COMPARISON OF VARIOUS INTERPOLATION METHODS

IN

THE FAST MULTIPOLE METHOD

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

NİLÜFER ASLIHAN ÖZDEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2000

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Fatih Canatan
Head of the Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Sencer Koç
Supervisor

Examining Committee Members

Prof. Dr. Canan Toker (Chairman)
Assoc. Prof. Dr. Sencer Koç
Prof. Dr. Nilgün Günalp
Prof. Dr. Ayhan Altıntaş (Bilkent University)
Assist. Prof. Dr. Arzu Tuncay Koç

ABSTRACT

COMPARISON OF VARIOUS INTERPOLATION METHODS IN THE FAST MULTIPOLE METHOD

Özdemir, Nilüfer Aslıhan

M.S., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Sencer Koç

August 2000, 94 pages

In this thesis, the problem of interpolation in the upward pass of the Multilevel Fast Multipole Algorithm (MLFMA) is studied. One method used in this problem is the local Lagrangian approximation which allows the use of Gauss-Legendre points, that are nonuniformly spaced, leading to accurate integration by the Gaussian quadrature in the downward pass of the MLFMA. In this work, the adaptive conjugate Toeplitz (ACT) method which is developed for the reconstruction of bandlimited functions from nonuniformly spaced samples is improved with the nonuniform fast Fourier transform (FFT) algorithm. This revised method is compared with the local Lagrangian approximation on the basis of their accuracy and their numerical complexities. The local Lagrangian approximation yields numerical complexity of $O(N)$, where N is the number of samples, whereas the numerical load due to the ACT method is $O(N \log_2 N)$. However, the interpolation error due to the ACT method is several orders of magnitude smaller than that due to the local

Lagrangian approximation. The order and error estimates are verified by numerical simulations.

Keywords: MLFMA, interpolation, nonuniform sampling, ACT, nonuniform FFT.

ÖZ

HIZLI ÇOK KUTUP YÖNTEMİNDE ARA KESTİRİM YÖNTEMLERİNİN KARŞILAŞTIRILMASI

Özdemir, Nilüfer Aslıhan

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Sencer Koç

Ağustos 2000, 94 Sayfa

Bu tezde çok katmanlı hızlı çok kutup algoritmasının (MLFMA) yukarı yönde hesaplamalarında ortaya çıkan ara kestirim problemi incelenmiştir. Bu amaçla kullanılan bir yöntem yerel Lagrange yakınlaştırmasıdır. Bu yöntem, MLFMA'nın aşağı yönde hesaplamalarında doğru sonuç elde edilmesini sağlayan Gauss sayısal tümlev kuralının kullandığı düzensiz Gauss-Legendre noktalarından ara kestirim yapılmasına olanak verir. Bu çalışmada düzensiz örneklenmiş bant sınırlı fonksiyonların yeniden oluşturulması için geliştirilen uyarlanır eşlenik Toeplitz (ACT) yöntemi, düzensiz hızlı Fourier dönüşümü (FFT) tekniği ile iyileştirilmiştir. Geliştirilen yöntem, doğruluk ve sayısal karmaşıklık açısından yerel Lagrange yakınlaştırması ile karşılaştırılmıştır. Örnek sayısı N olan bir durumda, Lagrange yakınlaştırması N, ACT yöntemi ise $N \log_2 N$ düzeyinde sayısal karmaşıklık içermektedir. Ara kestirim hatası açısından ise ACT yöntemi, yerel Lagrange

yakınlaştırmasından çok daha iyi sonuçlar vermektedir. Sayısal karmaşıklık ve hata tahminleri sayısal benzetimlerle doğrulanmıştır.

Anahtar Kelimeler: MLFMA, ara kestirim, düzensiz örnekleme, ACT, düzensiz hızlı Fourier dönüşümü.

To the Memory of My Father

ACKNOWLEDGEMENTS

I would like to thank Assoc. Prof. Dr. Sencer Koç for his patient supervision, guidance, suggestions and encouragement throughout the study of this thesis.

I am also grateful to my family and my colleagues in ASELSAN Inc., without whose support the writing of this thesis would not be possible.

TABLE OF CONTENTS

ABSTRACT.....iii

ÖZ.....v

ACKNOWLEDGEMENTS.....viii

TABLE OF CONTENTS.....ix

LIST OF TABLES.....xii

LIST OF FIGURES.....xiii

CHAPTER

1 INTRODUCTION

2.4.5. Near Interactions....13
2.5. Overall Numerical Complexity....13

3. INTERPOLATION IN MLFMA....15
3.1. MLFMA Formulation....15
3.1.1. Introduction to MLFMA....15
3.1.2. Nested Grouping Scheme....20
3.1.3. MLFMA Formulation....23
3.1.3.1. Upward Pass....24
3.1.3.2. Downward Pass....26
3.2. Interpolation in MLFMA....30
3.2.1. Definition of the Interpolation Problem.... 30
3.2.2. An Overview of Interpolation Methods....32
3.2.2.1. Cardinal Series Expansion....33
3.2.2.2. Convergence Factor....34
3.2.2.3. Self-Truncating Series Expansion....35
3.2.2.4. Approximate Prolate Series Expansion....36
3.2.2.5. Interpolation by Sampling Window....37
3.2.2.6. Lagrangian Interpolation....38
3.2.3. Numerical Results and Discussion....40
3.2.3.1. Simplification of the Interpolation Problem....41
3.2.3.2. Numerical Results....43

4. INTERPOLATION BY ACT METHOD....63
4.1. Introduction....64
4.2. Discrete Irregular Sampling....65
4.3. Reconstruction by the Frame Operator....66
4.4. Accelearion of the ACT Algorithm....67
4.4.1. Adaptive Weights Method....67
4.4.2. Conjugate Gradient Method....69

4.4.2.1. Algorithmic Prescription....69
4.4.2.2. Numerical Complexity....73
4.4.3. Nonuniform FFT Algorithm....74
4.4.3.1. Algorithmic Prescription....75
4.4.3.2. Numerical Complexity....76
4.5. Numerical Results and Discussion....77

5. CONCLUSION....90

REFERENCES....92

LIST OF TABLES

TABLE

3.1. N_α, n and N_θ for uniformly spaced points in each level in the 7-level FMM.....47

3.2. N , n and N_θ for the Gauss-Legendre points in each level in the 7-level FMM.....55

4.1. Best values of b and the corresponding maximum errors for each level when $m = 2$.....82

LIST OF FIGURES

FIGURES

2.1. The basic geometry of source and field points, R' and R respectively, and displacement vectors.....7

2.2. The source point $r'_{n'}$, the field point r_n, and their group centers, $R'_{n'}$ and R_n, respectively, and displacement vectors $\vec{d}_1$, $\vec{X}$, and $\vec{d}_2$.....8

2.3. Grouping for a simple surface.....10

3.1. Near and far interactions at level 2 in a 4-level division.....21

3.2. Near and far interactions at level 3 in a 4-level division.....22

3.3. Near and far interactions at level 4 in a 4-level division.....22

3.4. Upward pass at level 4 for a 4-level division.....26

3.5. Upward pass at level 3 for a 4-level division.....26

3.6. Far interactions at level 2 for a 4-level division.....29

3.7. Far interactions at level 3 for a 4-level division.....29

3.8. $P_2(\cos\theta)$ vs. θ. Interpolation of $P_2(\cos\theta)$.....43

3.9. $P_2(\cos\theta)$ vs. θ. Interpolated data when $\delta = 0$ 44

3.10. $P_2(\cos\theta)$ vs. θ. Interpolated data when $\delta = 0.25$ 45

3.11. $P_2(\cos\theta)$ vs. θ. Interpolated data when $\delta = 0.5$ 45

3.12. $P_2(\cos\theta)$ vs. θ. Interpolated data when $\delta = 0.75$ 46

3.13. Error in interpolation when $\delta = 0$ 48

3.14. Error in interpolation when $\delta = 0.25$ 49

3.15. Error in interpolation when $\delta = 0..5$ 50

3.16. Error in interpolation when $\delta = 0.75$ 51

3.17. Numerical load when δ = 0, 0.25, 0.5, 0.75 52

3.18. Error in local interpolation when $\delta = 0.75$ 53

3.19. Numerical load in local interpolation when $\delta = 0.75$ in comparison with the numerical load in global interpolation.....54

3.20. $P_2(\cos\theta)$ vs. θ. Interpolated data from the nonuniformly spaced samples when $\delta = 0.75$ 57

3.21. Error in interpolation from the nonuniformly spaced. samples.....57

3.22. Numerical load in interpolation from nonuniformly spaced samples compared with $O(N_\theta^2(q))$ curve.....59

3.23. $P_2(\cos\theta)$ vs. θ. 4-point Lagrange interpolation of $P_2(\cos\theta)$.....59

3.24. $P_7(\cos\theta)$ vs. θ. 4-point Lagrange interpolation of $P_7(\cos\theta)$.....60

3.25. Error in 4-point Lagrange interpolation.....60

3.26. Numerical load in 4-point Lagrange interpolation.....61

4.1. Maximum values of the error for $m = 2, 3, 4$.....83

4.2. The numerical load due to the implementation of the nonuniform FFT algorithm in the first part for $q = 2, q = 8, q = 14, q = 22$.....83

4.3. The numerical load due to the implementation of step1 by nonuniform FFT, and by direct computation.....84

4.4. Error due to the implementation of the CG or CG-FFT methods for tolerance of 10^{-6}.....85

4.5. The numerical load due to the implementation of the second step by Gaussian elimination, by the CG method, by the CG-FFT method.....86

4.6. Overall error due to the implementation of the nonuniform FFT algorithm in the third step.....87

4.7. Numerical load due to the implementation of the third step by direct evaluation, by the nonuniform FFT algorithm.....88

4.8. Comparison of the numerical load due to the 4-point Lagrange

interpolation, due to ACT algorithm.....89

4.9. Error due to 4-point Lagrange interpolation, error due to ACT algorithm...89

CHAPTER 1

INTRODUCTION

Solving complex wave scattering problems has preoccuppied scientists and engineers for many years because it helps to understand many physical phenomena and aids in a large number of engineering problems. Examples of such applications include optical waveguide design, signature prediction of aircraft, remote sensing, oil prospecting, inverse scatttering and imaging, inverse optics, and computer-aided engineering and design, e.g., in antennas [2].

The need to compute the scattered field in an accurate and efficient way has stimulated the development of fast algorithms for the solution of electromagnetic scattering problems. Chew, Lu and Yang [2] present a review of these algorithms, among which the Fast Multipole Method (FMM) takes an important part. Originally developed for particle simulations [9], FMM was soon applied to the boundary integral equations. For the Laplace's equation the numerical load can be reduced to $O(N)$ [6], while for the Helmholtz equation the numerical load is $O(N^{4/3})$ in two dimensions [5,16], and $O(N^{3/2})$ in three dimensions [3], where N is the number of scatterers. The numerical load in three dimensions can be further reduced to $O(N)$ by nested grouping of scatterers which leads to Multilevel Fast Multipole Algorithm (MLFMA) [13].

MLFMA consists of two parts: upward pass and downward pass. In the upward pass, outgoing multipole expansion inside each cell is obtained by using addition theorems starting from the cells at the lowest level. In the downward pass, the outgoing wave expansions of the cells that are in the far zone of the cell in

consideration are translated to a standing wave expansion at the center of the cell in consideration [13].

In this thesis, the interpolation problem in the upward pass of MLFMA is studied. In the upward pass, the multipole expansions of groups at a certain level can be obtained from the multipole expansions of groups at a lower level through interpolation. If the outgoing multipole expansions are evaluated at the Gauss-Legendre points for accurate integration in the downward pass, the sample points in θ, where θ is the colatitude angle in spherical coordinates, for different levels are quite different and nonuniform. One method is to apply local Lagrangian approximation in θ variables, which leads to a numerical load of $pN_\theta(q)$ for level q, where p is the number of local points around the point at which the function is to be evaluated, and $N_\theta(q)$ is the number of sample points at level q. If uniformly spaced points and appropriate interpolation schemes are used, the interpolation error bound becomes much smaller while the numerical load is almost the same as that of the Lagrangian approximation. However, for this case the quadrature in the downward pass is not accurate.

The ACT algorithm by Feichtinger, Gröchenig, and Strohmer [8], which is a combination of the adaptive weights method, the conjugate gradient method and the use of Toeplitz matrices, is introduced for the nonuniform interpolation in θ. In this work, the algorithm is improved by the nonuniform FFT and inverse FFT algorithms due to Dutt and Rokhlin for the critically sampled case [4]. The numerical load of the improved ACT algorithm is $O(mN_\theta(q)\log_2 N_\theta(q)+N_\theta(q)q')$, where m and q' are parameters to control numerical efficiency and accuracy in the nonuniform FFT algorithm, respectively. With the proper choice of these parameters, the error due to the ACT algorithm is several orders of magnitude smaller than the error due to the local Lagrangian approximation.

The outline of the thesis is as follows:

Chapter 2 gives an overview of FMM on a simple 3-D scattering problem adopted from [3]. Chapter 3 introduces the concept of MLFMA through a multiple scattering problem from a cluster of acoustical scatterers [13]. By separating the MLFMA algorithm into downward and upward passes, the definition of the interpolation problem, and its simplification are given in Chapter 3. The numerical load and accuracy due to the conventional uniform interpolation methods, and due to the Lagrangian approximation, which assumes the Gauss-Legendre points as sample points, are also illustrated and discussed in Chapter 3. Chapter 4 introduces the ACT method. The numerical load and error obtained by the implementation of this method and its comparison with the local Lagrangian interpolation are given in this chapter. Finally the concluding remarks can be found in Chapter 5.

CHAPTER 2

AN OVERVIEW OF FMM

2.1. Introduction

One of the approaches in the solution of electromagnetic scattering problem is to solve the large matrix equation arising from discretization of the surface or volume integral equation where the unknowns are the amplitudes of equivalent surface or volume current density distributions induced by the incident field on each discretized region. This approach has the advantage of producing scattering solution for an arbitrary geometry [19], and lets one model a large scatterer as a composite of small subscatterers. However, the numerical load is excessive -as high as $O(N^3)$- when Gaussian elimination is used, where N is the number of unknowns.

The use of iterative methods and the multipole expansion of Green's function leading to grouping of the scatterers, which is called *'The Fast Multipole Method'*, help this approach in numerical sense and reduce the numerical complexity of the 3-D problem to $O(N^{3/2})$ [3] and 2-D problem to $O(N^{4/3})$ [5,17]. Further reduction in numerical complexity to $O(N)$ is achieved by nested grouping of the scatterers in FMM, which is called *'The Multilevel FMM'* [13].

In this chapter, an overview of FMM will be given through the solution of a simple 3-D scattering problem adopted from [3]. Throughout the following analysis $e^{j\omega t}$ time convention is used and suppressed.

2.2 Formulation of Scattering Problem

A scattering problem can be defined by the scalar wave equation [3]:

$$(\nabla^2 + k^2)\Phi(\vec{r}) = 0 \tag{2.1}$$

and the boundary conditions

$$\Phi(\vec{r}) = 0 \;\; ; \;\; \vec{r} \text{ on S}, \tag{2.2}$$

$$\lim_{r\to\infty} r\left(\frac{\partial \Phi(\vec{r})}{\partial r} - jk\,\Phi(\vec{r})\right) = 0 \tag{2.3}$$

where S is the surface of the bounded scatterer, are imposed on equation (2.1) leading to the solution:

$$\Phi_{sca}(\vec{r}) = -\Phi_{inc}(\vec{r}) = -\int_{S'} ds' I(\vec{r}') \frac{e^{-jk|\vec{r}-\vec{r}'|}}{4\pi|\vec{r}-\vec{r}'|} \tag{2.4}$$

where $I(\vec{r})$ is the equivalent surface current density induced by the incident field .

The discretization of Eq. (2.4) by the method of moments results in a linear system of equations that can be formulated as a matrix equation:

$$\mathbf{\Phi} = \overline{\mathbf{Z}} \cdot \mathbf{I} \tag{2.5}$$

where

$\overline{\mathbf{Z}}$ is an $N \times N$ coefficient matrix whose entries are

$$Z_{nn'} = \int_{S_n} ds \int_{S_{n'}} ds' f_n(\vec{r}) \frac{e^{-jk|\vec{r}-\vec{r}'|}}{4\pi|\vec{r}-\vec{r}'|} f_{n'}(\vec{r}') \;, \tag{2.6}$$

$\mathbf{I}$ is a vector of length N whose elements are unknown amplitudes of current density distributions on each subdomain, and $\mathbf{\Phi}$ is a vector of length N whose elements are the scalar potential values due to the incident field [3].

The choice of same function, f_n, for both basis and weighting functions, named as the Galerkin method, results in rapid convergence [3].

It is obvious that solution of Eq. (2.5) by Gaussian elimination has numerical complexity of $O(N^3)$. As the scatterer size increases, the number of subscatterers increases rapidly, which means a rapid increase in solution time and memory requirements.

One can overcome these requirements using iterative methods such as the conjugate gradient method. In an iterative solution scheme, each matrix-vector multiplication requires $O(N^2)$ operations leading to overall numerical complexity of $O(pN^2)$, where p is the number of iterations required for desired accuracy [13].

Further reduction in the order of numerical complexity is possible by decreasing the order of number of operations in matrix-vector multiplication. FMM is a method that accelerates the repeated application of discretized integral operator, which is the coefficient matrix in our case, to candidate solution vectors in an iterative solution scheme [6].

2.3. Identities in FMM

FMM uses two identities. One of them is the multipole expansion of free space Green's function, which is the kernel of the integral in Eq. (2.4), using the Gegenbauer's Addition Theorem, [1]:

$$\frac{e^{-jk|\vec{X}+\vec{d}|}}{|\vec{X}+\vec{d}|} = -jk\sum_{n=0}^{\infty}(2n+1)j_n(kd)h_n^{(2)}(kX)P_n(\hat{d}\cdot\hat{X}) \qquad (2.7)$$

where

j_n : spherical Bessel function of the first kind of order n,

$h_n^{(2)}$: spherical Hankel function of the second kind of order n,

P_n : Legendre polynomial of order n,

and $d < X$

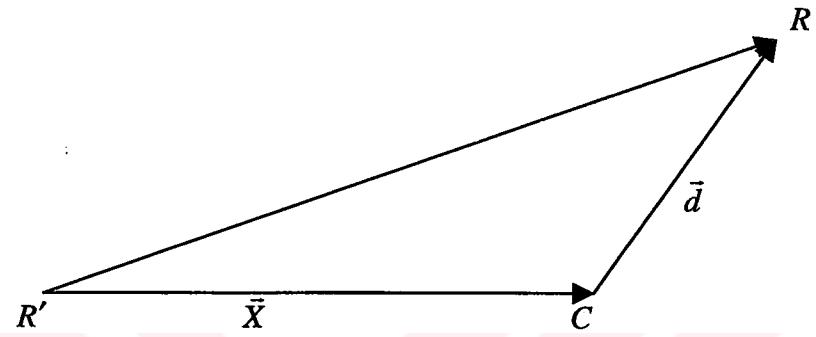

Figure 2.1. The basic geometry of source and field points, R' and R respectively, and displacement vectors

The second identity is the expansion of the product $j_n P_n$, [21]:

$$4\pi j^n j_n(kd) P_n\left(\hat{d}\cdot\hat{X}\right) = \int d^2\hat{k} e^{j\vec{k}\cdot\vec{d}} P_n\left(\hat{k}\cdot\hat{X}\right) \tag{2.8}$$

Substituting Eq. (2.8) into Eq. (2.7), we obtain

$$\frac{e^{-jk\left|\vec{X}+\vec{d}\right|}}{\left|\vec{X}+\vec{d}\right|} = \frac{-jk}{4\pi}\sum_{n=0}^{\infty}(-j)^n(2n+1)h_n^{(2)}(kX)\int d^2\hat{k} e^{j\vec{k}\cdot\vec{d}} P_n\left(\hat{k}\cdot\hat{X}\right). \tag{2.9}$$

The interchange of summation and integration is possible if the sum is truncated as it is intended in usual numerical practice. Then Eq. (2.9) becomes

$$\frac{e^{-jk\left|\vec{X}+\vec{d}\right|}}{\left|\vec{X}+\vec{d}\right|} = \frac{-jk}{4\pi}\int d^2\hat{k} e^{j\vec{k}\cdot\vec{d}} \sum_{n=0}^{N_g}(-j)^n(2n+1)h_n^{(2)}(kX)P_n\left(\hat{k}\cdot\hat{X}\right). \tag{2.10}$$

Now Eq. (2.10) can be computed in two steps. First, the summation

$$\tau_{N_g}(kX, \cos\theta) \equiv \sum_{n=0}^{N_g} (-j)^n (2n+1)\, h_n^{(2)}(kX) P_n(\cos\theta) \tag{2.11}$$

where, $\cos\theta = \hat{k} \cdot \hat{X}$, is computed for κ different directions of θ.

Second the integration is performed:

$$\frac{e^{-jk|\vec{X}+\vec{d}|}}{|\vec{X}+\vec{d}|} \approx \frac{-jk}{4\pi} \int d^2\hat{k} e^{jk\cdot\vec{d}} \tau_{N_g}(kX, \hat{k} \cdot \hat{X}) \tag{2.12}$$

Eq. (2.12) can be applied in the scattering problem solution using the geometry in Fig. 2.2.

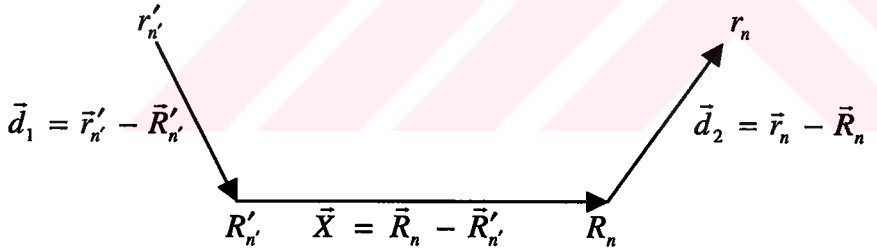

Figure 2.2. The source point $r'_{n'}$, the field point r_n and their group centers, $R'_{n'}$ and R_n, respectively and displacement vectors $\vec{d}_1$, $\vec{X}$ and $\vec{d}_2$.

Eq. (2.6) now becomes

$$Z_{nn'} \approx \frac{-jk}{(4\pi)^2} \int_S ds\, f_n(\vec{r}) \int_{S'} ds' f_{n'}(\vec{r}') \int d^2\hat{k} e^{jk\cdot(\vec{d}_1+\vec{d}_2)} \tau_{N_g}(kX, \hat{k} \cdot \hat{X}). \tag{2.13}$$

With the interchange of integrations, Eq. (2.13) can be rewritten as

$$Z_{nn'} \approx \frac{-jk}{(4\pi)^2} \int d^2\hat{k} \int_{S_n} ds\, f_n(\vec{r})\, e^{jk\cdot(\vec{r}_n - \vec{R}_n)}\, \tau_{N_g}\left(kX, \hat{k}\cdot\hat{X}\right) \int_{S_{n'}} ds' e^{-jk\cdot(\vec{r}'_{n'} - \vec{R}_{n'})} f_{n'}(\vec{r}'). \tag{2.14}$$

Keeping in mind that $\left|\vec{X}\right| > \left|\vec{d}\left(=\vec{d}_1 + \vec{d}_2\right)\right|$, $Z_{nn'}$ in Eq. (2.14) denotes the interactions between well-separated subscatterers. It is also clear that the last two integrations in Eq. (2.14) are the Fourier transforms of the basis functions, which means physically the far fields of the basis functions. These interactions or the far fields can be grouped together before the integration over $\hat{k}$.

For the near interactions, the entries should be computed directly using Eq. (2.6).

Here it must be noted that FMM is not restricted to MoM. The choice of MoM here is due to its simple representation and wide practice area.

2.4. Algorithmic Prescription

As can be inferred from Eq. (2.14), the idea of FMM is to divide the subscatterers into groups and distinguish the interactions between the subscatterers as *far* and *near* interactions. The algorithm can be explained in four steps. First, the multipole expansion of Green's function translates the scattered field of different scattering centers within a group into a single center, called the *aggregation* step. Second, for each group, field scattered by all the other group centers is first received by the group center, which is called the *group interactions* step. Third, the received field by the group centers are redistributed to the subcatterers belonging to the group, called the *disaggregation* step. As the fourth step, the interactions between the scatterers within the same group are computed directly, which is called the *near interactions* step [13,19].

2.4.1. Grouping Scheme

We assume that N basis functions are divided into M groups each containing N/M basis functions. The basis function index $n(m,\alpha)$ is related to a pair of indices (m,α), where α denotes the basis function in the m th group. The grouping and index relation for a simple surface is sketched in Figure 2.3 [3].

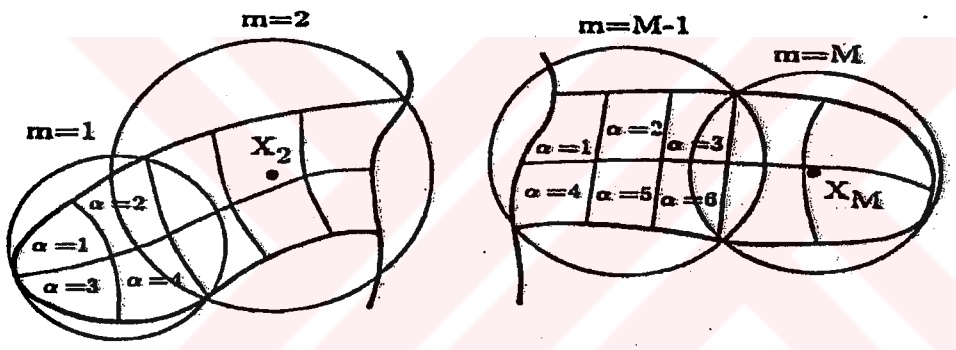

Figure 2.3. Grouping for a simple surface

Remembering the condition for the Gegenbauer's Addition Theorem, which is stated as $|\vec{X}| > |\vec{d}(=\vec{d}_1+\vec{d}_2)|$, the near interactions exist between the scatterers which belong to groups with centers separated by a distance less than D, where D is the diameter of larger of the groups considered. The far interactions occur when the groups are separated by a distance larger than D [3].

2.4.2. Aggregation

This step includes the translation of the scattered field by each subscatterer to the center of the group it belongs to. The last integration in Eq. (2.14) represents the aggregation step.

The Fourier transform of each basis function $f_{n'(m',\alpha)}$ is computed for κ directions of $\hat{k}$ and stored as

$$V_{m'\alpha}(\hat{k}) = \int_{S_{m'\alpha}} ds' e^{-j\vec{k}\cdot(\vec{r}'-\vec{R}_{m'})} f_{n'(m',\alpha)}(\vec{r}') \tag{2.15}$$

to be used in the first step of fast matrix-vector computation:

$$s_{m'}(\hat{k}) = \sum_{\alpha} V_{m'\alpha}(\hat{k}) I_{m'\alpha} \tag{2.16}$$

where $I_{m'\alpha}$ is the candidate solution in subdomain $n'(m',\alpha)$ [3].

The summation in equation (2.16) is to be computed for each basis function and for κ different directions, which results in numerical complexity of $O(\kappa N)$.

2.4.3. Group Interactions

This step includes the translation of far fields of the basis functions accumulated at each group center to other centers which are not nearby. The translation operator for each group pair mm' that are not in the neighborhood of each other is computed and stored as:

$$T_{mm'}(\hat{k}) = \frac{k}{(4\pi)^2} \sum_{n=0}^{N_g} (-j)^n (2n+1) h_n^{(2)}(kX_{mm'}) P_n(\hat{k}\cdot\hat{X}_{mm'}) \tag{2.17}$$

to be used in the second step of fast matrix-vector multiplication

$$g_m(\hat{k}) = \sum_{m'} T_{mm'}(\hat{k}) s_{m'}(\hat{k}). \tag{2.18}$$

The summation in Eq.(2.18) is to be computed for all group pairs in M groups and for K different directions, which results in numerical complexity of $O(\kappa M^2)$ [3].

The summation limit, N_g, in the transfer function $T_{mm'}$ depends on the desired accuracy and the value of kD, where D is the diameter of the sphere that confines larger of the groups considered. However, the limit cannot be too large since the Hankel functions start to oscillate when the order exceeds the argument, that is

$$N_g < kX_{mm'} \tag{2.19}$$

is required.

An emprical formula for N_g is

$$N_g = \lfloor kD + C_a \log(kD + \pi) \rfloor + 1 \tag{2.20}$$

where C_a is a parameter to adjust desired accuracy, and D is the diameter of the sphere that confines the largest group. If N_g calculated in Eq. (2.20) violates the condition formulated in Eq. (2.19), groups will be considered to be too close to use FMM, which is in fact the precise definition of nearby region [3,13].

2.4.4. Disaggregation

This step includes the distribution of the accumulated field at group centers to each individual scatterer in their neighborhood. The operator in this step is the conjugate transpose of the operator in the aggregation step. The field translated to the position of the subscatterer α in the m th group is computed as:

$$h_\alpha(\hat{k}) = V^*_{m\alpha}(\hat{k}) g_m(\hat{k}) \tag{2.21}$$

for κ different directions and for each subscatterer, which results in numerical complexity of $O(\kappa N)$.

2.4.5. Near Interactions

This step includes the computation of interactions between scatterers which belong to the same group or nearby groups using standard MoM, which is summarized in section 2.2.

$Z_{nn'}$ expressed in Eq. (2.6) is to be computed for each subscatterer neighborhood, which results in numerical complexity of $O(BN)$, where B is the typical number of neighboring scatterers. B is proportional to the number of scatterers in each group, hence the numerical complexity becomes $O(N^2/M)$ [13].

2.5. Overall Numerical Complexity

The overall numerical complexity coming from the four steps explained in section 2.4 is simply the sum of the numerical complexities of these steps:

$$O(\kappa N) + O(\kappa M^2) + O(\kappa N) + O(N^2/M) \tag{2.22}$$

Since the numerical complexity is expressed in terms of the number of unknowns, κ and M should be expressed in terms of N.

κ directions of $\hat{k}$ must be sufficient for exact integration in Eq. (2.14). The product formula, which is a form of Gauss-Legendre quadrature,

$$\oint f(x,y,z)d\hat{s} = \int_0^{2\pi}\int_0^{\pi} f(\theta,\phi)\sin\theta\, d\theta\, d\phi = \sum_{i=1}^{N_\theta}\sum_{j=1}^{2N_\theta} w_i w_j f(\theta_i,\phi_j) \tag{2.23}$$

is exact for polynomials $x^\alpha y^\beta z^\gamma$ if $\alpha+\beta+\gamma\langle 2N_\theta$, where θ_i are the Gauss-Legendre points, ϕ_j are $2N_\theta$ equally spaced points over the interval $[-\pi,\pi]$, w_i are the Gauss-Legendre weights, and $w_j = \pi/N_\theta$ [13]. Since P_n is a polynomial of order n, the above quadrature rule is exact for $n < 2N_\theta$, and $N_\theta = N_g + 1$ is sufficient for accurate integration [13]. The number of quadrature points can be found as

$$\kappa = 2N_g{}^2 \propto (kD)^2 \propto (N/M)\,. \tag{2.24}$$

The overall numerical complexity now becomes

$$O(N^2/M) + O(NM) \tag{2.25}$$

Optimum order is obtained when the orders in the summation above are balanced:

$$N^2/M = NM \tag{2.26}$$

When the group size N/M is chosen to be proportional to the xth power of the number of scatterers, $N/M \sim N^x$, Eq. (2.26) yields $x = 1/2$ leading to overall numerical complexity of $O(N^{3/2})$.

CHAPTER 3

INTERPOLATION IN MLFMA

3.1. MLFMA Formulation

3.1.1. Introduction to MLFMA

Another general approach in electromagnetic scattering problem solution is based on the multipole expansion of incident and scattered fields. The total field equation, which is the sum of incident and scattered fields, is transformed into a linear system of equations by utilizing the scalar addition theorems and the T-matrix formulation. The resulting matrix equation is solved by an appropriate iterative method. The following formulation describes briefly the mentioned method for a cluster of scatterers [13].

Multipole expansion of incident field about a point $\vec{p}_s$ is expressed as:

$$\Phi^{inc}(\vec{r}_s) = \sum_{n=0}^{n_s} \sum_{m=-n}^{n} a_{nm}^{s} j_n(kr_s) Y_{nm}(\theta_s, \phi_s) \tag{3.1}$$

where n_s is the order at which the expansion is truncated, a_{nm}^{s} are the multipole coefficients, $\vec{r}_s = \vec{r} - \vec{p}_s$ is the vector from the point $\vec{p}_s$ to the observation point $\vec{r}$. The spherical harmonics Y_{nm} are

$$Y_{nm}(\theta,\phi)=(-1)^m\left[\frac{(n-m)!}{(n+m)!}\frac{2n+1}{4\pi}\right]^{1/2}P_n^m(\cos\theta)e^{jm\phi} \tag{3.2}$$

where $P_n^m(x)$ is the Associated Legendre function.

For N scatterers at points described by the position vectors $\vec{p}_i$, $i=1,...,N$ field scattered by the ith scatterer can be expressed using the multipole expansion as

$$\Phi_i^{sca}(\vec{r}_i)=\sum_{n=0}^{n_i}\sum_{m=-n}^{n}b_{nm}^i h_n^{(2)}(kr_i)Y_{nm}(\theta_i,\phi_i) \tag{3.3}$$

where $\vec{r}_i=\vec{r}-\vec{p}_i$ is the vector from the ith scatterer to the observation point, and $h_n^{(2)}$ is the spherical Hankel function of the second kind.

When the two indices n and m are combined in a single index L: (0,0), (1,–1), (1,0), (1,1), ... , (n,m), ... , Eq. (3.1) and Eq. (3.3) can be written as [13]

$$\Phi^{inc}(\vec{r}_s)=\sum_{l=(0,0)}^{L}Rg\Psi_l(k,\vec{r}_s)a_l^s, \tag{3.4}$$

$$\Phi_i^{sca}(\vec{r}_i)=\sum_{l=(0,0)}^{L}\Psi_l(k,\vec{r}_i)b_l^i, \tag{3.5}$$

where $Rg\Psi_L(k,\vec{r}_s)=j_n(kr_s)Y_{nm}(\theta_s,\phi_s)$ are the regular wave functions, and $\Psi_L(k,\vec{r})=h_n^{(2)}(kr)Y_{nm}(\theta,\phi)$ are the outgoing wave functions. Eq. (3.4) and (3.5) can be rewritten using vector notation as [13]

$$\mathbf{\Phi}^{inc}(\vec{r}_s)=Rg\overline{\mathbf{\Psi}}^{\mathrm{t}}(k,\vec{r}_s)\cdot\mathbf{a}^s, \tag{3.6}$$

$$\mathbf{\Phi}_i^{sca}(\vec{r}_i)=\overline{\mathbf{\Psi}}^{\mathrm{t}}(k,\vec{r}_i)\cdot\mathbf{b}^i, \tag{3.7}$$

where $(\cdot)^t$ denotes the transpose of the vector.

The total scattered field is now

$$\mathbf{\Phi}^{sca} = \sum_{i=1}^{N} \overline{\mathbf{\Psi}}^{t}(k, \vec{r}_i) \cdot \mathbf{b}^{i} \tag{3.8}$$

leading to the total field expression as

$$\mathbf{\Phi} = \mathbf{\Phi}^{inc} + \mathbf{\Phi}^{sca} = Rg\overline{\mathbf{\Psi}}^{t}(k, \vec{r}_s) \cdot \mathbf{a}^{s} + \sum_{i=1}^{N} \overline{\mathbf{\Psi}}^{t}(k, \vec{r}_i) \cdot \mathbf{b}^{i}. \tag{3.9}$$

Eq (3.9) can be transformed to a linear system of equations by using the scalar addition theorems and the T - matrix formulation [13].

Scalar addition theorems let one translate the incident and scattered fields to the position of any one of the N scatterers as regular wave functions. An outgoing wave function at the position of the i th scatterer can be expanded into a sum of regular wave functions at the position of the x th scatterer as [13]

$$\Psi_{nm}(k, \vec{r}_i) = \sum_{\upsilon=0}^{\infty} \sum_{\mu=-\upsilon}^{\upsilon} Rg\Psi_{\upsilon\mu}(k, \vec{r}_x)\alpha_{\upsilon\mu,nm}(k, \vec{r}_{xi}) \tag{3.10}$$

where $\vec{r}_{xi} = \vec{p}_i - \vec{p}_x$ is the vector from the x th scatterer to the i th scatterer. The translation coefficients α are

$$\alpha_{\upsilon\mu,nm}(k, \vec{r}_{xi}) = \sum_{n'=|n-\upsilon|}^{n+\upsilon} (-j)^{(\upsilon+n'-n)} Y_{n',m-\mu}(\theta,\phi) h_{n'}^{(2)}(kr_{xi})(-1)^{\mu} \left[\frac{4\pi(2n+1)(2\upsilon+1)}{(2n'+1)}\right]^{1/2} (n\upsilon 00 \mid n\upsilon n' 0)(n\upsilon - m\mu \mid n\upsilon n' - m + \mu), \tag{3.11}$$

where $(j_1 j_2 m_1 m_2 \mid j_1 j_2 j_3 m_3)$ are the Clebsch-Gordan coefficients [13].

Similarly, a regular wave function with the origin at $\vec{p}_s$ can be expanded into a sum of regular wave functions at the position of the x th scatterer as [13]

$$Rg\Psi_{nm}(k, \vec{r}_s) = \sum_{\upsilon=0}^{\infty} \sum_{\mu=-\upsilon}^{\upsilon} Rg\Psi_{\upsilon\mu}(k, \vec{r}_x)\beta_{\upsilon\mu,nm}(k, \vec{r}_{xs}) \tag{3.12}$$

where $\vec{r}_{xs} = \vec{p}_x - \vec{p}_s$ is the vector from the source point to the position of the xth scatterer. β coefficients in Eq. (3.12) can be obtained by replacing the spherical Hankel functions in Eq. (3.11) by the spherical Bessel functions.

By using the addition theorems to express the scattered field at the position of the xth scatterer, one can get

$$\begin{aligned}\overline{\Psi}^t(k,\vec{r}_i)\cdot\mathbf{b}^i &= \sum_{nm}\Psi_{nm}(k,\vec{r}_i)b^i_{nm} \\ &= \sum_{nm}\left\{\sum_{\upsilon\mu} Rg\Psi_{\upsilon\mu}(k,\vec{r}_x)\alpha_{\upsilon\mu,nm}(k,\vec{r}_{xi})\right\}b^i_{nm} \\ &= \sum_{\upsilon\mu} Rg\Psi_{\upsilon\mu}(k,\vec{r}_x)\left\{\sum_{nm}\alpha_{\upsilon\mu,nm}(k,\vec{r}_{xi})b^i_{nm}\right\}. \end{aligned} \tag{3.13}$$

By adopting matrix and vector notation, Eq. (3.13) can be rewritten as [13]

$$\overline{\Psi}^t(k,\vec{r}_i)\cdot\mathbf{b}^i = Rg\overline{\Psi}^t(k,\vec{r}_x)\cdot\overline{\alpha}(k,\vec{r}_{xi})\cdot\mathbf{b}^i. \tag{3.14}$$

Similarly, the multipole expansion of the incident field translated to the position of the xth scatterer can be expressed using the notation in Eq. (3.14) as [13]

$$Rg\overline{\Psi}^t(k,\vec{r}_s)\cdot\mathbf{a}^s = Rg\overline{\Psi}^t(k,\vec{r}_x)\cdot\overline{\beta}(k,\vec{r}_{xs})\cdot\mathbf{a}^s. \tag{3.15}$$

The total field expression using Eq. (3.14) and (3.15) is now

$$\begin{aligned}\Phi = {} & Rg\overline{\Psi}^t(k,\vec{r}_x)\cdot\overline{\beta}(k,\vec{r}_{xs})\cdot\mathbf{a}^s + \sum_{\substack{i=1\\ i\neq x}}^{N} Rg\overline{\Psi}^t(k,\vec{r}_x)\cdot\overline{\alpha}(k,\vec{r}_{xi})\cdot\mathbf{b}^i \\ & + \overline{\Psi}^t(k,\vec{r}_x)\cdot\mathbf{b}^j. \end{aligned} \tag{3.16}$$

The first two terms in the summation in Eq. (3.16) are the incident fields onto the xth scatterer, and the third term is the field scattered by the xth scatterer.

T-matrix, denoted by $\overline{\mathbf{T}}_x$ for the xth scatterer, relates the multipole coefficients of the field scattered by the xth scatterer to the multipole coefficients of the field incident onto the xth scatterer when only the xth scatterer exists. Using T-matrix formulation Eq. (3.16) becomes [13]

$$\mathbf{b}^x = \overline{\mathbf{T}}_x \cdot \left(\overline{\boldsymbol{\beta}}(k, \vec{r}_{xs}) \cdot \mathbf{a}^s + \sum_{\substack{i=1 \\ i \neq x}}^{N} \overline{\boldsymbol{\alpha}}(k, \vec{r}_{xi}) \cdot \mathbf{b}^i \right), \qquad x = 1, \ldots, N. \tag{3.17}$$

In FMM far and near interactions are separated by plane wave expansion of the translation operator $\overline{\boldsymbol{\alpha}}$ for far interactions. Eq. (3.17) can be rewritten as [13]

$$\mathbf{b}^x - \overline{\mathbf{T}}_x \cdot \left[\sum_{l \in N_{l'}} \sum_{i \in G_l; i \neq x} \overline{\boldsymbol{\alpha}}_{near}(k, \vec{r}_{xi}) \cdot \mathbf{b}^i + \sum_{l \notin N_{l'}} \sum_{i \in G_l} \overline{\boldsymbol{\alpha}}_{far}(k, \vec{r}_{xi}) \cdot \mathbf{b}^i \right] = \overline{\mathbf{T}}_x \cdot \overline{\boldsymbol{\beta}}(k, \vec{r}_{xs}) \cdot \mathbf{a}^s. \tag{3.18}$$

where

l' : the index of the group which the xth scatterer belongs to,

$N_{l'}$: the set which contains the neighboring groups of l', including itself,

G_l : the set of indices belonging to the group l.

The first summation in Eq. (3.18) points at the near interactions in the neighborhood of the xth scatterer, and is denoted by $\mathbf{S}^x_{near}$. The second summation accounts for the far interactions and is denoted by $\mathbf{S}^x_{far}$.

$\overline{\boldsymbol{\alpha}}_{near}$ in Eq. (3.18) is the same as $\overline{\boldsymbol{\alpha}}$ in Eq. (3.11), and any element of $\overline{\boldsymbol{\alpha}}_{far}$ can be written as [13]:

$$\alpha^{far}_{\upsilon\mu,nm}(k, \vec{r}_{xi}) = \oint j^n Y_{nm}(\hat{k}) e^{-j\vec{k}\cdot\vec{r}_{xl}} \tilde{\alpha}_{N_\alpha;\lambda'\lambda}(\hat{k}) (-j)^\upsilon Y^*_{\upsilon\mu}(\hat{k}) e^{-j\vec{k}\cdot\vec{r}_{xl'}} d\hat{k}. \tag{3.19}$$

The formulation in Eq. (3.18) results in numerical complexity of $O(N^{10/7})$, where N is the number of scatterers. The complexity can further be reduced to

$O(N)$ by applying nested grouping scheme, which will be explained in section 3.1.2. The algorithm that combines FMM and the nested grouping scheme will be referred to as *Multilevel Fast Multipole Algorithm*, abbreviated as *MLFMA* [13].

3.1.2. Nested Grouping Scheme

Nested grouping is defined as a grouping scheme that divides each group into subgroups in a repeated manner. In this scheme, the cubical cell that confines all the scatterers is called the root cell. This cell is divided into eight cubical child cells, which are in turn divided into eight cells until one reaches the desired level of divisions. Levels consist of cells that are in the same level of division, and are indexed from 0 upwards. A level with a lower(higher) index number will be referred to as a higher(lower) level [13].

Index definitons of scatterers in nested grouping are stated below [13]:

m_q : the indices of the cells in the q th level,

N_{m_q} : the set of indices of the neighbors of the m_q th cell including itself,

I_{m_q} : the interaction list of m_q th cell, which consists of cells that are in the same level as m_q th cell, and are not neighbors of m_q th cell,

G_{m_q} : the index set of the m_q th cell's elements,

C_{m_q} : the index set of the m_q th cell's children.

The idea of nested grouping scheme is to translate the near interactions at a higher level to far interactions at a lower level, hence to reduce the numerical

complexity. By this way, calculation of near interactions is left to the lowest level, where it is done with the least cost.

Fig.s 3.1-3 illustrate the interaction list and neighborhood list at each level for a 4-level division. The interaction list and neighborhood lists are obtained for the cell with * inside and its parents. The cells denoted by F constitute the interaction list for the level considered, and the cells denoted by N constitute the neighborhood list. The final neighborhood list is obtained at level 4.

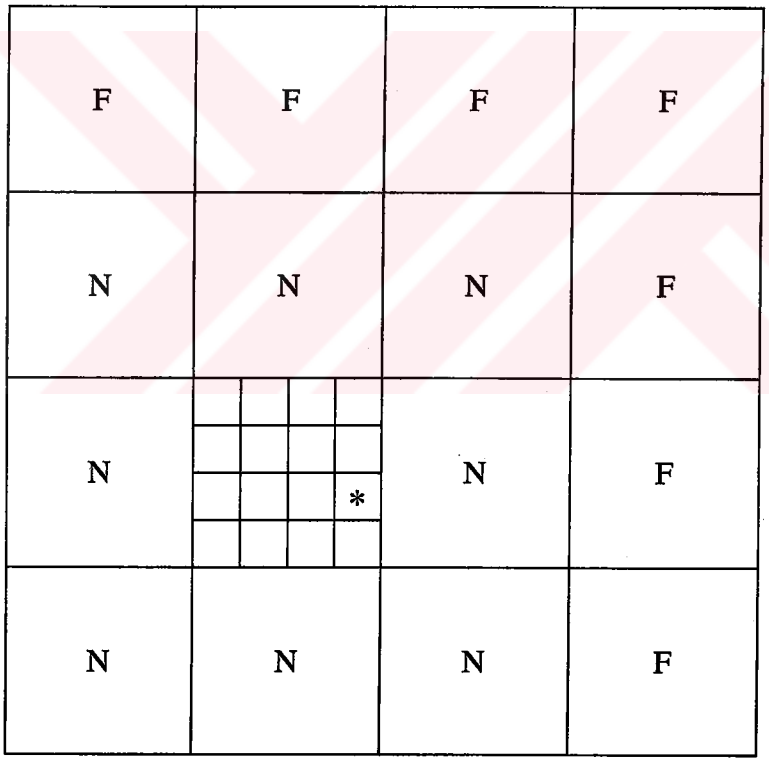

Figure 3.1. Near and far interactions at level 2 in a 4-level division

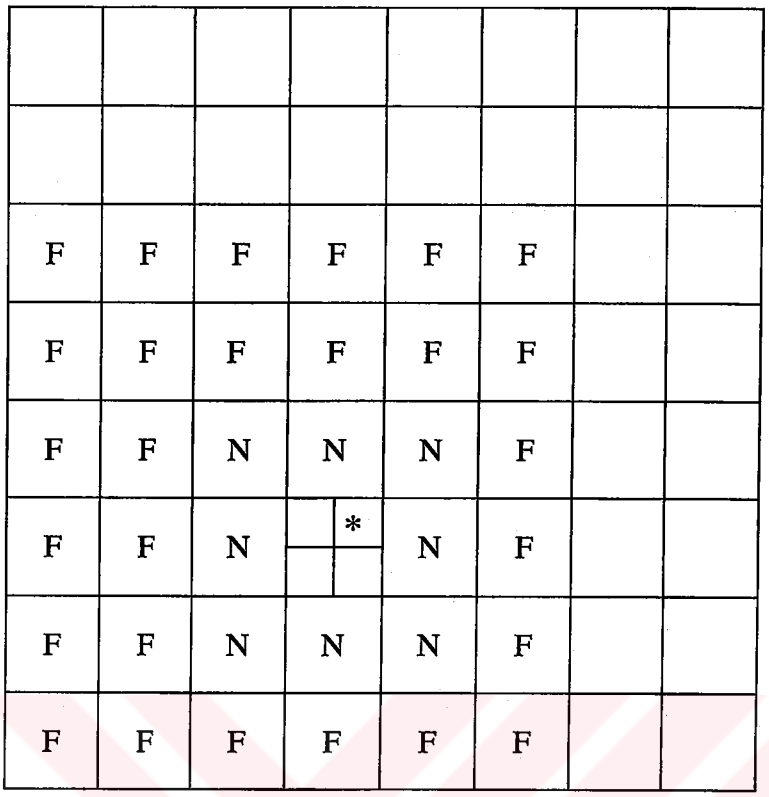

Figure 3.2. Near and far interactions at level 3 in a 4-level division

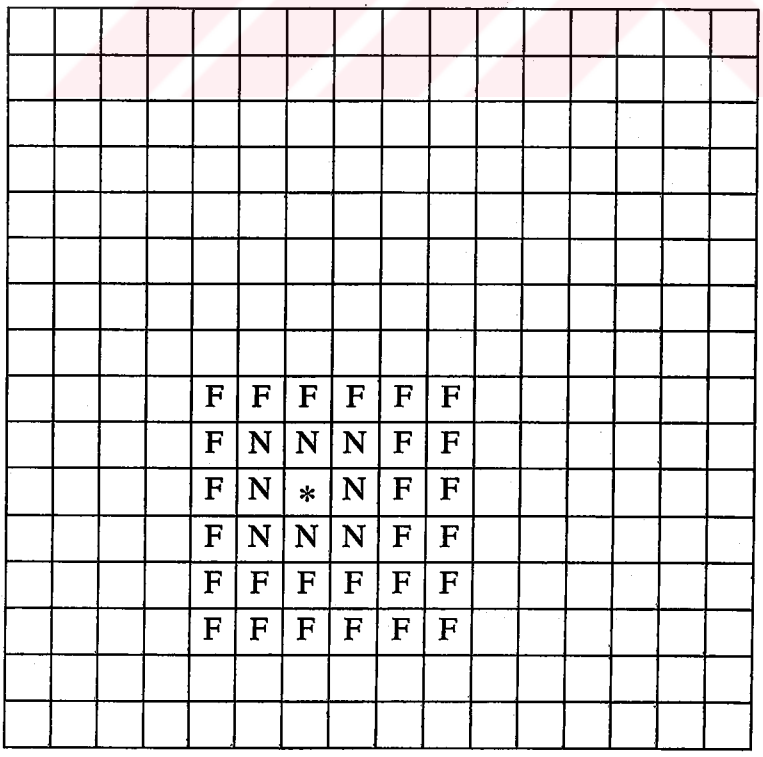

Figure 3.3. Near and far interactions at level 4 in a 4-level division

3.1.3. MLFMA Formulation

Using the definitions given in section 3.1.2 the total effect due to far and near interactions can be expressed as [13]:

$$\begin{aligned}\mathbf{S}^x_{total} &= \mathbf{S}^x_{near} + \mathbf{S}^x_{far} \\ &= \sum_{m'_Q \in N_{m_Q}} \sum_{x \in G_{m_Q}} \overline{\boldsymbol{\alpha}}(k, \vec{r}_{xi}) \cdot \mathbf{b}^i + \sum_{m'_Q \in I_{m_Q}} \oint \mathbf{t}_{m_Q}(\hat{k}) \tilde{\alpha}_{N_\alpha(Q);\lambda_{m_Q}\lambda_{m'_Q}}(\hat{k}) u_{m'_Q}(\hat{k}) d\hat{k} + \\ &\quad \sum_{m'_{Q-1} \in I_{m_{Q-1}}} \oint \mathbf{t}_{m_{Q-1}}(\hat{k}) \tilde{\alpha}_{N_\alpha(Q-1);\lambda_{m_{Q-1}}\lambda_{m'_{Q-1}}}(\hat{k}) u_{m'_{Q-1}}(\hat{k}) d\hat{k} + \ldots \end{aligned} \tag{3.20}$$

where

$$u_{m'_q}(\hat{k}) = \sum_{i \in G_{m'_q}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_q} i}} \mathbf{Y}^t(\hat{k}) \cdot \mathbf{b}^i, \tag{3.21}$$

which is defined as the outgoing multipole expansion of cell m'_q,

$$\tilde{\alpha}_{N_{\alpha(q)}\lambda_{m_q}m'_q}(\hat{k}) = \sum_{n=0}^{N_\alpha(q)} (-j)^n (2n+1) h_n^{(2)}(kr_{\lambda_{m_q}\lambda_{m'_q}}) P_n(\hat{k} \cdot \hat{r}_{\lambda_{m_q}\lambda_{m'_q}}), \tag{3.22}$$

which accounts for group interactions, and is defined as the translation operator that transforms the outgoing multipole expansion of cell m'_q to regular multipole expansion at the center of m_q th cell,

$$\mathbf{t}_{m_q}(\hat{k}) = \mathbf{Y}^*(\hat{k}) e^{-j\vec{k}\cdot\vec{r}_{x\lambda_{m_q}}}, \tag{3.23}$$

and is defined as the translation operator that distributes the regular multipole expansion at the center of the m_q th cell to regular multipole expansion at the position of the x th scatterer, where $(\,)^*$ denotes the conjugate of the vector, $\vec{r}_{\lambda_{m_q}}$ is the center of the m_q th cell, and m_{Q-1} is the parent of the cell m_Q which contains the x th scatterer [13].

As can be deduced from Fig.s 3.1-3.3, the highest level in which the far interactions begin to take effect is the second level. Thus the far interactions can be combined as a sum over q, where q is from 2 to Q;

$$\mathbf{S}_{far}^{x} = \sum_{q=2}^{Q} \sum_{m_q' \in I_{m_q}} \oint \mathbf{t}_{m_q}(\hat{k}) \tilde{\alpha}_{N_\alpha(q);\lambda_{m_q}\lambda_{m_q'}}(\hat{k}) u_{m_q'}(\hat{k}) d\hat{k} \tag{3.24}$$

Combination of the group interactions and the sum of outgoing wave expansions at each group center in each level is denoted by $\vartheta_{m_q}(\hat{k})$, and is expressed as [13] :

$$\vartheta_{m_q}(\hat{k}) = \sum_{m_q' \in I_{m_q}} \tilde{\alpha}_{N_\alpha(q),\lambda_{m_q m_q'}}(\hat{k}) u_{m_q'}(\hat{k}). \tag{3.25}$$

Eq. (3.20) can be rewritten using Eq. (3.24) and (3.25) as

$$\mathbf{S}_{total}^{x} = \mathbf{S}_{near}^{x} + \sum_{q=2}^{Q} \oint \mathbf{t}_{m_q}(\hat{k}) \vartheta_{m_q}(\hat{k}) d\hat{k} \ . \tag{3.26}$$

When the integral in Eq. (3.26) is replaced by a proper quadrature formula, Eq. (3.26) becomes

$$\mathbf{S}_{total}^{x} = \mathbf{S}_{near}^{x} + \sum_{q=2}^{Q} \sum_{n=1}^{K_q} w_n^q \mathbf{t}_{m_q}(\hat{k}_n^q) \vartheta_{m_q}(\hat{k}_n^q) \tag{3.27}$$

where $\hat{k}_n^q$ are the quadrature points, w_n^q are the weights corresponding to the quadrature points, K_q is the number of quadrature points at level q.

3.1.3.1. Upward Pass

While far interactions are considered, the first step is to calculate $u_{m_q}(\hat{k})$ at each quadrature point in each level. Outgoing multipole expansion in any level can be related to the expansion at a lower level as:

$$u_{m_q}(\hat{k}) = \sum_{i \in G_{m_q}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_q} i}} \mathbf{Y}^{\mathbf{t}}(\hat{k}) \cdot \mathbf{b}^i = \sum_{m_{q+1} \in C_{m_q}} \sum_{i \in G_{m_{q+1}}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_q} i}} \mathbf{Y}^{\mathbf{t}}(\hat{k}) \cdot \mathbf{b}^i$$

$$= \sum_{m_Q \in C_{m_q}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_q} m_{q+1}}} \sum_{i \in G_{m_{q+1}}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_{q+1}} i}} \mathbf{Y}^{\mathbf{t}}(\hat{k}) \cdot \mathbf{b}^i$$

$$= \sum_{m_{q+1} \in C_{m_q}} e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_q} \lambda_{m_{q+1}}}} u_{m_{q+1}}(\hat{k}). \tag{3.28}$$

Since outgoing multipole expansion at a lower level is translated to an expansion at a higher level, this step is called *upward pass.*

Because $u_{m_{q+1}}(\hat{k})$ is available at the quadrature points of level $q+1$, $u_{m_q}(\hat{k})$ is also available at these points by Eq. (3.28). To represent $u_{m_q}(\hat{k})$ at the quadrature points of level q, the value of $u_{m_{q+1}}(\hat{k})$ at quadrature points of level q must be expressed as a linear combination of the function values at the quadrature points of level $q+1$:

$$u_{m_{q+1}}(\hat{k}_{n'}^{q}) = \sum_{n=1}^{K_{q+1}} W_{n'n} u_{m_{q+1}}(\hat{k}_n^{q+1}). \tag{3.29}$$

It is obvious that Eq. (3.29) describes the general interpolation definition, where $W_{n'n}$ are the entries of the interpolation matrix.

Substituting Eq. (3.29) in Eq. (3.28), the latter becomes

$$u_{m_{q+1}}(\hat{k}_{n'}^{q}) = \sum_{m_{q+1} \in C_{m_q}} e^{-j\hat{k}_{n'}^{q}\cdot\vec{r}_{\lambda_{m_q} \lambda_{m_{q+1}}}} \sum_{n=1}^{K_{q+1}} W_{n'n} u_{m_{q+1}}(\hat{k}_n^{q+1}). \tag{3.30}$$

Upward pass is illustrated for a 4-level FMM in Fig.s 3.4-3.5.

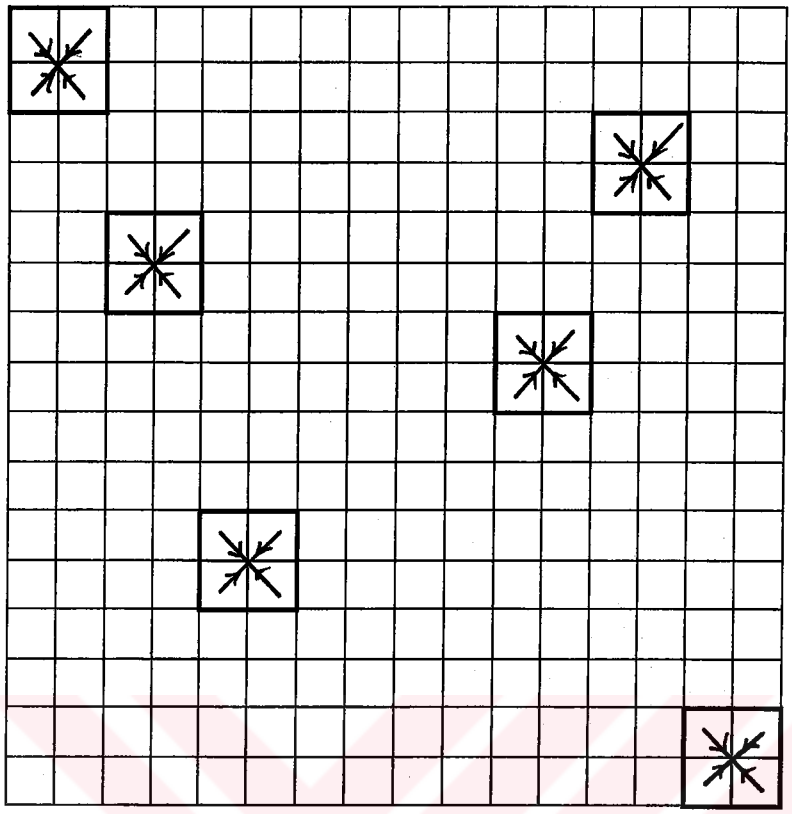

Figure 3.4. Upward pass at level 4 for a 4-level division

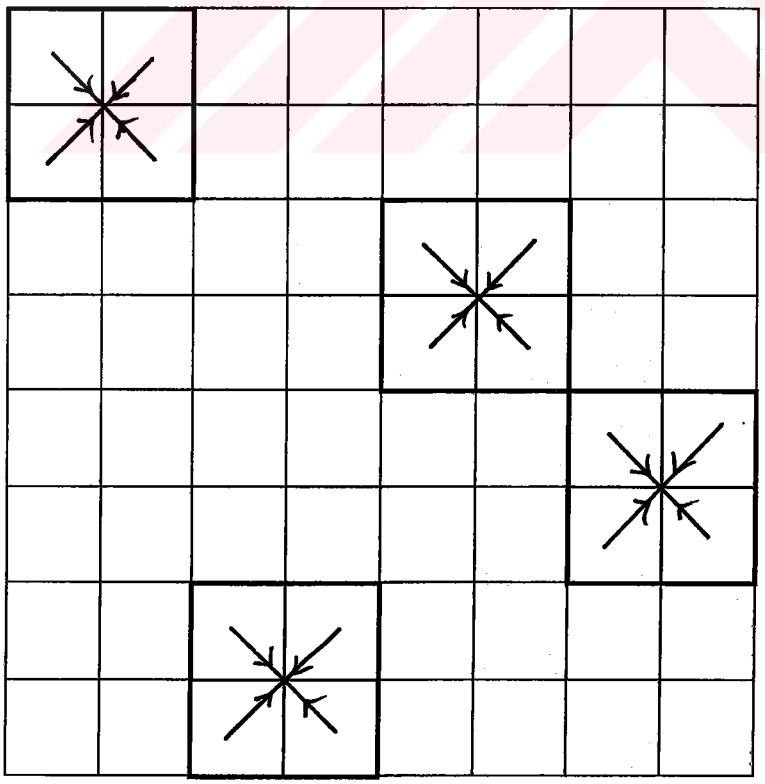

Figure 3.5. Upward pass at level 3 for a 4-level division

3.1.3.2. Downward Pass

The second step while calculating far interactions is to calculate $\vartheta_{m_q}(\hat{k})$ at each quadrature point in each level q, then to calculate the integral in Eq. (3.26).

The value of $\vartheta_{m_q}(\hat{k})$ at each quadrature point of level q can be written as

$$\vartheta_{m_q}(\hat{k}_n^q) = \sum_{m'_q \in I_{m_q}} \tilde{\alpha}_{N_\alpha(q),\lambda_{m_q m'_q}}(\hat{k}_n^q) u_{m'_q}(\hat{k}_n^q) \tag{3.31}$$

While calculating the integration in Eq. (3.26), first $\mathbf{t}_{m_{q-1}}(\hat{k})$ is related to a lower level as:

$$\begin{aligned} \mathbf{t}_{m_q}(\hat{k}) &= \mathbf{Y}^*(\hat{k}) e^{-j\vec{k}\cdot\vec{r}_{j\lambda_{m_q}}} = e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_{q+1}}\lambda_{m_q}}} \mathbf{Y}^*(\hat{k}) e^{-j\vec{k}\cdot\vec{r}_{j\lambda_{m_{q+1}}}} \\ &= e^{-j\vec{k}\cdot\vec{r}_{\lambda_{m_{q+1}}\lambda_{m_q}}} \mathbf{t}_{m_{q+1}}(\hat{k}). \end{aligned} \tag{3.32}$$

Using the same reasoning as in the representation of $u_{m_q}(\hat{k})$ at quadrature points of level q, Eq. (3.32) can be rewritten as

$$\mathbf{t}_{m_q}(\hat{k}_{n'}^q) = e^{-j\vec{k}_{n'}^q\cdot\vec{r}_{\lambda_{m_{q+1}}\lambda_{m_q}}} \sum_{n=1}^{K_q+1} W_{n'n} \mathbf{t}_{m_{q+1}}(\hat{k}_n^{q+1}) \tag{3.33}$$

When Eq. (3.33) is substituted in the integral representation in Eq. (3.27), the integral representation can be rewritten as:

$$\begin{aligned} \sum_{n'=1}^{K_q} w_{n'}^q t_{m_q}(\hat{k}_{n'}^q) \vartheta_{m_q}(\hat{k}_{n'}^q) &= \sum_{n'=1}^{K_q} w_{n'}^q \left(e^{-j\vec{k}_{n'}^q\cdot\vec{r}_{\lambda_{m_{q+1}}\lambda_{m_q}}} \sum_{n=1}^{K_{q+1}} W_{n'n} t_{m_{q+1}}(\hat{k}_n^{q+1}) \right) \vartheta_{m_q}(\hat{k}_{n'}^q) \\ &= \sum_{n=1}^{K_{q+1}} w_n^{q+1} t_{m_{q+1}}(\hat{k}_n^{q+1}) \left(\sum_{n'=1}^{K_q} \frac{w_{n'}^q}{w_n^{q+1}} W_{n'n} e^{j\vec{k}_{n'}^q\cdot\vec{r}_{\lambda_{m_{q+1}}\lambda_{m_q}}} \vartheta_{m_q}(\hat{k}_{n'}^q) \right). \end{aligned} \tag{3.34}$$

By Eq. (3.34), the integration in level q is translated to an integration in level $q+1$.

Denoting the second sum in Eq. (3.34) by $\tilde{\vartheta}_{m_q}\left(\hat{k}_{n'}^{q}\right)$ the integration in level $q+1$ becomes

$$\sum_{n=1}^{K_{q+1}} w_n^{q+1} t_{m_{q+1}}\left(\hat{k}_n^{q+1}\right)\left[\vartheta_{m_{q+1}}\left(\hat{k}_n^{q+1}\right)+\tilde{\vartheta}_{m_{q+1}}\left(\hat{k}_n^{q+1}\right)\right]. \tag{3.35}$$

The technique which transforms the integral in level q to an integral at level $q+1$ through interpolation of the kernel of the integral, $t_{m_q}\left(\hat{k}\right)$, then exchanging the order of integration and interpolation is defined as *anterpolation.* By the use of this technique, the integral in any level can be translated down to the lowest level, where it can be done with the least numerical complexity. Because the translation is in the downward direction, this step is called *downward pass.*

Downward passes for a 4-level FMM at levels 2 and 3 are illustrated in Fig.s 3.6-3.7.

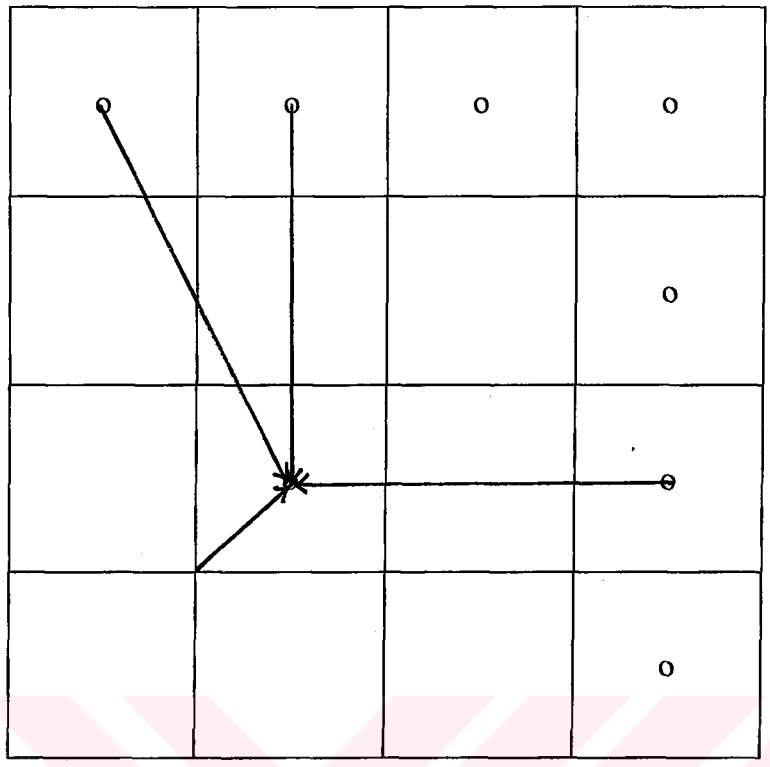

Figure 3.6 . Far interactions at level 2 for a 4-level division

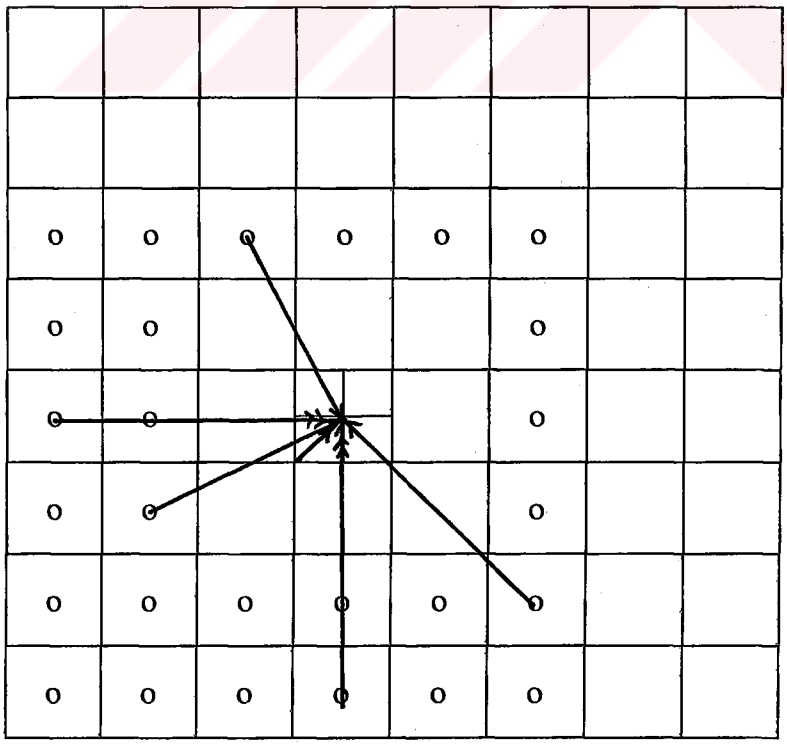

Figure 3.7. Far interactions at level 3 for a 4-level division

3.2. Interpolation in MLFMA

3.2.1. Definition of the Interpolation Problem

The interpolation problem in MLFMA can be defined as the calculation of outgoing multipole expansion through interpolation as explained in the upward pass in section 3.1.3.1 at each group center in each level with optimum error and numerical complexity. This problem is considered in two steps. The first step is to interpolate the outgoing multipole expansion in each group center in a level at quadrature points of its parent cell, and the second step is to translate this representation to the center of the parent cell, where the interpolated, then translated multipole expansions due to all children cells are summed [13]. For level q, the problem can be formulated as:

Step1:

$$u_{m_{q+1}}\left(\hat{k}_{n'}^{q}\right)=\sum_{n=1}^{K_{q+1}} W_{n'n} u_{m_{q+1}}\left(\hat{k}_{n}^{q+1}\right) \tag{3.36}$$

Step 2:

$$u_{m_q}\left(\hat{k}_{n'}^{q}\right)=\sum_{m_{q+1}\in C_{m_q}} u_{m_{q+1}}\left(\hat{k}_{n'}^{q}\right) e^{-j\vec{k}_{n'}^{q}\cdot\vec{r}_{\lambda_{m_q}\lambda_{m_{q+1}}}} \tag{3.37}$$

where $W_{n'n}$ are the entries of the interpolation matrix, and $\hat{k}_{n'}^{q}$ are the quadrature points of level q, which are defined on the unit sphere. This procedure starts from the lowest level Q where the outgoing multipole expansion in each cell is calculated directly as:

$$u_{m_Q}\left(\hat{k}_n^Q\right)=\sum_{i\in G_{m_Q}} e^{-j\bar{k}_n^Q\cdot\bar{r}_{\lambda_{m_Q}i}}\sum_{lm} Y_{lm}\left(\hat{k}_n^q\right)b_{lm}^i \tag{3.38}$$

where the spherical harmonics Y_{lm} are given by

$$Y_{lm}\left(\hat{k}_n^q\right)=(-1)^m\left[\frac{(l-m)!}{(l+m)!}\frac{2l+1}{4\pi}\right]^{1/2}P_l^m\left(\cos\theta_n^q\right)e^{jm\phi_n^q}. \tag{3.39}$$

The Legendre functions are polynomials of $\cos(\theta)$, therefore they are bandlimited. Since the Legendre functions are bandlimited, spherical harmonics in Eq. (3.39) are bandlimited, and u_{m_q}, which are linear combinations of spherical harmonics, are also bandlimited. This property is set as a constraint for the functions to be interpolated in uniform sampling theory which will be explained later in this chapter.

Because the sample points on the unit sphere are the Cartesian product of the samples in θ and ϕ, the first step of interpolation can be considered as the combination of two independent parts: interpolation in θ and interpolation in ϕ. Eq. (3.36) now becomes

$$u_{m_q}\left(\theta_{n'_\theta}^q,\phi_{n'_\phi}^q\right)=\sum_{n_\phi=1}^{2N_\theta^{q+1}}W_{n'_\phi n_\phi}\left(\theta_{n'_\theta}^q,\phi_{n'_\phi}^q;\theta_{n'_\theta}^q,\phi_{n_\phi}^{q+1}\right)\times\sum_{n_\theta=1}^{N_\theta^{q+1}}W_{n'_\theta n_\theta}\left(\theta_{n'_\theta}^q,\phi_{n_\phi}^{q+1};\theta_{n_\theta}^{q+1},\phi_{n_\phi}^{q+1}\right)u_{m_q}\left(\theta_{n_\theta}^{q+1},\phi_{n_\phi}^{q+1}\right) \tag{3.40}$$

The choice of quadrature points determines the accuracy of both interpolation and numerical quadrature. For accurate quadrature over the unit sphere $\hat{k}_n^q$ are chosen such that $\cos\left(\theta_n^q\right)$ are the $N_\theta(q)$ roots of Legendre polynomial, which are called the Gauss-Legendre points, and ϕ_n^q are the $2N_\theta(q)$ uniformly spaced points in $[0,2\pi)$. This choice of quadrature points describes the Gaussian quadrature. However, for accurate interpolation the quadrature points are preferred to be

uniformly spaced both in θ and ϕ, which implies a trade off between the accuracy of interpolation and the accuracy of quadrature.

It should be noted that the numerical complexity due to the interpolation scheme affects the numerical complexity of the MLFMA [13]. Hence, to obtain overall numerical complexity of at most $O(N_s \log N_s)$, which is sufficiently close to $O(N_s)$, the numerical load due to interpolation must also be at most $O(N_s \log N_s)$, where N_s is the number of scatterers.

Throughout this section general interpolation methods will be introduced, and the error and numerical complexity of each method will be discussed.

3.2.2. An Overview of Interpolation Methods

One of the well-known methods to reconstruct a bandlimited function from its regularly spaced samples is to use *Cardinal Series Expansion*, which is also known as the *Shannon Sampling Theorem*. This expansion is an infinite series expansion which must be truncated for numerical evaluation leading to a truncation error. Both the rate of convergence and the upper bound of the truncation error are measures of the accuracy of interpolation. To increase the rate of convergence of the truncation error, cardinal series expansion can be modified with a convergence factor, which yields *self-truncating series*, or *approximate prolate series*, which has an improved convergence factor obtained by using an approximation to spheroidal wave functions. Interpolation using the concept of *sampling window* is introduced in order to minimize the truncation error bound [11,12].

Another method of interpolation is to use polynomial approximation. The idea of polynomial approximation is to find the unique polynomial of degree n which assumes the given values at the sample points, where n is the number of sample points. Lagrangian approximation is the commonly used polynomial approximation and will be discussed here [15].

Following subsections summarize the interpolation methods mentioned above.

3.2.2.1. Cardinal Series Expansion

The sampling theorem can be stated as follows [16] :

Let $f(t)$ be a bandlimited function with $F(\omega) = 0$ for $|\omega| > \omega_M$. Then $f(t)$ is uniquely determined by its samples $f(k\tau_s)$, $k = 0,\pm 1,\pm 2, \ldots$ if $\omega_s > 2\omega_M$, where $\omega_s = \frac{2\pi}{\tau_s}$.

Given these samples, $f(t)$ can be reconstructed by generating a periodic impulse train in which successive impulses have amplitudes that are successive sample values. This impulse train is then processed through an ideal lowpass filter with gain τ_s and cutoff frequency greater than ω_M and less than $(\omega_s - \omega_M)$. The resulting output function will exactly equal $f(t)$.

The above statement can be formulated as

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s)\,\tau_s \frac{\omega_c}{\pi} \frac{\sin(\omega_c(t - k\tau_s))}{\omega_c(t - k\tau_s)} . \qquad (3.41)$$

For $\omega_c = \frac{\omega_s}{2}$, Eq. (3.41) can be rewrittten as

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s) \frac{\sin\left(\frac{\omega_s}{2}(t - k\tau_s)\right)}{\frac{\omega_s}{2}(t - k\tau_s)} \qquad (3.42)$$

$\frac{1}{\tau_0} = \frac{\omega_0}{2\pi} = \frac{\omega_M}{\pi}$ is the lowest sampling frequency for perfect reconstruction, and defined as the Nyquist sampling frequency. Sampling at frequency greater than the Nyquist frequency is called oversampling.

It is not possible in numerical practice to reconstruct $f(t)$ using Eq. (3.41) or (3.42) since an infinite number of samples would be needed. Thus, the expansion in Eq. (3.42) is truncated leading to the truncation error series [11, 12] :

$$e_N(t) = f(t) - \hat{f}_N(t) = \sum_{|k|>N} f(k\tau_s) \frac{\sin\left(\frac{\omega_s}{2}(t - k\tau_s)\right)}{\frac{\omega_s}{2}(t - k\tau_s)} \tag{3.43}$$

where

$$\hat{f}_N(t) = \sum_{|k|\le N} f(k\tau_s) \frac{\sin\left(\frac{\omega_s}{2}(t - k\tau_s)\right)}{\frac{\omega_s}{2}(t - k\tau_s)}. \tag{3.44}$$

The error bound for cardinal series with oversampling is given by [11]

$$|e_N(t)| \le \frac{M}{\pi N} \frac{1}{\cos\frac{\pi}{2}(1-\delta)} \tag{3.45}$$

where $\delta = 1 - \tau_s/\tau_0$ represents the amount of oversampling relative to the Nyquist sampling rate, and M is the bound of $f(t)$, i.e. $|f(t)| < M$. The case $\delta = 0$ corresponds to sampling at Nyquist rate, and as δ approaches unity, oversampling increases.

3.2.2.2. Convergence Factor

If $f(t)$ is a bandlimited function, δ is any real number between zero and unity, i.e., $0 < \delta < 1$, and $\tau_s = \frac{\pi(1-\delta)}{\omega_M}$, then

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s) \frac{\sin\left(\frac{\omega_M}{1-\delta}(t - k\tau_s)\right)}{\frac{\omega_M}{1-\delta}(t - k\tau_s)} \Psi(t - k\tau_s) \tag{3.46}$$

for any $\Psi(t)$ bandlimited to $\frac{\delta}{1-\delta}\omega_M$ with $\Psi(0)=1$.

The truncation error magnitude now becomes

$$|e_N(t)| = \left| \sum_{|k|>N} f(k\tau_s) \frac{\sin\left(\frac{\omega_M}{1-\delta}(t - k\tau_s)\right)}{\frac{\omega_M}{1-\delta}(t - k\tau_s)} \Psi(t - k\tau_s) \right|, \tag{3.47}$$

and $\Psi(t)$ is chosen such that the convergence of the error series is increased; that is, $e_N(t)$ is forced to approach zero for large $|t|$. Therefore, $\Psi(t)$ is called *convergence factor.* Self-truncating series and approximate prolate series are the results of two different choices for $\Psi(t)$ [11,12].

Here it should be noted that $0 < \delta < 1$ implies oversampling, that is use of a convergence factor is possible only in the case of oversampling.

3.2.2.3. Self-Truncating Series Expansion

One possible choice for $\Psi(t)$ as convergence factor is

$$\Psi(t) = \left[\frac{\sin\left(\frac{\delta\omega_M}{(1-\delta)m} t\right)}{\frac{\delta\omega_M}{(1-\delta)m} t} \right]^m \tag{3.48}$$

where m is a positive integer. $\Psi(t)$ in Eq. (3.48) is $O(t^{-m})$ as $t \to \infty$, which is a measure of how fast it approaches zero. The sampling expansion that uses $\Psi(t)$ in Eq. (3.48) as convergence factor is called self-truncating series, and is expressed as

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s) \frac{\sin\left(\frac{\omega_M}{1-\delta}(t-k\tau_s)\right)}{\frac{\omega_M}{1-\delta}(t-k\tau_s)} \left[\frac{\sin\frac{\delta\omega_M}{(1-\delta)m}(t-k\tau_s)}{\frac{\delta\omega_M}{(1-\delta)m}(t-k\tau_s)}\right]^m . \quad (3.49)$$

The error bound for the self-truncating series is given by [11]

$$e_N(t) \le \frac{2M}{m\pi}\left[\frac{m}{c}\right]^m \quad (3.50)$$

where

$$c = \pi N\delta \quad (3.51)$$

and M is chosen such that $|f(t)| < M$. This class of functions yields smaller truncation error bounds than cardinal series when m is properly chosen. The approximate minimum error is obtained by letting $m = c/e$ in Eq. (3.50) [11] :

$$e_N(t) \le \frac{2eM}{c\pi} e^{-0.3679c}. \quad (3.52)$$

3.2.2.4. Approximate Prolate Series Expansion

In Eq. (3.46), $\Psi(t)$ is subject only to the constraints that $\Psi(t)$ be bandlimited to $\frac{\delta\omega_M}{1-\delta}$, and $\Psi(0)=1$. A desirable property for $\Psi(t)$ is that it be as small as possible for $|t| > N\tau_s$ besides its convergence to zero for large t. The problem to be solved is to find $\Psi(t)$ which has the largest fraction of its energy in the interval $[-T,T]$, where $T = N\tau_s$, [11].

An approximate solution to this problem as given by [11] is

$$\Psi(t) = \frac{\sin\left[c\sqrt{(t/N\tau_s)^2 - 1}\right]}{(\sinh c)\sqrt{(t/N\tau_s)^2 - 1}} \tag{3.53}$$

where c is as in Eq. (3.51). $\Psi(t)$ in Eq. (3.53) is referred to as the approximate prolate (AP) convergence factor, and is $O(t^{-1})$. The new sampling expansion using AP convergence factor is called the AP series and is expressed as [11]

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s) \frac{\sin\left(\frac{\omega_M}{1-\delta}(t - k\tau_s)\right)}{\frac{\omega_M}{1-\delta}(t - k\tau_s)} \frac{\sin\left[c\sqrt{\left(\frac{t - k\tau_s}{N\tau_s}\right)^2 - 1}\right]}{(\sinh c)\sqrt{\left(\frac{t - k\tau_s}{N\tau_s}\right)^2 - 1}}. \tag{3.54}$$

By choosing N in Eq. (3.54) as the truncation limit, truncation error is improved.

Truncation of $f(t)$ by N leads to $e_N(t)$, for which the bound for central interpolation is given by

$$|e_N(t)| \le \frac{M}{\sinh c} |\sin \pi t/\tau_s| \le \frac{M}{\sinh c} \tag{3.55}$$

where c and M are as defined in the self-truncating series case [11,12].

It has been shown in [11] that AP series error bound is smaller than the error bounds of both the cardinal series and the self-truncating series .

3.2.2.5. Interpolation by Sampling Window

The idea of interpolation by sampling window is to choose $\Psi(t)$ such that it minimizes the bound of Eq. (3.46). $\Psi(t)$ which satisfies this condition is given by [12]

$$\Psi(t) = \frac{\cos\left(\pi N\delta\sqrt{(t/N\tau_s)^2 - 1}\right)}{\cosh(\pi N\delta)}. \tag{3.56}$$

It should be noted that $\Psi(t) \to \frac{\cos(\delta\omega_M)}{\cosh(\pi N\delta)}$ as $t \to \infty$ and is $O(t)$. Hence, it will not force the error series converge to zero for large t as opposed to the convergence factors of self-truncating series and approximate prolate series.

The sampling expansion with the sampling window becomes [12]

$$f(t) = \sum_{k=-\infty}^{\infty} f(k\tau_s) \frac{\sin\left(\frac{\omega_M}{1-\delta}(t - k\tau_s)\right)}{\frac{\omega_M}{1-\delta}(t - k\tau_s)} \frac{\cos\left(c\sqrt{(t/N\tau_s)^2 - 1}\right)}{\cosh c}. \tag{3.57}$$

The bound of $e_N(t)$ for central interpolation is given by [12]

$$|e_N(t)| \le \frac{\sqrt{2}}{\pi} \frac{M \sin(\pi|t/\tau_s|)}{\sqrt{1-\delta}\cosh c} \left(N - \frac{(t/\tau_s)^2}{N}\right)^{-1/2} \tag{3.58}$$

where M and c are as in the self-truncating series.

The error bounds in Eq.s (3.45), (3.52) and (3.55) are valid for finite-power signals; however, Eq. (3.58) is obtained for finite-energy signals. Associated Legendre functions are finite-power signals, hence the error bound in Eq. (3.58) cannot be applied to our case.

3.2.2.6. Lagrangian Interpolation

A standard idea in interpolation is to find a polynomial $p_n(t)$ of degree n (or less) that assumes the given values :

$$p_n(t_0) = f_0,\ p_n(t_1) = f_1\ , \ldots, p_n(t_n) = f_n \tag{3.59}$$

$p_n(t)$in Eq. (3.59) is called the interpolation polynomial and $t_0,\ldots,t_n$ are called nodes. If $f(t)$ is a mathematical function, p_n is defined to be the polynomial approximation of f. p_n satisfying Eq. (3.59) for given data exists and is unique.

Lagrangian interpolation is the commonly used polynomial interpolation. The idea of Lagrange interpolation is as follows: given $(t_0,f_0),\ldots,(t_n,f_n)$ with arbitrarily spaced t_j, the unique interpolation polynomial of degree n or less can be obtained by multiplying each f_j by a polynomial that is 1 at t_j and 0 at all the other n nodes, then by adding these $n+1$ polynomials. This idea can be formulated for general n as [15]

$$f(t)\approx p_n(t)=\sum_{k=0}^{n}L_k(t)f_k=\sum_{k=0}^{n}\frac{l_k(t)}{l_k(t_k)}f_k \tag{3.60}$$

where

$L_k(t_k)=1$ and 0 at the other nodes,

$$l_k(t)=\prod_{\substack{m=0\\ m\neq k}}^{n}(t-t_m),\qquad k=0,1,\ldots,n. \tag{3.61}$$

The error estimate of Lagrange interpolation is given by [15]

$$e_n(t)=f(t)-p_n(t)=(t-t_0)(t-t_1)\cdots(t-t_n)\frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{3.62}$$

where ξ is properly chosen such that $t_0\le\xi\le t_n$. From Eq. (3.62) one can conclude that if f is a polynomial of degree n or less, the error is zero. In fact, it is clear that $n+1$ data uniquely determine a polynomial of degree n or less .

For the case of trigonometric polynomials, the following estimate for error is useful because ξ which is not known in general, does not exist in this expression [14]:

$$e_n(x) \leq M \frac{(2p)!}{(p!)^2} \left(\frac{Nh}{4}\right)^{2p}. \tag{3.63}$$

Here, M is the bound of $f(x)$, i.e. $|f(x)| < M$, $n = 2p$ is the number of samples used in interpolation, $h \overset{\Delta}{=} \max_i \{x_i - x_{i-1}\}$, $i = 2,\ldots,2p$, and N is the degree of the trigonometric polynomial.

3.2.3. Numerical Results and Discussion

3.2.3.1. Simplification of the Interpolation Problem

To compare the different interpolation methods mentioned, several simplifications can be done without loss of generality:

1) The contribution of only a single scatterer, which belongs to any group in the lowest level, to the multipole expansion in each level is considered. This helps one discard the summations in Eq.s (3.37) and (3.38).

2) The multipole expansion can be interpolated in θ and ϕ seperately as expressed in Eq. (3.40), which means that interpolation in θ can be done by fixing ϕ, and interpolation in ϕ can be done by fixing θ. This observation also clears the fact that methods used for the interpolation in θ and in ϕ need not be identical.

3) The multipole expansion at any quadrature point is a linear combination of spherical harmonics. Therefore, interpolation of a multipole expansion can be simplified to interpolation of a spherical harmonic of any degree and order, which exists in the linear combination.

4) Considering 2 and 3 together, one can conclude that fixing ϕ interpolation in θ can be simplified to interpolation of $P_n^m(\cos\theta)$ for any degree n and order m in the linear combination, and fixing θ interpolation in ϕ can be simplified to interpolation of $e^{jm\phi}$ for any order m.

3.2.3.2. Numerical Results

Considering the simplified problem, the first part of the numerical results is obtained from the interpolation of the Associated Legendre functions of the largest possible degree and any order in each level by cardinal series, self-truncating series, approximate prolate series, and sampling window. It can be remembered from the previous sections that these interpolation methods are valid for regularly spaced data. It can also be remembered that the other constraint for these methods to be applicable is bandlimitedness. It is clear that the Associated Legendre functions are bandlimited because they are polynomials of cosines.

The sample points in θ lie in $[0,\pi]$ because of the spherical coordinates. The constraint that $P_n^m(\cos\theta)$ be bandlimited implies that there should be no discontinuities when the samples are extended to infinity with period π. However, $P_n^m(\cos\theta)$ becomes discontinuous for n odd when $m=0$. To avoid this phenomena sampled data used in interpolation should be folded around π to cover the interval $[0,2\pi]$.

Interpolation at Nyquist rate implies that perfect reconstruction of $P_n^m(\cos\theta)$ is possible having at least $2n+1$ regularly spaced points in one period of the function, that is in $[0,2\pi]$. It is easily seen that the discrete Fourier transform of $P_n^m(\cos\theta)$ is bandlimited to n.

Interpolation at a higher rate implies oversampling. The oversampling rate is determined by $\delta = 1 - \frac{h}{\tau}$. $\delta = 0$ implies sampling at Nyquist rate, and as δ approaches unity, oversampling rate increases. By simple algebra the number of regular samples needed in $[0,2\pi]$ for the oversampled case is obtained as $\frac{2n}{1-\delta}+1$.

One can conclude that the sampled data in each level be $\frac{n}{1-\delta}+1$ regularly spaced points in $[0,\pi]$. It must be noted that these data are sample points of the higher level, and interpolation points of the lower level. Here n can take any value from 0 to the highest degree of Legendre function possible for the level considered. Knowing that harmonic content in level q is [13]

$$N_\alpha(q) = \lfloor kD_q + C_a \log(kD_q + \pi) \rfloor + 1, \tag{3.64}$$

and

$$N_\alpha(q) > 2n, \tag{3.65}$$

then $n = \left\{0,\ldots,\left\lfloor \frac{N_\alpha(q)-1}{2} \right\rfloor\right\}$. In these expressions D_q is the diameter of the cell considered in level q, and C_a is a factor to control the accuracy.

It is clear that for exact interpolation infinite number of samples is needed; which means that the samples in $[0,2\pi]$ must be extended to infinity with period 2π around the interpolation point. However, for numerical evaluation the number of samples must be finite, which introduces truncation error. In the following analysis the truncation limit around each interpolation point is

$$N = \frac{n}{1-\delta} \tag{3.66}$$

Fig. 3.8 shows interpolation from a level with $d = \lambda/2$, where $N_\alpha = 6$, $n = 2$, $N = 5$, to a level with $d = \lambda$, where $N_\alpha = 10$, $n = 4$, $N = 9$. Because $2n+1$ samples are needed around the interpolation point, the samples in $[0,\pi]$ are also folded around 0.

It is clear from Fig. 3.8 that the midpoint of the samples is the point that lies nearest to the interpolation point. That is why interpolation here is considered as central interpolation, which imposes the constraint that the distance between the interpolation point and the midpoint of samples to be less than or equal to the half space between the samples. Hence, the error bounds in Eq.s (3.45), (3.52), and (3.55) are valid.

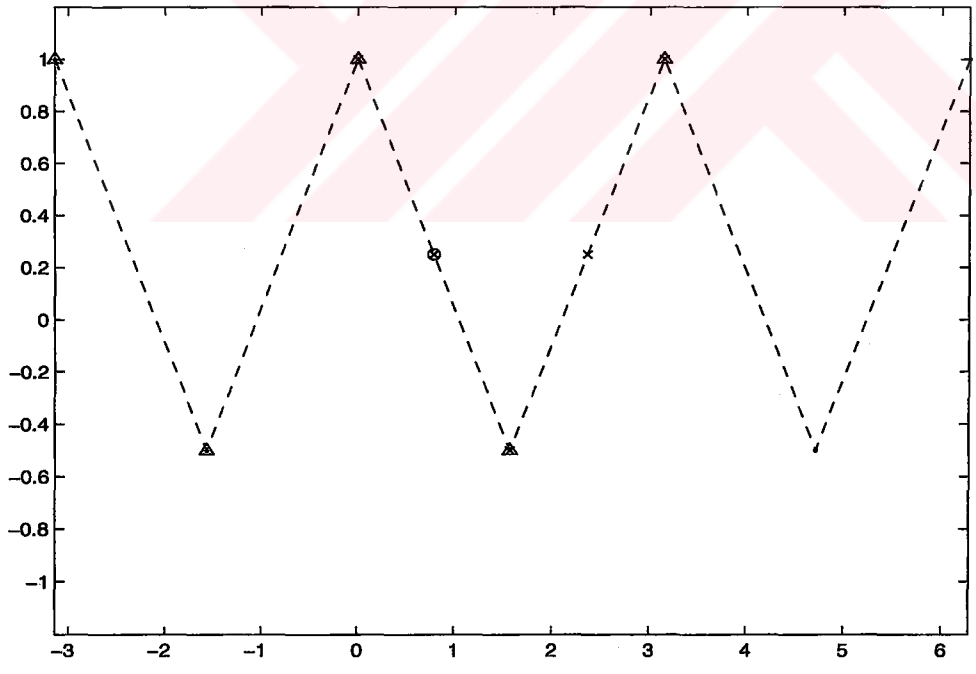

Figure 3.8. $P_2(\cos\theta)$ vs. θ.

Interpolation of $P_2(\cos\theta)$; • : sample points, × : interpolation points, o : point to be interpolated, Δ : sample points used in interpolation of the point o.

In Fig.s 3.9-12 interpolated $P_2(\cos\theta)$ by cardinal series, self-truncating series, AP series, and sampling window are plotted together for $\delta = 0, 0.25, 0.5, 0.75$, respectively. It is seen that $\delta = 0.5$ provides enough oversampling for interpolation by the self-truncating series, AP series, and sampling window to coincide with the exact values. For the cardinal series expansion $\delta = 0.75$ is needed.

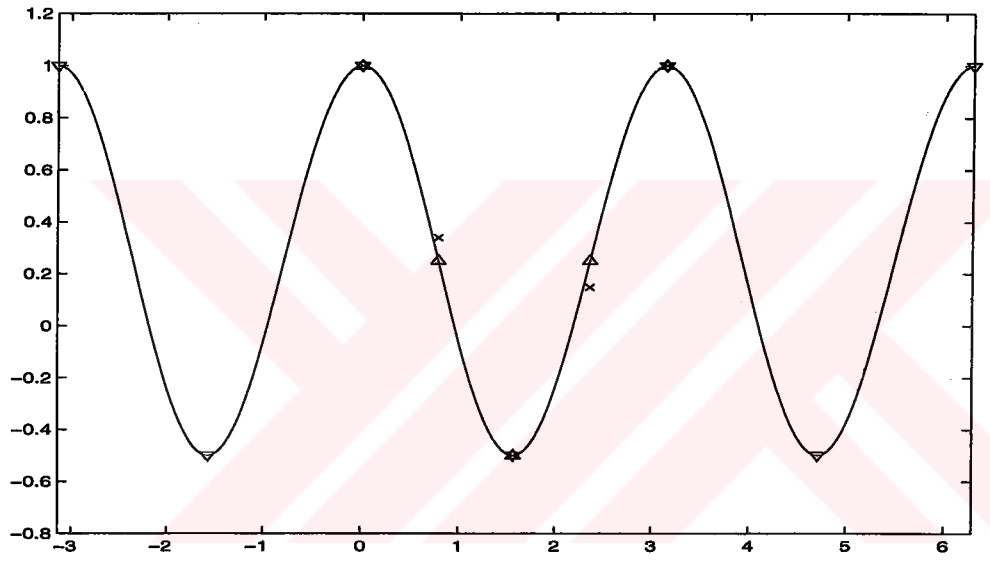

Figure 3.9. $\mathbf{P_2(\cos\theta)}$ **vs.** $\boldsymbol{\theta}$.
Interpolated data when $\delta = 0$; x : by cardinal series, ∇ : sampled data, Δ : exact data at the interpolation points.

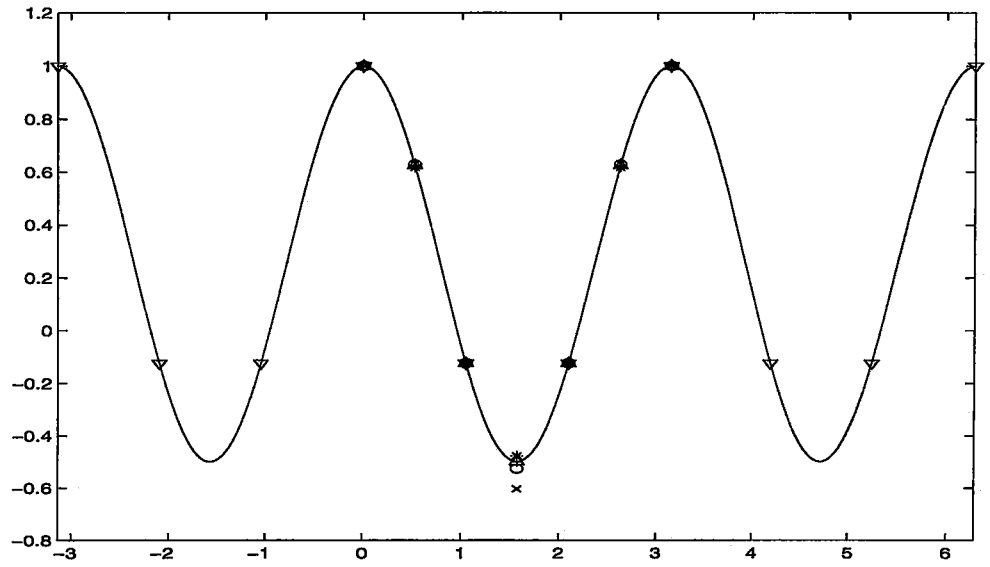

Figure 3.10. $\mathbf{P_2(\cos\theta)}$ **vs.** $\boldsymbol{\theta}$.

Interpolated data when $\delta = 0.25$; x : by cardinal series, o : by AP series, + : by self-truncating series, * : by sampling window, ∇ : sampled data, Δ : exact data at the interpolation points.

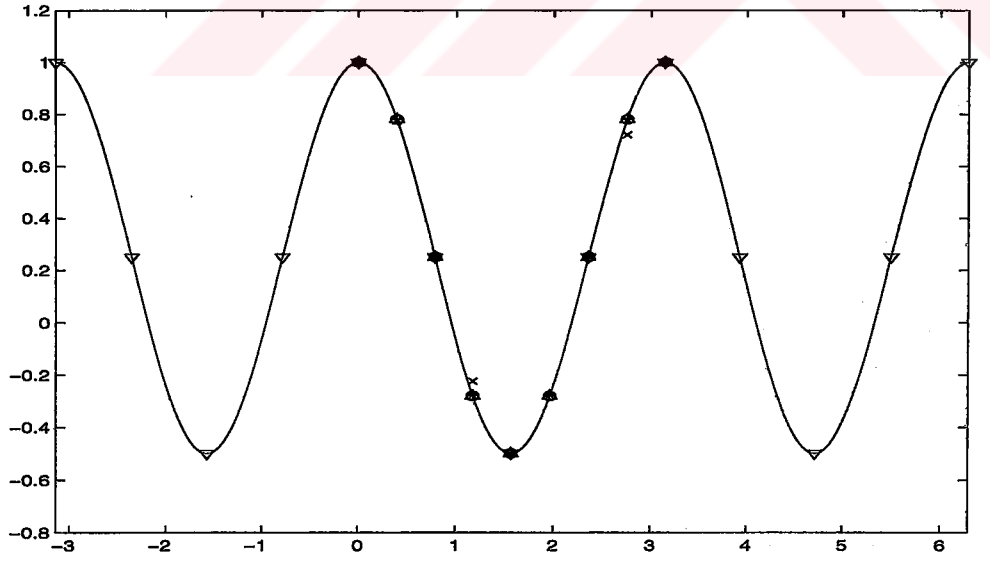

Figure 3.11. $\mathbf{P_2(\cos\theta)}$ **vs.** $\boldsymbol{\theta}$.

Interpolated data when $\delta = 0.5$; x : by cardinal series, o : by AP series, + : by self-truncating series, * : by sampling window, ∇ : sampled data, Δ : exact data at the interpolation points.

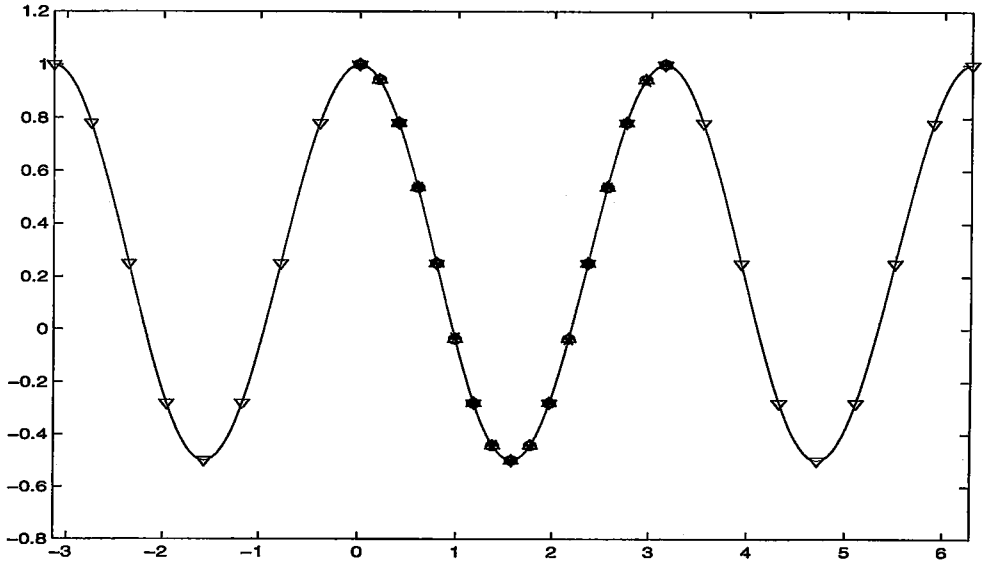

Figure 3.12. $\mathbf{P_2(\cos\theta)}$ **vs.** $\boldsymbol{\theta}$.
Interpolated data when $\delta = 0.75$; x : by cardinal series, o: by AP series, + : by self- truncating series, * : by sampling window, ∇ : sampled data, Δ : exact data at the interpolation points.

The best way to compare the accuracy of the interpolation results is to compare the error from each method. The error comparison is done for a 7-level FMM starting from $d_7 = \lambda/4$, and $d_q = d_7 \cdot 2^{7-q}$ for level q. Error is calculated for the highest degree polynomial in each level. N_α in Eq. (3.64), n in Eq. (3.65), and N in Eq. (3.66) for this problem are tabulated in Table 3.1.

Table 3.1. N_α , n and N in each level for the 7-level FMM.

	N_α	n	N			
			$\delta = 0$	$\delta = 0.25$	$\delta = 0.5$	$\delta = 0.75$
$d = \lambda/4$	4	1	1	2	2	4
$d = \lambda/2$	5	2	2	3	4	8
$d = \lambda$	9	4	4	6	8	16
$d = 2\lambda$	16	7	7	10	14	28
$d = 4\lambda$	29	14	14	19	28	56
$d = 8\lambda$	55	27	27	36	54	108
$d = 16\lambda$	106	52	52	70	104	208

The interpolation error in the 7-level FMM problem due to the cardinal series, self-truncating series, AP series, and the sampling window are plotted with the error bounds in Fig.s 3.13-16 for $\delta = 0, 0.25, 0.5, 0.75$, respectively. From the plots it can be seen that the AP series is the most accurate one as expected, and the errors due to all methods are below their bounds. It must be noted that for the error bounds smaller than the floating point error, which is 2.2204×10^{-16} in MATLAB, the error stays around the floating point error.

It is seen from the figures that as δ increases, number of samples used in interpolation increases, which results in higher accuracy in interpolation, and later in

quadrature. However, the more accurate results due to oversampling are not only due to the increased number of samples in total, but also due to the increased number of samples in the local neighborhood of the interpolation points.

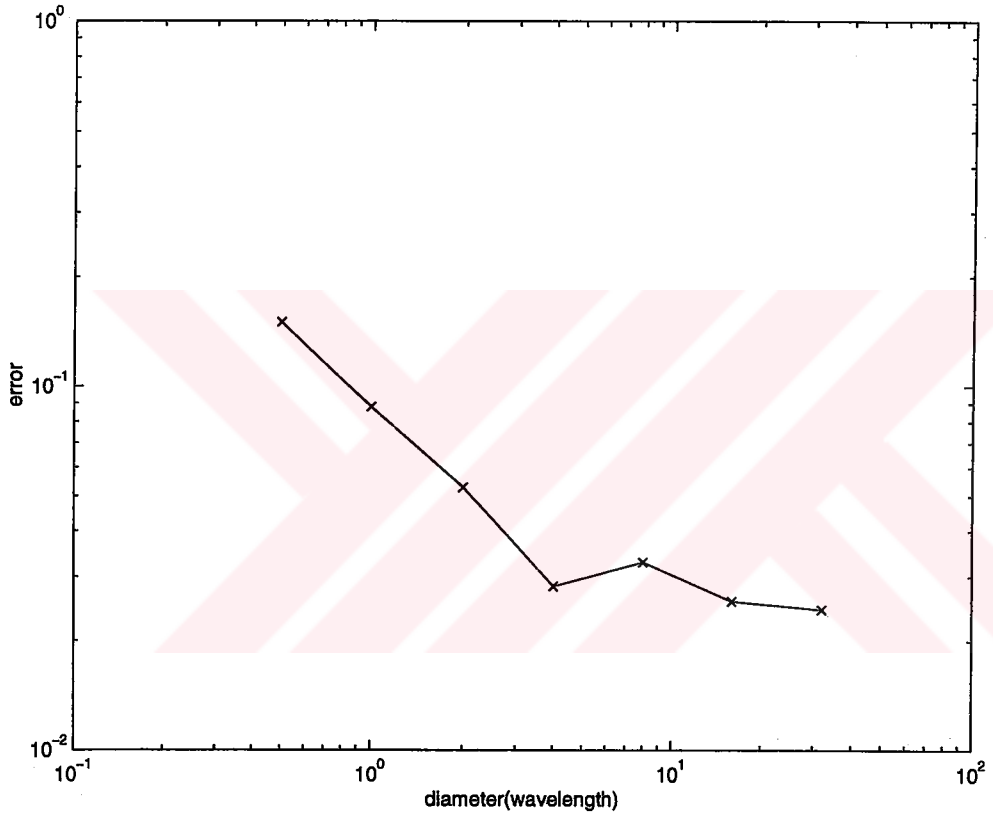

Figure 3.13. Error in interpolation when $\delta = 0.x$: by cardinal series.

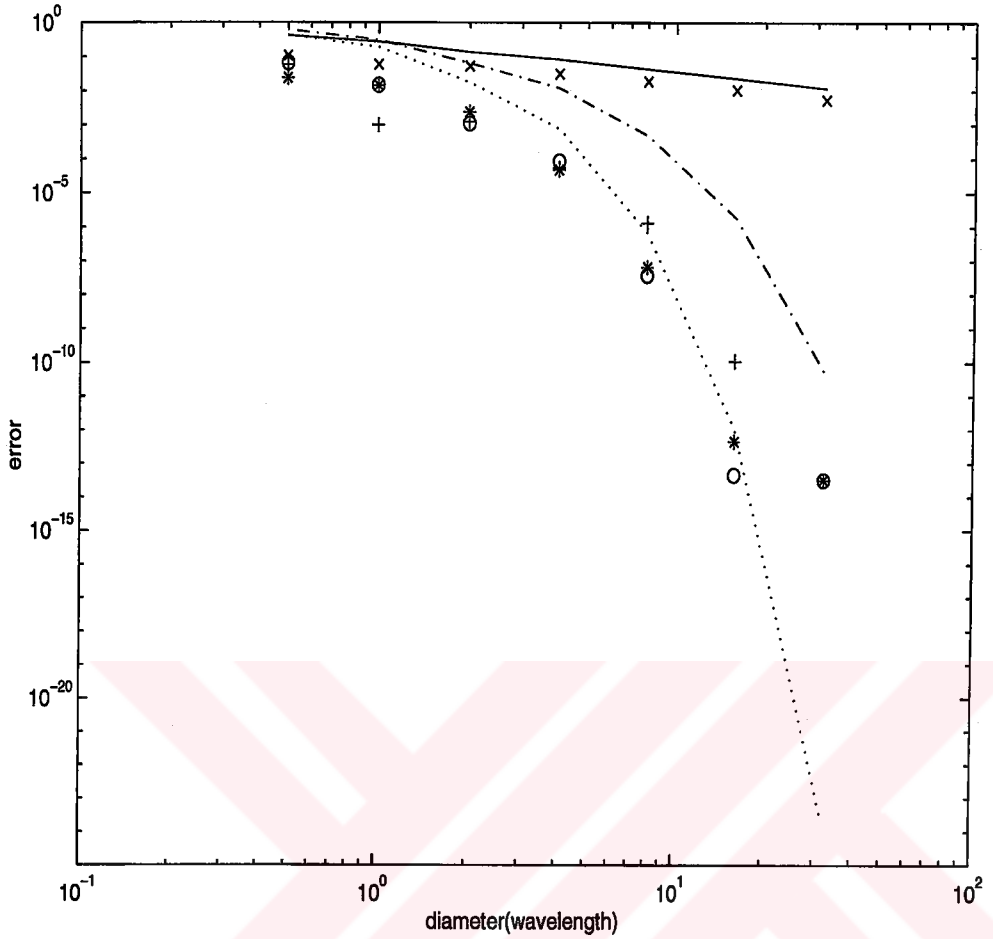

Figure 3.14. Error in interpolation when $\delta = 0.25$. x : by cardinal series, + : by self-truncating series, o : by AP series, * : by sampling window. Error bounds are also shown; solid : by cardinal series, dash-dotted : by self-truncating series, dotted : by AP series.

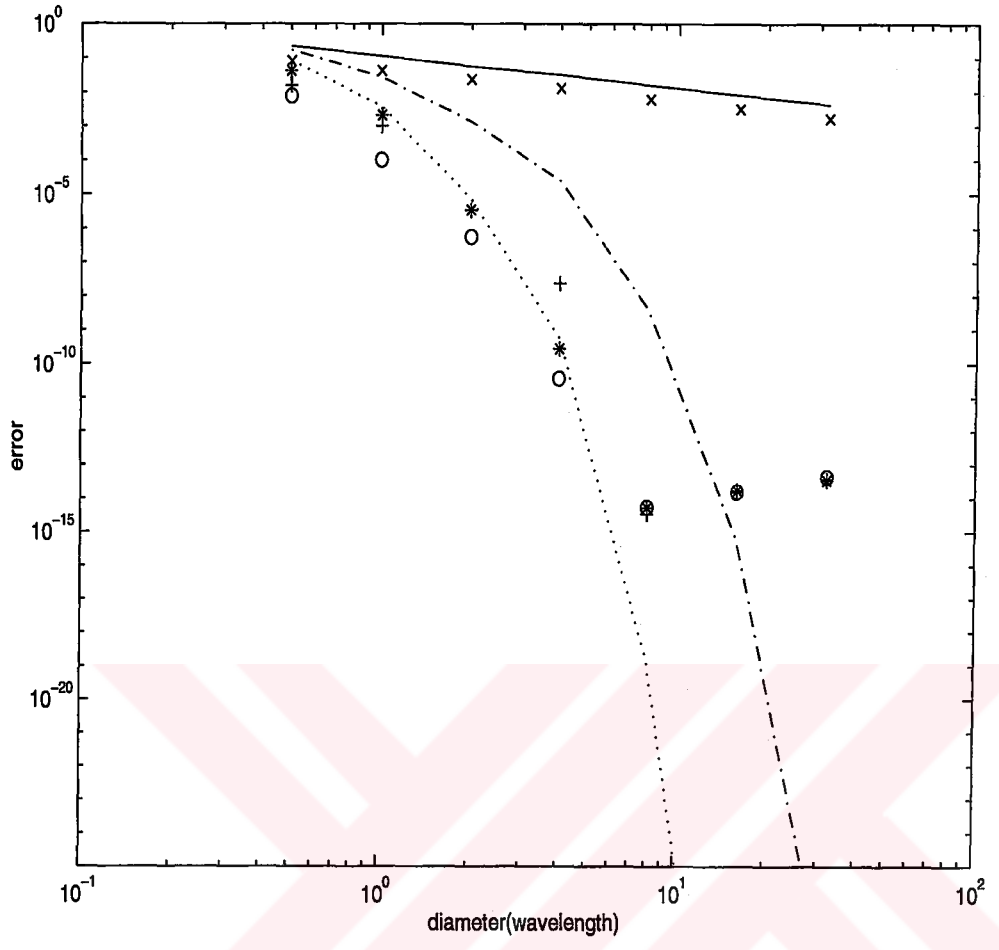

Figure 3.15. Error in interpolation when $\delta = 0.5$. x : cardinal series, + : self-truncating series, o : AP series, * : sampling window. Error bounds are also shown; solid : by cardinal series, dash-dotted : by self-truncating series, dotted : by AP series.

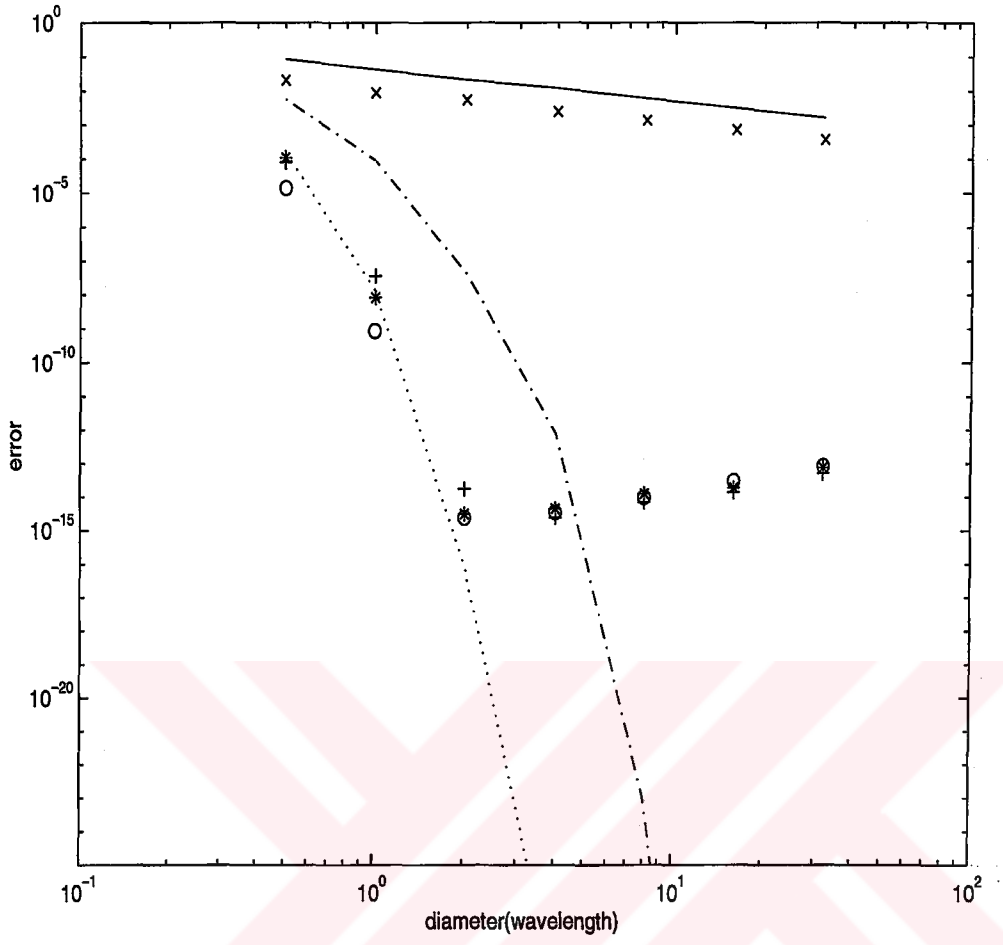

Figure 3.16. Error in interpolation when $\delta = 0.75$. x : cardinal series, + : self-truncating series., o : AP series, * : sampling window. Error bounds are also shown ; solid : by cardinal series, dash-dotted : by self-truncating series, dotted : by AP series.

One can easily conclude that the increased number of samples due to oversampling increases the numerical load which is illustrated in Fig. 3.17. Increased number of samples in the local neighborhood of the interpolation points due to oversampling gives the chance to optimize the numerical load and accuracy by introducing local interpolation. Local interpolation suggests to use the samples in the local neighborhood of the interpolation points, while in global interpolation all

samples in the sampling set are used. In local interpolation the number of samples in the local neighborhood is determined according to the accuracy desired. For error to be less than 10^{-3} when $\delta = 0.75$, the number of samples in the neighborhood is chosen to be 7 for the self-truncating series, AP series, and the sampling window. Error and numerical load due to these three methods in this case are shown in Fig.s 3.18-19. In Fig. 3.19 the numerical load due to the global interpolation is also plotted. When one compares the slopes of the lines due the local and global interpolation which are close to 1 and 2 respectively, one can immediately conclude that the numerical load due to the local interpolation is $O(N)$, and due to the global interpolation is $O(N^2)$.

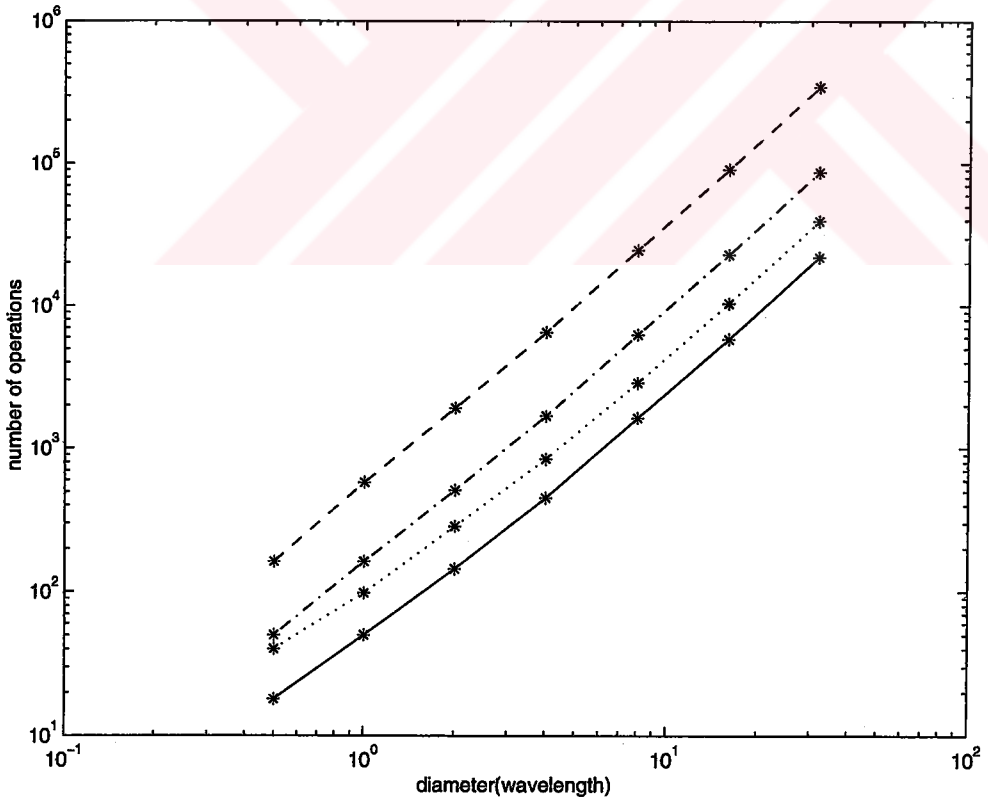

Figure 3.17. Numerical load when $\delta = 0, 0.25, 0.5, 0.75$ by solid , dotted, dash-dotted, dashed lines respectively .

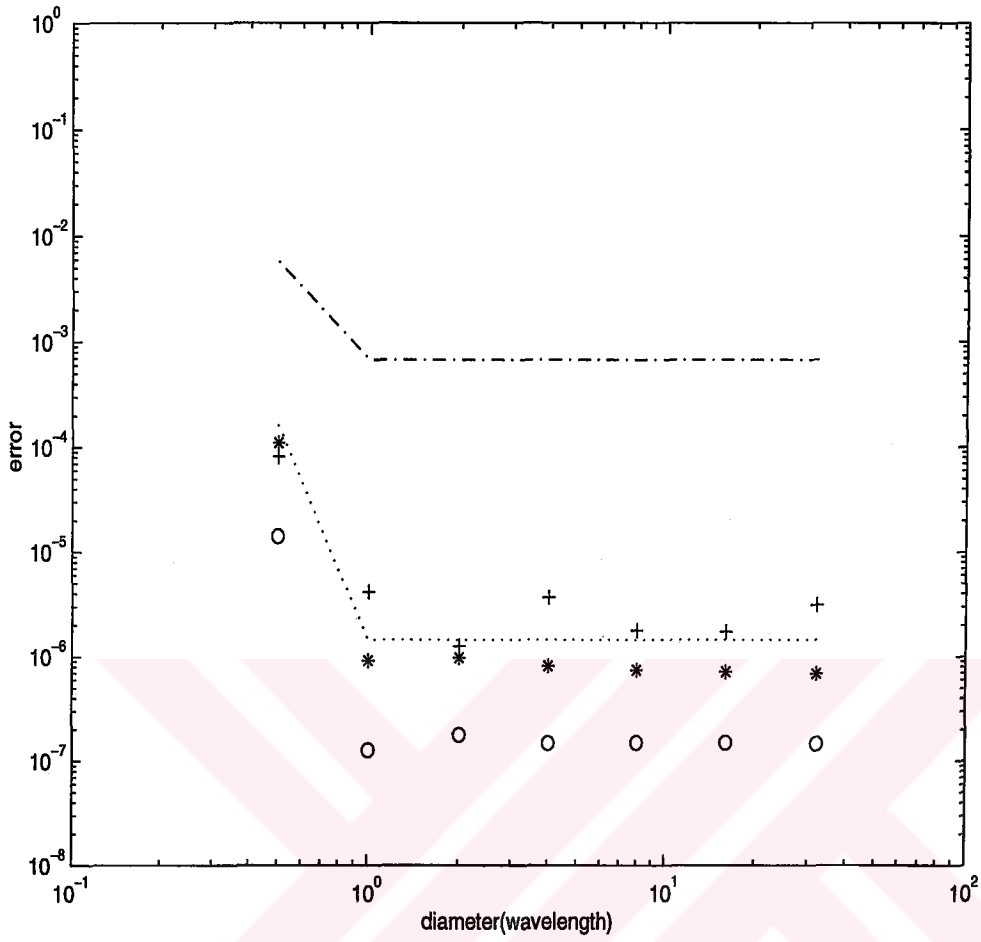

Figure 3.18. Error in local interpolation when $\delta = 0.75$. + : self-truncating series, o : AP series, * : sampling window. Error bounds are also shown; dash-dotted : by self-truncating series, dotted : by approximate prolate series.

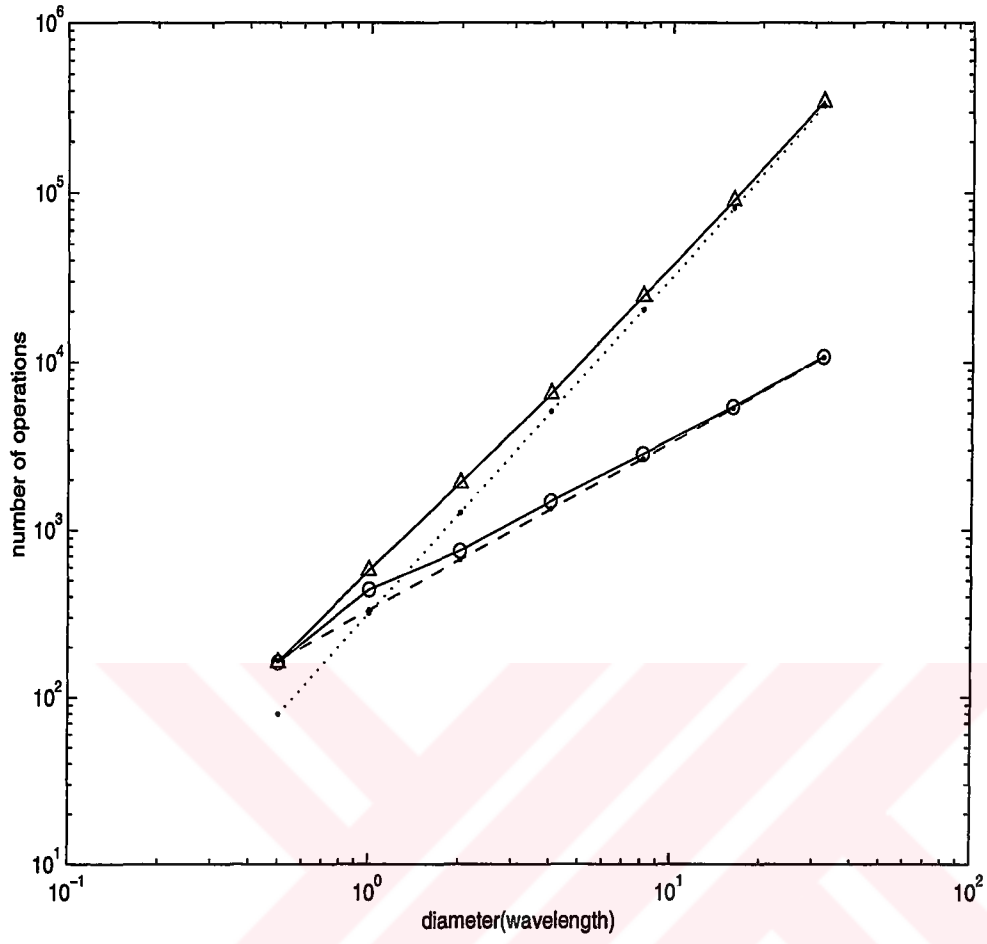

Figure 3.19 . Numerical load in local interpolation when $\delta = 0.75$: o in comparison with the numerical load in global interpolation : Δ. $O(N)$ curve : dashed, $O(N^2)$ curve : dotted.

In the second part of the analysis, $P_n^m(\cos\theta)$ is interpolated from the nonuniformly spaced Gauss-Legendre points. The results are obtained considering the 7-level FMM problem defined in the first part. The number of sample points N_θ which are related to N_α in Eq. (3.64) as

$$N_\theta = N_\alpha + 1, \tag{3.67}$$

and n in Eq. (3.66) are tabulated in Table 3.2. Here it must be remembered that the interpolation points in level q are the sample points in level $q-1$.

Table 3.2. N_α, N_θ, n for the Gauss-Legendre points in each level in 7-level FMM

	N_α	N_θ	n
$d = \lambda/4$	4	5	1
$d = \lambda/2$	5	6	2
$d = \lambda$	9	10	4
$d = 2\lambda$	16	17	7
$d = 4\lambda$	29	30	14
$d = 8\lambda$	55	56	27
$d = 16\lambda$	106	107	52
$d = 32\lambda$	207	208	103

Although the methods mentioned in the first part are defined for the uniformly spaced samples, one can consider the Gauss-Legendre points in $[0,\pi]$ almost uniform, and develop a heuristic approach. If the mean difference between the Gauss-Legendre points is calculated, and the sampling set is extended to $[-\pi,2\pi]$ keeping the same difference between the samples, the results illustrated in Fig.s 3.20-22 are obtained.

It is seen from Fig. (3.20) that the interpolated data due to the self-truncating series, AP series and the sampling window are closer to the exact results than the interpolated data due to the cardinal series. This observation is supported in Fig. (3.21) where the interpolation error is plotted versus the cell size. The interpolation error due to the cardinal series decreases by an order of magnitude as the cell size increases while the error due to the other methods remain almost constant for all cell sizes. Although $\delta = 0.5$ in this case because $2N_\theta(q)$ points constitute the sampling set at each level q, the error is in the order of 10^{-3}. Comparison of these results with the ones obtained in the first part shows the effect of the use of nonuniformly spaced samples on accuracy. The numerical loads due to these methods are equal, and they are $O(N_\theta^2(q))$ as shown in Fig. 22.

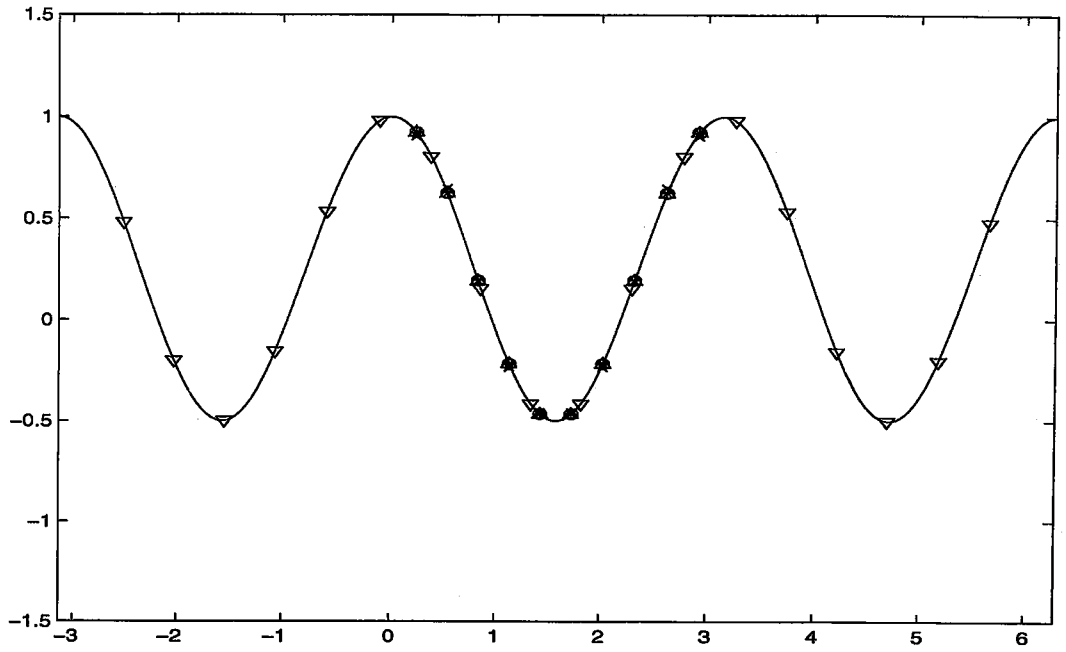

Figure 3.20. $P_2(\cos\theta)$ **vs.** θ.
Interpolated data from the nonuniformly spaced samples when $\delta = 0.75$. x : by cardinal series, o: by AP series, + : by self-truncating series, * : by sampling window, ∇ : sampled data, Δ : exact data at the interpolation points.

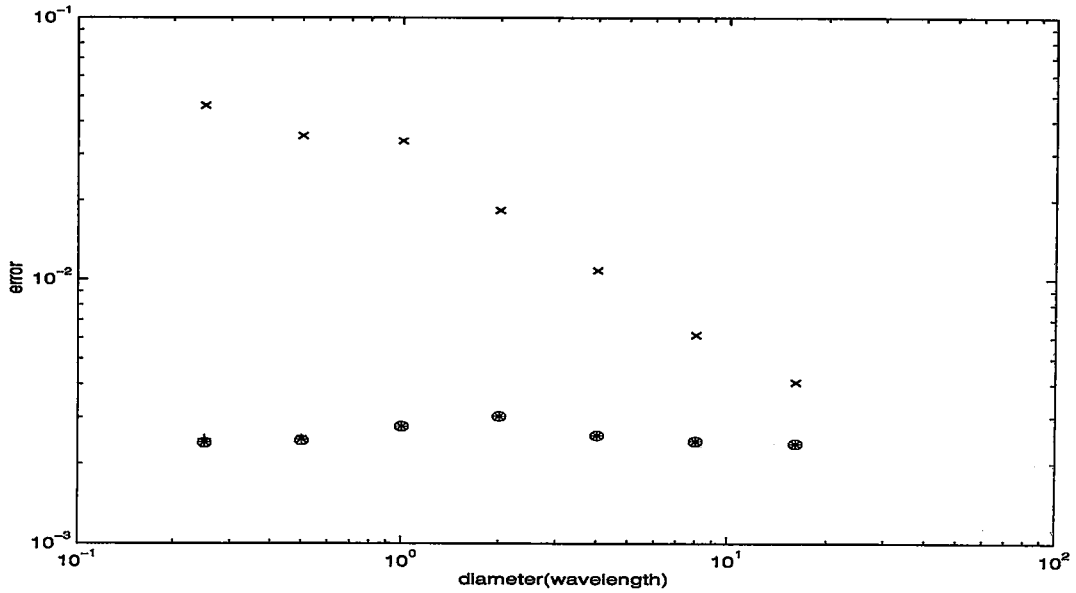

Figure 3.21. Error in interpolation from the nonuniformly space samples. + : by self-truncating series, o : by AP series, * : by sampling window.

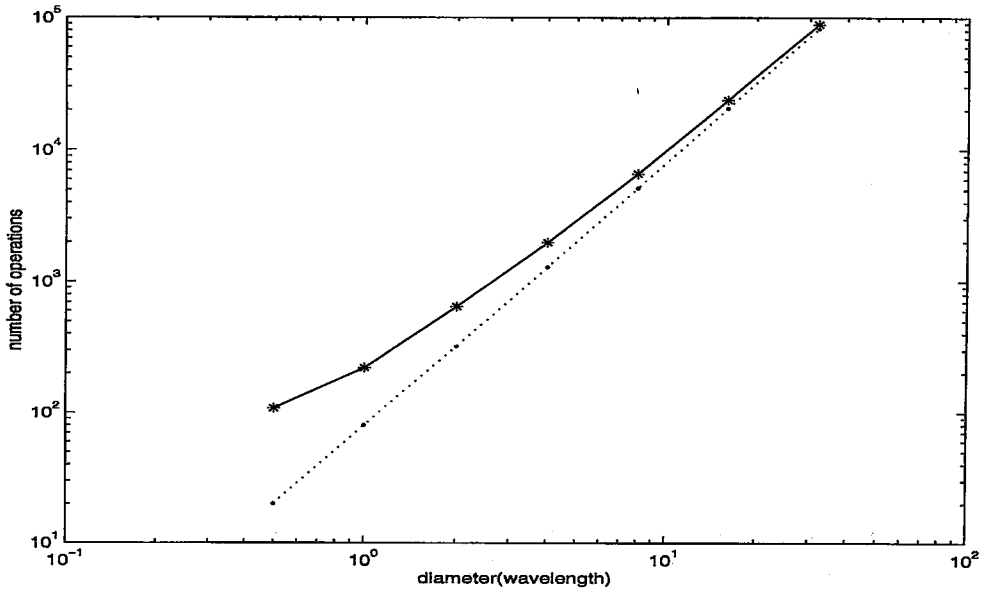

Figure 3.22. Numerical load in interpolation from nonuniformly spaced samples is compared with $O\left(N_\theta^2(q)\right)$ curve.

Because Lagrange interpolation is a polynomial approximation, sample points used in this interpolation can be the nonuniformly spaced Gauss-Legendre points. Throughout the analysis, 4-point Lagrange interpolation, which approximates $P_n(\cos\theta)$ to a polynomial of degree 4 is applied. Interpolated $P_2(\cos\theta)$ and $P_7(\cos\theta)$ using this method in this problem are illustrated in Fig.s 3.23-24.

It is expected that 4-point Lagrange interpolation produces exact results for $P_n(\cos\theta)$, where $n<4$, because $n+1$ data uniquely determine a polynomial of degree n, and that error increases with n increasing above 4. Error for the 7-level FMM is plotted in Fig. 3.25 along with the error bound for the interpolation.

Numerical load of the problem is obtained in terms of the number operations denoted by N_{op}. Number of operations in level q can be obtained as

$$N_{op}(q) \propto 4N_{\theta}(q) \tag{3.68}$$

where $N_{\theta}(q) \propto kD_q$. Hence

$$N_{op}(q) \propto 4kD_q. \tag{3.69}$$

Numerical load for this problem is shown in Fig. 3.26. Comparison to the $O(N_{\theta}(q))$ curve in Fig. (3.26) shows that the numerical complexity of the 4-point Lagrangian interpolation is $O(N_{\theta}(q))$.

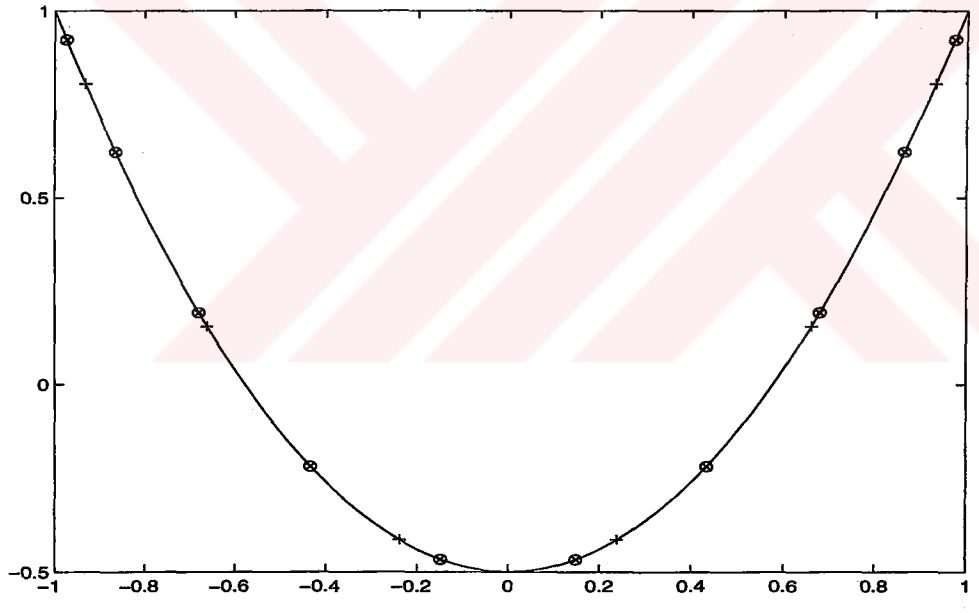

Figure 3.23. $P_2(\cos\theta)$ vs. cos θ.
4- point Lagrange interpolation of $P_2(\cos\theta)$. + : sampled data , x : interpolated data, o : exact value at the interpolation point.

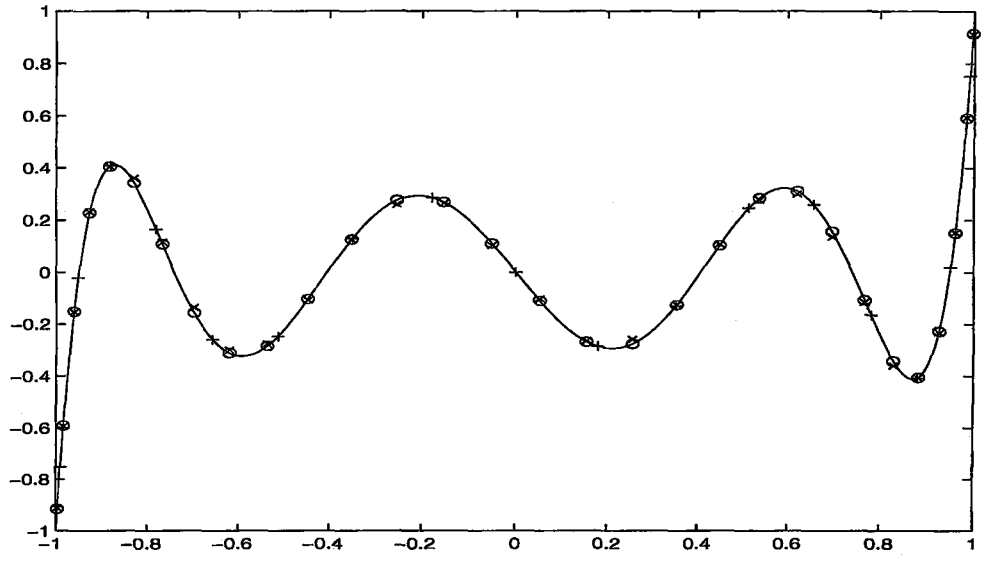

Figure 3.24. $P_7(\cos\theta)$ **vs. cos θ.**

4- point Lagrange interpolation of $P_7(\cos\theta)$. + : sampled data , x : interpolated data, o : exact value at the interpolation point

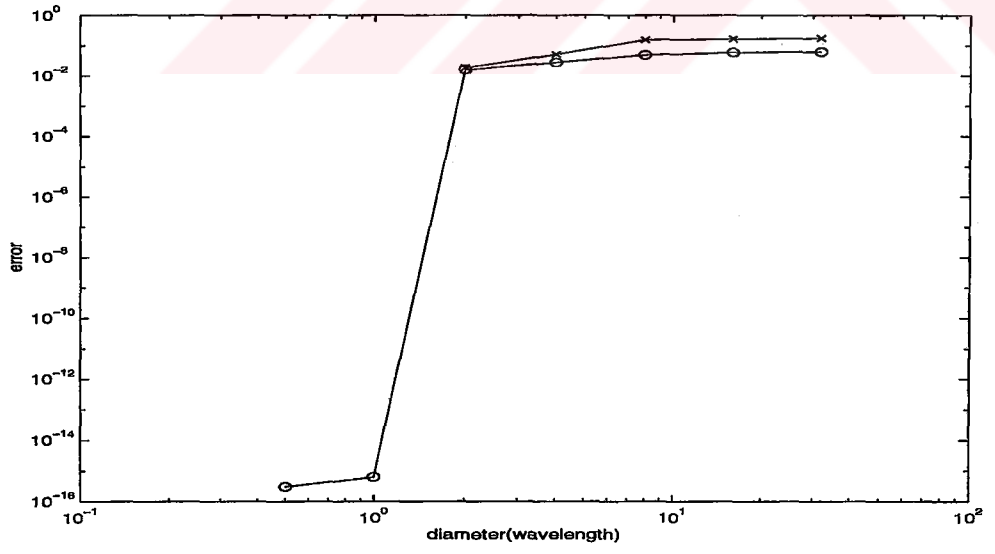

Figure 3.25. Error in 4-point Lagrange interpolation. Error due to Lagrange interpolation : o, error bound : x.

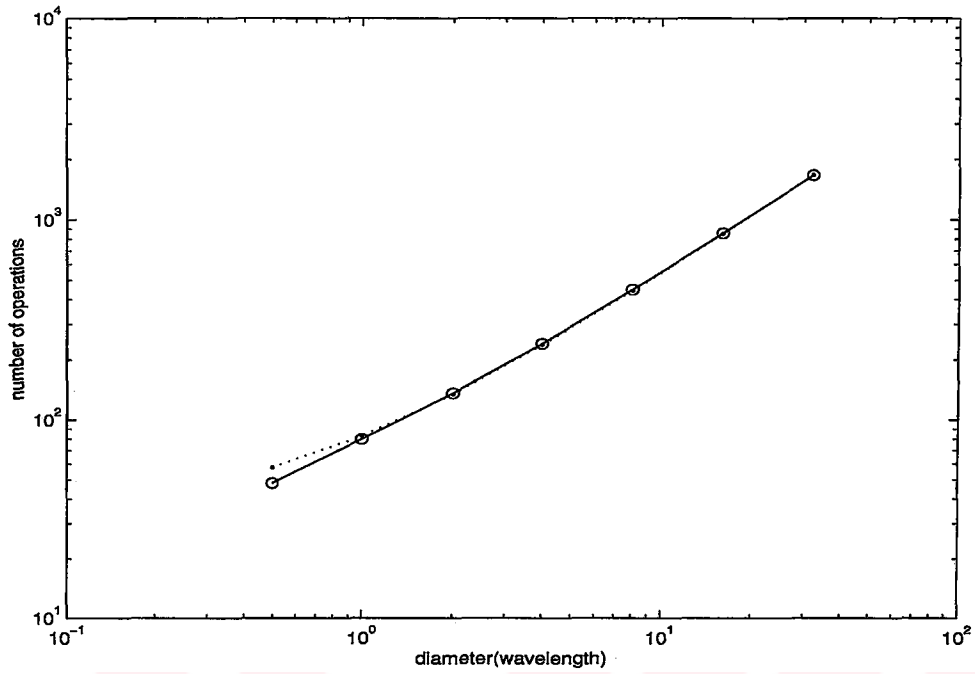

Figure 3.26. Numerical load in 4-point Lagrange interpolation. Numerical load due to Lagrange interpolation : solid, $O(N_\theta(q))$ curve :dotted.

CHAPTER 4

INTERPOLATION BY ACT METHOD

4.1. Introduction

In this chapter an efficient method, by Feichtinger, Gröchenig, and Strohmer [8], for the problem of reconstructing a bandlimited function from its nonuniformly spaced samples is introduced. The solution to the problem is based on the representation of the bandlimited function as a trigonometric polynomial, and the approximation of the polynomial by the frame operator, which can be formulated as a *Toeplitz system.* Since the method is a combination of the *adaptive weights method*, developed by Feichtinger and Gröchenig [7, 8], the *method of conjugate gradients* for the solution of positive definite linear systems, and the use of *Toeplitz matrices*, it is called the *ACT* (adaptive conjugate Toeplitz) algorithm. Adaptive weights method acts as a simple, but an efficient method of preconditioning. Efficient implementation of the conjugate gradients method is possible by utilizing the Toeplitz type structure of the system matrix, which allows the use of FFT routines in matrix-vector multiplication [8, 20]. Further acceleration in the algorithm is due to the application of nonuniform FFT formulations developed by Rokhlin and Dutt [4].

4.2. Discrete Irregular Sampling

Any discrete function $f[n]$ of length N, and bandwidth M can be expressed as

$$f[n]=\frac{1}{N}\sum_{k=-M}^{M}\hat{f}(k)e^{jk2\pi\frac{n}{N}},\quad n=0,...,N-1 \tag{4.1}$$

where

$$\hat{f}[k]=\sum_{n=0}^{N-1}f[n]e^{-jk2\pi\frac{n}{N}},\quad k=0,...,N-1. \tag{4.2}$$

$f[n]$ in Eq.(4.1) can be considered as the restriction of a trigonometric polynomial of period 1, and degree M to an arithmetic sequence in $[0,1)$. The generalized form of Eq.(4.1) is

$$p(t)=\sum_{k=-M}^{M}a_k e^{jk2\pi t}. \tag{4.3}$$

where $t \in [0,1)$, and $(a_k)_{k=-M}^{M} \in C^{2M+1}$, where C^{2M+1} is the complex vector space of dimension $2M + 1$.

Comparing Eq.s (4.1) and (4.3), it can be concluded that reconstruction of $f[n]$ given $0 \le n_1 < ... < n_r \le N - 1$, and the samples $f[n_i]$, $i=1,...,r$, is equivalent to reconstruction of $p(t)$ given $0 \le t_1 < ... < t_r < 1$, and the samples $p(t_i)$, $i = 1, \ldots, r$, where $t_i = \frac{n_i}{N}$. It is clear that the sampling points t_i do not need to be uniformly spaced.

Because p is a trigonometric polynomial of highest degree M, p is uniquely determined by its r samples $p(t_i)$ if and only if $r \ge 2M + 1$ [8].

4.3. Reconstruction by the Frame Operator

The efficient reconstruction of trigonometric polynomials by the frame operator as defined in [7] is possible by choosing Dirichlet Kernel as frame [10]. The following formulation shows that for $p(t)$ bandlimited to M, i.e. $p(t) = \sum_{k=-M}^{M} a_k e^{jk2\pi t}$ where $(a_k)_{k=-M}^{M} \in C^{2M+1}$, the frame operation is equivalent to a matrix operation, where the system matrix has a Toeplitz structure of size $(2M+1)\times(2M+1)$ [8].

Dirichlet kernel denoted by D_M is defined as

$$D_M(t) = \sum_{l=-M}^{M} e^{jl2\pi t} = \frac{\sin[(M+1/2)2\pi t]}{\sin(\pi t)}. \tag{4.4}$$

It is easily observed that

$$p(t) = \int_0^1 p(u) D_M(t-u)\, du \tag{4.5}$$

for the continuous case. Combining Eq.s (4.4) and (4.5) for the discrete case the frame operator S is defined as

$$S_p(t) = \sum_{i=1}^{r} p(t_i)\, D_M(t - t_i). \tag{4.6}$$

Substituting Eq.s (4.3) and (4.4) in Eq. (4.6) $S_p(t)$ can be rewritten as

$$\begin{aligned} S_p(t) &= \sum_{i=1}^{r} p(t_i) D_M(t-t_i) = \sum_{i=1}^{r} \sum_{k=-M}^{M} a_k e^{jk2\pi t_i} D_M(t-t_i) \\ &= \sum_{i=1}^{r} \sum_{k=-M}^{M} a_k e^{jk2\pi t_i} \sum_{l=-M}^{M} e^{jl2\pi(t-t_i)} \\ &= \sum_{l=-M}^{M} \left[\sum_{k=-M}^{M} \left(\sum_{i=1}^{r} e^{j(k-l)2\pi t_i} \right) a_k \right] e^{jl2\pi t}. \end{aligned} \tag{4.7}$$

Defining the $(2M+1)\times(2M+1)$ Toeplitz matrix $\overline{\mathbf{T}}$ as [8]

$$T_{lk} = T_{l-k} = \sum_{i=1}^{r} e^{-j(l-k)2\pi t_i} \text{ for } |l|, |k| \le M, \tag{4.8}$$

the coefficients of the trigonometric polynomial S_p can be found as $\overline{\mathbf{T}} \cdot \mathbf{a}$, where $\mathbf{a}$ is the column vector with entries a_k as defined in Eq. (4.3).

The formulation of the algorithm can now be seperated into three parts [8]:

$$1) b_k = \sum_{i=1}^{r} p(t_i)\, e^{jk2\pi t_i} \text{ for } |k| \le M, \tag{4.9}$$

$$2) \mathbf{a} = \overline{\mathbf{T}}^{-1} \cdot \mathbf{b} \in C^{2M+1} \tag{4.10}$$

$$3) p(t) = \sum_{k=-M}^{M} a_k e^{jk2\pi t} \tag{4.11}$$

4.4. Acceleration of the Reconstruction Algorithm

Acceleration of the algorithm, which is formulated in Eq.s (4.9-11), is achieved in two steps:

1) The matrix inversion in Eq. (4.10) can be solved by either direct methods for the complete inversion of $\overline{\mathbf{T}}$, or by iterative methods for an approximation of $\overline{\mathbf{T}}^{-1}$. Direct methods result in a higher precision; however, the numerical load is as high as $O(M^3)$. Iterative methods can be restricted to have lower complexity while having less accuracy. In this case the numerical load is $O(pM^2)$, where p is the number of iterations required for the desired accuracy. The use of the adaptive weights methods improves the condition number of $\overline{\mathbf{T}}$, hence reduces p [8]. The numerical complexity of the iterative methods can be further reduced to $O(pM \log_2 M)$ by

applying FFT routines in matrix-vector multiplications, where the system matrix has a Toeplitz structure.

Because $\overline{\mathbf{T}}$ is a positive-definite matrix, the conjugate gradients method is a simple, but a powerful and flexible choice for the iterative method. Moreover, it is well-suited for the solution of Toeplitz systems [20].

2) Eq.s (4.9) and (4.11) are simply the matrix-vector multiplications, which result in a numerical complexity of $O(M^2)$ in the critically sampled case. In fact, one can easily see that these equations are the Fourier and inverse Fourier transforms for irregularly spaced data. Hence the numerical load can be reduced to $O(M \log_2 M)$ using the generalized FFT algorithms developed by Dutt and Rokhlin [4].

4.4.1. Adaptive Weights Method

The *adaptive weights method* is a simple, but an efficient preconditioning method, which improves the condition number of the system matrix $\overline{\mathbf{T}}$ in Eq. (4.8), and provides explicit estimates for the rate of convergence, and therefore gives useful stopping criteria [7,8].

The weight vector $\mathbf{w}$ for the sampling set $0 \leq t_1 < \ldots < t_r < 1$ is defined as [7, 8]

$$w_i = \frac{1}{2}(t_{i+1} - t_{i-1}) \qquad \text{for } i = 1, \ldots, r, \tag{4.12}$$

where $t_0 = t_r - 1$, and $t_{r+1} = t_1 + 1$.

The frame operator is revised with the weight factors as [7, 8]

$$S_{wp}(t) = \sum_{i=1}^{r} p(t_i)\, w_i D_M(t - t_i), \tag{4.13}$$

and is called weighted frame operator.

If one repeats the computation in Eq. (4.7) with the weighted frame operator, the entries of the weighted system matrix $\overline{\mathbf{T}}_{\mathbf{w}}$ can be obtained as [8]

$$(T_w)_{lk} = (T_w)_{l-k} = \sum_{i=1}^{r} w_i e^{-j(l-k)2\pi t_i} \qquad \text{for } |l|, |k| \le M. \tag{4.14}$$

Moreover, defining the maximal gap δ as [7, 8]

$$\delta := \max(t_{i+1} - t_i) < \frac{1}{2M} \tag{4.15}$$

the condition number of $\overline{\mathbf{T}}_{\mathbf{w}}$ can be estimated by [8]

$$cond(\overline{\mathbf{T}}_{\mathbf{w}}) \le \left(\frac{1 + 2\delta M}{1 - 2\delta M}\right)^2 \tag{4.16}$$

It is clear from Eq. (4.16) that the rate of convergence is independent of clustering effects of the sampling set, and depends essentially on the maximal gap between the samples. As the rate of oversampling increases, the condition number of $\overline{\mathbf{T}}_{\mathbf{w}}$ decreases. The use of weights w_i compansates the local variations in the sampling density [7,8].

Reconstruction algorithm in Eq.s (4.9-11) revised with the adaptive weights method is as follows [8] :

For $\delta < \frac{1}{2M}$, let $p(t_i)$ be r samples of some p bandlimited to M,

$$1)\quad b_k = \sum_{i=1}^{r} p(t_i)\, w_i\, e^{jk2\pi t_i} \text{ for } \mid k \mid \leq M, \tag{4.17}$$

$$2)\quad \mathbf{a} = \overline{\mathbf{T}}_{\mathbf{w}}^{-1} \cdot \mathbf{b} \in C^{2M+1}, \tag{4.18}$$

$$3)\quad p(t) = \sum_{k=-M}^{M} a_k e^{jk2\pi t}. \tag{4.19}$$

It must be noted that $\overline{\mathbf{T}}_{\mathbf{w}}$ is a positive definite matrix because its entries are obtained by multiplying the the entries of $\overline{\mathbf{T}}$ with elements of $\mathbf{w}$, which are positive.

4.3.2. Conjugate Gradient Method

4.3.2.1. Algorithmic Prescription

The conjugate gradient method is an effective iterative method for solving large linear system of equations of the form

$$\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b} \tag{4.20}$$

where $\mathbf{x}$ is an unknown vector, $\mathbf{b}$ is a known vector, and $\overline{\mathbf{A}}$ is a known, square, symmetric, positive-definite matrix. A matrix is said to be positive-definite if for all non-zero vectors $\mathbf{x}$,

$$\mathbf{x}^{\mathbf{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{x} > \mathbf{0} \tag{4.21}$$

is satisfied. In fact, the solution to Eq. (4.20) is a critical point of $f(x)$ defined by the quadratic form

$$f(x) = \frac{1}{2}\mathbf{x}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{x} - \mathbf{b}^{\mathrm{T}} \cdot \mathbf{x} + c \tag{4.22}$$

where c is a scalar constant. If $\overline{\mathbf{A}}$ is positive-definite as well as symmetric, then the solution to Eq. (4.20) is a global minimum of f in Eq. (4.21) [18].

Starting from an arbitrary point $\mathbf{x}_{(0)}$ the iterates $\mathbf{x}_{(i)}$ are updated in each iteration by a multiple (α_i) of the search direction vector $\mathbf{d}_{(i)}$:

$$\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \alpha_{(i)}\mathbf{d}_{(i)} \tag{4.23}$$

In conjugate gradient method these search directions are chosen such that they are $\overline{\mathbf{A}}$ orthogonal, i.e.,

$$\mathbf{d}_{(i)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)} = 0. \tag{4.24}$$

$\alpha_{(i)}$ is chosen such that $f(\mathbf{x}_{(i+1)})$ be the minimum point along the search direction $\mathbf{d}_{(i)}$ [18] :

$$\frac{d}{d\alpha_{(i)}}\left(f(\mathbf{x}_{(i+1)})\right) = 0. \tag{4.25}$$

Eq. (4.25) yields

$$f'(\mathbf{x}_{(i+1)})^T \cdot \frac{d}{d\alpha_{(i)}}\mathbf{x}_{(i+1)} = -\mathbf{r}_{(i+1)}^T \cdot \mathbf{d}_{(i)} = \mathbf{d}_{(i)}^T \cdot \overline{\mathbf{A}} \cdot \mathbf{e}_{(i+1)} = 0 \ . \tag{4.26}$$

where $\mathbf{r}_{(i+1)} = \mathbf{b} - \overline{\mathbf{A}} \cdot \mathbf{x}_{(i+1)}$ is the residual vector at the $i+1$th step, and $\mathbf{e}_{(i+1)} = \mathbf{x}_{(i+1)} - \mathbf{x}$ is the error vector at the $i+1$th step. Noting that $\mathbf{e}_{(i+1)} = \mathbf{e}_{(i)} + \alpha_{(i)}\mathbf{d}_{(i)}$ and $\mathbf{r}_{(i)} = -\overline{\mathbf{A}} \cdot \mathbf{e}_{(i)}$, $\alpha_{(i)}$ can be obtained from Eq. (4.26) as [18]

$$\alpha_{(i)} = \frac{\mathbf{d}_{(i)}^T \cdot \mathbf{r}_{(i)}}{\mathbf{d}_{(i)}^T \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(i)}} \tag{4.27}$$

The set of $\overline{\mathbf{A}}$-orthogonal search directions $\{\mathbf{d}_{(i)}\}$ is generated by the *conjugate Gram-Schmidt process.* For n linearly independent vectors $\mathbf{u}_{(0)}, \mathbf{u}_{(1)}, \ldots, \mathbf{u}_{(n-1)}$, one sets $\mathbf{d}_{(0)} = \mathbf{u}_{(0)}$, and for $i > 0$, the search direction at the ith step can be related to the search directions at the previous steps as [18]

$$\mathbf{d}_{(i)} = \mathbf{u}_{(i)} + \sum_{k=0}^{i-1} \beta_{ik}\mathbf{d}_{(k)} \tag{4.28}$$

where β_{ik} are defined for $i > k$. Multiplying the transpose of each side of Eq. (4.28) by $\overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}$ for $j < i$, then using the $\overline{\mathbf{A}}$-orthogonality of the search directions, β_{ij} can be found as

$$\beta_{ij} = -\frac{\mathbf{u}_{(i)}^T \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}}{\mathbf{d}_{(j)}^T \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}} \tag{4.29}$$

The difficulty in Gram-Schmidt conjugation is that all the old search vectors must be kept in memory to construct each new one, and furthermore $O(n^3)$ operations are required to generate the full set. This difficulty can be overcome when the search vectors are constructed by conjugation of the residuals. By replacing $\mathbf{u}$ with the residual vector $\mathbf{r}$ Eq. (4.26) can be rewritten as

$$\beta_{ij} = -\frac{\mathbf{r}_{(i)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}}{\mathbf{d}_{(j)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}} \tag{4.30}$$

By taking the inner product of $\mathbf{r}_{(j)}$ with $\mathbf{r}_{(\,)}$ and noting that $\mathbf{r}_{(j+1)} = \mathbf{r}_{(j)} - \alpha_{(j)}\overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}$, $\mathbf{r}_{(i)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)}$ can be obtained as

$$\mathbf{r}_{(i)}^{T} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(j)} = \begin{cases} \dfrac{1}{\alpha_{(i)}} \mathbf{r}_{(i)}^{\mathrm{T}} \cdot \mathbf{r}_{(i)}, & i = j, \\ -\dfrac{1}{\alpha_{(i-1)}} \mathbf{r}_{(i)}^{\mathrm{T}} \cdot \mathbf{r}_{(i)}, & i = j+1, \\ 0, & \textit{otherwise.} \end{cases} \tag{4.31}$$

Substituting Eq. (4.31) in Eq. (4.30), the latter can be expressed as:

$$\beta_{ij} = \begin{cases} \dfrac{1}{\alpha_{(i-1)}} \dfrac{\mathbf{r}_{(i)}^{\mathrm{T}} \cdot \mathbf{r}_{(i)}}{\mathbf{d}_{(i-1)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(i-1)}}, & i = j+1 \\ 0, & i > j+1. \end{cases} \tag{4.32}$$

By Eq. (4.32) it is no longer necessary to store old search vectors to guarantee the $\overline{\mathbf{A}}$-orthogonality of the new search vectors, hence $\beta_{i(i-1)}$ can be simplified to $\beta_{(i)}$.

The algorithmic prescription for the method of conjugate gradient is now [18]:

1) $\mathbf{d}_{(0)} = \mathbf{r}_{(0)} = \mathbf{b} - \overline{\mathbf{A}} \cdot \mathbf{x}_{(0)}$, (4.33)

2) Using the identity $\mathbf{d}_{(i)}^{T} \cdot \mathbf{r}_{(i)} = \mathbf{r}_{(i)}^{T} \cdot \mathbf{r}_{(ig)}$ and Eq. (4.27) $\alpha_{(i)}$ can be obtained as

$$\alpha_{(i)} = \frac{\mathbf{r}_{(i)}^{\mathrm{T}} \cdot \mathbf{r}_{(i)}}{\mathbf{d}_{(i)}^{\mathrm{T}} \cdot \overline{\mathbf{A}} \cdot \mathbf{d}_{(i)}}, \tag{4.34}$$

3) $\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \alpha_{(i)}\mathbf{d}_{(i)}$, (4.35)

$$4)\ \mathbf{r}_{(i+1)} = \mathbf{r}_{(i)} - \alpha_{(i)}\overline{\mathbf{A}} \cdot \mathbf{d}_{(i)}, \tag{4.36}$$

$$5)\ \beta_{(i+1)} = \frac{\mathbf{r}_{(i+1)}^{\mathrm{T}} \cdot \mathbf{r}_{(i+1)}}{\mathbf{r}_{(i)}^{\mathrm{T}} \cdot \mathbf{r}_{(i)}}, \tag{4.37}$$

$$6)\ \mathbf{d}_{(i+1)} = \mathbf{r}_{(i+1)} + \beta_{(i+1)}\mathbf{d}_{(i)}. \tag{4.38}$$

4.4.2.2. Numerical Complexity

Conjugate gradient method reaches the exact result after n iterations, where n is the dimension of the system matrix $\overline{\mathbf{A}}$; however, in practice accumulated floating point roundoff error causes the residual to lose accuracy, and cancellation error causes to lose $\overline{\mathbf{A}}$-orthogonality [18].

The convergence behavior of conjugate gradient depends on the distribution of eigenvalues of the system matrix. Given infinite floating point precision, the number of iterations required to compute an exact solution is at most the number of distinct eigenvalues, and convergence is quicker when the eigenvalues are closely rather than irregularly distributed.

The dominating numerical load in the conjugate gradient stems from the matrix-vector multiplications in the first and second steps of the algorithmic prescription. If the number of iterations required to reach the desired accuracy is p, then the numerical complexity is $O(pn^2)$.

Reduction in numerical load is possible for the linear systems where the system matrix has a Toeplitz structure, which allows the use of FFT routines in matrix-vector multiplicatons. The Toeplitz matrix $\overline{\mathbf{A}}$ of size $n \times n$ can be extended to a circulant matrix $\tilde{\mathbf{A}}$ of size $(2n-1) \times (2n-1)$ whose entries in the ith row are related to the those of $\overline{\mathbf{A}}$ in the following way:

$$\tilde{A}_{ij} = \begin{cases} A_{ij}, & j = 1,\ldots, n \\ A^*_{i(2n+1-j)}, & j = n+1,\ldots,2n-1. \end{cases} \tag{4.39}$$

The vector $\mathbf{x}$ is completed to $\tilde{\mathbf{x}}$ by $n-1$ zeros. The entries of the circulant matrix $\tilde{\mathbf{A}}$ satisfy $\tilde{A}_{ij} = \tilde{A}_{i-j} = \tilde{A}_{i-j+n}$, hence the linear system $\tilde{\mathbf{A}} \cdot \tilde{\mathbf{x}} = \tilde{\mathbf{b}}$ is equivalent to the circular convolution $\tilde{\mathbf{a}} * \tilde{\mathbf{x}} = \tilde{\mathbf{b}}$, where $\tilde{\mathbf{a}}$ is the first column of $\tilde{\mathbf{A}}$. Applying the discrete Fourier transform to the circular convolution one gets $\hat{\mathbf{a}} \bullet \hat{\mathbf{x}} = \hat{\mathbf{b}}$, where $\hat{\mathbf{a}}$, $\hat{\mathbf{x}}$, and $\hat{\mathbf{b}}$ are the Fourier transforms of the $\tilde{\mathbf{a}}$, $\tilde{\mathbf{x}}$, and $\tilde{\mathbf{b}}$ respectively, then $\tilde{\mathbf{b}}$ is recovered from $\hat{\mathbf{b}}$ by inverse Fourier transform. The first n elements of $\tilde{\mathbf{b}}$ constitute $\mathbf{b}$, where $\overline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$. The efficient implementation of Fourier transform is possible when the length of the vectors is a power of 2, then the numerical complexity of the matrix-vector multiplication becomes $O(n \log_2 n)$ leading to the overall complexity $O(pn \log_2 n)$ [8,18].

4.4.3. Nonuniform FFT Algorithm

The nonuniform FFT algorithm is developed to accelerate the computation of the discrete Fourier transform of the nonuniformly spaced data, which can be generalized as [4]

$$f_l = F(\alpha)_l = \sum_{k=0}^{N} \alpha_k e^{j\omega_k \frac{2\pi l}{N}} \tag{4.40}$$

for $l = -N/2,\ldots, N/2$.

4.4.3.1. Algorithmic Prescription

The nonuniform FFT algorithm depends on the approximation theory. The principal idea is to analyze the Fourier series of functions $\phi : [-\pi, \pi] \rightarrow C$, where C is the complex set of numbers, given by the formula:

$$\phi(x) = e^{-bx^2} \cdot e^{jcx} \tag{4.41}$$

where $b > \frac{1}{2}$ and c are real numbers.

Following is a series of theorems and a corollary from [4] which form the basis of nonuniform FFT algorithm.

Theorem 1: Let $\phi(x) = e^{-bx^2} \cdot e^{jcx}$ for any real $b > \frac{1}{2}$ and c. Then for any $x \in (-\pi, \pi)$,

$$\left| \phi(x) - \sum_{k=-\infty}^{\infty} \rho_k e^{jkx} \right| < e^{-b\pi^2} \cdot \left(4b + \frac{70}{9} \right), \tag{4.42}$$

$$\rho_k = \frac{1}{2\sqrt{b\pi}} e^{-(c-k)^2/4b} \text{ for } k = -\infty, \ldots, \infty \tag{4.43}$$

According to Theorem 1, functions of the form $e^{-bx^2} \cdot e^{jcx}$ can be approximated by a Fourier series whose coefficients are given analytically by Eq. (4.43), and the error of the approximation decreases exponentially as b increases.

The coefficients ρ_k have a peak at $k = [c]$, the nearest integer to c, and decays exponentially as $k \rightarrow \pm\infty$. So one can intuitively expect that the series in Eq. (4.42) can be expressed as a finite-term series. This observation leads to Theorem 2.

Theorem 2 : Let $\phi(x) = e^{-bx^2} \cdot e^{jcx}$ for any real $b > \frac{1}{2}$ and c, and let q be an even integer such that $q \geq 4\pi b$. Then for any $x \in (-\pi, \pi)$,

$$\left| \phi(x) - \sum_{[c]-q/2}^{[c]+q/2} \rho_k e^{jkx} \right| < e^{-b\pi^2}(4b + 9) \tag{4.44}$$

The following corollary describes a formula for approximating e^{jcx}, which will be used in the nonuniform discrete Fourier transform expression, using a series of $q + 1$ terms:

Corollary : Suppose that $m \geq 2$ is an integer and that the conditions of Theorem 2 are satisfied. Then multiplying both sides by e^{bx^2}, we can obtain

$$\begin{aligned} \left| e^{jcx} - e^{bx^2} \sum_{k=[c]-q/2}^{[c]+q/2} \rho_k e^{jkx} \right| &< e^{bx^2} e^{-b\pi^2}(4b + 9) \\ &< e^{b\pi^2/m^2} e^{-b\pi^2}(4b + 9). \end{aligned} \tag{4.45}$$

for any $x \in [-\pi/m, \pi/m]$.

Theorem 3 makes use of a simple linear scaling to generalize the inequality in Eq. (4.45) from $\left[-\frac{\pi}{m}, \frac{\pi}{m}\right]$ to any interval $[-d, d]$.

Theorem 3: Let $b > 1/2$, $c, d > 0$ be real numbers, and let $m \geq 2$, $q \geq 4\pi b$ be integers. Then for any $x \in [-d, d]$

$$\left| e^{jcx} - e^{b(x\pi/md)^2} \sum_{k=[cmd/\pi]-q/2}^{[cmd/\pi]+q/2} \rho_k e^{jx\pi/md} \right| < e^{-b\pi^2(1-1/m^2)} \tag{4.46}$$

Following is an algorithmic prescription for the formulation obtained above.

1) For an integer $m \geq 2$ and a real number $b > 0$, we define a real number $\varepsilon > 0$, which denotes the desired accuracy in Eq. (4.46) :

$$\varepsilon = e^{-b\pi^2(1-1/m^2)}(4b + 9) \tag{4.47}$$

2) We denote by q the smallest even natural number such that

$$q \geq 4\pi b. \tag{4.48}$$

3) For an integer m, and a set of real numbers $\{\omega_k\}$, we will denote by μ_k the nearest integer to $m\omega_k$ for $k = 0,..., N$, and by $\{P_{lk}\}$ a set of real numbers defined by the formula

$$P_{lk} = \frac{1}{2\sqrt{b\pi}} e^{-(m\omega_k - (\mu_k + l))^2/4b} \tag{4.49}$$

for $k = 0, \ldots, N$, and $l = -q/2, \ldots, q/2$.

Setting $d = \pi$ in Theorem 3 we obtain

$$\left| e^{j\omega_k x} - e^{b(x/m)^2} \sum_{-q/2}^{q/2} P_{lk} e^{j(l+\mu_k)x/m} \right| < \varepsilon \tag{4.50}$$

for any $k = 0,..., N$, and any $x \in [-\pi, \pi]$.

4) For any given set of complex numbers $\{\alpha_k\}$, we will denote by $\{\tau_j\}$ the unique set of complex coefficients such that

$$\sum_{k=1}^{N} \alpha_k \sum_{l=-q/2}^{q/2} P_{lk} e^{j(\mu_k + l)\frac{x}{m}} = \sum_{n=-mN/2}^{mN/2-1} \tau_n e^{jnx/m} \tag{4.51}$$

so that

$$\tau_n = \sum_{k,l,\mu_k + l = n} \alpha_k P_{lk}. \tag{4.52}$$

5) We will denote by $\{T_l\}$ a set of complex numbers defined by the formula:

$$T_l = \sum_{n=-mN/2}^{n=mN/2} \tau_n e^{jl2\pi n/mN} \text{ for } n = -mN/2,..., mN/2 - 1 \tag{4.53}$$

6) $\tilde{f}_l = e^{b(2\pi l/mN)^2} T_l$, for $l = -N/2,..., N/2$. (4.54)

4.4.3.2. Numerical Complexity

When the steps in Eq.s (4.47-54) are followed, the first three steps are considered as the initialization stages, and the steps 4-6 are considered as the evaluation stages. The numerical complexity depends on the numerical loads in the evaluation stages, which are $O(Nq)$, $O(mN \log_2 N)$, if FFT is used, and $O(N)$ respectively. Hence the overall numerical complexity of the algorithm, which is the sum of the numerical loads of each step, becomes $O(mN \log_2 N + Nq)$. It must be noted that the values q, and m determine both the accuracy and the numerical complexity of the algorithm.

4.5. Numerical Results and Discussion

The numerical results in this section are obtained from the interpolation of the associated Legendre function of the largest degree and any possible order at each level for the 7-level FMM problem described in section 3.2.3.2 by the ACT method. N_α, N_θ, n for each level are tabulated in Table 3.2. The constraint for this method to be applicable is the bandlimitedness of the function to be interpolated. The associated Legendre functions already satisfy this constraint. The sample points are the Gauss-Legendre points which allow accurate integration by the Gaussian quadrature as mentioned before.

The constraint that $P_n^m(\cos\theta)$ be bandlimited implies that there should be no discontinuities when the samples are extended to infinity with period π. However, $P_n^m(\cos\theta)$ becomes discontinuous for n odd when $m = 0$. To avoid this phenomena samples in θ should be folded around π to cover the interval $[0,2\pi]$.

The sample and the interpolation points in the interval $[0,2\pi]$ must be scaled to the interval $[0,1)$, hence 2π is a proper scaling factor. It can be seen that with this choice of the scaling factor, the Nyquist Criterion in Eq. (4.15) is satisfied.

The following analysis consists of three steps. In the first step the nonuniform FFT algorithm is applied to evaluate the nonuniform Fourier sum in Eq. (4.17). In the second step the solution to the linear system in Eq. (4.18) is obtained by CG-FFT, which makes use of the Toeplitz structure of the system matrix. In the third step the nonuniform inverse FFT algorithm is applied to evaluate the nonuniform inverse Fourier sum in Eq. (4.19). The error and numerical load due to all three parts are illustrated, and compared with the results obtained from direct evaluation of these three steps.

The implementation of the nonuniform FFT and IFFT algorithms in steps 1 and 3, and the CG-FFT algorithm in step 2 consists of two main stages. The first stage is called the initialization stage in which the matrix and vector operators of the algorithms are precomputed and stored. The second stage is the evaluation stage in which these operators are applied. Considering the algorithm in two stages provides the advantage of requiring the initialization stage to be performed only once when the transformation is successively applied to multiple vectors. This feature of the algorithm is consistent with the grouping idea of FMM, which lets the initialization stage to be computed only once at each level.

The above discussion suggests that the numerical loads of the nonuniform FFT and IFFT algorithms depend heavily on the numerical load of the second stage, which is given as $O(mN \log_2 N) + Nq$, where m and q are as defined in Theorem 2 in section 4.4.3.1, and N is the length of the input vector. The parameters m and q affect the numerical efficiency, and the parameters b and q affect the accuracy of the algorithm.

The nonuniform FFT and IFFT algorithms involve the computation of a uniform FFT of size proportional to the length of the input vector, thus the efficient performance of the algorithms depends on the efficient performance of FFT. It is

known that if the length of the input vector is a power of 2, the fast radix-2 FFT algorithm can be applied. Hence the input vector is extended to length $2^{\lceil \log_2 N \rceil}$, i.e. the smallest power of 2 which is greater than N, by padding with zeros.

Both the nonuniform FFT and IFFT algorithms require the evaluation of the sums of the form

$$f_l = \sum_{k=-N/2}^{N/2-1} \alpha_k e^{jk2\pi l/N} \tag{4.57}$$

for $l = -N/2, \ldots, N/2 - 1$, whereas the general expression for the discrete Fourier transform is given as

$$f_l = \sum_{k=0}^{N-1} \alpha_k e^{jk2\pi l/N} \tag{4.58}$$

for $l = 0, \ldots, N - 1$. By defining $\hat{\alpha}_k = \alpha_k$ for $k = 0, \ldots, N/2 - 1$, $\hat{\alpha}_k = \alpha_{k-N}$ for $k = N/2, \ldots, N - 1$, and $\hat{f}_l = f_l$ for $l = 0, \ldots, N/2 - 1$, $\hat{f}_l = f_{l-N}$ for $l = N/2, \ldots, N - 1$, where $\hat{f}$ is the discrete Fourier transform of $\hat{\alpha}$, the sum in Eq. (4.57) is converted to the sum in Eq. (4.58).

The numerical load of the CG-FFT algorithm also depends heavily on the numerical load of the evaluation stage, which is given as $O(pN \log_2 N)$, where p is the number of iterations required to reach the desired accuracy, and N is the length of the column vector of the system matrix. The CG-FFT algorithm involves the computation of uniform FFT of size proportional to N, hence the column vector of the system matrix must also be extended to the smallest power of 2 which is greater than N by padding with zeros.

The error and numerical load due to all three parts are illustrated, and compared with the results obtained from direct evaluation of these three steps. The measure of accuracy chosen in the following analysis is defined by the formula:

$$\text{error} = \sqrt{\sum_{j=0}^{N} \left| \tilde{f}_j - f_j \right|^2 \Big/ \sum_{j=0}^{N} \left| f_j \right|^2} \tag{4.56}$$

where f is the result of a direct computation, and $\tilde{f}$ is the result of the computation being considered.

The overall error due to the implementation of the nonuniform FFT algorithm in the first step depends on the values of q and b. One can obtain emprically q and the corresponding b values which produce the least error as in Table 4.1, where $m = 2$. The errors shown in Table 4.1 are plotted as a function of q, along with the corresponding results for the cases $m = 3$ and $m = 4$ in Fig. 4.1. It is clear that q has a significant effect on accuracy, while $m = 2, 3, 4$ produce almost the same results. Hence for computational efficiency $m = 2$ is preferred. The effect of q on the numerical complexity of the first part is also investigated and illustrated in Fig. 4.2.

Table 4.1. Best values of b and the corresponding maximum errors for each level when $m = 2$.

q	b	error
2	0.15	1.31×10^{-2}
4	0.25	5.732×10^{-4}
6	0.35	2.645×10^{-5}
8	0.4	1.150×10^{-6}
10	0.5	3.325×10^{-8}
12	0.6	2.375×10^{-9}
14	0.7	2.529×10^{-10}
16	0.8	2.551×10^{-10}
18	0.9	2.241×10^{-12}
20	0.95	8.019×10^{-14}
22	1	1.450×10^{-14}
24	1.05	1.408×10^{-14}

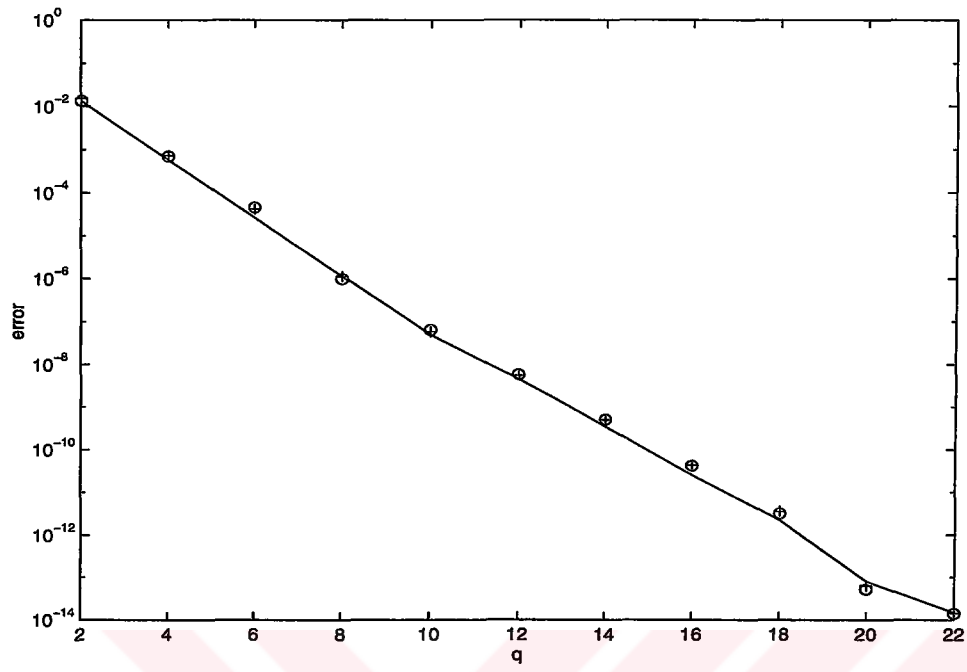

Fig. 4.1. Maximum values of the error for $m = 2$: solid, $m = 3$: +, $m = 4$: o.

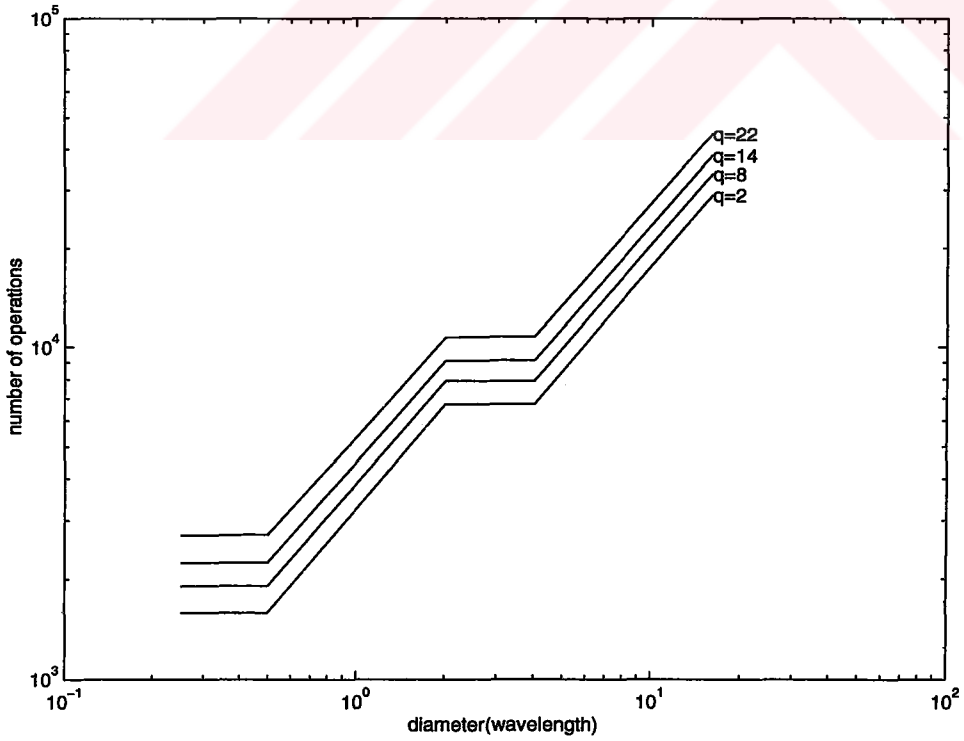

Fig. 4. 2. The numerical load due to the implementation of the nonuniform FFT algorithm in the first part for $q = 2$, $q = 8$, $q = 14$, $q = 22$.

Comparing Fig.s 4.1 and 4.2 one can optimize the error and the numerical complexity by choosing $q = 8$, hence $b = 0.4$, leading to an error in the order of 10^{-6} fot the single precision arithmetic. In Fig. 4.3 the numerical load of the first step when the nonuniform FFT is applied is compared with the numerical load of the direct evaluation. Remembering that $N_\theta(m') \propto N_\alpha(m') \propto kD_{m'}$, where $D_{m'}$ is the diameter of the cell considered at level m', the slopes of the curves in Fig. 4.3 give an idea about the numerical loads. The slope of the curve due to direct evaluation is close to 2, hence the numerical load is proportional to $N_\theta^2(m')$ at each level m'. The slope of the curve due to the implementation of the nonuniform FFT algorithm is close to unity, hence the numerical load is proportional to $N_\theta(m')\log_2 N_\theta(m')$ at each level m'.

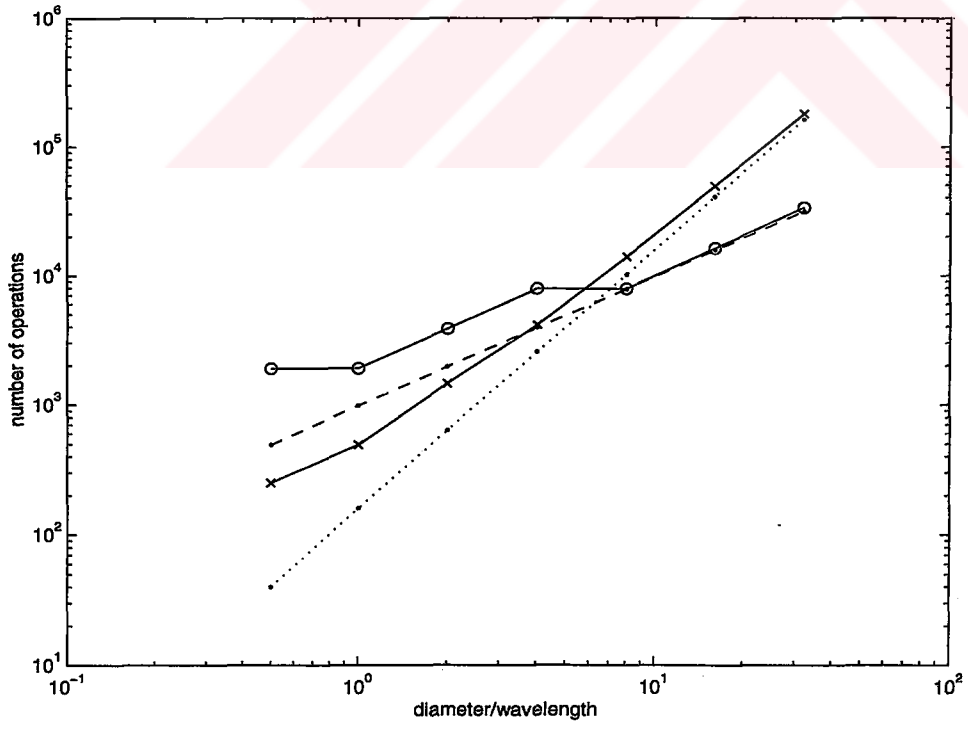

Figure 4.3. The numerical load due to implementation of step 1 by nonuniform FFT : o, and by direct computation : x. $O(N_\theta(m'))$ curve : dotted, $O(N_\theta^2(m'))$ curve : dashed.

Because the error bound of the first step is 10^{-6}, the desired accuracy in the second step need not be less than 10^{-6}, hence one can limit the desired tolerance in CG-FFT to 10^{-6}. It was observed that the number of iterations to reach this accuracy is at most 3, which occurs in the highest level, where the system matrix has the largest dimension.The error due to the implementation of the CG or CG-FFT methods for the tolerance 10^{-6} is illustrated in Fig. 4.4.

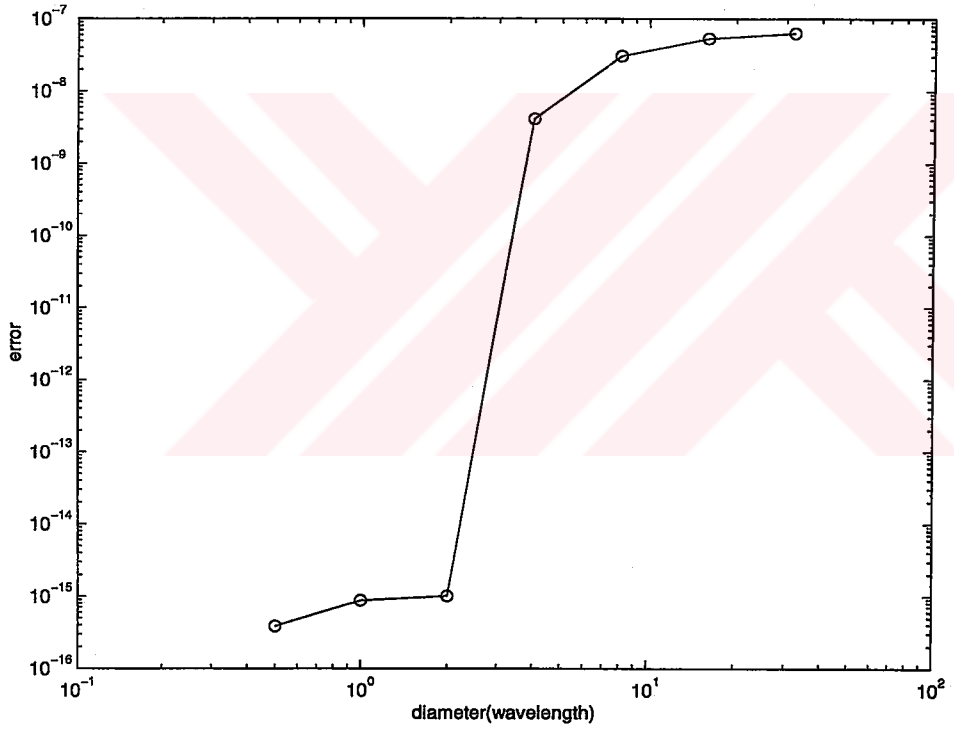

Figure 4.4. Error due to the implementation of the CG or CG-FFT methods for tolerance of 10^{-6}.

The quick convergence of the CG/CG-FFT is due to the almost uniform distribution of the Gauss-Legendre points in θ, and two-fold oversampling for the

critically sampled case. It is observed that the preconditioning effect of the adaptive weights method for this problem is insignificant, i.e. the number of iterations required to reach error in the order of 10^{-6} is the same when the frame operator is weighted, and when it is not.

The numerical loads due to the implementation of the second step by Gaussian elimination, by the CG method and by the CG-FFT method are illustrated in Fig. 4.5. From the slopes of the curves it is verified that direct inversion involves $O(N_\theta^3(m'))$ operations, the CG method involves $O(N_\theta^2(m'))$ operations, and the CG-FFT method involves $O(N_\theta(m')\log_2 N_\theta(m'))$ operations.

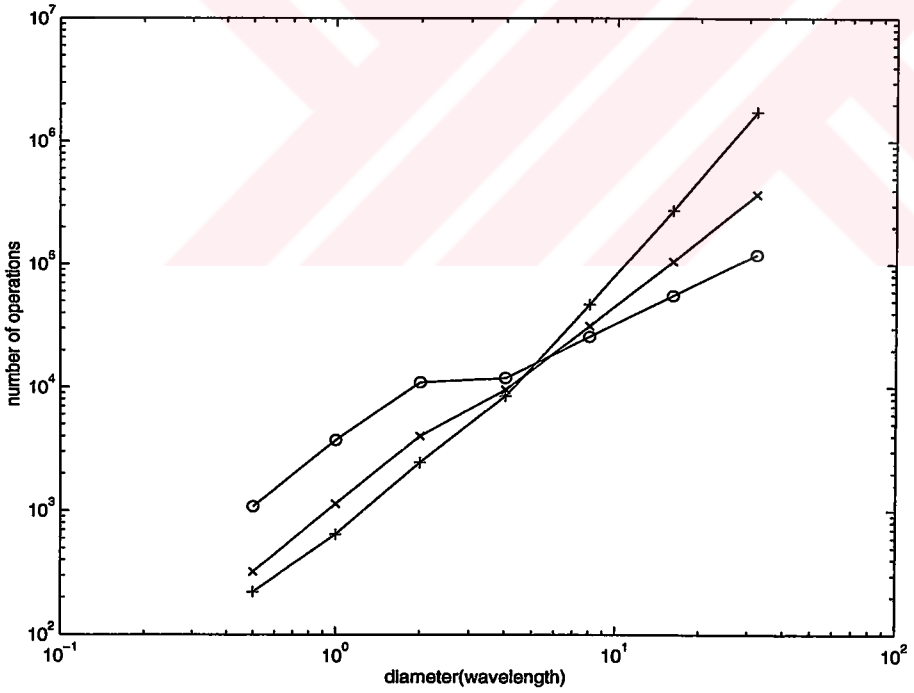

Figure 4.5. The numerical load due to the implementation of the second step by Gaussian elimination : +, by the CG method : x, by the CG-FFT method : o

In Fig. 4.6 the overall error due to the implementation of the nonuniform IFFT algorithm is illustrated with the same choice of b and q in the first step. In Fig. 4.7. the numerical load due to the implementation of the nonuniform IFFT in the third step and due to direct evaluation are shown. From the slopes of the curves due to IFFT algorithm and due to direct evaluation, which are very close to 1 and 2, respectively, it is verified that the number of operations are proportional to $O(N_\theta(m')\log_2 N_\theta(m'))$ and $O(N_\theta^2(m'))$, respectively.

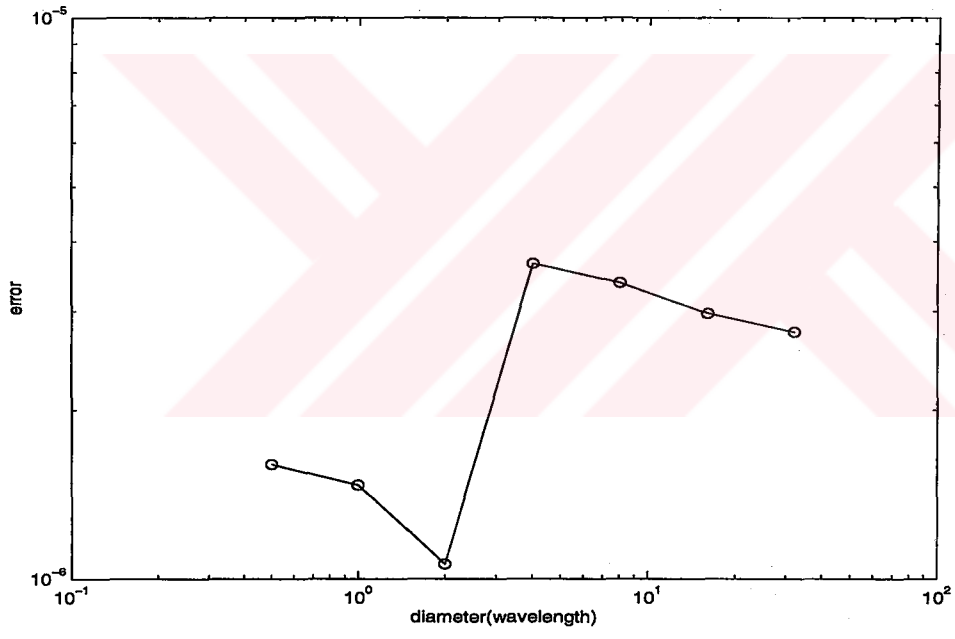

Figure 4.6. Overall error due to the implementation of the nonuniform FFT algorithm in the third step

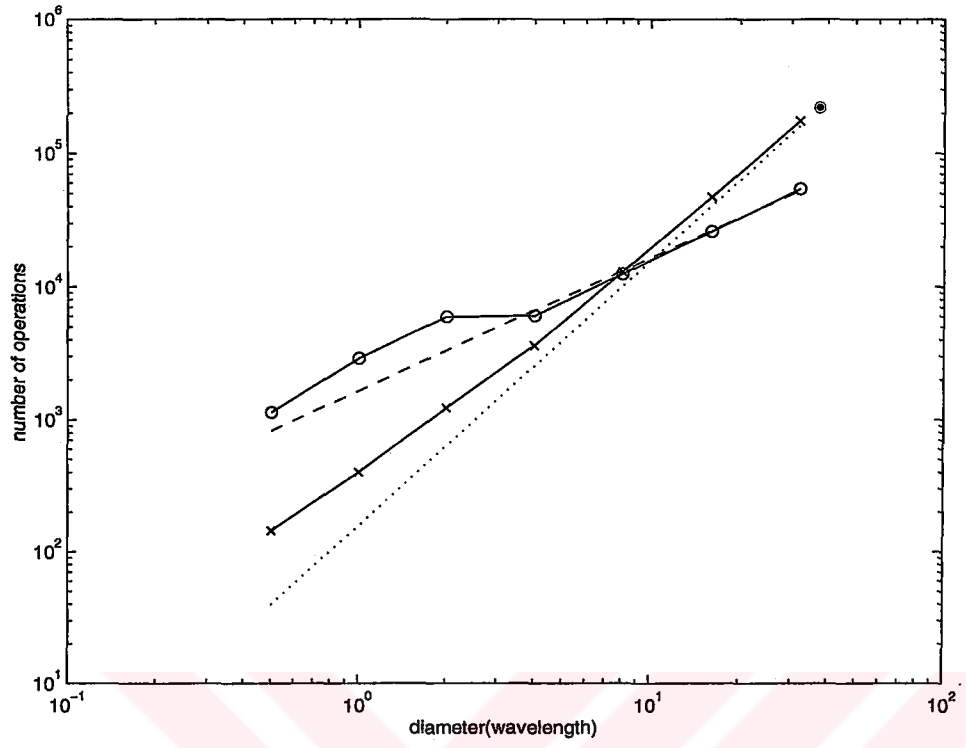

Figure 4.7. Numerical load due to the implementation of the third step by direct evaluation : x, by the nonuniform FFT algorithm : o . $O\left(N_\theta^2(m')\right)$ curve : dotted, $O\left(N_\theta(m')\right)$ curve : dashed.

Finally the overall numerical complexity and error of the ACT algorithm is compared to the numerical complexity and error of the 4-point Lagrange interpolation which is the best candidate for nonuniform interpolation in Fig.s 4.8 and 4.9, respectively. Although the numerical load due to the ACT algorithm is much higher than the numerical load due to the 4-point Lagrange interpolation, the slopes of the curves are almost same, which means that the numerical complexities of the methods can be considered to be the same. The error due to the ACT algorithm is much lower for the parameters chosen, i.e. $m = 2$, $q = 8$, $b = 0.4$, and is adjustable without a significant change in the numerical load.

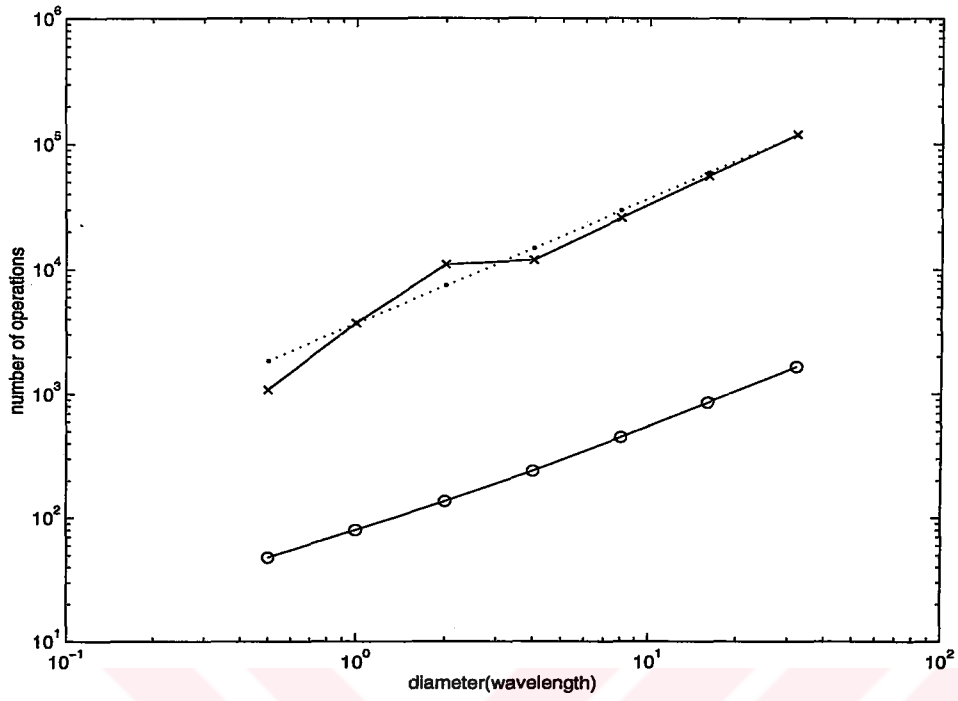

Fig. 4. 8. Comparison of numerical load due to 4-point Lagrange interpolation : o, due to ACT algorithm : x . $O(N_\theta(m'))$ curve : dotted is also plotted.

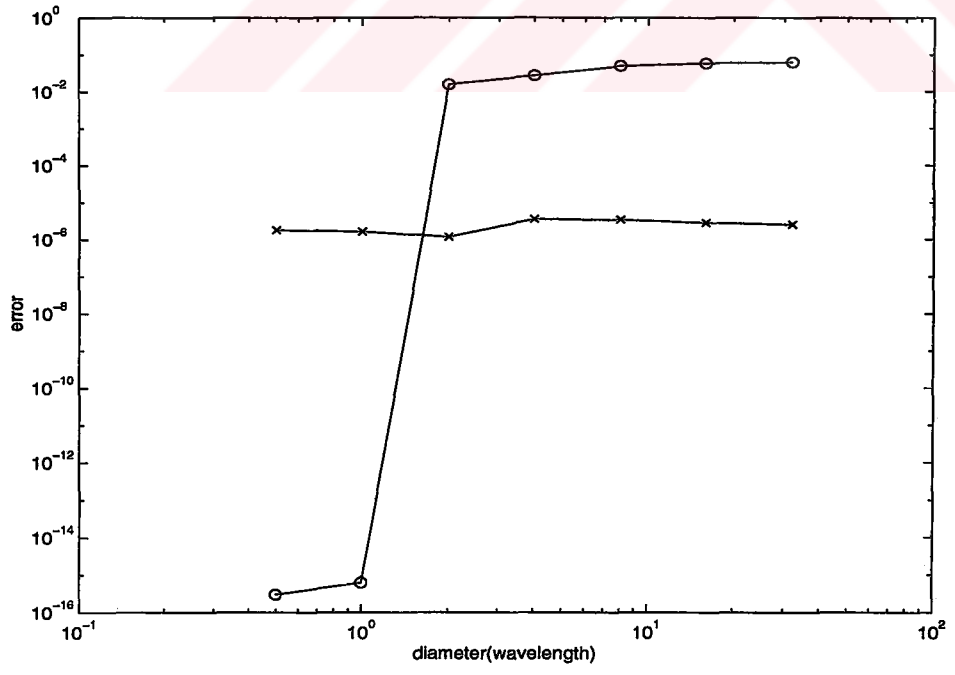

Fig. 4.9 Error due to 4- point Lagrange interpolation : o, error due to ACT algorithm : x

CHAPTER 5

CONCLUSION

In this thesis, the interpolation problem in the upward pass of the MLFMA is studied. The sample points in this problem are the Gauss-Legendre points, which allow accurate integration by the Gaussian quadrature in the downward pass of the MLFMA. The ACT method, which is a combination of the adaptive weights method, the conjugate gradient method, and the use of Toeplitz matrices, has been suggested for the interpolation. The method is further improved by the use of nonuniform FFT and inverse FFT algorithms, which is the unique contribution of this study.

The conventional interpolation methods, which are cardinal series expansion, self-truncating series, AP series, sampling window, and the Lagrangian approximation are also studied. The first four methods are defined for the uniformly spaced samples, hence are not suitable for the Gaussian quadrature. However, considering that the Gauss-Legendre points are almost uniform, one can apply these methods on the irregularly spaced data, and obtain acceptable results. For the conventional methods global interpolation introduces numerical complexity of $O(N^2)$, where N is the number of samples, hence local interpolation, in which only r samples around the interpolation point constitute the sampling set, should be preferred resulting in a numerical load of $O(N)$.

The performance of the ACT method has been demonstrated on the interpolation of the associated Legendre functions for the largest degree, and any order possible for each level considered in the 7-level FMM problem as described in section 3.2.3.2. The numerical complexity due to this method is $O(N \log_2 N)$, where N is the number of samples in each level considered. The interpolation error

is determined by the choice of the parameter q, which determines the truncation limit of the finite-term Fourier series, and b, which is a parameter over accuracy in the nonuniform FFT and IFFT algorithms. For the change of q from 2 to 24 with b changing from 0.15 to 1.05, the error changes from an order of 10^{-2} to 10^{-14}.

The results of the ACT method improved with the nonuniform FFT and IFFT algorithms are compared to the results of the local Lagrangian approximation, which assumes four points in the local neighborhood of the interpolation point. Although the numerical load due to the local Lagrangian interpolation and ACT is $O(N)$ and $O(N \log_2 N)$ repsectively, the error due to the ACT method is several orders of magnitude smaller when q and b are properly chosen.

An important remark would be one can discard the adaptive weights in the ACT algorithm for this problem as the samples in θ are almost uniformly distributed. In this case the condition number of T matrix is close to unity leading to rapid convergence of CG-FFT.

The overall numerical complexity of the MLFMA depends on the numerical complexities of the downward and upward passes. By applying the ACT algorithm on nonuniformly spaced samples in θ, the numerical complexity of the upward pass becomes $O(N \log_2 N)$, which is a significant improvement compared to $O(N^2)$. Although the interpolation in ϕ, which are uniformly distributed, has not been mentioned, it is obvious that the implementation of the uniform interpolation schemes locally yields $O(N)$ numerical complexity, which is less than $O(N \log_2 N)$, with accurate results.

REFERENCES

[1] Abramowitz, M., Stegun I. A.. Handbook of Mathematical Functions, New York: Dover Publications Inc., 1970, pp.363.

[2] Chew W.C., Lu C. C., Wang Y. M.. 'Efficient Computation of Three-Dimensional Scattering of Vector Electromagnetic Waves', J. Opt. Soc. Am., Vol. 11, No. 4, April 1994, pp. 1528-37.

[3] Coifman, R., Rokhlin V., Wandzura S.. 'Fast Multipole Method for the Wave Equation: Pedestrian Prescription', IEEE Trans. Antennas and Propagation, Vol. 35, No. 3, June 1993, pp. 7-12.

[4] Dutt A., Rokhlin V.. 'Fast Fourier Transforms for Nonequispaced Data', SIAM J. Sci. Comput., Vol. 14, No. 6, November 1993, pp. 1368-93.

[5] Engheta N., Murphy W.D., Rokhlin V., and Vassiliou M.S.. 'The Fast Multipole Method for Electromagnetic Scattering Problems', IEEE Trans. Antennas and Propagation., Vol. AP-40, No.6, June 1992 pp. 634-41.

[6] Epton M.A., Dembart B.. 'Multipole Translation Theory for the Three-Dimensional Laplace and Helmholtz Equations', SIAM J. Sci. Comput., Vol. 16, No. 4, July 1995, pp. 865-97.

[7] Feichtinger H.G., Gröchenig K.. Theory and Practice of Irregular Sampling, In : Benedetto, J. Frazier M., eds., Wavelets: Mathematics and Applications CRC Press, 1993, pp. 305-363.

[8] Feichtinger H.G., Gröchenig K., Strohmer, T.. 'Efficient Numerical Methods in Nonuniform Sampling Theory', Numerische Mathematik, Vol. 69, 1995, pp. 423-40.

[9] Greengard, L. and Rokhlin, V.. 'A Fast Algorithm for Particle Simulations', J. Comput. Phys., vol. 73, 1987, pp. 325-48.

[10] Gröchenig, K.. 'Acceleration of the Frame Algorithm', IEEE Trans. Signal Proc., Special Issue on Wavelets, Vol. 41, No.12, 1993, pp. 3331-40.

[11] Knab, J.. 'Interpolation of Bandlimited Functions Using Approximate Prolate Series', IEEE Trans. Inform. Theory, Vol. It-25, No. 6, November 1979, pp.717-20.

[12] Knab, J.. 'The Sampling Window', IEEE Trans. Inform. Theory, Vol. It-29, No.1, January 1983, pp. 157-9.

[13] Koc, S., Chew, W.C.. 'Calculation of Acoustical Scattering from a Cluster of Scatterers', J. Acoust. Soc. Am., Vol. 103, No. 2, February 1998, pp. 721-34.

[14] Koc S., Song J., and Chew W.C., 'Error Analysis for the Numerical Evaluation of the Diagonal Forms of the Scalar Addition Theorem', SIAM J. Numer. Anal., Vol. 36, No. 3, 1999, pp. 906-21.

[15] Kreyszig, E.. Advanced Engineering Mathematics, John Wiley & Sons Inc., 1993, pp. 937-9.

[16] Oppenheim, A., Willsky, A. S., Young, I. T.. Signals and Systems, Prentice/Hall International, Inc., 1983, pp. 514-26.

[17] Rokhlin, V.. 'Rapid Solution of Integral Equations of Scattering Theory in Two Dimenions', J. Comput. Phys., Vol. 86, 1990, pp. 414-39.

[18] Shewchuck, J. R.. 'An Introduction to the Conjugate Gradient Method without Agonizing Pain', School of Computer Science, Carnegie Mellon University Pittsburgh, August 1994.

[19] Song J.M., Chew, W.C.. 'Fast Multipole Method Using Parametric Geometry', Microwave Opt. Tech. Lett., Vol. 7, No. 16, November 1994, pp. 760-5.

[20] Strang, G., Chan, R. H.. 'Toeplitz Equations by Conjugate Gradients with Circulant Preconditioner', SIAM J. Sci. Stat. Comput., Vol. 10, No.1, January 1989, pp.104-9.

[21] Stratton , J.. Electromagnetic Theory, McGraw-Hill, New York, 1941, pp. 410.