -axis, and impinging from the x -axis, the E-plane corresponds to the θ -axis with $\phi = 0^\circ$ for conventional spherical coordinates, and the co-polarized component of the E-field is represented by E_θ . We represent the E-plane as the elevation axis of the PEC sphere, with the elevation angle EL defined in degrees as

$$EL = 90^\circ - \theta. \quad (5.1)$$

With the coordinate definition given in (5.1), the back-scattering direction for a plane-wave impinging from the x -axis becomes $EL = 0^\circ$, while the forward-scattering direction becomes $EL = 180^\circ$.

We present the far-zone co-polarized E-fields along the elevation axis for $f = 15.63$ MHz, $f = 250$ MHz, and $f = 4$ GHz at $\epsilon_d = 0.01$ in Figs. 5.9–5.11.

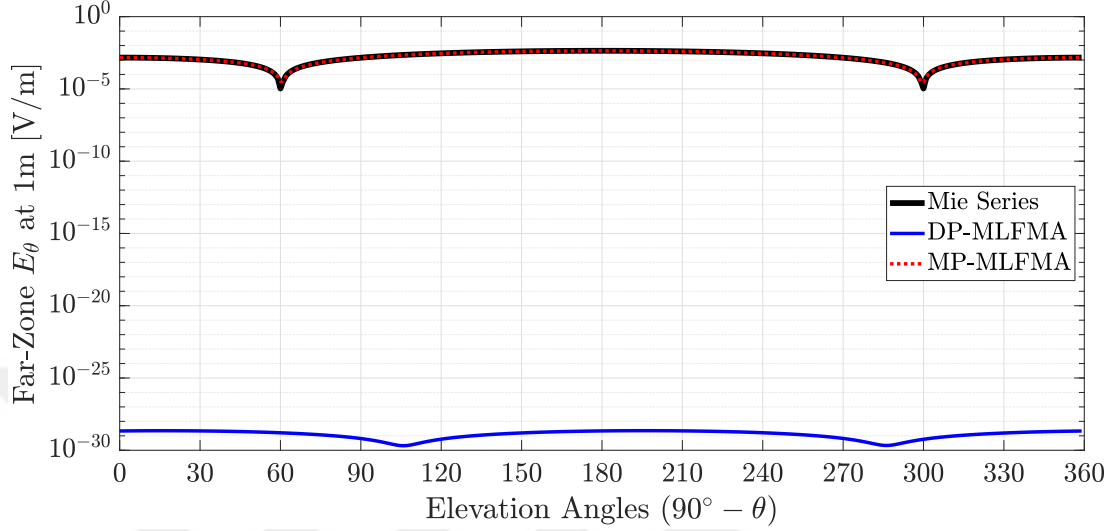


Figure 5.9: Far-zone co-polarized E-fields at $f = 15.63$ MHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$.

As seen in Figs. 5.9–5.11, MP-MLFMA agrees well with the Mie series solution at a wide range of frequencies, thanks to its built-in MPA framework. DP-MLFMA results differ greatly –and in fact do not converge– for $f = 15.63$ MHz and $f = 250$ MHz, as the standard double-precision implementation cannot handle those problems. Note that for $\epsilon_d = 0.01$, solutions at $f = 15.63$ MHz and $f = 250$ MHz require $MP_3 = 55$ and $MP_3 = 29$ (see Table 5.1), respectively, which are well beyond the minimum precision of 16 decimal digits offered by the double-precision. Another observation from Fig. 5.11 is that there is a slight mismatch between the Mie series solution and DP-MLFMA/MP-MLFMA, although DP-MLFMA and MP-MLFMA agree well with each other. This is due to the relatively large triangular mesh size at $f = 4$ GHz ($l_{RWG} = 25\text{mm} = \lambda/3$), and could be mitigated by using a smaller mesh element size.

After we illustrate some of the individual cases of Table 5.1 in Figs. 5.9–5.11, we now present the far-field pattern errors for the entirety of Table 5.1. We define the average relative percentage error (ARPE) of the far-field radiation patterns as

$$ARPE = 100 \times \frac{1}{M} \sum_{i=1}^M \frac{|E_\theta[i] - E_{Mie}[i]|}{|E_{Mie}[i]|}, \quad (5.2)$$

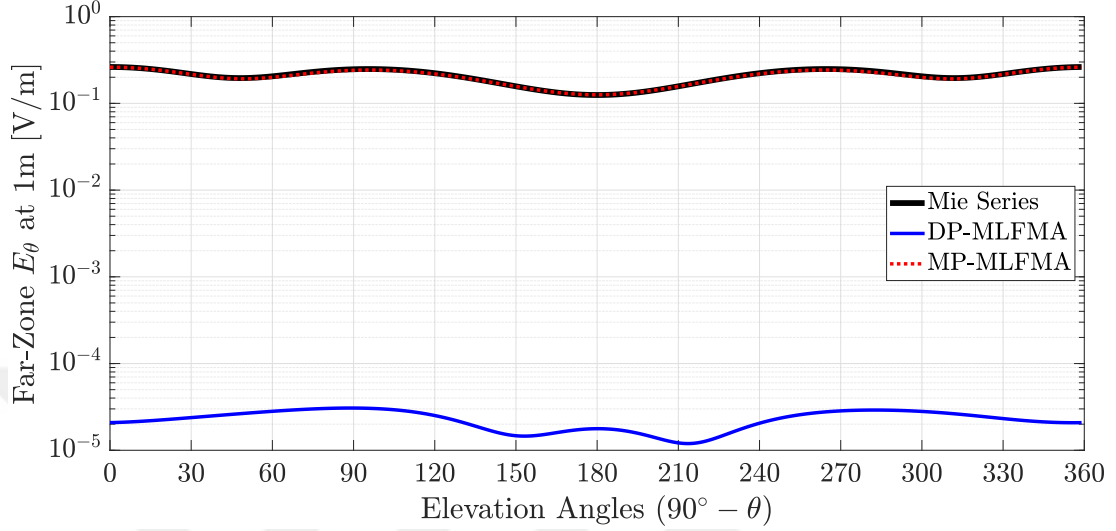


Figure 5.10: Far-zone co-polarized E-fields at $f = 250$ MHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$.

where M is the number of far-field samples and E_{Mie} corresponds to the reference Mie series solution. We have selected $M = 360$ in our analyses, which corresponds to an angular spacing of 1° in the elevation axis. The ARPE far-field errors for all the cases in Table 5.1 are presented in Fig. 5.12.

Fig. 5.12 perfectly demonstrates the broadband capabilities of MP-MLFMA, as the ARPE of far-field patterns stay below 20% while scanning over a 256:1 frequency bandwidth. Note that the error breakdown of DP-MLFMA occurs at $f = 500$ MHz for $\epsilon_d = 0.01$, which corresponds to the point after where MP_3 becomes larger than 16; hence the double-precision cannot handle lower frequencies. A similar breakdown occurs at $f = 62.5$ MHz for lower ϵ_d values, as expected from the MP_3 values in Table 5.1. Also note that the ARPE values are very close for different ϵ_d values for MP-MLFMA, as the desired error threshold values are defined for the free-space Green's function. Since far-field patterns are *integrated* values, the Green's function errors are diluted and usually a larger error in Green's function can be tolerated, as explained in Section 5.1. Finally, the relatively large error at $f = 4$ GHz for both DP-MLFMA and MP-MLFMA can be attributed to the electrically large discretization, as shown in Table 5.1 and as illustrated in Fig. 5.11.

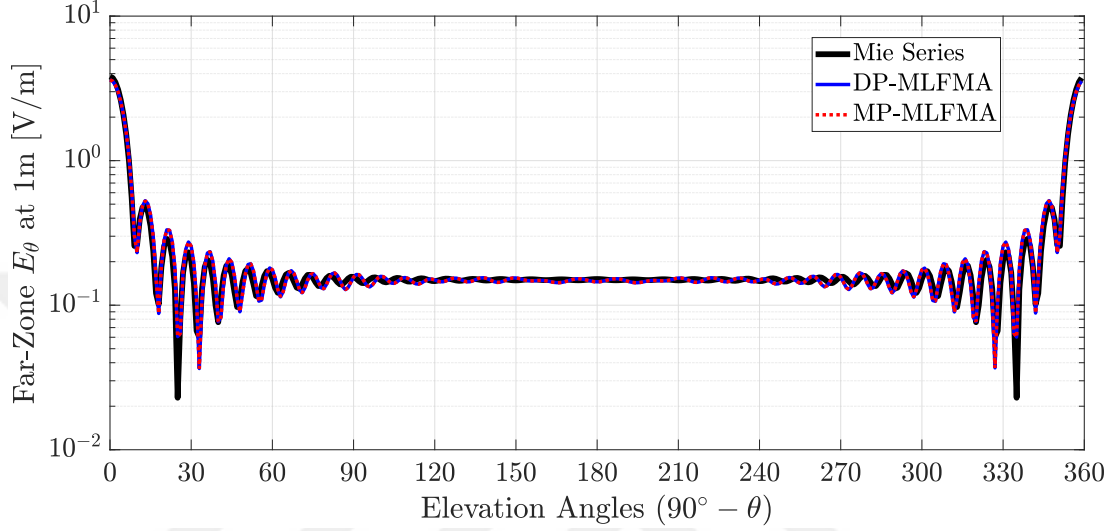


Figure 5.11: Far-zone co-polarized E-fields at $f = 4$ GHz solved by double-precision MLFMA (DP-MLFMA) and MP-MLFMA using the parameters given in Table 5.1 for $\epsilon_d = 0.01$.

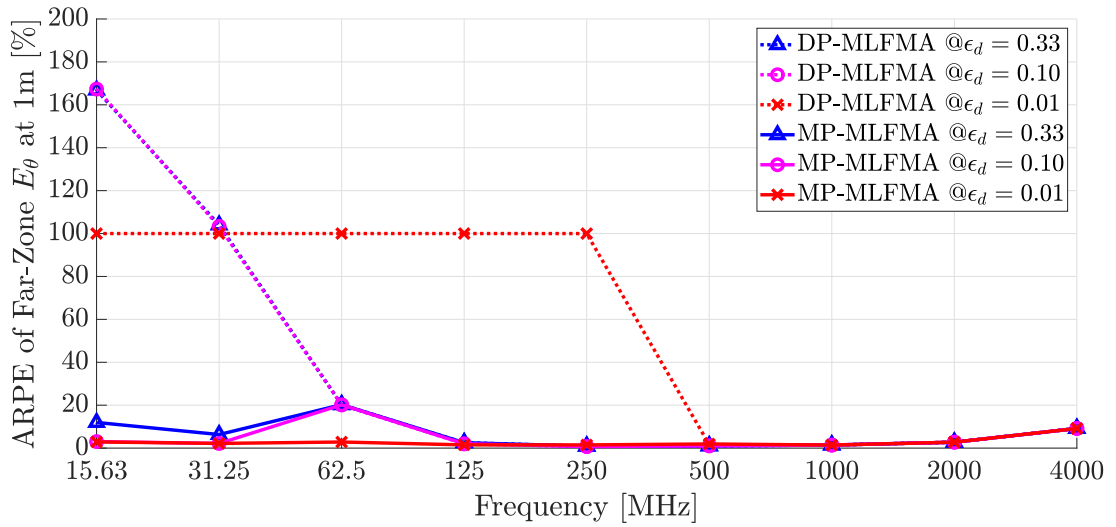


Figure 5.12: ARPE of Far-zone E_θ for all the cases listed in Table 5.1.

5.2.3 MP-MLFMA Solutions for Deep Tree-Structures

We now present the capabilities of MP-MLFMA for *deep* tree-structures with $N > 3$, where both accuracy for electrically small boxes and efficiency for electrically large boxes can be attained in an MPA framework. As examples of deep tree-structure problems, we now analyze *arrays* of PEC spheres placed along the z -axis and illuminated by a z -polarized plane wave impinging from the x -axis with an amplitude of 1V/m. The general array geometry is presented in Fig. 5.13.

As shown in Fig. 5.13, we place K spheres with diameters of D , uniformly with a spacing of d . With this configuration, the overall size of the object becomes

$$D_{max} = (K - 1)d + D. \quad (5.3)$$

Using the array architecture given in Fig. 5.13, we analyze the scenarios listed in Table 5.2, where we list the array geometries, as well as the lists of number of harmonics ($\bar{\tau}$) and machine precisions ($\overline{\text{MP}}$) as given by the error control scheme in Chapter 3.

As shown in Table 5.2, we selected 4 cases solved at $f = 125$ MHz so that the maximum size of each case is an 2^k multiple of the previous case. Due to this selection, the leaf level box size is kept constant for every case at $a_N = \lambda/16$, while increasing the number of levels for the MP-MLFMA tree structure. Note that the leaf-level box size of $\lambda/16$ and the desired error threshold of $\epsilon_d = 0.01$ are

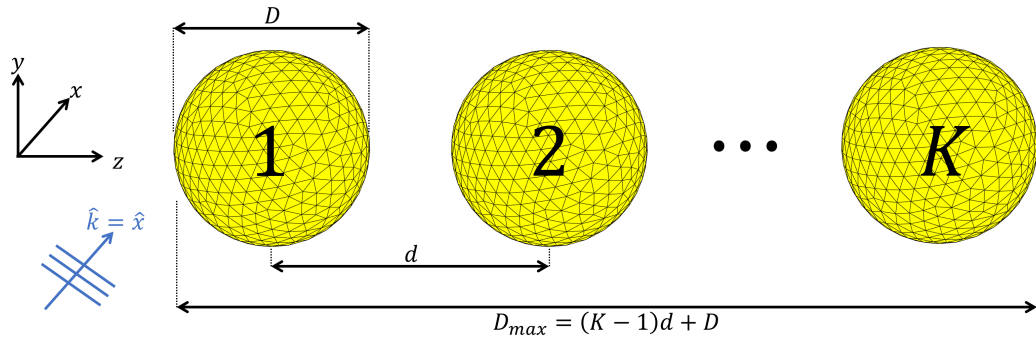


Figure 5.13: Sphere array geometry for the demonstration of deep tree-structure capabilities of MP-MLFMA. Spheres are placed along the z -axis, with a z -polarized plane-wave impinging from x -direction.

Table 5.2: Deep tree-structure scenarios under analysis for $f = 125$ MHz and $\epsilon_d = 0.01$.

Case #	K	d	D_{max}	N	$a_{3,...,N}$	$\bar{\tau}$	MP
1	1	-	0.6m	3	$\{\lambda/16\}$	$\{20\}$	$\{35\}$
2	2	1.8m	2.4m	5	$\{\lambda/4, \lambda/8, \lambda/16\}$	$\{20, 20, 20\}$	$\{22, 29, 35\}$
3	4	1.4m	4.8m	6	$\{\lambda/2, \lambda/4, \lambda/8, \lambda/16\}$	$\{20, 20, 20, 20\}$	$\{16, 22, 29, 35\}$
4	10	1m	9.6m	7	$\{\lambda, \lambda/2, \lambda/4, \lambda/8, \lambda/16\}$	$\{24, 20, 20, 20, 20\}$	$\{13, 16, 22, 29, 35\}$

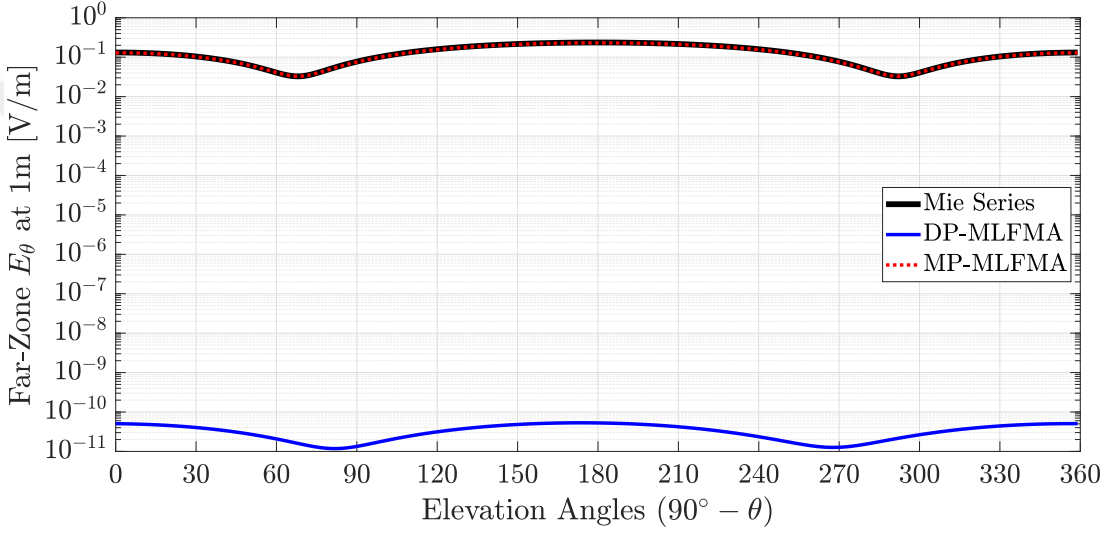


Figure 5.14: Far-zone E_θ fields obtained by MP-MLFMA for Case-1 at $f = 125$ MHz with $D = 0.6m$, $K = 1$, and $N = 3$, compared to the Mie series and DP-MLFMA.

specifically chosen so that the double-precision (16 decimal digits) is not enough to handle the leaf-level translations, while lower precision levels than double-precision are employed at larger boxes. As a result, these cases can demonstrate the deep tree-structure solution capabilities of MP-MLFMA, while DP-MLFMA cannot handle any of the cases due to high precision requirements of the smaller box sizes.

For each case given in Table 5.2, we present the far-zone co-polarized E-fields scaled at 1m, recorded along the E-plane, i.e., the elevation axis as defined in the previous subsection. The far-field patterns are compared with the Mie series solution for Case-1 (a single sphere), and the MoM solution for Cases-2–4, with the same CFIE formulation with a combination parameter of $\alpha = 0.5$. The far-field patterns for Cases-1–4 are given in Figs. 5.14–5.17.

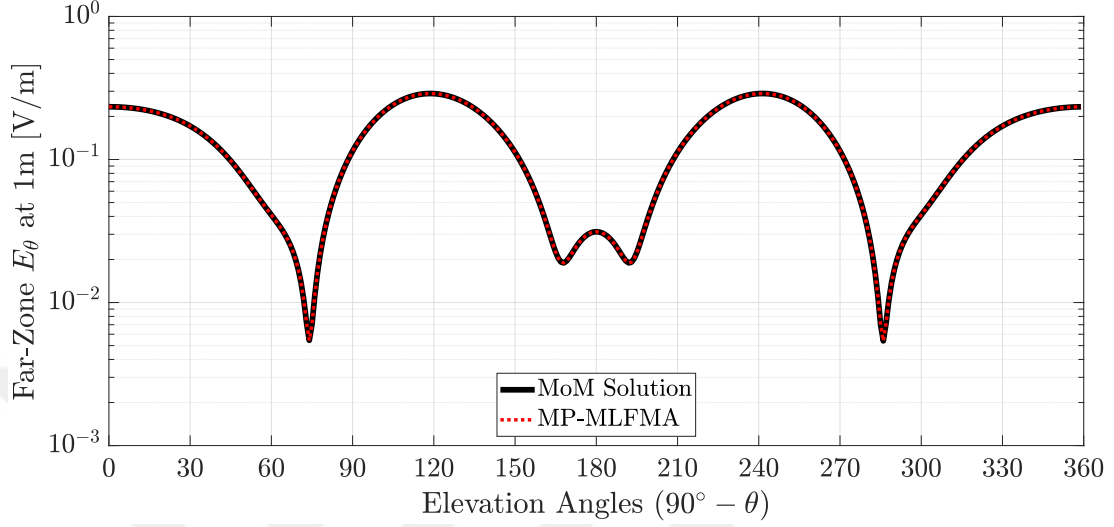


Figure 5.15: Far-zone E_θ fields obtained by MP-MLFMA for Case-2 at $f = 125$ MHz with $D = 0.6m$, $K = 2$, $d = 1.8m$, and $N = 5$, compared to the MoM solution.

As seen in Figs. 5.14–5.17, MP-MLFMA results agree perfectly with Mie series and MoM solutions for all cases. Note that for Case-1 in Fig. 5.14, we also present the DP-MLFMA solution that fails to converge due to the low-frequency breakdown. For all other cases, we omit the DP-MLFMA solutions as they all fail due to having the same leaf-level box size that requires a machine precision level of $MP_N = 35$, as shown in Table 5.2.

Note that as we move from Case-1 to Case-4, the overall size of the problem D_{max} increases 16 times, as the number of levels of MP-MLFMA increases from $N = 3$ to $N = 7$. For the largest problem in Case-4 with $N = 7$, MP-MLFMA elegantly handles the low-frequency problems at the lower levels, while offering a potential increase in efficiency at the largest level with $MP_3 = 13$. Therefore, scenarios defined in Table 5.2 and results given in Figs. 5.14–5.17 clearly show the deep tree-structure capabilities offered by MP-MLFMA.

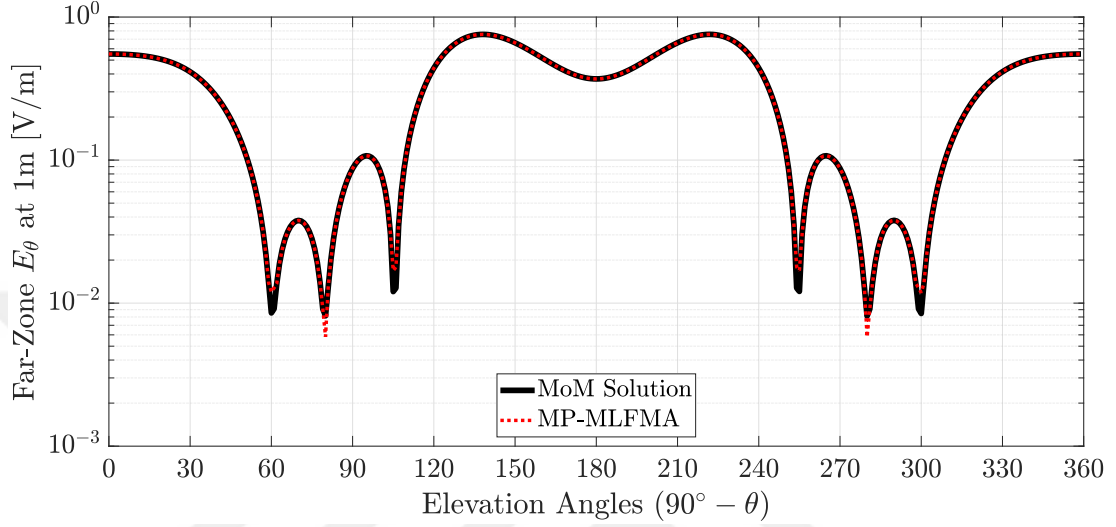


Figure 5.16: Far-zone E_θ fields obtained by MP-MLFMA for Case-3 at $f = 125$ MHz with $D = 0.6m$, $K = 4$, $d = 1.4m$, and $N = 6$, compared to the MoM solution.

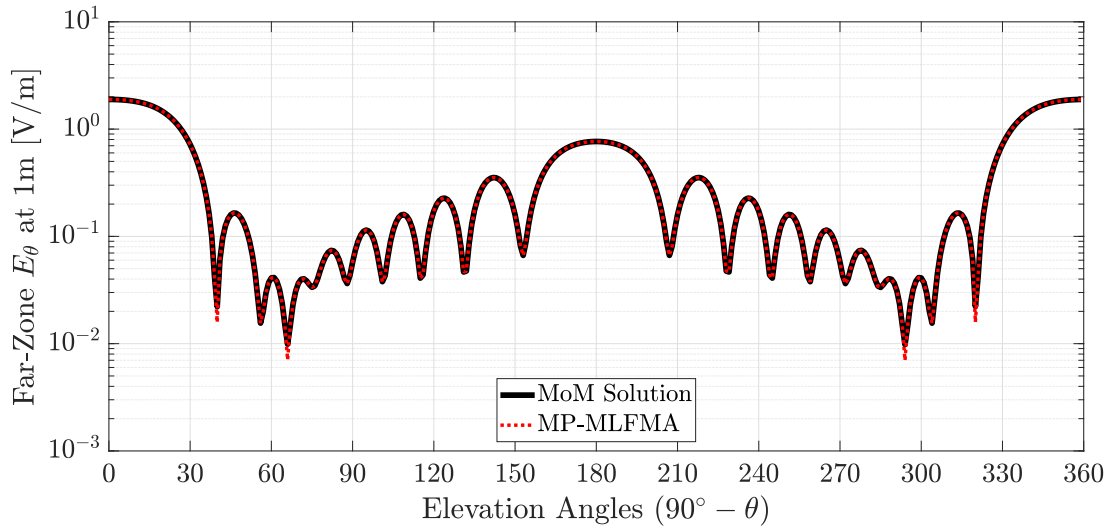


Figure 5.17: Far-zone E_θ fields obtained by MP-MLFMA for Case-4 at $f = 125$ MHz with $D = 0.6m$, $K = 10$, $d = 1m$, and $N = 7$, compared to the MoM solution.

Chapter 6

Conclusion

In this dissertation, we proposed and validated a novel Multiple-Precision Multilevel Fast Multipole Algorithm (MP-MLFMA) that enables the accurate and efficient solution of extremely large problems with deep tree-structures over arbitrarily large frequency bandwidths. With the introduction of a multiple-precision framework within the conventional aggregation, translation, and disaggregation stages of MLFMA, arbitrarily low-frequencies are handled without the well-known low-frequency breakdown simply by increasing the machine precision of the computing platform. On the other side of the frequency spectrum, the machine precision can be reduced below the native precision (single- or double-precision, depending on the platform) for high-frequency problems, offering increased efficiency without sacrificing the accuracy. For a deep tree-structure problem, a distinct precision can be assigned to each level of the tree structure of MP-MLFMA, which elegantly combines the accuracy for lower levels with electrically small features and the efficiency for higher levels with electrically large features.

To select the required machine precision for any MP-MLFMA tree structure, as well as to control the translation errors due to the truncation of the diagonal form of the free-space Green's function, we proposed a novel error control scheme. Our novel scheme offers error control at all frequencies and takes into account

the machine precision of the computing platform, whereas the previous state-of-the-art on error control was limited to high-frequencies without including the machine precision. The proposed scheme is validated for a wide range of MP-MLFMA tree structures (i.e., box sizes or frequencies). And for the first time in the literature, we introduced extremely accurate design curves for MLFMA users on the achievable errors and smallest box sizes, that can be used either for MP-MLFMA or conventional MLFMA for fixed-precision computers.

Combined with our novel error control scheme, we validated MP-MLFMA for a wide range of scattering problems that demonstrate its capabilities for broadband and deep tree-structure solutions. Extremely accurate results that showcase the MP-MLFMA are presented for all cases.

To showcase the true potential of the MP-MLFMA implementations, we first analyzed the capabilities and limitations of a software-defined MPA implementation, which is especially advantageous if an arbitrarily large machine precision is required. Then we implemented a hardware-defined MPA framework for simple arithmetic operations that illustrate a super-linear speed up, when lower precision levels are attained compared to the native single- or double-precision of the computer.

This dissertation introduces the MP-MLFMA for the first time, demonstrates its capabilities over extremely large bandwidths with deep tree-structures, and *hints* at its true potential through preliminary hardware implementations. The future work on MP-MLFMA should focus on the efficient implementation of the MPA framework over a large range of precision levels, using hardware-defined schemes for low-precisions and software-defined schemes for arbitrarily high-precisions.

Bibliography

- [1] R. Coifman, V. Rokhlin, and S. Wandzura, “The fast multipole method for the wave equation: a pedestrian prescription,” *IEEE Antennas and Propagation Magazine*, vol. 35, pp. 7–12, Jun. 1993.
- [2] B. A. Cipra, “The best of the 20th century: editors name top 10 algorithms,” *SIAM News*, vol. 33, May 2000.
- [3] J. Song, C.-C. Lu, and W. C. Chew, “Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects,” *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 1488–1493, Oct. 1997.
- [4] C. C. Lu and W. C. Chew, “Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects,” *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 1488 – 1493, Oct. 1997.
- [5] Ö. Ergül and L. Gürel, *The Multilevel Fast Multipole Algorithm for Solving Large-Scale Computational Electromagnetics Problems*. Hoboken, NJ, USA: Wiley-IEEE Press, 2014.
- [6] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.
- [7] J. L. Volakis, A. Chatterjee, and L. C. Kempel, *Finite Element Method for Electromagnetics*. IEEE Press, 1998.

- [8] K. Yee, "Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," *IEEE Transactions on Antennas and Propagation*, vol. 14, no. 3, pp. 302–307, 1966.
- [9] P. H. Aoyagi, Jin-Fa Lee, and R. Mittra, "A hybrid Yee algorithm/scalar-wave equation approach," *IEEE Transactions on Microwave Theory and Techniques*, vol. 41, no. 9, pp. 1593–1600, 1993.
- [10] J.-P. Berenger, "Perfectly matched layer for the FDTD solution of wave-structure interaction problems," *IEEE Transactions on Antennas and Propagation*, vol. 44, no. 1, pp. 110–117, 1996.
- [11] L. Greengard, J. Huang, V. Rokhlin, and S. Wadzura, "Accelerating fast multipole methods for the Helmholtz equation at low frequencies," *IEEE Computing in Science and Engineering*, vol. 5, pp. 32–38, Jul.–Sep. 1998.
- [12] S. Koc, J. M. Song, and W. C. Chew, "Error analysis for the numerical evaluation of the diagonal forms of the scalar spherical addition theorem," *SIAM Journal on Numerical Analysis*, vol. 36, no. 3, pp. 906–921, 1999.
- [13] J. M. Song and W. C. Chew, "Error analysis for the truncation of multipole expansion of vector Green's functions," *IEEE Microwave and Wireless Components Letters*, vol. 11, pp. 311–313, Jul. 2001.
- [14] W. C. Chew, J.-M. Jin, E. Michielssen, and J. M. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*. Boston, USA: Artech House, 2001.
- [15] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. New York, USA: Dover, 1964.
- [16] M. L. Hastriter, S. Ohnuki, and W. C. Chew, "Error control of the translation operator in 3D MLFMA," *Microwave and Optical Technology Letters*, vol. 37, pp. 184–188, May 2003.
- [17] J.-S. Zhao and W. C. Chew, "Three dimensional multilevel fast multipole algorithm from static to electrodynamic," *Microwave and Optical Technology Letters*, vol. 26, pp. 43–48, Jul. 2000.

- [18] J.-S. Zhao and W. C. Chew, “Applying matrix rotation to the three-dimensional low-frequency multilevel fast multipole algorithm,” *Microwave and Optical Technology Letters*, vol. 26, pp. 105–110, Jul. 2000.
- [19] J.-S. Zhao and W. C. Chew, “Applying LF-MLFMA to solve complex PEC structures,” *Microwave and Optical Technology Letters*, vol. 28, pp. 155–160, Feb. 2001.
- [20] Y.-H. Chu and W. C. Chew, “A multilevel fast multipole algorithm electrically small composite structures,” *Microwave and Optical Technology Letters*, vol. 43, pp. 202–207, Nov. 2004.
- [21] Ö. Ergül and L. Gürel, “Efficient solutions of metamaterial problems using a low-frequency multilevel fast multipole algorithm,” *Progress in Electromagnetics Research*, vol. 108, pp. 81–99, 2010.
- [22] V. Melapudi, B. Shanker, S. Seal, and S. Aluru, “A scalable parallel wide-band MLFMA for efficient electromagnetic simulations on large scale clusters,” *IEEE Transactions on Antennas and Propagation*, vol. 59, pp. 2565–2577, Jul. 2011.
- [23] L. J. Jiang and W. C. Chew, “Low-frequency fast inhomogeneous planewave algorithm (LF-FIPWA),” *Microwave and Optical Technology Letters*, vol. 40, pp. 117–122, Jan. 2004.
- [24] I. Bogaert, J. Peeters, and F. Olyslager, “A nondirective plane wave MLFMA stable at low frequencies,” *IEEE Transactions on Antennas and Propagation*, vol. 56, pp. 3752–3767, Dec. 2008.
- [25] I. Bogaert and F. Olyslager, “A low frequency stable plane wave addition theorem,” *Journal of Computational Physics*, vol. 228, pp. 1000–1016, Mar. 2009.
- [26] B. Khalichi, Ö. Ergül, and V. B. Ertürk, “Broadband solutions of potential integral equations with NSPWMLFMA,” *IEEE Trans. Antennas Propag.*, vol. 67, no. 6, pp. 4307–4312, 2019.

- [27] Ö. Ergül and B. Karaosmanoğlu, “Low-frequency fast multipole method based on multiple-precision arithmetic,” *IEEE Transactions on Antennas and Propagation*, vol. 13, pp. 975–978, 2014.
- [28] “IEEE Standard for Floating-Point Arithmetic,” *IEEE Std 754-2008*, pp. 1–70, Aug 2008.
- [29] D. Göddeke, R. Strzodka, and S. Turek, “Performance and accuracy of hardware-oriented native-, emulated- and mixed-precision solvers in FEM simulations,” *International Journal of Parallel, Emergent and Distributed Systems*, vol. 22, no. 4, pp. 221–256, 2007.
- [30] A. Buttari, J. Dongarra, J. Langou, J. Langou, P. Luszczek, and J. Kurzak, “Mixed precision iterative refinement techniques for the solution of dense linear systems,” *The International Journal of High Performance Computing Applications*, vol. 21, no. 4, pp. 457–466, 2007.
- [31] J. Sun, G. D. Peterson, and O. O. Storaasli, “High-performance mixed-precision linear solver for FPGAs,” *IEEE Transactions on Computers*, vol. 57, no. 12, pp. 1614–1623, 2008.
- [32] D. H. Bailey, R. Barrio, and J. M. Borwein, “High-precision computation: Mathematical physics and dynamics,” *Applied Mathematics and Computation*, vol. 218, pp. 10106–10121, jun 2012.
- [33] A. Haidar, S. Tomov, J. Dongarra, and N. J. Higham, “Harnessing GPU tensor cores for fast FP16 arithmetic to speed up mixed-precision iterative refinement solvers,” in *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 603–613, 2018.
- [34] E. Carson and D. Gergelits, “Initial exploration of the use of mixed precision in iterative solvers,” Tech. Rep. LLNL-SR-811988 B639388, Lawrence Livermore National Laboratory, June 2020.
- [35] T. P. Stefański and K. Krzyzanowska, “Implementation of FDTD-compatible Green’s function on graphics processing unit,” *IEEE Antennas and Wireless Propagation Letters*, vol. 11, pp. 1422–1425, 2012.

- [36] T. P. Stefański, “Implementation of FDTD-compatible Green’s function on heterogeneous CPU-GPU parallel precessing system,” *Progress In Electromagnetics Research*, vol. 135, pp. 297–316, 2013.
- [37] T. P. Stefański, “Electromagnetic problems requiring high-precision computations,” *IEEE Antennas and Propagation Magazine*, vol. 55, no. 2, pp. 344–353, 2013.
- [38] T. J. Giese and D. M. York, “Extension of adaptive tree code and fast multipole methods to high angular momentum particle charge densities,” *Journal of Computational Chemistry*, vol. 29, no. 12, pp. 1895–1904, 2008.
- [39] M. Vikram, H. Huang, B. Shanker, and T. Van, “A novel wideband FMM for fast integral equation solution of multiscale problems in electromagnetics,” *IEEE Transactions on Antennas and Propagation*, vol. 57, pp. 2094–2104, Jul. 2009.
- [40] Y. Liu, X. Pan, and X. Sheng, “Skeletonization accelerated MLFMA solution of volume integral equation for plasmonic structures,” *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 3, pp. 1590–1594, 2018.
- [41] J. Wei, Z. Peng, and J. Lee, “Multiscale electromagnetic computations using a hierarchical multilevel fast multipole algorithm,” *Radio Science*, vol. 49, no. 11, pp. 1022–1040, 2014.
- [42] M. Takrimi, Ö. Ergül, and V. B. Ertürk, “A novel broadband multilevel fast multipole algorithm with incomplete-leaf tree structures for multiscale electromagnetic problems,” *IEEE Transactions on Antennas and Propagation*, vol. 64, no. 6, pp. 2445–2456, 2016.
- [43] S. Velamparambil, W. C. Chew, and J. Song, “10 million unknowns: Is it that big?,” *IEEE Antennas and Propagation Magazine*, vol. 45, pp. 43 – 58, Apr. 2003.
- [44] S. Velamparambil and W. C. Chew, “Analysis and performance of a distributed memory multilevel fast multipole algorithm,” *IEEE Transactions on Antennas and Propagation*, vol. 53, pp. 2719 – 2727, Aug. 2005.

- [45] Ö. Ergül and L. Gürel, “Efficient parallelization of the multilevel fast multipole algorithm for the solution of large-scale scattering problems,” *IEEE Transactions on Antennas and Propagation*, vol. 56, no. 8, pp. 2335–2345, 2008.
- [46] Ö. Ergül and L. Gürel, “A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm,” *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 6, pp. 1740–1750, 2009.
- [47] B. Michiels, J. Fostier, I. Bogaert, and D. De Zutter, “Weak scalability analysis of the distributed-memory parallel MLFMA,” *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 11, pp. 5567–5574, 2013.
- [48] R. F. Harrington, *Field Computation by Moment Methods*. IEEE Press, 1993.
- [49] W. C. Chew, J. M. Jin, C. C. Lu, E. Michielssen, and J. M. Song, “Fast solution method in electromagnetics,” *Transactions on Antennas and Propagation*, vol. 45, no. 3, pp. 533–543, 1997.
- [50] V. Rokhlin, “Rapid solution of integral equations of scattering theory in two dimensions,” *Journal of Computational Physics*, vol. 86, no. 2, pp. 414–439, 1990.
- [51] J. M. Song, C. C. Lu, W. C. Chew, and S. W. Lee, “Fast Illinois solver code (FISC),” *IEEE Antennas and Propagation Magazine*, vol. 40, pp. 27 – 34, Oct. 1998.
- [52] Ö. Ergül, “Basics of MLFMA.” CEMETU Lecture Notes, Middle East Technical University, 2013.
- [53] Y. Saad and M. H. Schultz, “GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems,” *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 3, pp. 856–869, 1986.
- [54] “Multiprecision computing toolbox for MATLAB.” Accessed: Aug. 11, 2017 [Online]. Available: <http://www.advanpix.com>.
- [55] M. Kalfa, Ö. Ergül, and V. B. Ertürk, “Error control of multiple-precision MLFMA,” vol. 66, pp. 5651–5656, Oct. 2018.

- [56] G. Venkatesh, E. Nurvitadhi, and D. Marr, “Accelerating deep convolutional networks using low-precision and sparsity,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2861–2865, 2017.
- [57] Z. Cai, X. He, J. Sun, and N. Vasconcelos, “Deep learning with low precision by half-wave Gaussian quantization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [58] M. J. Schulte and E. E. Swartzlander, “A family of variable-precision interval arithmetic processors,” *IEEE Transactions on Computers*, vol. 49, no. 5, pp. 387–397, 2000.
- [59] A. Haidar, H. Bayraktar, S. Tomov, J. Dongarra, and N. J. Higham, “Mixed-precision iterative refinement using tensor cores on GPUs to accelerate solution of linear systems,” tech. rep., Royal Society Open Science, 2014.
- [60] K. Rudnicki and T. P. Stefański, “FPGA implementation of the multiplication operation in multiple-precision arithmetic,” in *Mixed Design of Integrated Circuits and Systems*, June 2017.
- [61] D. F. Zucker and R. B. Lee, “Re-use of high-precision arithmetic hardware to perform multiple concurrent low-precision calculations,” tech. rep., NASA Technical Report, April 1994.
- [62] S. M. Rao, D. R. Wilton, and A. W. Glisson, “Electromagnetic scattering by surfaces of arbitrary shape,” *IEEE Transactions on Antennas and Propagation*, vol. AP-30, pp. 409–418, May. 1982.

Appendix A

Floating-Point Arithmetic Standard

In contemporary digital processors, storage and arithmetic operations are performed on fixed-size registers. These registers are built at a hardware-level to keep binary numbers with a certain number of bits and perform operations on them. Although the contemporary architectures allow arbitrary number formats, two main classes of number formats are widely used; namely, the fixed-point and floating-point numbers.

The fixed-point numbers are basically signed integer numbers with a pre-defined scaling factor. For b -bits, the signed integer range is $[-2^{b-1}, 2^{b-1} - 1]$, while the scaling factor is chosen so that the fixed-point numbers are representing fractional numbers. The scaling factor is generally kept constant throughout the whole computation. The fixed-point arithmetics is common in early generation of processors, Application Specific Integrated Circuits (ASICs), and simple micro-controllers. For scientific computation, the fixed-point arithmetic operations are not usually preferred, due to their limited range compared to floating-point numbers for the same number of bits.

The floating-point number format is essentially a trade-off for an extremely

large range by sacrificing resolution for large numbers. The term *floating-point* comes from the fact that the radix point (decimal or binary point), and hence the *scaling factor* actually moves automatically throughout the numbers' range. Most modern CPUs, GPUs, and other processor architectures employ floating-point arithmetics as defined by the IEEE-754 standard [28]. The IEEE-754 standard separates the total number of bits into three regions: the sign bit (s), the exponent (x), and the fraction (f), as shown in Fig. A.1.

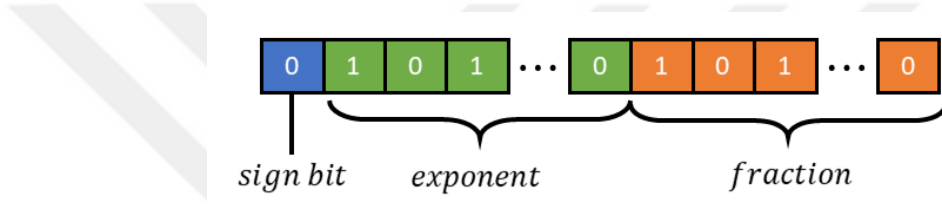


Figure A.1: Floating-point number representation defined in IEEE-754 standard [28].

With the s , x , and f defined as in Fig. A.1, and the number bits in each region defined as $b_s = 1$, b_x , and b_f , respectively; the represented number A can be found as

$$A = (-1)^s \left(1 + \frac{f}{2^{b_f}} \right) \times 2^{x-2^{(b_x-1)}}. \quad (\text{A.1})$$

Note that in (A.1), the exponent itself is also defined as a signed integer, represented by an *exponent bias* of 2^{b_x-1} . With this definition, the approximate range R and the precision P in decimal digits can be calculated as

$$R \approx \pm 2^{2^{b_x-1}}, \quad (\text{A.2})$$

$$P \approx 2^{-b_f}. \quad (\text{A.3})$$

The properties of the popular precision definitions given in IEEE-754 standard are listed in Table A.1. Note that the range of half-precision is asymmetrical in Table A.1, as the exponent bias in IEEE-754 is defined as 15 rather than the expected 2 or 3.

In most contemporary CPUs, the double-precision standard is used for 64-bits, whereas most commercial GPUs employ single-precision arithmetics for 32-bits.

Table A.1: Definitions and properties of some popular precision levels in IEEE-754 [28]

Definition	b_s	b_x	b_f	Total	Range	Precision (decimal digits)
Half-Precision	1	5	10	16-bits	$6 \times [10^{-8}, 10^5]$	~ 3.3
Single-Precision	1	8	23	32-bits	$\pm 10^{38}$	~ 7.2
Double-Precision	1	11	52	64-bits	$\pm 10^{308}$	~ 15.9
Quad-Precision	1	15	112	128-bits	$\pm 10^{4932}$	~ 34.0

GPUs specifically built for scientific computation (most popularly used in deep learning research) usually offer 64- and even 16-bits (half-precision) Arithmetic Logic Units (ALUs).

Appendix B

Asymptotic Expressions for Bessel and Hankel Functions

Bessel functions of the first kind $J_\nu(\rho)$, second kind $Y_\nu(\rho)$, and their linear combinations are the solutions for the radial component of the homogeneous cylindrical wave equation given as

$$\frac{1}{\rho} \frac{\partial}{\partial \rho} \left(\rho \frac{\partial \Omega(\rho, \phi, z)}{\partial \rho} \right) + \frac{1}{\rho^2} \frac{\partial^2 \Omega(\rho, \phi, z)}{\partial \phi^2} + \frac{\partial^2 \Omega(\rho, \phi, z)}{\partial z^2} + k^2 \Omega(\rho, \phi, z) = 0, \quad (\text{B.1})$$

where (ρ, ϕ, z) are the cylindrical coordinates, with k representing the wavenumber. Hankel functions of the first and second kind, which are linear combinations of Bessel functions, can be defined as

$$H_\nu^{(1)}(\rho) = J_\nu(\rho) + iY_\nu(\rho), \quad (\text{B.2})$$

$$H_\nu^{(2)}(\rho) = J_\nu(\rho) - iY_\nu(\rho), \quad (\text{B.3})$$

where i is the imaginary unit number with $i^2 = -1$.

Each of the cylindrical Bessel and Hankel functions are related to their spherical counterparts via the following relations:

$$j_\nu(\rho) = \sqrt{\frac{\pi}{2\rho}} J_{\nu+0.5}(\rho), \quad (\text{B.4})$$

$$y_\nu(\rho) = \sqrt{\frac{\pi}{2\rho}} Y_{\nu+0.5}(\rho), \quad (\text{B.5})$$

$$h_\nu^{(1)}(\rho) = \sqrt{\frac{\pi}{2\rho}} H_{\nu+0.5}^{(1)}(\rho), \quad (\text{B.6})$$

$$h_\nu^{(2)}(\rho) = \sqrt{\frac{\pi}{2\rho}} H_{\nu+0.5}^{(2)}(\rho). \quad (\text{B.7})$$

Note that the lowercase letters j, y, h represent the spherical Bessel/Hankel functions, while the uppercase letters J, Y, H are used for the cylindrical functions.

In MLFMA, using a time convention of $e^{-i\omega t}$, with $\omega = 2\pi f$ representing the angular frequency, the spherical Bessel and Hankel functions of the first kind are used to expand the free-space Green's function into spherical harmonics as given in (2.2). To derive accurate and simple expressions for the error control of MLFMA as discussed in Chapter 3, we use different asymptotic forms of the Bessel and Hankel functions of the first kind. These asymptotic forms are provided in the following sections.

B.1 Large Argument Approximations

In the derivation of the well-known EBF [12], the large argument approximation of the Bessel and Hankel functions of the first kind [15, eq. (9.2.1)] is used for $|\rho| \rightarrow \infty$ as

$$J_\nu(\rho) \approx \sqrt{\frac{2}{\pi\rho}} \left[\cos\left(\rho - \frac{\nu\pi}{2} - \frac{\pi}{4}\right) + e^{|\Im(\rho)|} O(|\rho|^{-1}) \right], \quad \text{if } |\angle\rho| < \pi, \quad (\text{B.8})$$

$$H_\nu^{(1)}(\rho) \approx \sqrt{\frac{2}{\pi\rho}} e^{i\left(\rho - \frac{\nu\pi}{2} - \frac{\pi}{4}\right)}, \quad \text{if } -\pi < \angle\rho < 2\pi, \quad (\text{B.9})$$

where $\Im$ is the imaginary part operator. Note that the last term in (B.8) disappears when the order is an half-integer [15], i.e., for the spherical Bessel functions.

B.2 Small Argument Approximations

To derive the error control scheme of MLFMA for small box sizes and low-frequencies, we use the small argument approximations of the Bessel and Hankel functions of the first kind [15, eq. (9.1.7)] as

$$J_\nu(\rho) = \frac{(\rho/2)^\nu}{(\nu+1)!}, \quad (\text{B.10})$$

$$H_\nu^{(1)}(\rho) = \left(\frac{-i}{\pi}\right) \frac{(\nu)!}{(\rho/2)^\nu}. \quad (\text{B.11})$$

B.3 Large Order Approximations

One of the most important contributions of this dissertation is the development of a novel error control scheme for MLFMA that is valid at all frequencies, corresponding to all possible arguments of the Bessel and Hankel functions in (2.2). To achieve this, we use the large order approximation given in [15, eq. (9.3.2)] as

$$J_\nu(\nu \operatorname{sech} \gamma_j) \approx \frac{e^{\nu(\tanh \gamma_j - \gamma_j)}}{\sqrt{2\pi\nu \tanh \gamma_j}}, \quad (\text{B.12})$$

$$Y_\nu(\nu \operatorname{sech} \gamma_y) \approx -i \frac{e^{\nu(\gamma_y - \tanh \gamma_y)}}{\sqrt{\frac{1}{2}\pi\nu \tanh \gamma_y}}. \quad (\text{B.13})$$

Note that the large order approximation for the Hankel function of the first kind can be found by substituting (B.12) and (B.13) into (B.2).

Appendix C

Discussion on Legendre Polynomials

In MLFMA, Legendre polynomials are used to represent the spherical harmonics of the free-space Green's function, together with the spherical Bessel and Hankel functions of the first kind. The error control schemes for MLFMA in the literature [12, 16], and the novel error control scheme we introduce in Chapter 3 use the truncation error given in (3.2), which is repeated here as

$$\hat{\epsilon} \approx kR(2\tau + 3) \left| j_{\tau+1}(kv) h_{\tau+1}^{(1)}(kw) P_{\tau+1}(\hat{w} \cdot \hat{v}) \right|, \quad (\text{C.1})$$

where τ is the truncation number and P_ν is the Legendre polynomial of order ν . Legendre polynomials can be calculated using the following recurrence relation as

$$(\nu + 1)P_{\nu+1}(x) = (2\nu + 1)xP_\nu(x) - \nu P_{\nu-1}(x), \quad (\text{C.2})$$

with

$$P_0(x) = 1, \quad (\text{C.3})$$

$$P_1(x) = x. \quad (\text{C.4})$$

The error control schemes used in [12–14] which are also given in Chapter 3 assume $|P_{\tau+1}(\hat{w} \cdot \hat{v})| = 1$ for all arguments as the worst case when estimating

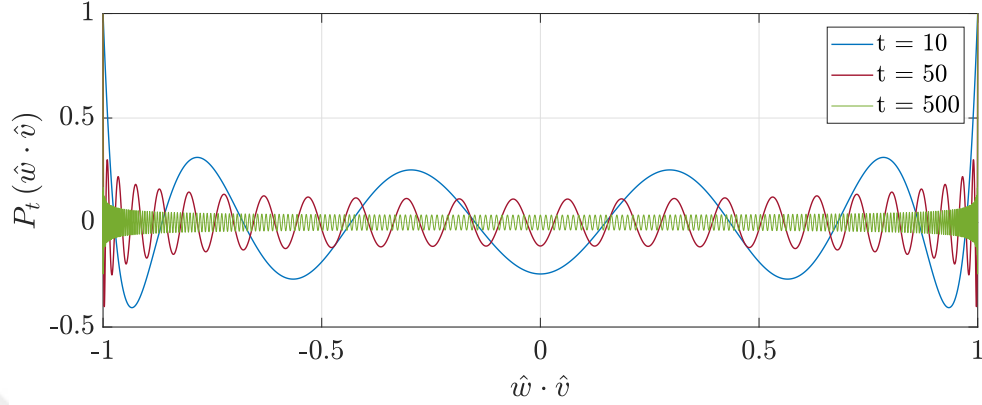


Figure C.1: The value of Legendre polynomials for various orders.

the relative error. Although this assumption simplifies the expressions greatly, it is especially erroneous at large orders, which is exactly the region where these previous error control schemes operate for large arguments. To illustrate this, we present the values of Legendre polynomials for various orders in Fig. C.1.

As seen in Fig. C.1, the absolute value of Legendre polynomials is strictly below unity for most of its range ($|\hat{w} \cdot \hat{v}| \leq 1$). Moreover, the arguments where the absolute value of the Legendre polynomial reaches unity ($\hat{w} \cdot \hat{v} = \pm 1$) are not necessarily the points where the largest errors will occur; e.g., $\hat{w} \cdot \hat{v} = 1/\sqrt{3}$ for the worst case as illustrated in Fig. 2.2. Therefore, the omission of the Legendre polynomial leads to overestimation of the truncation numbers across all frequencies (i.e., box sizes). The omission of the Legendre polynomial may be desirable for a simpler formulation and a closed-form solution for the truncation number. However, even slightly overestimated truncation numbers cause a much higher required machine precision especially for low-frequencies because of the increased numerical instability of the Hankel functions at small arguments. As a result, the Legendre polynomial must be kept intact for an accurate error control scheme.

For the novel simplified expressions we present in Section 3.3, we require a simple analytical representation of Legendre polynomials, namely the RMS value of Legendre polynomials across its argument range. To derive the RMS value, we start with the compact representation of Legendre polynomials, called the

Rodrigues' formula [15, eq. (8.6.18)] given by

$$P_\nu(x) = \frac{1}{2^\nu \nu!} \frac{d^\nu}{dx^\nu} (x^2 - 1)^\nu. \quad (\text{C.5})$$

Using (C.5), we can define the RMS value of Legendre polynomials as a function of their orders as

$$P_{RMS}(\nu) = \sqrt{\int_{-1}^{+1} (P_\nu(x))^2 dx}. \quad (\text{C.6})$$

Writing the integral in (C.6) explicitly we obtain,

$$\int_{-1}^{+1} (P_\nu(x))^2 dx = \frac{1}{2^{2\nu} (\nu!)^2} \int_{-1}^{+1} \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} dx. \quad (\text{C.7})$$

Using integration by parts on the right-hand-side (RHS) of (C.7), and defining $U = \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu}$ and $dV = \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} dx$, we get

$$\begin{aligned} \int_{-1}^1 \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} dx &= \left[\frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} \frac{d^{\nu-1} (x^2 - 1)^\nu}{dx^{\nu-1}} \right] \Big|_{-1}^{+1} \\ &\quad - \int_{-1}^1 \frac{d^{\nu+1} (x^2 - 1)^\nu}{dx^{\nu+1}} \frac{d^{\nu-1} (x^2 - 1)^\nu}{dx^{\nu-1}} dx. \end{aligned} \quad (\text{C.8})$$

Note that the first term in the RHS of (C.8) will always have a factor of $(x^2 - 1)$ due to the term $\frac{d^{\nu-1} (x^2 - 1)^\nu}{dx^{\nu-1}}$; hence, the first term of RHS is equal to zero for $x = 1$ or $x = -1$, reducing (C.8) to

$$\int_{-1}^1 \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} \frac{d^\nu (x^2 - 1)^\nu}{dx^\nu} dx = - \int_{-1}^1 \frac{d^{\nu+1} (x^2 - 1)^\nu}{dx^{\nu+1}} \frac{d^{\nu-1} (x^2 - 1)^\nu}{dx^{\nu-1}} dx. \quad (\text{C.9})$$

What we essentially show in (C.7)–(C.9) is that we can shift the derivative order of one term to the other, inside the integrand. If we repeat this integration by parts procedure a total of ν times, we can express (C.7) as

$$\int_{-1}^{+1} (P_\nu(x))^2 dx = \frac{1}{2^{2\nu} (\nu!)^2} (-1)^\nu \int_{-1}^{+1} \frac{d^{2\nu} (x^2 - 1)^\nu}{dx^{2\nu}} (x^2 - 1)^\nu dx. \quad (\text{C.10})$$

The first term in the integrand of (C.10) is actually a scalar, since the largest degree factor in $(x^2 - 2)^\nu$ (i.e., $x^{2\nu}$) is equal to the derivative order. Therefore, we can simply write the derivative term as

$$\frac{d^{2\nu} (x^2 - 1)^\nu}{dx^{2\nu}} = \frac{d^{2\nu} x^{2\nu}}{dx^{2\nu}} = (2\nu)!. \quad (\text{C.11})$$

Substituting (C.11) into (C.10), we obtain

$$\int_{-1}^{+1} (P_\nu(x))^2 dx = \frac{(-1)^\nu (2\nu)!}{2^{2\nu} (\nu!)^2} \int_{-1}^{+1} (x^2 - 1)^\nu dx. \quad (\text{C.12})$$

Writing the integral on the RHS of (C.12) explicitly, we have

$$\int_{-1}^{+1} (x^2 - 1)^\nu dx = \int_{-1}^{+1} (x - 1)^\nu (x + 1)^\nu dx. \quad (\text{C.13})$$

Using integration by parts with $U = (x - 1)^\nu$ and $dV = (x + 1)^\nu dx$, we obtain

$$\int_{-1}^{+1} (x^2 - 1)^\nu dx = \frac{(x - 1)^\nu (x + 1)^{\nu+1}}{\nu + 1} \Big|_{-1}^{+1} - \frac{\nu}{\nu + 1} \int_{-1}^{+1} (x - 1)^{\nu-1} (x + 1)^{\nu+1} dx. \quad (\text{C.14})$$

Similar to (C.8), the first term in the RHS of (C.14) is equal to zero due to the presence of $(x - 1)$ and $(x + 1)$ factors. By repeating the integration by parts scheme a total of ν times, we can again shift the power of the $(x - 1)$ term in the integrand of (C.14) to the $(x + 1)$ term, resulting in

$$\int_{-1}^{+1} (x^2 - 1)^\nu dx = (-1)^\nu \frac{\nu!}{(\nu + 1)(\nu + 2) \cdots (2\nu - 1)(2\nu)} \int_{-1}^{+1} (x + 1)^{2\nu} dx. \quad (\text{C.15})$$

Evaluating the simple integral in the RHS of (C.15), and after further simplifications, we have

$$\int_{-1}^{+1} (x^2 - 1)^\nu dx = (-1)^\nu \frac{(\nu!)^2}{(2\nu + 1)!} 2^{2\nu+1}. \quad (\text{C.16})$$

Substituting (C.16) into (C.12) results in

$$\int_{-1}^{+1} (P_\nu(x))^2 dx = \frac{(-1)^\nu (2\nu)!}{2^{2\nu} (\nu!)^2} \frac{(-1)^\nu 2^{2\nu+1} (\nu!)^2}{(2\nu + 1)!} = \frac{2}{2\nu + 1}. \quad (\text{C.17})$$

Finally, we substitute (C.17) into (C.6) to obtain the RMS value of Legendre polynomials over an integration range of $[-1, 1]$ as

$$P_{RMS}(\nu) = \sqrt{\int_{-1}^{+1} (P_{\nu}(x))^2 dx} = \sqrt{\frac{2}{2\nu + 1}}. \quad (\text{C.18})$$

Equation (C.18) can be used as a simple alternative to the Legendre polynomial in (C.1), enabling computationally tractable error control schemes as shown in Chapter 3.