A Spatially Resolved Study of the KATRIN Main Spectrometer Using a Novel Fast Multipole Method

by

John Patrick Barrett

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2017

Signature redacted

Author . . . .
Department of Physics
September 19, 2016

Signature redacted

Certified by . . . .
Joseph A. Formaggio
Professor of Physics
Thesis Supervisor

Signature redacted

Accepted by . . . .
Nergis Mavalvala
Associate Department Head, Physics

A Spatially Resolved Study of the KATRIN Main Spectrometer Using a Novel Fast Multipole Method

by

John Patrick Barrett

Submitted to the Department of Physics
on September 19, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

The Karlsruhe Tritium Neutrino (KATRIN) experiment is intended to make a sensitive ($\sim$ 200 meV) model-independent measurement of the neutrino mass through high precision electrostatic spectroscopy of the tritium β-decay spectrum. One of the principle components in this experiment is the main spectrometer which serves as an integrating MAC-E filter with $\mathcal{O}(1)$ eV resolution. Thorough understanding of the transmission properties of the main spectrometer system is an inextricable challenge associated with this effort, and requires a very accurate and fast method for calculating the electrostatic fields created within its volume. To this end, the work described in this thesis documents the development of a novel variation on the Fast Multipole Method (FMM), which is a hybrid of the canonical algorithm and the Fast Fourier Transform on Multipoles (FFTM) method. This hybrid technique has been implemented to take advantage of scalable parallel computing resources and has been used to solve the Laplace boundary value problem using the Boundary Element Method with millions of degrees of freedom. Detailed measurements taken during the KATRIN main spectrometer commissioning phase are used to validate the fully three-dimensional electrostatic field calculation and the hybrid fast multipole method. Then, the hybrid method is used to greatly accelerate charged particle tracking in a high-statistics Monte Carlo simulation. The data from this simulation is then used to develop a spatially resolved model of the main spectrometer transmission function. This full transmission function model is then used to evaluate the performance of several of approximate transmission function models, the results of which show that a purely axially symmetric treatment of the main spectrometer is not sufficient. We conclude by addressing the appropriate level of measurement detail needed in order to reconstruct a realistic, non-axially symmetric transmission function model.

Thesis Supervisor: Joseph A. Formaggio
Title: Professor of Physics

Acknowledgments

First of all, I would to thank my advisor Joe Formaggio, whose constant and thoughtful guidance helped me develop an understanding of what it means to do research. I especially need to thank you for always being happy to listen, for asking questions I'd never thought to ask myself, and for your unrestrained patience in shepherding me away from (and sometimes entertaining) the odd tangents on which I have occasionally found myself throughout this process.

I also want to thank my committee members Janet Conrad and Washington Taylor for their comments and critical eye, and the many people who have helped me along the way:

– T.J. Corona, for teaching me an incredible amount about coding and life in general, putting up with my sometimes indelicate touch when merging and modifying KEMField, and for always being willing to lend a hand when wrestling with a dirty Julabo.

– D. Furse, for his much appreciated camaraderie and insight on KATRIN and MIT, his patience in answering my sometimes confusing or ill-posed coding questions, and reminding me not to take things too seriously.

– N. Oblath, for his pleasant company in Seattle and Karlsruhe, help with the veto system, and ensuring the functionality of our workstation.

– A. Poon, for help in arranging access to the computational resources at NERSC needed to complete this thesis.

– My office-mates, especially R. Ott, A. Schmidt, B. Henderson, R. Russel, C. Epstein and J. Hardin for always providing a welcoming environment in which to commiserate, timely alerts about free food in the area, and the occasional, but much-needed, distraction from the day-to-day grind.

– S. Groh and M. Kleesiek who have always to be incredibly gracious hosts throughout many visits to Germany, and for their help in data taking and coding.

– M. Erhard, J. Behrens, P. Ranitzsch, M. Krauss, K. Wierman, and the rest of the control room and Münster electron gun crew who helped make the SDS (both phases) measurements successful, taking data without your help and good natured company would have been immensely more difficult and equally less enjoyable.

– F. Glück, D. Hilk and W. Gosda, for many interesting discussions about math, code, E&M, and for carrying the KEMField torch onward.

I also want to thank my parents John and Mary Barrett, whose continuous support and encouragement throughout my life have shaped who I am today. If I were to make a list of all the reasons to thank you, this acknowledgements section would likely be longer than the rest of this thesis, I would not have been able to do this without you!

Next I would like to thank the rest of my family:

– My sisters Bessie and Katie, for reminding me to take a break once in a while and always keeping the tea kettle at the ready.

– My godparents Nancy and Kevin Whittemore, who have always encouraged my interest in science and bailed us out of many a childcare crisis.

– My second set of parents, Chung and Yen Liu, who have treated me as one of their own, and whether it be meals, childcare or much needed time away from Cambridge, have always happily done their utmost to help out at every turn.

– James and Michelle Liu for always being willing to lend a helping hand, especially when things got stressful.

– My daughters Emma and Annabelle, for both reminding me of why I am doing this in the first place and for making sure I don't spend all my time glued to a computer screen and make it outside to get some fresh air.

Last but most definitely not least, I need to thank my wonderful wife, Cyndi, without whose unconditional love and support over the past eight years I never would have found my way to the end. Without your careful proof-reading I'm afraid this thesis would have been one, extremely long run-on sentence. This would have been an impossible task without you and I am so very blessed to have found you and had you by my side.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the richest sub-fields of modern particle physics is the study of the subatomic particle known as the neutrino. The variety of experiments dedicated to extricating the properties of this particle is astoundingly broad, which is due in part to the extraordinarily small likelihood of neutrino interactions with matter. While the study of the neutrino is challenging, since the capstone discovery of the Higg's boson, it remains one of the most active and promising frontiers in the search for physics beyond the standard model.

1.1 Theoretical Prediction of Neutrinos

The existence of the neutrino was first postulated in 1930 by Wolfgang Pauli in a letter as an explanation for the shape of the energy spectrum of beta-decay [184]. His postulate was a "desperate remedy" intended to save the bedrock principle of energy conservation. At the time, the process of beta-decay was assumed to be a two-body decay, which kinematically fixes the energies of the decay products. However, a three-body decay allows a distribution of energies among the decay products and can produce a non-monoenergetic spectrum. Of course, at the time, physicists where aware that the beta-decay spectrum was not mono-energetic [131], but they had been unsuccessful in detecting a third particle involved in the process. Pauli postulated that this third particle was electrically neutral and therefore had a low probability of interaction with any experimental apparatus. However, since this newly proposed particle was neutral and not a photon, it could not participate in an entirely electromagnetically mediated interaction. So by extension, Pauli had also proposed an entirely new force of nature. This new mediating force would later become know as the weak force.

Enrico Fermi, an Italian physicist who excelled at both theoretical and experimental work, devised a theory for β-decay as a point-like four particle interaction shortly after learning of Pauli's idea in 1934 [233]. It is Fermi who is responsible for the name "neutrino" as the diminutive of neutron [215] (the name which Pauli had proposed in his letter but had recently been expropriated for the particle discovered by Chadwick [43]). Fermi constructed his theory using creation and annihilation operators for the electron and neutrino in a way that was analogous to the existing theory of atomic photon emission. Incidentally, Fermi's paper was rejected from the journal *Nature* for being too speculative to be of interest [148], despite correctly reproducing the shape of the beta spectrum and accounting for the extreme spread in half-lives of known beta emitters though the concept of 'allowed' and 'forbidden' transitions [233].

Fermi's theory of the weak interaction would undergo further development throughout the next several decades with contributions from Gamow and Teller [87], Lee and Yang [153], and Feynman and Gell-Mann [80]. Eventually the weak force would be unified with the electromagnetic force by Glashow, Weinberg, and Salam to form the basis of the standard model of particle physics [95, 227, 207].

1.2 First Detection

The neutrino is notoriously difficult to study because it only interacts with matter via the weak force and gravity. Since neutrino's weak force cross section is exceedingly small it took nearly 25 years before direct experimental evidence of this particle was available. The first experiment to put the existence of the neutrino on solid footing was called Project Poltergeist [55]. Lead by C. Cowan and F. Reines, Poltergeist exploited the relatively recent invention of the nuclear fission reactor to provide a large flux of high energy anti-neutrinos. These ant-neutrinos could then be detected in a target composed of alternating layers of liquid scintillator and a solution of cadmium chloride dissolved in water. The process that Poltergeist was searching for was that of inverse β-decay:

$$\overline{\nu}_e + p \rightarrow n + e^+ \, , \tag{1.1}$$

through which a proton absorbs an anti-neutrino, converting into a neutron and emitting a positron. This process produces prompt scintillation through the annihilation of the positron, followed by a signal from the capture of the free neutron on a cadmium nucleus and its subsequent gamma emission. This type of event, exhibiting delayed coincidence, provided a unique signature which allowed them to significantly reject background due to other radiation from the reactor and cosmic rays which would have otherwise obscured the anti-neutrino induced events.

1.3 Discovery of Flavors in the Neutrino Sector

The decade following the first direct observation of the electron (anti-)neutrino was a good one for experimental neutrino physics and the development of the theory of the weak force. It was followed in rapid succession by Lee and Yang's proposal [153] of parity violation in 1956 and its discovery by Wu [235] in 1957. Later that same year, Goldhaber devised an amazingly beautiful table-top experiment to measure the helicity of neutrinos produced in the β-decay of Eu^{152} and found them to be left handed [101]. Meanwhile, the development of accelerators capable of reaching energies above the pion production threshold allowed high statistics observations of the decays of charged pions and muons, which had previously only been observed in cosmic ray events. From the observation of their decay branching ratios, the question as to whether the neutrinos produced in muon production and decay were the same particle as that involved in nuclear β-decay arose. To explore this question, an experiment making use of the first accelerator produced neutrino beam at the Brookhaven Alternating Gradient Synchrotron (AGS) [57] was carried out, resulting in the discovery of the muon neutrino. This discovery lead to confirmation of the suspicion that the different lepton 'flavors' exhibited by the electron and muon were also present among the neutrinos. The discovery of the τ lepton in 1975 [187] prompted a search for the corresponding τ-flavored neutrino. The ν_τ was finally observed by the DONUT collaboration 25 years later, filling out the standard model picture of the leptonic sector [141].

1.4 The Solar Neutrino Problem

For a short time in the early 1960s, it may have briefly seemed that the weak force theory was surprisingly consistent with the available experimental evidence. However, cracks were beginning to show in the contemporary understanding of neutrinos. This was particularly exacerbated by the Homestake mine experiment headed by R. Davis. The Homestake experiment was designed with the rather audacious goal of measuring the electron neutrino flux from the sun [59, 49]. To accomplish this, a massive 390,000 liter tank of tetrachloroethylene was placed nearly a mile underground in the Homestake gold mine in Lead, SD. The tetrachloroethylene contained large quantities of chlorine which served as the target of neutrino capture. The massive size of the target was necessary because of the extremely low probability of inverse β-decay process, while the thick overburden was needed to reduce the cosmic ray induced backgrounds. The inverse β-decay process of interest was neutrino capture on the Cl^{37} isotope present in the tetrachloroethylene. This would convert the nucleus to Ar^{37} along with the emission of an electron, according to:

$$\nu_e + \mathrm{Cl}^{37} \longrightarrow \mathrm{Ar}^{37} + e^- \ . \tag{1.2}$$

The Homestake experiment relied entirely on radio-chemical methods instead of real time observation of the neutrino capture events. The Ar^{37} atoms were collected by bubbling helium gas through the liquid, yielding about one Ar^{37} every two days. Since Ar^{37} is radioactive with a half-life of about 35 days, its decay was then detected by a proportional counter, with each event corresponding to a neutrino capture. Strangely, the number of Ar^{37} atoms collected, and thus the neutrino flux seen by the detector, were only about a third of the rate predicted by the standard solar model of J. Bahcall [20]. This discrepancy became known as the solar neutrino problem.

Following the Homestake experiment, the water Cherenkov detectors Kamiokande and its enormous successor, Super-Kamiokande, were the next to observe a deficit in the solar neutrino rate [178]. Originally designed for a proton decay search, it was realized that the low background and excellent ability to resolve low energy electrons in these detectors could provide a means to observe neutrino-electron scattering from solar neutrinos. These detectors were a significant improvement over the radiochemical methods of Davis, because real time event observation allowed an active veto and provided additional information, such as directionality, which could be used to further reduce background processes. However, rather than exactly confirming the results of Davis's search, Kamiokande instead observed about half the expected flux, which further complicated the puzzle.

One explanation for the solar neutrino problem was provided by the development of the theory of neutrino oscillations, which proceeded in parallel with experiment throughout the middle of the 20th century [177]. The existence of neutrino oscillations allows the neutrinos generated by the fusion reactions in the Sun's interior to convert to other flavors as they travel on their way to the Earth. Since the flavors (ν_e, ν_μ, ν_τ) do not interact with the matter composing the detectors in exactly the same fashion, oscillations allow for the "disappearance" of some of the solar neutrino flux. However, these oscillations can only exist if neutrinos have mass, and specifically, only if their mass eigenstates are different from their flavor eigenstates.

Experimentally, the solar neutrino problem was finally resolved by the SNO collaboration [6]. Since the energy of solar neutrinos is only a few MeV or lower, the ν_τ and ν_μ flavored neutrinos cannot participate in the charged current reaction, as they do not have enough energy to produce a μ or τ particle. The Homestake and (Super-)Kamiokande detectors were only sensitive to these charged-current reactions so τ and μ flavored neutrinos were undetectable. However, SNO was also sensitive to neutral-current reactions through its use of a heavy water, D_2O, target. This is because in addition to the processes of inverse β-decay and neutrino-electron scattering, heavy water is also subject to the dissociation of its deuterium atoms, through [196]:

$$\nu_x + D \rightarrow p + n + \nu_x \ . \tag{1.3}$$

Because of its very low threshold (2.2 MeV), all neutrino flavors can participate in this process. By measuring the flux in both the neutral and charged-current channels, the SNO collaboration was able to demonstrate that the deficit in the solar neutrino flux was in fact due to the oscillation of ν_e into ν_μ and ν_τ.

1.5 Neutrino Oscillations

In 1958, Pontecorvo [191] was the first to propose an oscillation model in the neutrino sector. At the time it was still not known that there were multiple flavors of neutrino. Rather than flavor oscillations, Pontecorvo's first model was devised in order to explain the existence of events of the form:

$$\overline{\nu_e} + \mathrm{Cl}^{37} \longrightarrow \mathrm{Ar}^{37} + e^- \ , \tag{1.4}$$

which he erroneously believed had been observed by Ray Davis [36]. However, as more experimental evidence was accumulated, (particularly the distinct nature of the flavors ν_e and ν_μ [57]) a flavor oscillation model was developed by Maki, Nakagawa, and Sakata [166] and subsequently developed into the form we are familiar with today [37].

A thorough description of the mechanism of neutrino oscillations can be found in [94, 105], but we will give a brief summary following the notation of [94]. Neutrino oscillations rely on the weak interaction eigenstates, $|\nu_a\rangle = (\nu_e, \nu_\mu, \nu_\tau)$, being an admixture of the free space propagation eigenstates, $|\nu_k\rangle = (\nu_1, \nu_2, \nu_2)$, which have different masses. This mixing is described by the the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) matrix U, through [94]:

$$|\nu_a\rangle = \sum_k U^*_{ak}|\nu_k\rangle \ , \tag{1.5}$$

$$|\nu_k\rangle = \sum_a U_{ak}|\nu_a\rangle \ . \tag{1.6}$$

In the three neutrino model, U can be parameterized as a unitary matrix in the commonly used form of [171]:

$$U = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_{23} & \sin\theta_{23} \\ 0 & -\sin\theta_{23} & \cos\theta_{23} \end{bmatrix} \begin{bmatrix} \cos\theta_{13} & 0 & \sin\theta_{13}e^{i\delta_{cp}} \\ 0 & 1 & 0 \\ -\sin\theta_{13}e^{i\delta_{cp}} & 0 & \cos\theta_{13} \end{bmatrix} \begin{bmatrix} \cos\theta_{12} & \sin\theta_{12} & 0 \\ -\sin\theta_{12} & \cos\theta_{12} & 0 \\ 0 & 0 & 1 \end{bmatrix} \ . \tag{1.7}$$

This parameterization can be viewed as a series of three rotations, along with the presence of an additional complex CP-violating phase δ_{cp}. If neutrinos are in fact Majorana particles, there is also the possibility of two additional complex phases. In which case, we have $U \longrightarrow U \times \mathrm{diag}(1, e^{i\gamma_2}, e^{i\gamma_3})$ [171].

Under the assumption that neutrinos are produced in a flavor eigenstate and freely propagate as a plane wave, the neutrino's flavor content will evolve in time according to the Schrödinger equation:

$$i\hbar\frac{d}{dt}|\nu(t)\rangle = \mathcal{H}|\nu(t)\rangle \ . \tag{1.8}$$

The mass eigenstates are the stationary states of the free particle Hamiltonian, so they evolve with a simple energy dependent phase according to:

$$|\nu_k(t)\rangle = \exp\left(-i\frac{E_k}{\hbar}t\right)|\nu_k(t)\rangle \ . \tag{1.9}$$

Combining 1.6 and 1.9 implies a neutrino which is produced at time $t = 0$ in the state:

$$|\nu(t=0)\rangle = \nu_a\ , \tag{1.10}$$

which, as an admixture of several mass states, can be described at some later time t by [94]:

$$|\nu(t)\rangle = \sum_k U^*_{ak} \exp\left(-i\frac{E_k}{\hbar}t\right)|\nu_k\rangle\ . \tag{1.11}$$

Therefore, given a beam of neutrinos generated in the state ν_a, there is some non-zero probability to observe a neutrino, ν_b, of a different flavor, $a \neq b$, which is given by:

$$P_{\nu_a \longrightarrow \nu_b}(t) = |\langle \nu_b|\nu(t)\rangle|^2\ . \tag{1.12}$$

In the ultra-relativistic limit, $m_k \ll E_k$, the energy is dominated by the neutrinos momentum, $E \approx |p|c$, so E_k can be expressed to first order in the mass as:

$$E_k = \sqrt{p^2c^2 + m_k^2c^4} \approx E\left(1 + \frac{m_k^2c^4}{2E^2}\right)\ . \tag{1.13}$$

Similarly, as the neutrino is moving near the speed of light, c, time may be replaced by $t \approx L/c$. Therefore, the transition probability can be approximated quite well by [94]:

$$P_{\nu_a \longrightarrow \nu_b}(L/E) = \sum_{k,j} U^*_{ak}U_{bk}U_{aj}U^*_{bj} \exp\left(-i\frac{\Delta m^2_{kj}c^3L}{2\hbar E}\right)\ , \tag{1.14}$$

where Δm^2_{kj} is the mass squared difference of the two propagation eigenstates ν_k and ν_j, given by:

$$\Delta m^2_{kj} = m_k^2 - m_j^2\ . \tag{1.15}$$

As a matter of terminology, if an experiment is designed to observe neutrinos of the flavor ν_b from a source emitting ν_a, it is termed an "appearance" experiment, while if it is designed to measure the deficit of the expected flux of ν_a, it is called a "disappearance" experiment.

Over the past several decades, there has been an enormous effort to extract the parameters describing the PMNS matrix , involving a variety of techniques and wide ranging length and energy scales. The observation of solar neutrinos by the SNO experiment has produced an extremely precise measurement of the mass splitting Δm_{21}, [24]. Atmospheric neutrino oscillations are sensitive to the Δm_{31} mass splitting [136]. Examinations of reactor produced neutrinos and long baseline accelerator experiments have complemented atmospheric and solar neutrino measurements. This has allowed precision measurements of the mixing angles and the mass differences [41]. A summary of these parameters is given in table 1.1.

Name	Value	3σ lower limit	3σ upper limit
θ_{12}	$33.72°$	$31.52°$	$36.18°$
θ_{23}	$49.3°$	$38.6°$	$53.1°$
θ_{13}	$8.47°$	$7.86°$	$9.1°$
Δm^2_{21} (eV^2)	7.49×10^{-5}	7.02×10^{-5}	8.08×10^{-5}
Δm^2_{31} (eV^2)	2.484×10^{-3}	2.351×10^{-3}	2.618×10^{-3}

Table 1.1: Parameters of the PMNS matrix and neutrino mass differences from oscillation measurements. Values given are the best fit for the three neutrino normal hierarchy model of NuFit 2.1 2016 [104].

The observation of neutrino oscillations provides strong evidence for the existence of non-zero neutrino mass and indicates that at least one of the propagation eigenstates must have a mass greater or equal to the largest mass splitting:

$$m_{\nu_{\text{max}}} \geq \sqrt{|\Delta m^2_{31}|} \,. \tag{1.16}$$

However, while the introduction of neutrino mass and oscillations has lead to the resolution of the solar neutrino problem, it has also opened the door to new questions. The current state of knowledge of physics in the neutrino sector leaves us with the following open questions:

1. What is the mechanism by which neutrino mass is produced? Are they Dirac or Majorana particles?

2. Is the mass hierarchy normal $(m_1, m_2 < m_3)$, inverted $(m_3 < m_1, m_2)$, or degenerate $(m_1 \simeq m_2 \simeq m_3)$?

3. What is the value of the CP-violating phase δ_{cp}?

4. To what extent is the three neutrino PMNS matrix, U, unitary? Are there sterile neutrino flavors? If so, how many are there?

5. What is the absolute mass scale of the neutrinos?

These questions can only be informed through more experimentation and remain one of the most promising avenues in the quest for new physics.

The answer to the first question can only be definitively answered in favor of Majorana particles with the observation of neutrino-less double β-decay. Within the standard model, the existence of two neutrino double beta ($2\nu\beta\beta$) decay as a second order weak process is a well established fact, and has been demonstrated to exist for a variety of isotopes [197]. However, while $2\nu\beta\beta$-decay provides a useful extension to our series of tests of the current model of nuclear structure, it does not provide any instruction on the possibility of physics beyond the standard model. One exciting possibility which might give a glimpse into new physics, and one which is predicted by several classes of grand unified theories [238], is that of zero neutrino double beta ($0\nu\beta\beta$) decay. So far, the experimental search for $0\nu\beta\beta$-decay has failed to yield any indisputable evidence for its existence [1, 138, 237]. If $0\nu\beta\beta$-decay were to be observed, this phenomenon would give a clear indication of the nature of neutrinos as Majorana particles, as well as allowing us to establish an estimate of the absolute mass scale of the neutrino.

To answer the second question, there are several competing methods for the determination of the neutrino hierarchy (see [123], [164], and [156]). In the near term, one promising technique may be that of the Noνa and T2K collaborations, which are using long base line ν_μ neutrino beams to look for $\nu_\mu \longrightarrow \nu_e$ oscillation [19, 90]. In addition, since U_{13} has been measured to be non-zero[3, 9], the passage of a neutrino beam through matter can provide some sensitivity (by way of the MSW effect) to the sign of Δm_{13} [60]. This will enable another generation of oscillation experiments currently getting under way to probe the mass hierarchy, as well as the value of the CP-violating phase δ_{cp} [174, 209].

Measurements of the Z boson width and its invisible decay fraction has constrained the number of active light neutrinos to the currently known three flavors [62]. However, this does not preclude the existence of 'sterile' neutrinos which may not participate in W,Z-boson mediated interactions. The observation of the reactor neutrino anomaly [172] and anomalies in disappearance experiments [143] both suggest that the presence of an additional "sterile" neutrino flavor(s) is a distinct possibility. The resolution of these anomalies awaits further experimental results, and a variety of techniques to explore the existence of sterile neutrinos is underway [2]. Since the presence of a massive sterile neutrino would modify the shape of the β-decay spectrum, one interesting possibility is the use of the KATRIN experiment to explore this effect [83].

The last question cannot be addressed through oscillation experiments as they are only sensitive to the mass differences. Instead, it can only be probed indirectly through cosmological measurements, or directly from kinematic measurements. The mass scale of the neutrinos is a question of utmost importance since it can inform models of the structure formation of the universe and may provide an indication of the energy scale at which the standard model breaks down as an effective theory [228].

1.6 Probes of the Neutrino Mass scale

While there is currently little doubt that neutrinos have non-zero mass, direct experimental searches have so far failed to demonstrate the sensitivity needed to make a discovery. This leaves us unable to make a distinction between the several competing candidate theories which propose how to incorporate neutrino mass into the standard model. The two most prominent mechanisms for producing massive neutrinos incorporate them into the standard model in either a semi ad-hoc manner as Dirac fermions, or as Majorana particles through the see-saw mechanism [175]. Between these two mechanisms, the see-saw mechanism is generally seen as more compelling, since it provides a natural explanation for the lightness of the neutrino masses.

Whether neutrinos are Majorana has been an ongoing line of inquiry throughout the 20th century and in some ways is a more fundamentally important question than that of determining the absolute mass scale. However, proof of the Majorana mechanism can only be provided through the observation of $0\nu\beta\beta$-decay. The converse, however, is not true, and the absence of the direct observation of $0\nu\beta\beta$-decay does not necessarily indicate that neutrinos are Dirac particles. In fact, if this turns out to be the case, the Majorana-Dirac determination can only be made possible if both a stringent lower limit on $0\nu\beta\beta$-decay lifetimes is made in conjunction with a definitive measurement of the absolute neutrino mass scale [21].

1.6.1 Cosmology

One approach to determine the mass scale of the neutrinos is through the use of cosmological observations. Since the discovery of the cosmic microwave background [186], subsequent satellite based measurements [33, 118] have allowed a more refined understanding of the matter density anisotropies in the early universe dating back to the epoch of recombination. Understanding the evolution of these early anisotropies into the large scale galactic structures we observe later in the universe's history requires knowledge of several parameters describing the matter content and energy density of space. These parameters dictate the evolution of the universe and structure formation according to Einstein's field equations and Newtonian dynamics. This parameterization and the behavior of each constituent is collectively referred to as the ΛCDM model. This is on account of the dominance of dark energy (Λ), and cold (non-relativistic) dark matter (CDM) which heavily influence structure formation. In this model, baryonic matter is assigned only a small fraction ($\sim 5\%$) of the energy density of the universe, along with the cosmic microwave background and cosmic relic neutrinos.

Cosmic relic neutrinos exist as an analog of the cosmic microwave background. They are remnants leftover from the time when matter had cooled enough for the weak force to decouple. Recent measurements by the Planck collaboration have put a best limit on the sum over all neutrino mass eigenstates of [5]:

$$\sum_{\nu} m_{\nu} < 0.23 \ \text{ eV} . \tag{1.17}$$

However, this limit is determined through a complex analysis of many correlated parameters and varies considerably depending on the choice of data sets used in the analysis. Such a limit is termed model-dependent, since it relies on the choice of a large number of free parameters which are not well constrained. Furthermore, given the nature of astrophysical observation, where the system under observation cannot be controlled in a way so as to explore systematic errors or the presence of unknown observational biases, then existence of such a limit does not make laboratory tests dispensable. So while astrophysical data provides much needed insight into areas which are otherwise inaccessible to experiment, they are not yet sufficient by themselves to definitively constrain the neutrino mass scale.

1.6.2 Single and Double β-Decay

It is fitting that the same process which lead to the neutrino hypothesis in the first place would also be an appropriate way to probe their mass scale. In order to briefly review review single β-decay, we will restrict ourselves to β^--decay, and ignore the similar β^+-decay, and electron-capture processes. Single β-decay occurs in nuclei, ${}^A_Z X_N$, which have a mass excess greater than m_e of the corresponding daughter nucleus, ${}^A_{Z+1}Y_{N-1}$, it proceeds according to:

$${}^A_Z X_N \longrightarrow {}^A_{Z+1} Y_{N-1} + e^- + \bar{\nu}_e . \tag{1.18}$$

The differential rate with respect to the emitted electron's energy at which this process occurs is dictated by Fermi's golden rule [193]:

$$d\Gamma = \frac{2\pi}{\hbar}|\mathcal{M}|^2 \frac{d\rho(E_{tot}, E_e)}{dE_e} dE_e , \tag{1.19}$$

where $|\mathcal{M}|$ is the matrix element describing the transition amplitude of the relevant dynamics of the decay process, $\rho(E_{tot}, E_e)$ is the differential element of phase space, E_e is the energy of the emitted electron, and E_{tot} is the total energy available for the phase space of the electron and neutrino. In the simplified nucleon picture of β-decay, this matrix element can be factored into nuclear and leptonic parts under the assumption that the nuclear part does not depend upon the electron's state (true for super-allowed and allowed decays). We will not go into detail on the

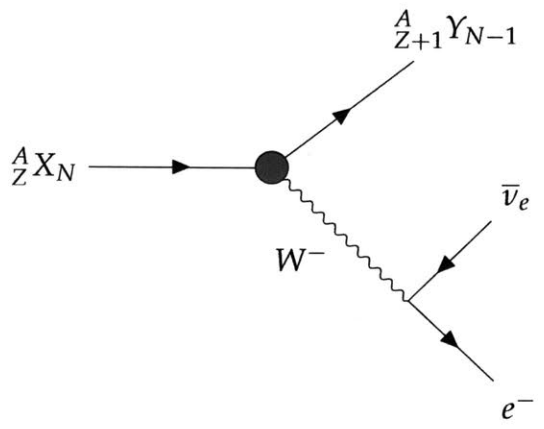


Figure 1-1: The β^- decay of the nucleus ${}^{A}_{Z}X_{N}$. The blob represents the effective weak interaction involving the entire nucleus.

construction of the transition amplitude from the Feynman diagram in figure 1-1, but will instead quote the result [193], which has been factored into these two parts:

$$d\Gamma = \frac{2\pi}{\hbar} |\mathcal{M}_{nuc}|^2 |\mathcal{M}_{lep}|^2 \frac{d\rho(E_{tot}, E_e)}{dE_e} dE_e \ . \tag{1.20}$$

The leptonic matrix element essentially boils down to the probability of finding the electron and anti-neutrino at the same place within the nucleus, as proposed in Fermi's original point-like interaction theory. Assuming the wave functions of the electron and anti-neutrino to be plane waves normalized over the volume of the nucleus, the leptonic matrix element is given by [181]:

$$|\mathcal{M}_{lep}|^2 = \frac{1}{V^2} F(E_e, Z+1) \ , \tag{1.21}$$

where V is the volume of the nucleus and $F(E_e, Z+1)$ is the so-called Fermi function which corrects for the interaction of the electron with the Coulomb charge of the daughter nucleus. The Fermi function is given by [38]:

$$F(E_e, Z+1) = \frac{2\left(\frac{(Z+1)\alpha}{\beta_e}\right)\pi}{1 - \exp\left(-\left(\frac{(Z+1)\alpha}{\beta_e}\right)\pi\right)} , \tag{1.22}$$

where $\beta_e = v_e/c$, α is the fine structure constant, and $Z+1$ is the nuclear charge of the daughter. The nuclear matrix element, $|\mathcal{M}_{nuc}|^2$, is a measure of the overlap between the final and initial nuclear states. It contains two contributions:

$$|\mathcal{M}_{nuc}|^2 = g_v^2|\mathcal{M}_F|^2 + g_a^2|\mathcal{M}_{GT}|^2 . \tag{1.23}$$

That which is due to the vector part of the weak interaction, $\mathcal{M}_F$, is known as the Fermi matrix element, and that which is due to the axial-vector part $\mathcal{M}_{GT}$, is known as the Gamow-Teller matrix element. The relative strength of the Fermi and Gamow-Teller elements is dictated by the vector and axial-vector coupling constants; g_v and g_a. As might be expected, a transition involving the change of a nucleon from a neutron to a proton involves the isospin raising operator. In fact, in the case of free neutron decay, the matrix elements are simply:

$$|\mathcal{M}_F| = |\langle p|\tau_+\mathbb{I}|n\rangle| , \tag{1.24}$$

$$|\mathcal{M}_{GT}| = |\langle p|\tau_+\sigma|n\rangle| . \tag{1.25}$$

However, when we are in the confines of a nucleus, we must sum over all the nucleons which can participate in the reaction, so the matrix elements become:

$$|\mathcal{M}_F| = \sum_{m_f}\left|\langle f|\sum_k^A \tau_+(k)\mathbb{I}|i\rangle\right| , \tag{1.26}$$

$$|\mathcal{M}_{GT}| = \sum_{m_f}\left|\langle f|\sum_k^A \tau_+(k)\sigma(k)|i\rangle\right| , \tag{1.27}$$

where m_f is the z-projection of the nuclear angular momentum, and $|i\rangle$ and $\langle f|$ are the initial and final nuclear states.

The form of the nuclear matrix elements in equations 1.26 and 1.27 appear quite simple (though it hides the true details of the quark structure), and in some cases they are fairly easy to calculate. In the case of a super allowed decay, the decaying neutron in the parent nucleus has the same quantum numbers (n, l, j) as the produced proton in the daughter nucleus. Therefore, the remaining nucleons do not participate and the matrix element essentially reduces to the free neutron case [193]. A particularly important example of a super allowed decay is that of the isospin doublet:

$$^3_1H_2 \longrightarrow \, ^3_2He_1 + e^- + \bar{\nu}_e \, . \tag{1.28}$$

However, the vast majority of β-decaying nuclei do not participate in super allowed decay. This is generally on account of the Coulomb repulsion of the protons, which raises their energy and allows for a neutron excess to be present in nuclei of appreciable size. For nuclei which decay through allowed or forbidden processes, the calculation of the spectral shape becomes more complicated through additional shape factors. These shape factors depend on the linear and angular momentum of the emitted electron and neutrino [145]. However, computing the nuclear matrix elements in these nuclei cannot always rely on a simple treatment of the decay operator as an isospin raising/lower operator, because decays in heavier nuclei often involve a transition between nucleon shells or the production of an excited state in the daughter nucleus. In the general case, the process to compute nuclear matrix elements consists of the following: solving for the single-nucleon states in some effective potential, constructing the nuclear wave-function, constructing the Fock Space of nuclear states, expressing the decay operators in terms of single particle annihilation/creation operators acting on this Fock space, and finally, evaluating the single particle transition matrix elements. The full procedure is beyond the scope of this introduction, but its complexity underscores the advantage of using much simpler super-allowed decays when probing the neutrino mass from the shape of the β-decay spectrum.

In single β-decay, it is the spectral shape of the emitted electron's energy distribution that allows us to probe the neutrino mass. However, so far neither the leptonic nor nuclear contributions to the matrix element have provided a term with any dependence on the neutrino mass. In fact, the neutrino mass enters entirely through kinematical constraints on the available phase space of the emitted electron. The phase space term to equation 1.20 is given by [193]:

$$d\rho(E_{tot}, E_e) = V^2 \frac{(4\pi)^2}{(2\pi\hbar)^6} p_e^2 \left(\frac{dp_e}{dE_e}\right) p_\nu^2 \left(\frac{dp_\nu}{dE_{tot}}\right) dE_e \,. \tag{1.29}$$

The phase space of the daughter nucleus does not contribute, as it is fixed by energy and momentum conservation. Furthermore, the recoil energy with which it is imparted is very small since its mass isn't much larger than the electron and neutrino. In the region of interest near the end-point this energy, E_{rec}, varies very little and can be treated as a constant, modifying the total energy, E_0, available to the electron and neutrino. The modified end-point is then given by [181]:

$$E_0 = Q - E_{rec} \tag{1.30}$$

where Q is the total energy available to all products of the decay. Applying the relativistic energy-momentum relation, rewriting the phase space in terms of the electrons kinetic energy and combining it with the nuclear and leptonic matrix elements yields the final description of the shape of the β-decay spectrum near the end-point:

$$\begin{aligned} \frac{d\Gamma}{dK_e} = &\frac{g_v^2|\mathcal{M}_F|^2 + g_a^2|\mathcal{M}_{GT}|^2}{2\pi^3 c^6 \hbar^7} \\ &\times F(E_e, Z+1) E_e p_e (E_0 - K_e) \sqrt{(E_0 - K_e)^2 - m_{\nu_\beta}^2 c^4} \,. \end{aligned} \tag{1.31}$$

Note that equation 1.31 treats the interacting neutrino as if it were a particle with a single mass, m_{ν_β}. Including the full set of mass eigenstates modifies the differential decay rate so that it becomes [94, 83]:

$$\frac{d\Gamma}{dK_e} = \frac{g_v^2|\mathcal{M}_F|^2 + g_a^2|\mathcal{M}_{GT}|^2}{2\pi^3 c^6 \hbar^7} F(E_e, Z+1) E_e p_e (E_0 - K_e)$$
$$\times \sum_{i=1}^{N_\nu} |U_{ei}|^2 \left[\Theta(E_0 - K_e - m_i) \sqrt{(E_0 - K_e)^2 - m_i^2} \right] , \quad (1.32)$$

where $\Theta(E_0 - K_e - m_i)$ is the Heaviside step function restricting the phase space to the physical region. Clearly, an experiment with an energy resolution that is much larger than the differences between the mass eigenstates cannot resolve the influence of any single m_i. Instead, it sees the influence of the coherent sum, given by [181]:

$$m_{\nu_\beta}^2 = \sum_{i=1}^{N_\nu} |U_{ei}|^2 m_i^2 , \quad (1.33)$$

where U_{ei} is the element of the PMNS matrix dictating the mixing between the electron neutrino and the i-th neutrino mass eigenstate. It is this value that we refer to as the "neutrino mass" in reference to limits obtained from β-decay. The current best limit on the neutrino mass from single β-decay comes from tritium and is held by the Mainz experiment which yielded a limit of $m_{\nu_\beta} < 2.2$ eV at the 95% confidence level [146].

The neutrino mass can also be probed through the study of double β-decay. The process of $2\nu\beta\beta$-decay is a second order weak process predicted by the standard model. It was first predicted by Goeppert-Mayer in 1935 [99]. However, because of the smallness of the weak force coupling constant, second order processes are exceedingly rare (with $t_{1/2} \gtrsim 10^{18} - 10^{22}$ years [93]) and were not observed until more than 50 years after their prediction [74].

Under the model dependent assumption that neutrino's are in fact Majorana particles, with $\bar{\nu} = \nu$, the $2\nu\beta\beta$-decay spectrum is modified by a neutrino-less double β-decay process, given by:

$${}^A_Z X_N \longrightarrow {}^A_{Z+2} Y_{N-2} + e^- + e^- . \quad (1.34)$$

Of course, this decay is only possible if the neutrino is a massive Majorana particle that is able to serve as both the "anti-neutrino" the "neutrino" at both vertices in figure 1-2. In order for this decay to proceed, the parent nucleus ${}^A_Z X_N$ and the

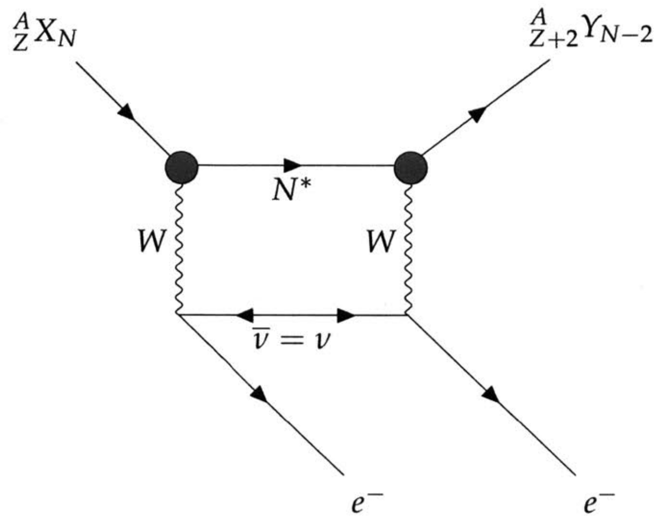


Figure 1-2: Neutrinoless $\beta\beta$-decay mediated by massive majorana neutrino, $\bar{\nu} = \nu$. The virtual nuclear state is denoted by N^*.

daughter ${}^A_{Z+2} Y_{N-2}$ must both be more bound than the intermediate nucleus. This is usually satisfied in the situation where the parent and daughter nuclei are both even-even, and the intermediate nucleus is odd-odd [76]. Once again, the decay rate is dictated by Fermi's golden rule, and according to Elliott, [76] reduces to:

$$\Gamma = 2\pi \int \sum_{spins} |R_{0\nu}|^2 \delta(\epsilon_1 + \epsilon_2 + E_f - M_i) \frac{d\vec{p}_1}{(2\pi)^3} \frac{d\vec{p}_2}{(2\pi)^3} , \tag{1.35}$$

where M_i and E_f are the mass-energy of the intial and final state nuclei respectively, ϵ_i and $\vec{p}_i$ are the energy and momenta of the i-th final state electron, and $|R_{0\nu}|^2$ is the transition amplitude. Fortunately, the phase space integral over the electron final states factors out of the transition amplitude $|R_{0\nu}|^2$ [76]. However, the leptonic portion of the transition amplitude does not completely separate from the nuclear matrix element due to the presence of the neutrino propagator. The leptonic component contains terms of the form [76]:

$$-i\int \frac{d^4q}{(2\pi)^4}e^{-iq(x-y)}\bar{e}(x)\gamma_\rho\left(\frac{1-\gamma_5}{2}\right)\left(\frac{q^\mu\gamma_\mu+m_j}{q^2-m_j^2}\right)\left(\frac{1-\gamma_5}{2}\right)\gamma_\sigma e^c(y)\ , \quad (1.36)$$

where $\bar{e}(x)$ and $e^c(y)$ are the electron creation operators, q is the virtual neutrino four-momentum, and m_j is the mass of the j-th neutrino mass eigenstate. The term involving $q^\mu\gamma_\mu$ drops out once we have applied Casimir's trick in summing over the spins, because γ_5 anti-commutes with γ_μ, and because terms involving the trace of an odd number of γ-matrices are zero, as well as terms of the form $Tr(\gamma_\mu\gamma_\nu\gamma_5)$. This leaves the transition amplitude proportional to the square of a linear combination of the neutrino mass eigenstates, known as the effective Majorana neutrino mass [76]:

$$\langle m_{\nu_{\beta\beta}}\rangle^2 = \left|\sum_j m_j U_{ej}^2\right|^2\ . \quad (1.37)$$

This indicates that measuring or placing a limit on the rate of $0\nu\beta\beta$ decay is directly sensitive to the absolute mass scale of the (Majorana) neutrino (but not the shape of the spectrum). However, there are additional complications when determining the rate which arise from evaluating the nuclear matrix element. This is because of the virtual nuclear state N^*. The full description of these issues is beyond the scope of this introduction, but they lead to a wide range in theoretical values for the nuclear matrix elements. In fact, a review of the calculations for nuclear matrix elements in the decay of the single isotope Ge^{76} shows that they vary by a factor of ~ 3, which leads to large uncertainties on the mass limits obtained from double β-decay [75]. The current best limit on $\langle m_{\nu_{\beta\beta}}\rangle^2$, which comes from a combined analysis is [113]:

$$\langle m_{\nu_{\beta\beta}}\rangle^2 < 0.13 - 0.31\ \text{ eV}\ . \quad (1.38)$$

This limit is of course only valid if neutrinos are in fact Majorana particles.

Table 1.2 summarizes the current neutrino mass limits available from cosmology, $0\nu\beta\beta$-decay, and single β-decay. Each of these methods is sensitive to a somewhat different mass term, which complicates a direct comparison between each of them. However, as there is currently no definitive claim of a non-zero mass discovery (other than oscillations), there is a continuing effort on all fronts to develop more sensitive experiments to explore the neutrino mass.

Method	Mass term	Current limit	Reference
Cosmology	$\sum_i m_i$	< 0.23 eV	[5]
$0\nu\beta\beta$-decay	$\sqrt{\left\|\sum_i m_i U_{ei}^2\right\|^2}$	$< 0.13 - 0.31$ eV	[113]
β-decay	$\sqrt{\sum_i \|U_{ei}\|^2 m_i^2}$	< 2.2 eV	[146]

Table 1.2: Current best limits on the absolute mass scale of the neutrino from various techniques

Chapter 2

The KATRIN Experiment

The Karlsruhe tritium Neutrino (KATRIN) experiment is intended to further our understanding of the neutrino sector by making a model independent measurement of the absolute neutrino mass scale. Its planned sensitivity is an order of magnitude better than the current best limit of $\sim$ 2.2 eV [146, 15]. It is important to note that the KATRIN experiment, being a Tritium β-decay spectroscopy experiment, is insensitive to the exact neutrino mass mechanism, since it is solely dependent on the decay kinematics. In addition, unlike cosmological observations which involve a plethora of variables, direct measurements like KATRIN only require a minimal number of orthogonal parameters describing the shape of the β-decay spectrum in order to extract the neutrino mass.

To accomplish this, KATRIN will examine the energy spectrum of tritium β-decay with unprecedented precision, in order to look for any tell-tale distortion near the end-point, indicative of a non-zero neutrino mass. On account of the small size of this effect, KATRIN must attain an energy resolution on the order of 1 eV, while imaging a high intensity gaseous molecular tritium source and maintaining a low background rate of less than 10mHz [51]. While the basic technique of the KATRIN experiment mirrors the approach of previous neutrino mass searches at Mainz [29] and Troitsk [161], its sheer size brings a number of new challenges which must first be resolved.

2.1 Tritium β-Decay

Tritium β-decay makes an excellent probe of the neutrino mass and has been a mainstay of model independent searches for neutrino mass for the past 60 years [149, 208, 34, 232, 146, 15]. The reasons for this are several fold. The first is that it is a super-allowed decay, making the matrix elements of the nuclear transition independent of the energy of the emitted electron. In addition, the tritium half life of 12.3 years [163] is short enough to allow a source with high rates, yet long enough to enable high statistics measurements over a time scale of years. Furthermore, tritium has the second lowest end-point energy, $E_0 = 18.6$keV, of any β-decay. This is important, since it maximizes the proportion of the spectrum which is sensitive to the neutrino mass, which scales like $\propto 1/E_0^3$ [94]. In addition, for some required absolute energy resolution ΔE (which is dictated by the neutrino mass scale), a lower end-point energy permits a less demanding relative energy resolution, $\Delta E/E_0$. This is an important consideration for electrostatic spectrometers, which must maintain high voltages within a narrow stability range.

However, with better energy resolution comes greater sensitivity to effects due to the final state of the daughter system. These modify the shape of the spectrum near the end-point because of molecular excitations. The influences of such final states can have a deleterious effect on neutrino mass sensitivity when they are poorly understood and on the same order as the spectrometer resolution. In the past, this has lead to unreproducible claims of a non-zero neutrino mass discovery. An example of which is the $\sim$ 30 eV neutrino observed by the Moscow group [165]. This experiment used a tritiated valine source which had a very complex and difficult to calculate final state distribution. While this claim eventually proved unfounded, it spurred the use of better understood tritium sources in subsequent neutrino mass searches.

Of the predecessors of the KATRIN experiment, the earliest to use a gaseous tritium source was the Los Alamos experiment [201, 232], which placed an upper limit on the neutrino mass of < 9.3 eV in the late 1980s. The novel use of a gaseous molecular T_2 source is particularly notable, since it reduces the spectral broadening due to the final state distribution and eliminates complications due to solid state effects. While the Los Alamos experiment used a Tretyakov type spectrometer similar to that of the Moscow experiment [165], the subsequent experiments at Troitsk [161] and Mainz [146] were the first tritium β-decay experiments to use spectrometers of the MAC-E type (Magnetic Adiabatic Collimation with Electrostatic Filter). The Mainz experiment, however, used a tritium film quench condensed on highly-oriented pyrolytic graphite in favor of a more complex gaseous source. The Troitsk experiment was instead equipped with a windowless gaseous tritium source as well as a MAC-E filter, making it something of a direct ancestor of the KATRIN experiment.

Neglecting relativistic and radiative corrections (see [170] for the full relativistic treatment), the differential decay rate of molecular tritium can be described through a summation of the simple (multi-neutrino) spectrum of equation 1.32 over all the final states. The final states modify the end-point as follows [66]:

$$\frac{d\Gamma}{dK_e} = \frac{g_v^2|\mathcal{M}_F|^2 + g_a^2|\mathcal{M}_{GT}|^2}{2\pi^3 c^6 \hbar^7} \sum_j P_j F(E_e, Z+1) E_e p_e (E_j - K_e) \times \sum_{i=1}^{N_\nu} |U_{ei}|^2 \left[\Theta(E_j - K_e - m_i) \sqrt{(E_j - K_e)^2 - m_i^2} \right] , \quad (2.1)$$

where the sum over j is over all final states which have an end-point $E_j = E_0 - \epsilon_j$, and P_j is the probability of occurrence of the j-th final state. The probability of an excitation with energy, ϵ_j, occurring is typically calculated in the so-called sudden approximation, which assumes that probability is given solely by the overlap in the electronic wave functions of the initial T_2 molecule and the resulting He^3T ion [66]. The spectrum of discrete final states in molecular T_2 decay according to the model of Saenz et. al. [206] is shown in figure 2-1. Precise knowledge of the final states spectrum is of critical importance for KATRIN and a large effort is currently underway to study them theoretically and experimentally [39].

Of course, barring the existence of sterile neutrinos with large mass and non-

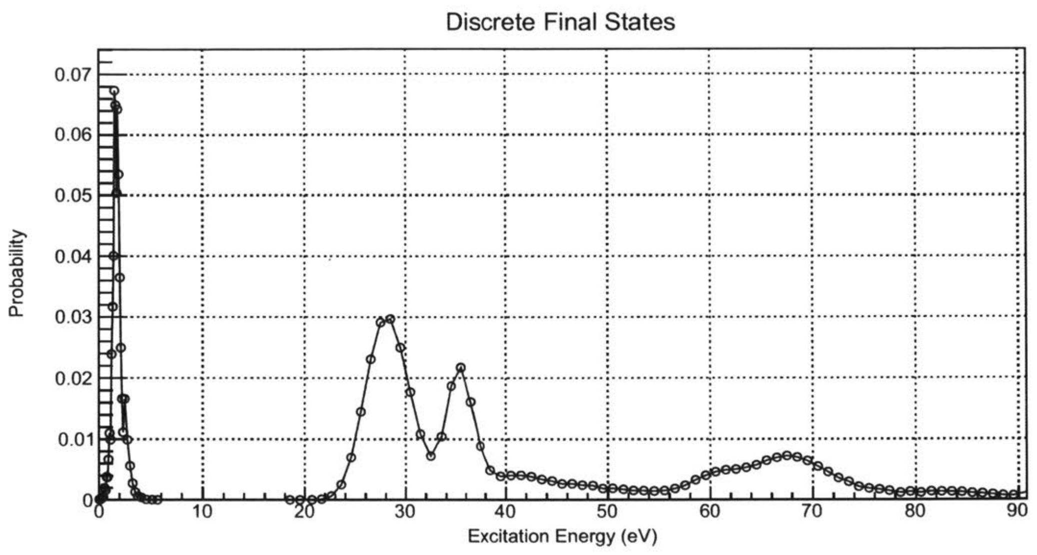


Figure 2-1: The spectrum of discrete final states of the He^{3}T daughter ion in T$_2$ β-decay. Model shown is from the calculation of Saenz et al. [206].

vanishing coupling [173], the KATRIN experiment cannot resolve the influence of any single m_i but only the effective mass m_{ν_β}. However, the fact that KATRIN is sensitive to the absolute neutrino mass scale through the coherent sum of equation 1.33 is very advantageous, since it is entirely model independent and makes no assumptions about whether neutrinos are Majorana or Dirac particles. Furthermore, unlike $0\nu\beta\beta$-decay based searches, the effective mass, m_{ν_β}, has no dependence on the the complex phases of the matrix elements, U_{ei}, which could, in theory, allow cancellations that may make the measurement of $m_{\nu_{\beta\beta}}$ difficult or impossible [234].

Unfortunately, the fraction of β-decay events emitted in the region sensitive to the neutrino mass is exceedingly small, being on the order of 10^{-13} as shown in figure 2-2 [51]. Therefore, any experiment which aims to measure the neutrino mass through tritium β-decay must be able to image a source with very high luminosity.

2.2 Basic Operating Principles (MAC-E Filter)

Since a high luminosity is necessary to gain enough statistics in the sensitive region of the spectrum, a spectrometer which can accept β-decay electrons over a large

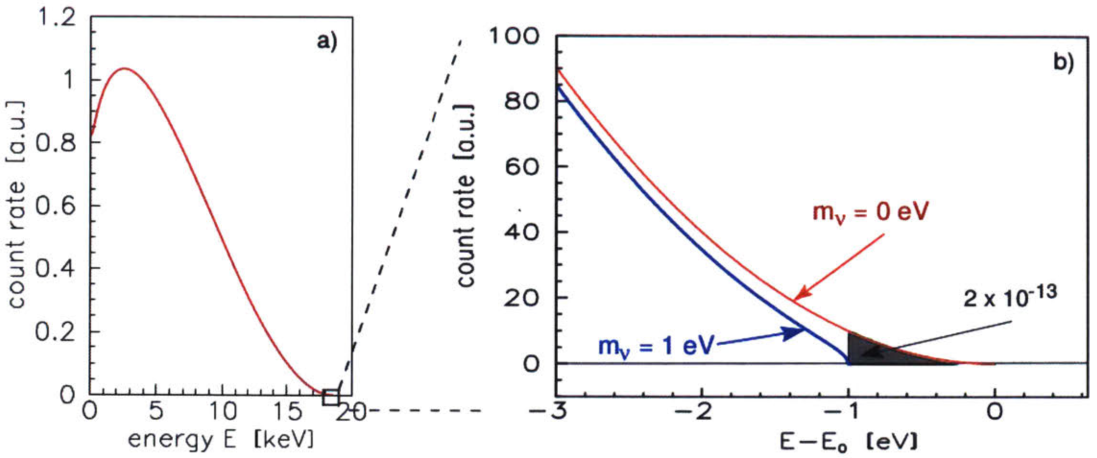


Figure 2-2: The fraction of events in the sensitive portion of the spectrum of tritium β-decay. Figure taken from [51].

solid angle is essential. One such type is the Magnetic Adiabatic Collimation with Electrostatic filter (MAC-E) spectrometer, initially proposed for photo-electron spectroscopy [31] and first applied to the problem of probing the anti-neutrino rest mass by Troitsk [162]. The basic operating principle behind the MAC-E filter relies on the conservation of the orbital magnetic moment, μ, of a charged particle in a magnetic field. In the non-relativistic limit, μ, is given by [181]:

$$\mu = \frac{mv_\perp^2}{2|\mathbf{B}|} = \frac{E_\perp}{|\mathbf{B}|}, \tag{2.2}$$

where $E_\perp$ is the kinetic energy associated with the particle's transverse motion with respect to the direction of the magnetic field $\mathbf{B}$. When the particle traverses a slowly varying magnetic field which satisfies the condition [181]:

$$\frac{1}{|\mathbf{B}|}\frac{d|\mathbf{B}|}{dt} < \omega_c = \frac{q|\mathbf{B}|}{m}, \tag{2.3}$$

the orbital magnetic moment becomes the first adiabatic invariant of its motion. The conservation of μ can then be exploited to collimate the momentum of particles emitted from the source, so that their energy may be analyzed by electrostatic means. This can be accomplished by slowly reducing the magnetic field from a large magnitude in the source region, $\mathbf{B}_s$, to a low value in the analyzing region, $\mathbf{B}_a$, as depicted in figure 2-3.

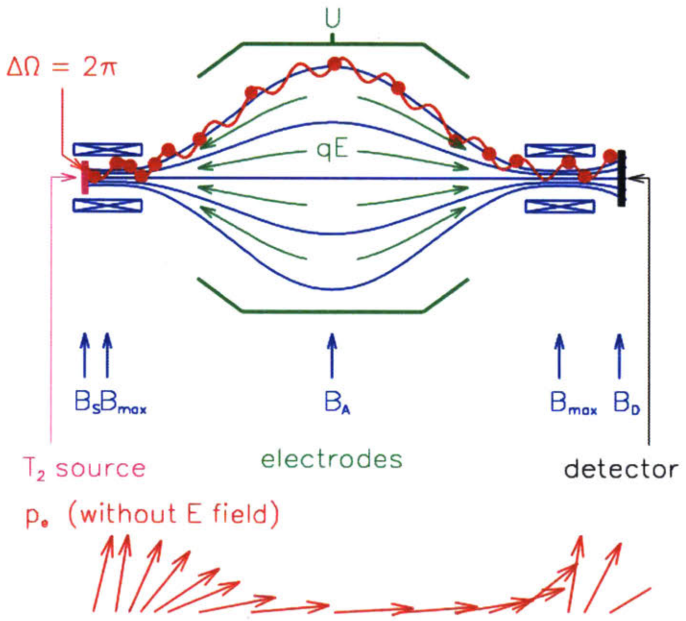


Figure 2-3: Basic operating principle of a MAC-E filter spectrometer. The momentum of the particles emitted from the source is adiabatically transformed by the slowly varying magnetic field, so that the longitudinal energy can be analyzed by the electrostatic field. Figure taken from [51].

From the conservation of equation 2.2, we may equate the orbital magnetic moment in the source and analyzing regions:

$$\frac{E_{\perp a}}{|\mathbf{B}_a|} = \frac{E_{\perp s}}{|\mathbf{B}_s|}\,, \tag{2.4}$$

$$\Longrightarrow E_{\perp a} = \frac{|\mathbf{B}_a|}{|\mathbf{B}_s|} E \sin^2 \phi_s\,. \tag{2.5}$$

Since adiabatic collimation is used to align the particles momentum against that of the electric field, only the transverse component of the particle's energy is unanalyzable. This implies that the irreducible fractional energy resolution of such a spectrometer is given by:

$$\frac{\Delta E}{E} = \frac{E_{\perp a}}{E} = \frac{|\mathbf{B}_a|}{|\mathbf{B}_s|} \sin^2 \phi_s \leq \frac{|\mathbf{B}_a|}{|\mathbf{B}_s|} \,. \tag{2.6}$$

In practice, it is often helpful to ensure that the maximum magnetic field encountered by the particle, $\mathbf{B}_{\text{max}}$, is not located in the source region. This modifies the energy resolution to be:

$$\frac{\Delta E}{E} = \frac{|\mathbf{B}_a|}{|\mathbf{B}_{\text{max}}|} \,. \tag{2.7}$$

This is done so that we may reject particles that have a high pitch angle, ϕ_s, in the source region. These particles have a greater path length in the source and an increased probability of scattering, which increases the uncertainty on their original energy. In KATRIN, the maximum magnetic field is provided by the pinch magnet located just before the focal plane detector (FPD) so as to reject these high-pitch angle particles through the magnetic mirror effect. Therefore, the fractional energy resolution of KATRIN is roughly:

$$\frac{\Delta E}{E} = \frac{3 \times 10^{-4} T}{6T} = 0.5 \times 10^{-5} \,, \tag{2.8}$$

which near the tritium end-point of 18.6 keV, yields an absolute energy resolution of approximately 0.93 eV.

The selection of particles which do reach the detector is governed by the transmission function of the spectrometer. For an ideal spectrometer and a completely isotropic source, the transmission function can be described analytically in terms of the particle's kinetic energy, E, charge, q, and the spectrometer potential, U, by [51]:

$$T(E, qU) = \begin{cases} 0 & \text{if } E - qU \leq 0 \,, \\ \frac{1-\sqrt{1-\frac{E-qU}{E} \cdot \frac{B_S}{B_a}}}{1-\sqrt{1-\frac{\Delta E}{E} \cdot \frac{B_S}{B_a}}} & \text{if } 0 \leq E - qU \leq \Delta E \,, \\ 1 & \text{if } E - qU > \Delta E \,. \end{cases} \tag{2.9}$$

Clearly, the MAC-E filter is an integrating spectrometer, so care must be taken to reduce any low energy backgrounds which may exist between the analyzing region and the detector, as these will be indistinguishable from the source electrons. This disadvantage can be mitigated through the use of a MAC-E time-of-flight mode as described in [214]. However, this mode of operation brings some additional challenges which have yet to be resolved. In addition, while the transmission function of an ideal spectrometer may appear quite simple, this expression is only valid assuming a fixed value for U and B_a. However, the analyzing potential, U_a, and magnetic field, B_a, can and do vary over the flux tube which images the source. These inhomogeneities can adversely affect the energy resolution of the spectrometer if they are not measured or modeled accurately. Fortunately, they can be compensated to some degree through the discrete pixelation of the detector.

2.3 The KATRIN Beam line

The KATRIN beam line consists of a long collection of modules designed to isolate those rare decay events which probe the scale of the neutrino mass. Collectively, the beam line is approximately 70 meters long and consists of four main parts: the source section, transport and pumping section, spectrometers, and detector region. It is shown in figure 2-4.

The KATRIN experiment's tritium source is based on the Windowless Gaseous Tritium Source (WGTS) concept that was pioneered by [232] and [161, 15]. The WGTS serves to localize a dilute gas of T_2 molecules while allowing β-decay electrons to escape to the spectrometers guided along the magnetic field. The gas is maintained at a steady temperature of $\sim$ 30K by a dual phase Neon gas-liquid refrigeration system [111]. The T_2 gas is continuously pumped and re-injected to keep a uniform column density of 5×10^{17} molecules/cm^2. This density must be maintained with a stability of 0.1% [51] and is monitored by the rear-wall section. Before re-injection, the tritium gas passes through a recycling system in order to maintain the strength and purity of the source. The fraction of other hydrogen isotopologues present is monitored by a Laser Raman scattering system in order

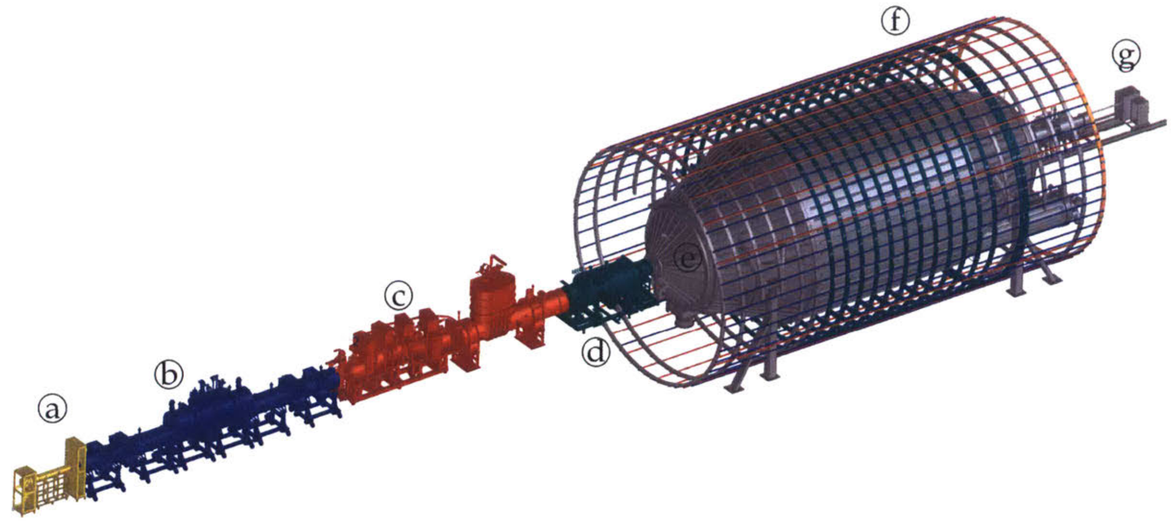


Figure 2-4: The complete KATRIN Beam line. The rear-wall section is denoted by (a), (b) is the WGTS, (c) is the transport and pumping section, (d) is the pre-spectrometer, (e) is the main spectrometer, (f) is the magnetic field shaping system and (g) is the detector section. Image adapted from [50].

to reduce the systematics associated with these impurities [82]. Another relevant systematic of the source section is the knowledge of the energy loss function from electrons scattering off the T_2 gas as they exit the source. The energy loss function has been measured at Troitsk [17], but an additional measurement program at KATRIN will be necessary to meet its demanding design sensitivity.

The transport and pumping section consists of two main parts: the Differential Pumping Section (DPS) and the Cyrogenic Pumping Section (CPS). The differential pumping section uses a long beam pipe bent into a chicane, coupled with large aperture turbo-molecular pumps in order to reduce the remaining tritium gas by several orders of magnitude. The CPS then further reduces the gas load through the use of an Argon frost coated tube to absorb any remaining tritium in order to reach the final vacuum level required by the spectrometer section.

After the transport and pumping section comes the pre-spectrometer. The pre-spectrometer's primary purpose is to reduce the flux of β-decay electrons entering the main spectrometer. This diminishes ionization of any residual gas molecules which remain in the large main spectrometer volume. Ionization leads to low energy secondary electrons being emitted in the analyzing region. These can mimic signal electrons, so it is essential to reduce this process.

The main spectrometer is a massive stainless steel vacuum chamber approximately 10 meters in diameter and 24 meters in length. Such large dimensions are necessary in order to accommodate the expansion of the magnetic field as it decreases from 6 T to roughly 0.3 mT in magnitude. The vessel is pumped through three large ports using a combination of turbo-molecular pumps and Non-Evaporable-Getter (NEG) strips and must be baked out to a high temperature of $\sim 300°\text{C}$ in order to attain the design pressure of $< 10^{-11}$ mbar [12]. The main spectrometer is surrounded by the Large Field Compensation System (LFCS), which is a series of axially symmetric ring magnets designed to shape the field so that it is both confined within the spectrometer and has an appropriately located minimum. In addition to the LFCS is the Earth's Magnetic field Compensation System (EMCS), which consists of a series of linear current elements that serve to cancel out the Earth's magnetic field. This is necessary because the Earth's magnetic field would otherwise seriously distort the flux tube. The electrode system of the main spectrometer serves to apply and shape the electric potential for the energy analysis of the incident electrons. It consists of many double and single-layer wire array modules, as well as shaping electrodes designed to mitigate Penning traps. A detailed description can be found in [222] and [236]. Figure 2-5 shows the construction of the electrodes in the main spectrometer, exhibiting the combs upon which the wire arrays are strung and the support structures upholding each module. The use of wire arrays instead of solid electrodes reduces the electrically active surfaces exposed to the interior volume of the spectrometer and protects this volume from low energy backgrounds arising from the vessel hull. Unfortunately, as a result of the high vacuum bake out process some electric shorts appeared between some of the individual wire modules and wire layers. This was caused by the deformation of several of the copper-beryllium distribution rods [221]. While some of the short circuits have been repaired, the remaining shorts may necessitate the main

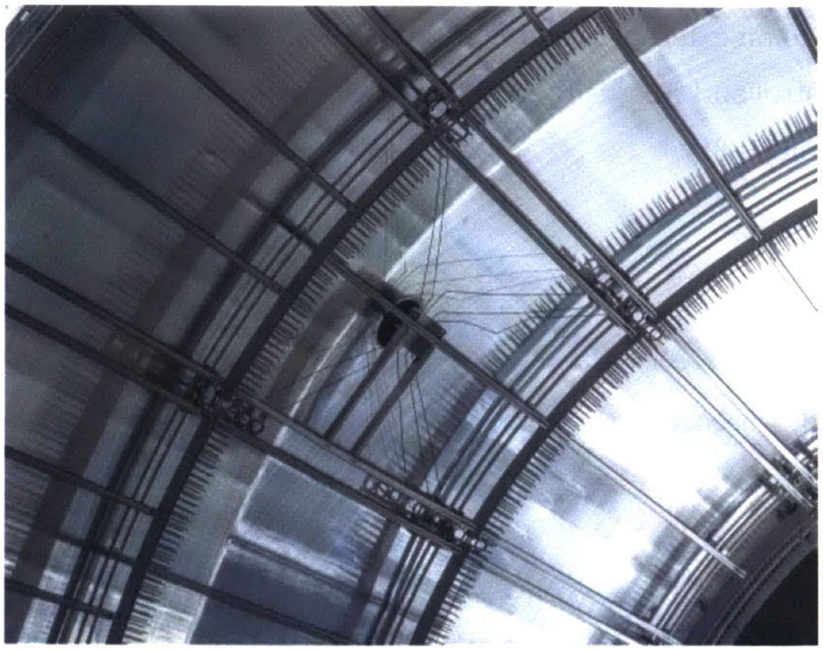


Figure 2-5: Close up of the electrodes inside the main spectrometer. The CuBe distribution rods are visible emerging from a vacuum port just behind the wire array. Figure taken from [12].

spectrometer being utilized in a mode where the double-layer wire arrays must be operated at a single voltage. This might increase backgrounds originating from the vessel hull but also alters the spatial homogeneity of the electrostatic field from its original design.

At the end of the beam line lies the Focal Plane Detector (FPD) system, shown in figure 2-6. It registers electrons which have passed through the MAC-E filter. It consists of a 148 pixel silicon PIN diode detector array, roughly 9 cm in diameter, arranged into 12 rings of 12 pixels each, along with a central 4 pixel "bullseye" [8]. The FPD is preceded by a cylindrical post-acceleration electrode which increases incident electron energy by roughly 10keV. This aids in decreasing the background, reducing the likelihood of backscattering, and also improves the energy resolution. The signals produced from the PIN diode array are then amplified and sent by an optical link from the high voltage region to the Digital Acquisition (DAQ) system. The DAQ then digitizes the raw signal at 20MHz and passes it through a trapezoidal filter in order to provide a triggering signal [8]. Triggered events are then collected and exported to an external machine running the software ORCA (Object-oriented Real-time Control and Acquisition), where they are packaged and saved to disk for later processing [188].

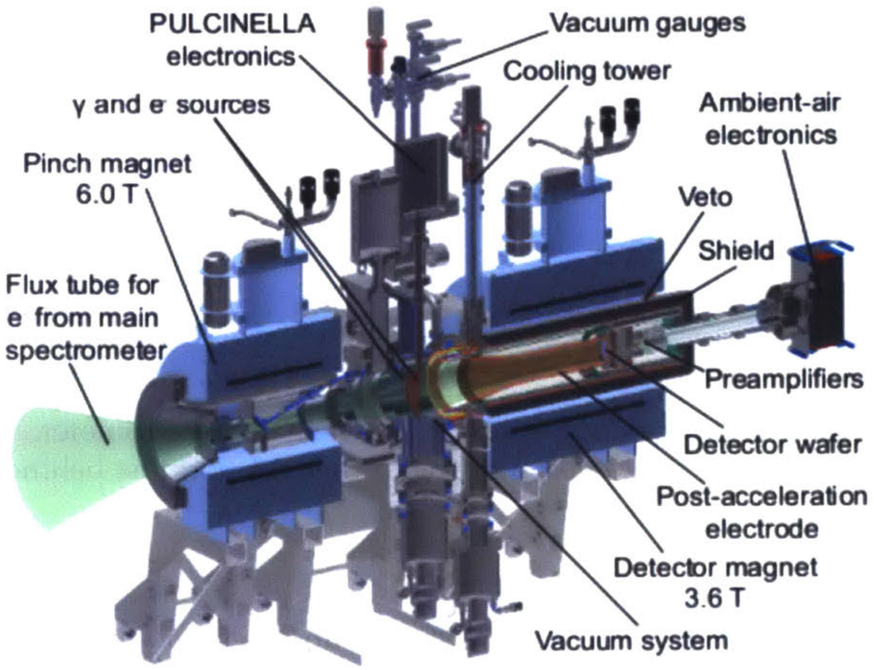


Figure 2-6: The components of the focal plane detector system. Image taken from [8].

2.4 Simulation of the KATRIN Experiment

In order to understand the systematic uncertainties involved in a measurement performed by a device as complex as the KATRIN experiment, a very detailed Monte Carlo simulation package is required. This package must be able to accurately model the electromagnetic fields involved, propagate particles through those fields and apply any discrete stochastic interactions which may occur during propagation. To this end, the KATRIN collaboration has developed the C++ software package `Kassiopeia` [85, 212].

Kassiopeia is the modular and extensively configurable (through a very flexible XML interface [85]) front-end of the KATRIN simulation code. Its primary responsibility is solving the equations of motion of particles propagating through the experiment. However, it also serves as an interface through which a user can access various sub-programs. Kassiopeia is supported by a large collection of such sub-programs which are designed for a variety of specific purposes. These include many basic tasks such as approximating the solid geometry of the experiment (KGeoBag) or calculating the electromagnetic fields (KEMField) [53], as well as more specialized projects such as modeling the gaseous tritium source section (SSC) [119, 135, 139], or the silicon detector's response (KESS) [199].

One of the most basic dependencies of Kassiopeia is the geometry library KGeoBag. This package is responsible for the basic physical modeling of the experimental components and answering shape, location, and navigation queries about them. KGeoBag also handles the Boundary Element Method (BEM) mesh generation for use by the field package KEMField. The common dependence on KGeoBag across modules allows for a consistent representation of the geometry by multiple tools.

The purpose of the field solving package KEMField is to compute the electrostatic and magnetostatic fields produced by the charges and currents of the experimental apparatus. In order to supply information about the electrostatic fields, KEMField must first solve the Laplace boundary value problem in order to determine the configuration of charges. Since the run time of all charged particle simulations in the Kassiopeia package is dominated by the computation of the electrostatic field, it is of primary importance that this portion of the code be both accurate and extremely fast. In order to satisfy these two goals, the KEMField package has been augmented with a new fast multipole method. This method has been developed so that a fully realistic three dimensional field model of the KATRIN system can be simulated within a reasonable time frame.

Chapter 3

The Laplace Equation and the Boundary Element Method

The fundamental theory of classical electromagnetism is described by Maxwell's equations in conjunction with the Lorentz force law. Maxwell's equations, in vacuum, are given in differential form by:

$$\nabla \cdot \mathbf{E} = \rho/\epsilon_0 \,, \tag{3.1}$$

$$\nabla \cdot \mathbf{B} = 0 \,, \tag{3.2}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \,, \tag{3.3}$$

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \epsilon_0 \mu_0 \frac{\partial \mathbf{E}}{\partial t} \,, \tag{3.4}$$

where **E** and **B** are the electric and magnetic fields respectively, ρ is the charge density and **J** is the current density, all of which are explicit functions of position and time. The Lorentz force law tells us that a particle with charge q and velocity **v** experiences a force **F**, due to the electric and magnetic fields at its position, given by:

$$\mathbf{F} = q(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \,. \tag{3.5}$$

In short, Maxwell's equations state the how electric and magnetic fields are generated from the spatial distribution of charged particles and their currents, while in turn, the Lorentz force law tells us how the motion of charged particles is influenced by the fields surrounding them. These equations, coupled with relativistic kinematics can, in principle, describe every classical electromagnetic interaction. However, while these equations represent a complete and consistent theory, they are merely the starting point for exploring an incredibly rich variety of phenomena and computational techniques.

3.1 The Laplace equation

While the full description provided by Maxwell's equations is always correct classically, there remain much simpler approximations to the full system of equations that are still applicable in many situations. For example, in the case of the KATRIN experiment, we can treat all of the charge and current sources of the spectrometer fields as being completely static. This ignores any contribution to the fields from the charged particles that we wish to track through the experiment, but this is a negligible correction. In this approximation, Maxwell's equations reduce to:

$$\nabla \cdot \mathbf{E} = \rho/\epsilon_0 \, , \tag{3.6}$$

$$\nabla \cdot \mathbf{B} = 0 \, , \tag{3.7}$$

$$\nabla \times \mathbf{E} = 0 \, , \tag{3.8}$$

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} \, . \tag{3.9}$$

Solving for the magnetic field is quite simple since in the static case (with no magnetizable materials) the magnetic field can be computed directly from the current sources using the Biot-Savart law (which follows directly from (3.7) and (3.9) in the Coulomb gauge):

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int \frac{\mathbf{J}(\mathbf{r}') \times (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^3} d^3 r' \, . \tag{3.10}$$

Since the placements and magnitudes of the current sources are directly controllable and measurable within the context of a physical experiment, the Biot-Savart law is sufficient for the calculation of all the relevant magnetic fields. It should be noted that for axially symmetric sources, such as solenoids and loops, the magnetic field can be computed much more quickly than the naive numerical evaluation of Biot-Savart. This is because axially symmetric sources can be easily approximated using zonal harmonic expansions. A full description of this technique can be found in [97] and [54].

From the point of view of an experimentalist, solving for the electric field is a somewhat more complicated problem. Like the Biot-Savart law, the electrostatic field can be computed directly from charged sources via Coulomb's law:

$$\mathbf{E}(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int \frac{\rho(\mathbf{r}')(\mathbf{r}-\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3} d^3r' \,, \tag{3.11}$$

where the integral is taken over all space. However, unlike current sources, we have no direct control over the placement of free charges within the experiment. We only control the voltages on various electrode surfaces. From equations (3.6) and (3.8), it follows that the electric field can be written as the gradient of a scalar potential Φ:

$$\mathbf{E} = -\nabla\Phi \,, \tag{3.12}$$

and that Φ obeys what is known as the Poisson equation:

$$\nabla \cdot (\nabla\Phi) = \nabla^2\Phi = -\rho/\epsilon_0 \,. \tag{3.13}$$

For regions which contain no free charge ($\rho = 0$), this becomes the so-called Laplace equation:

$$\nabla^2\Phi = 0 \,. \tag{3.14}$$

Using the Laplace equation to compute the electric fields of the KATRIN experiment is appropriate, since with the possible exception of the source region, there is negligible space charge build up and all charge sources are confined to surfaces.

To compute the electric fields, we must first specify the boundary conditions on the border of the region where we wish to determine the field. Once the boundary conditions are specified, the Laplace equation uniquely determines the potential Φ everywhere.

Since the geometry of the KATRIN experiment is sufficiently complex to make an analytic solution impossible, it is necessary to solve the Laplace equation using a numerical approach. There are two classes of numerical techniques that are commonly used for solving the Laplace equation on a computer. These are the finite difference method and the boundary element method.

The finite difference method is possibly the most widely known technique; brief summaries of which can be found in [129] and [71]. This method is at first enticing because of its ability to calculate Φ directly from the quantities we experimentally control (the potentials on electrode surfaces). Also, the application of this method generally results in a sparse system of linear equations which can be efficiently solved using a variety of iterative techniques (e.g. relaxation methods). However, it is unlikely to be capable of modeling KATRIN with sufficient accuracy. The reasons for this are severalfold. The first is that the finite difference method requires a volume discretization. This discretization must be sufficiently fine grained that it is possible to resolve the smallest electrically active features on the boundary, while also covering the entire volume of interest. Since the smallest features of the KATRIN main spectrometer are on the order of micrometers[1], while the volume of the spectrometer is roughly 1400 m^3 [51], the memory requirement for such a volume discretization is on the order of a petabyte for a naive uniform grid. Even if the volume were adaptively meshed, memory usage could easily reach tens to hundreds of terabytes. An algorithm with such large memory requirements is clearly not feasible on modern computer hardware. Secondly, the finite difference method only accurately determines the potential at the mesh points of the volume discretization. Since in order to track charged particles we must know the potential at any arbitrary point in space, this implies we need to interpolate the field between

[1]The wires of the inner array are roughly 200 μm in diameter.

grid points which can result in a loss of accuracy. Furthermore, in order to determine the electric field, we must compute a numerical derivative from the potential values which results in a further loss of accuracy. Therefore, despite its initially appealing simplicity, the finite difference method is wholly inadequate for the task of modeling the electric field of the KATRIN experiment.

The boundary element method is an alternative approach which allows us to avoid a volume discretization in favor of a boundary mesh. This is immediately appealing since it reduces the dimensionality of the problem and results in a much lower memory requirement. Additionally, the boundary element method allows us to solve for the source charges instead of a derived quantity such as the potential. Knowing the source charges allows us to compute the electric field directly without needing to rely on an accuracy impairing interpolation or numerical differentiation step. However, as we will see in the following sections, these advantages come at the cost of a dense (rather than sparse) system of linear equations which requires a substantially different set of tools in order to solve efficiently.

3.2 The Laplace Boundary Value Problem

To specify the problem we are trying to solve, it is necessary to introduce a more precise description of the input data and the desired solution as follows. Let Ω be a compact subset of $\mathbb{R}^3$ whose boundary is the orientable manifold $\Gamma = \partial\Omega$. We wish to find a function, $\Phi(\mathbf{r}) : \mathbb{R}^3 \to \mathbb{R}$, which satisfies the Laplace equation $\nabla^2\Phi(\mathbf{r}) = 0$, for all points $\mathbf{r} \in \Omega$. In order to obtain a solution for Φ, we are required to specify the boundary conditions for all points $\mathbf{r} \in \Gamma$.

The boundary conditions can be specified though the use of either Dirichlet or Neumann conditions. To define Dirichlet boundary conditions, we must prescribe a function that the potential Φ must match on the surface Γ:

$$\Phi(\mathbf{r}) = D(\mathbf{r}), \ \forall\, \mathbf{r} \in \Gamma. \tag{3.15}$$

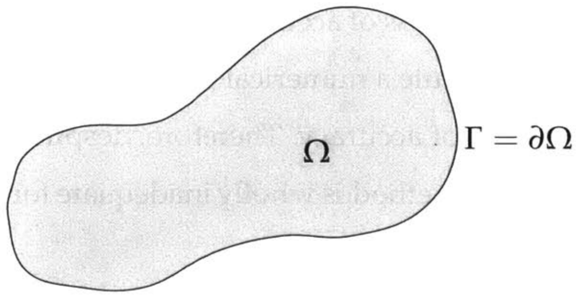


Figure 3-1: A two-dimensional projection of the domain of interest $\Omega \subset \mathbb{R}^3$, and its boundary Γ.

For Neumann boundary conditions, we must define a function for the normal derivative of Φ over the surface:

$$\nabla\Phi(\mathbf{r}) \cdot \hat{\mathbf{n}} = \left.\frac{\partial\Phi}{\partial n}\right|_{\mathbf{r}} = N(\mathbf{r}), \;\; \forall\, \mathbf{r} \in \Gamma. \tag{3.16}$$

It should be noted that in the case of pure Neumann boundary conditions, the solution for Φ is only unique up to a globally constant value. This is, however, unimportant since this does not affect the physically observable field $\mathbf{E}$ which is what determines charged particle motion. It is also possible to specify mixed Dirichlet-Neumann boundary conditions. That is to say, for some $\{D\}$, and $\{N\}$, where $\{D\},\ \{N\} \subset \Gamma$, and $\{D\} \cup \{N\} = \Gamma$ we may define:

$$\Phi(\mathbf{r}) = D(\mathbf{r}), \;\; \forall\, \mathbf{r} \in \{D\} \tag{3.17}$$

and

$$\left.\frac{\partial\Phi}{\partial n}\right|_{\mathbf{r}} = N(\mathbf{r}), \;\; \forall\, \mathbf{r} \in \{N\}. \tag{3.18}$$

Another possible boundary condition is that of the Robin type, which is a restriction on the value of a linear combination of Φ and $\frac{\partial\Phi}{\partial n}$ on Γ such as:

$$\alpha\Phi(\mathbf{r}) + \beta \left.\frac{\partial\Phi}{\partial n}\right|_{\mathbf{r}} = R(\mathbf{r}), \;\; \forall\, \mathbf{r} \in \Gamma. \tag{3.19}$$

However, we will not consider the Robin case further, as it represents impedance conditions, and in our idealized electrostatic model, we treat all of the metallic surfaces as perfect conductors (Dirichlet conditions), and all of the insulating surfaces as perfect (linear) dielectrics (Neumann conditions).

For the aforementioned types of boundary conditions, it is well known that solutions for the Laplace equation exist and are unique [129]. We should also note that imposing a fourth type of boundary condition, that of the Cauchy type, results in an ill-posed problem for the Laplace equation. Cauchy conditions require simultaneously fixing a value for both Φ and $\frac{\partial \Phi}{\partial n}$ on Γ. Since the solution to the Laplace equation with Dirichlet boundary conditions exists and is unique, separately specifying Neumann conditions for the same section of boundary would either be superfluous or result in an overdetermined problem with no solution.

3.3 The Boundary Element Method

In order to develop a computational procedure for determining the solution to the Laplace boundary value problem (LBPV), we first need to convert the governing partial differential equation (PDE) into a boundary integral equation (BIE). We will favor an approach which appeals to physical arguments, but more mathematically rigorous derivations can be found in [190], [160], [53], or [132].

To transform the PDE into a BIE, we start with the Laplace equation $\nabla^2 \Phi(\mathbf{r}) = 0$ and multiply it by an as of yet unspecified function $G(\mathbf{r}, \mathbf{r}') : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$, and then integrate over the domain Ω:

$$\int_\Omega \nabla^2 \Phi(\mathbf{r}') G(\mathbf{r}, \mathbf{r}') d\Omega = 0 \,. \tag{3.20}$$

We will defer the introduction of the exact function $G(\mathbf{r}, \mathbf{r}')$ until later, but require it to be square-integrable and twice differentiable over the domain Ω. In order to convert the volume integral into a surface integral, we would like to be able to apply the divergence theorem of Gauss:

$$\int_\Omega \nabla \cdot \mathbf{F} d\Omega = \oint_\Gamma \mathbf{F} \cdot \hat{\mathbf{n}} d\Gamma \,. \tag{3.21}$$

While the divergence theorem can't be directly applied to equation 3.20, we note that taking the divergence of a function $\mathbf{F} = \nabla\Phi G$, produces:

$$\nabla \cdot (\nabla\Phi G) = (\nabla^2\Phi)G + (\nabla\Phi) \cdot (\nabla G) \,, \tag{3.22}$$

which allows us to rewrite the left hand side of 3.20 as:

$$\int_\Omega (\nabla^2\Phi)G d\Omega = \int_\Omega \left[\nabla \cdot (\nabla\Phi G) - (\nabla\Phi) \cdot (\nabla G)\right] d\Omega \,. \tag{3.23}$$

Rearranging and applying the divergence theorem produces Green's first identity:

$$\int_\Omega (\nabla^2\Phi)G d\Omega + \int_\Omega (\nabla\Phi) \cdot (\nabla G) d\Omega = \oint_\Gamma (\hat{\mathbf{n}} \cdot \nabla\Phi) G d\Gamma \,. \tag{3.24}$$

Now we can express the second term on the left hand side as:

$$\begin{align} \int_\Omega (\nabla\Phi) \cdot (\nabla G) d\Omega &= \int_\Omega (\nabla G) \cdot (\nabla\Phi) d\Omega \tag{3.25} \\ &= \int_\Omega \nabla \cdot (\nabla G\Phi) d\Omega - \int_\Omega (\nabla^2 G)\Phi d\Omega \tag{3.26} \\ &= \oint_\Gamma (\hat{\mathbf{n}} \cdot \nabla G)\Phi - \int_\Omega (\nabla^2 G)\Phi d\Omega \,. \tag{3.27} \end{align}$$

Inserting the above expression into 3.24 yields Green's second identity:

$$\int_\Omega \left[(\nabla^2\Phi)G - (\nabla^2 G)\Phi\right] d\Omega = \oint_\Gamma \left[(\hat{\mathbf{n}} \cdot \nabla\Phi)G - (\hat{\mathbf{n}} \cdot \nabla G)\Phi\right] d\Gamma \,. \tag{3.28}$$

Clearly, from the fact that Φ satisfies 3.20, the first term on the left hand side of 3.28 is zero, so we obtain:

$$\int_\Omega (\nabla^2 G)\Phi d\Omega = \oint_\Gamma \left[(\hat{\mathbf{n}} \cdot \nabla\Phi)G - (\hat{\mathbf{n}} \cdot \nabla G)\Phi\right] d\Gamma \,. \tag{3.29}$$

In order to proceed further, it is now necessary to make a particular choice for the function G. This function is known as a Green's function. A natural choice is to seek a solution to the so-called fundamental equation:

$$\nabla^2 G(\mathbf{r},\mathbf{r}') = -\delta(\mathbf{r}-\mathbf{r}') \,, \tag{3.30}$$

where $\delta(\mathbf{r}-\mathbf{r}')$ is the three dimensional Dirac δ-function. In three dimensions, this has the solution [129] [2]:

$$G(\mathbf{r},\mathbf{r}') = \frac{1}{4\pi|\mathbf{r}-\mathbf{r}'|} \,. \tag{3.31}$$

While we should note that the Dirac δ-function is not a function in the strictest sense, a precise mathematical definition can be formulated in terms of the theory of distributions (see [147]). However, for our purpose, it suffices to define it by the manner in which it acts under integration[160]:

$$\int_V \psi(\mathbf{r}')\delta(\mathbf{r}-\mathbf{r}')dV = \begin{cases} \psi(\mathbf{r}) & \text{if } \mathbf{r} \in V \\ 0 & \text{if } \mathbf{r} \notin V \end{cases} \,, \tag{3.32}$$

where $V \in \mathbb{R}^3$ is some volume. Inserting 3.30 into the left hand side of 3.29 and applying property 3.32 allows us to express the value of the solution Φ at the point $\mathbf{r} \in \Omega$ as:

$$\int_\Omega \nabla^2 G(\mathbf{r},\mathbf{r}')\Phi(\mathbf{r}')d\Omega = \int_\Omega -\delta(\mathbf{r}-\mathbf{r}')\Phi(\mathbf{r}')d\Omega = -\Phi(\mathbf{r}). \tag{3.33}$$

Therefore,

$$\Phi(\mathbf{r}) = \oint_\Gamma (\hat{\mathbf{n}}\cdot\nabla G(\mathbf{r},\mathbf{r}'))\Phi(\mathbf{r}')d\Gamma - \oint_\Gamma (\hat{\mathbf{n}}\cdot\nabla\Phi(\mathbf{r}'))G(\mathbf{r},\mathbf{r}')d\Gamma \,, \tag{3.34}$$

from which we can see that the value of potential Φ, is solely a function of the boundary condition data on the surface Γ and the Green's function $G(\mathbf{r},\mathbf{r}')$. Using 3.34 to solve for Φ from the values of Φ and $\frac{\partial\Phi}{\partial n}$ on Γ, is known as the "direct"

[2] Note: An addition term of $K(\mathbf{r},\mathbf{r}')$ may be present on the right hand side of 3.31, where $\nabla^2 K(\mathbf{r},\mathbf{r}') = 0 \;\; \forall \mathbf{r} \in \Omega$. However, if we take the domain of the fundamental solution to be over all space $\mathbb{R}^3$ and make the reasonable assumption that the potential at infinity goes to zero, then this additional term is zero and can be ignored

boundary element method [160]. However, it is generally simpler to implement, and more physically intuitive to use what is known as the "indirect" method. This is the approach used in `KEMField`. The indirect method solves for the value of Φ not directly from the given boundary data, but rather from an unknown source function on the boundary. Upon inserting the expression for the fundamental Green's function into 3.34

$$\Phi(\mathbf{r}) = \frac{1}{4\pi}\left[-\oint_\Gamma \frac{\Phi(\mathbf{r}')\hat{\mathbf{n}}\cdot(\mathbf{r}-\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3}d\Gamma - \oint_\Gamma \frac{\hat{\mathbf{n}}\cdot\nabla\Phi(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|}d\Gamma\right], \tag{3.35}$$

and making the suggestive definitions:

$$\mathbf{p}(\mathbf{r}')/\epsilon_0 = \Phi(\mathbf{r}')\hat{\mathbf{n}}, \tag{3.36}$$

$$\sigma(\mathbf{r}')/\epsilon_0 = -\hat{\mathbf{n}}\cdot\nabla\Phi(\mathbf{r}'), \tag{3.37}$$

the nature of these source terms becomes clear:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0}\left[\oint_\Gamma \underbrace{\frac{\sigma(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|}}_{\mathcal{S}(\mathbf{r},\mathbf{r}')} d\Gamma - \oint_\Gamma \underbrace{\frac{\mathbf{p}(\mathbf{r}')\cdot(\mathbf{r}-\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3}}_{\mathcal{D}(\mathbf{r},\mathbf{r}')} d\Gamma\right]. \tag{3.38}$$

The first term $\mathcal{S}(\mathbf{r},\mathbf{r}')$, known as a single layer potential, is nothing other than the potential due to a thin layer of charge, $\sigma(\mathbf{r}')$, affixed to the boundary Γ. Whereas the second term $\mathcal{D}(\mathbf{r},\mathbf{r}')$ is known as a double layer potential and is the potential that would arise from a surface density of infinitesimal dipoles, $\mathbf{p}(\mathbf{r}')$ [129]. Since the presence of a dipole layer on the surface of a conductor or a dielectric material is unphysical,[3] we are motivated to discard this term as a possible source function. However, in addition to physical arguments against this term we note that it introduces a discontinuity in the potential Φ [129], which is unacceptable for our purposes since it leads to an undefined electric field $\mathbf{E}$ on the boundary. Instead,

[3]We are disregarding certain unrelated problems involving mobile charges in solution, where such dipole layers may form, as they do not obey the Laplace equation. Nevertheless, the freedom afforded by the term $K(\mathbf{r},\mathbf{r}')$ in the Green's function (2), allows us to selectively eliminate either the single or double layer terms, up to the introduction of a constant [129].

we are motivated to seek a solution Φ which is only due to the presence of surface charges $\sigma(\mathbf{r}')$ on the boundary Γ:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \oint_\Gamma \frac{\sigma(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} d\Gamma \,. \tag{3.39}$$

Arriving at such an ansatz for the solution comes as a rather unsurprising result, since with the knowledge that there is no net charge in the bulk, it could also have been derived by the direct integration of Coulomb's law 3.11. Or if the quantity of interest is the electric field (e.g. mixed or Neumann conditions), Coulomb's law itself:

$$\mathbf{E}(\mathbf{r}) = -\frac{1}{4\pi\epsilon_0} \oint_\Gamma \frac{\sigma(\mathbf{r}')(\mathbf{r}-\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3} d\Gamma \,. \tag{3.40}$$

One advantage afforded by the indirect approach using the single layer potential ansatz is that it naturally enforces continuity in Φ. Therefore, it can be applied simultaneously to solve both the interior problem (finding Φ within the domain Ω) as well as the exterior problem (determining Φ in $\mathbb{R}^3 \setminus \Omega$), while satisfying the homogeneous condition at infinity. Additionally, since Φ is continuous across the boundary Γ, we can do away with the requirement that Γ be a closed surface. While an infinitely thin, open surface is certainly unphysical, it can be a very useful approximation. When dealing with geometries where thin shells of conducting material are present (e.g. vacuum chambers), such an approximation often allows us to reduce the problem size by roughly a factor of two.

3.4 Linearization by Collocation

The results of the preceding section have allowed us to write down a solution as an integral over the boundary Γ, but they have not directly helped us determine it, as we have merely substituted an unknown potential Φ for an unknown surface charge density σ. In order to determine the surface charge density, σ, we need to apply the boundary conditions. For the moment, let us assume that we are solving a problem with purely Dirichlet boundary conditions. This results in what is known as a Fredholm integral equation of the first kind [160]:

$$D(\mathbf{r}) = \int_{\Gamma} K_D(\mathbf{r}, \mathbf{r}')\sigma(\mathbf{r}')d\Gamma \,, \tag{3.41}$$

where $D(\mathbf{r})$ is the boundary conditions and $K_D(\mathbf{r}, \mathbf{r}')$ is known as the kernel, where:

$$K_D(\mathbf{r}, \mathbf{r}') = \frac{1}{4\pi|\mathbf{r} - \mathbf{r}'|} \,. \tag{3.42}$$

To solve this equation, we need to do two things. The first is to compute the electric potential according to 3.39. The second is to choose a measure for the error on the boundary conditions. Unfortunately, unless we are dealing with extremely simple geometries (e.g. sphere, plane, etc.), it is not generally possible to write down an analytic form for either the surface or the surface charge density over which to perform the integration. Similarly, since the boundary conditions are specified in a continuous manner, it is difficult to compute the degree to which they are violated for an arbitrary geometry. To deal with these difficulties, we must resort to making an approximation of the original surface by discretizing it into simpler component shapes. Let us assume that the original surface, Γ, can be approximated

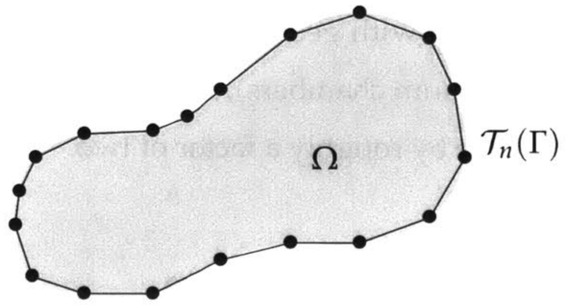


Figure 3-2: The boundary Γ of the domain Ω is approximated by a discretization $\mathcal{T}_n(\Gamma)$.

by the union of n simpler two-dimensional geometric entities u_i, over which the charge density takes the form $\sigma_i(\mathbf{r})$. We will refer to this set of n shapes u_i with the basis functions $\sigma_i(\mathbf{r})$, as a mesh or discretization, denoted by $\mathcal{T}_n(\Gamma)$. Figure 3-2 demonstrates a two dimensional projection of a boundary approximated by a mesh.

It is desirable that the functions u_i and σ_i have the following properties. The first is that the shape functions u_i must be composed of two-dimensional patches which can be positioned, without overlap or intersection, in such a way as to reproduce the original surface Γ within some error. That is to say, for some given error, s, it is possible to find a discretization $\mathcal{T}_n(\Gamma)$ such that:

$$\sum_{u_i \in \mathcal{T}_n(\Gamma)} \int_{u_i} \inf_{\mathbf{r} \in \Gamma} \left| \mathbf{r} - \mathbf{r}' \right| du < s \,, \tag{3.43}$$

provided n is sufficiently large. Secondly, we would like the shape and basis functions to be composed of simple forms for which inexpensive integration rules can be found. A particularly simple choice for the basis functions, which is used in `KEMField`, are the so-called pulse functions [91], which take an appealingly simple form:

$$\sigma_i(\mathbf{r}) = \begin{cases} \sigma_i & \text{if } \mathbf{r} \in u_i \\ 0 & \text{if } \mathbf{r} \notin u_i \end{cases} , \tag{3.44}$$

where σ_i is a real number representing a constant charge density over the surface element, u_i. By replacing the continuous integral over Γ with a sum over a discretization with a pulse function basis, the single layer ansatz of 3.39 and 3.40 then respectively become:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \left[\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_i} \frac{1}{|\mathbf{r} - \mathbf{r}'|} du \right] , \tag{3.45}$$

and

$$\mathbf{E}(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \left[\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_i} \frac{(\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^3} du \right] . \tag{3.46}$$

The shape functions that are employed in `KEMField` for the solution of three dimensional electrostatic problems are of the rectangular or triangular type. One dimensional line segments are also used for efficient representation of wire elements. The full description of these shape functions can be found in the thesis of

T. J. Corona [53]. The final property needed by the mesh is that the solution space spanned by the basis functions provides enough freedom to accurately represent the variation in the surface charge density. An exact definition of this property is difficult to give a priori, since calculating the error of a particular representation would require knowing the exact solution of the surface charge density in advance of solving the problem. However, in light of the existence and uniqueness theorems, it is not unreasonable to expect that if the boundary conditions are satisfied to within the required accuracy, then the solution has acceptably converged. A precise description of this notion of convergence can be formulated for elliptic partial differential equations [13, 229, 121], but this requires an introduction to Sobolev spaces, which is far beyond the scope of this thesis. As a practical means of determining how well the boundary conditions are satisfied, we will use the method of weighted residuals.

Dirichlet Conditions

For Dirichlet conditions, the residual is defined as the difference between the boundary value to be enforced and the potential calculated from the surface charge distribution. The Dirichlet residual of the potential at $\mathbf{r} \in \Gamma$ is given by:

$$R_D(\mathbf{r}) = V(\mathbf{r}) - \Phi(\mathbf{r}) \, . \tag{3.47}$$

In order to satisfy the boundary conditions we wish to make this residual function zero over the whole surface Γ. To do this, we choose a metric defined by some set of weight functions, f_j, against which we form an inner product over the surface Γ, requiring that:

$$\int_\Gamma f_j(\mathbf{r}) R_D(\mathbf{r}) d\Gamma = 0 \, . \tag{3.48}$$

Since practically speaking, we can only do this in an approximate way, it is necessary to replace the surface, Γ, with the discretization $\mathcal{T}_n(\Gamma)$. With this substitution, it is convenient to define the weight functions in such a way that they are orthogonal to each other with respect to the shape functions composing the discretization, such that:

$$f_j(\mathbf{r}) = \begin{cases} w_j(\mathbf{r}) & \text{if } \mathbf{r} \in u_j \\ 0 & \text{if } \mathbf{r} \notin u_j \end{cases} . \tag{3.49}$$

Inserting our definitions of the potential 3.39 and the weight functions 3.49, and approximating Γ with the mesh $\mathcal{T}_n(\Gamma)$, we can decompose 3.48 into n equations of the following form:

$$\int_{u_j} w_j(\mathbf{r}) V(\mathbf{r}) du_j - \int_{u_j} w_j(\mathbf{r}) \left[\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right] du_j = 0 \, . \tag{3.50}$$

Upon manipulation, this becomes:

$$\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_j} \int_{u_i} \frac{w_j(\mathbf{r}) du_i du_j}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} = \int_{u_j} w_j(\mathbf{r}) V(\mathbf{r}) du_j \, . \tag{3.51}$$

It is now clear that this equation can be written as a matrix equation of the form $A\mathbf{x} = \mathbf{b}$, by making the identification of the matrix and vector elements as:

$$x_i = \sigma_i \, , \tag{3.52}$$

$$b_j = \int_{u_j} w_j(\mathbf{r}) V(\mathbf{r}) du_j \, , \tag{3.53}$$

and

$$A_{ij} = \int_{u_j} \int_{u_i} \frac{w_j(\mathbf{r}) du_i du_j}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \, . \tag{3.54}$$

In order to evaluate the integrals A_{ij}, it is now necessary to make an explicit choice for the weight functions, w_j. The BEM literature has many options for the type of weight function we may use, each with varying degrees sophistication. One possible choice is the so-called Galerkin method [40], where the weight functions, w_j, are chosen to be the adjoint of the basis functions, u_j. In this way, the residual error on the boundary condition is minimized in an average sense over the entire surface of the discretization. A major advantage of the Galerkin method is that the resulting system matrix A is symmetric. This both reduces the memory required to store the coefficients and allows a wider range of algorithms for solving the equation $A\mathbf{x} = \mathbf{b}$. The primary disadvantages of the Galerkin technique is that evaluation of the double integrals, A_{ij}, is computationally expensive and the implementation of the Galerkin method in computer code is relatively complex.

A simpler alternative, which is used by `KEMField`, is the collocation method. In the collocation method, rather than performing a weighted minimization of the residual for all points on the boundary discretization, we choose a set of points, $\{\mathbf{y}_i\}$, where we wish the residual to be exactly zero. This results in a simple choice for the weight functions w_j, since they are nothing but Dirac δ-functions:

$$w_j(\mathbf{r}) = \delta(\mathbf{y}_j - \mathbf{r}) \tag{3.55}$$

Inserting this choice for w_j into 3.54 and integrating reduces the double integral for the matrix elements into a single integral:

$$A_{ij} = \int\limits_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{y}_j - \mathbf{r}'|}, \tag{3.56}$$

whereas the integral of equation 3.53 simplifies to the value of the boundary data at the collocation points: $b_i = V(\mathbf{y}_i)$.

While any set of points on the boundary (provided there are as many as the number of degrees of freedom) can be used as the collocation points, a common choice is to use the centroids of the elements of the discretization. The centroid $\mathbf{y}_i$ of the element u_i is given by the integral:

$$\mathbf{y}_i = \int\limits_{u_i} \mathbf{r}' du_i \,, \tag{3.57}$$

which in the case of a planar polygonal element can be computed simply by averaging the m vertices $\mathbf{v}_k$ of the element u_i:

$$\mathbf{y}_i = \frac{1}{m} \sum_{k=1}^{m} \mathbf{v}_k \,. \tag{3.58}$$

Choosing the centroids as the collocation points is advantageous for two reasons. The first is that, given convex polygonal shape functions, u_i, the centroid always exists away from the edges, in a region with a well defined tangent plane and normal vector. The second reason is that the centroid is always well separated from neighboring elements, which simplifies the treatment of singularities arising in the evaluation of 3.56 when $\mathbf{r}' \longrightarrow \mathbf{y}_i$. Since the centroids $\mathbf{y}_i$ are well contained within their respective elements, all integrals of the form A_{ij}, with $i \neq j$, are singularity free. The matrix elements along the diagonal, A_{ii}, remain somewhat problematic, since $\mathbf{y}_i$ is in the domain u_i. However, it is possible to treat them consistently in a limiting sense, through the use of the Cauchy principle value (CPV) [42]. This is done by excising the region contained by a small disk, $B(\mathbf{y}_i)$, about the point, $\mathbf{y}_i$, from the domain of integration and computing the resulting integral in the limit that the radius of the disk, $B(\mathbf{y}_i)$, goes to zero. It suffices to say that the CPV of the integral, A_{ii}, exists and is readily calculable, however for the sake of brevity we will not detail this calculation here, as it has been dealt with extensively elsewhere (see [53] and [84]).

Neumann Conditions

We can follow a similar procedure to apply Neumann boundary conditions. In the case of linear dielectrics, the boundary condition is a statement about the discontinuity in the normal component of the electric field. This discontinuity is due to the polarization charge present on the interface between two materials with different permittivities. We assume that the surface, Γ, representing the interface between the materials is a smooth, orientable surface with a normal, $\hat{\mathbf{n}}$, defined everywhere. Labeling the permittivity of the material above the interface (in the direction of $\hat{\mathbf{n}}$) as ϵ_+ and the permittivity of the material below the interface as ϵ_-, then for all points $\mathbf{r} \in \Gamma$, the normal component of the electric field satisfies the following:

$$\epsilon_+ E_+^\perp - \epsilon_- E_-^\perp = 0 \,. \tag{3.59}$$

Considering that the above expression 3.59 is defined to be zero over the whole interface, it is a logical choice for the residual function used to enforce the Neumann condition:

$$R_N(\mathbf{r}) = \epsilon_+ E_+^\perp - \epsilon_- E_-^\perp \,, \tag{3.60}$$

with the limiting values of normal component of the electric field $E_\pm^\perp$ given by [4]:

$$E_\pm^\perp = \lim_{\Delta \longrightarrow 0} [\hat{\mathbf{n}}(\mathbf{r}) \cdot \mathbf{E}(\mathbf{r} \pm \Delta \hat{\mathbf{n}}(\mathbf{r}))] \tag{3.61}$$

$$= - \lim_{\Delta \longrightarrow 0} [\hat{\mathbf{n}}(\mathbf{r}) \cdot \nabla \Phi(\mathbf{r} \pm \Delta \hat{\mathbf{n}}(\mathbf{r}))] = -\frac{\partial_\pm}{\partial n} [\Phi(\mathbf{r})] \,. \tag{3.62}$$

As before, we want the integral of this residual function multiplied against the weight functions, f_j, over the surface, Γ, to be zero:

$$\int_\Gamma f_j(\mathbf{r}) R_N(\mathbf{r}) d\Gamma = \int_\Gamma f_j(\mathbf{r}) \left[\epsilon_+ E_+^\perp - \epsilon_- E_-^\perp\right] d\Gamma = 0 \,. \tag{3.63}$$

To carry out this integration in our solution space, we again replace, Γ, with its discretization $\mathcal{T}_n(\Gamma)$ and use the orthogonality of the weight functions to decompose it into to the n equations:

$$\int\limits_{u_j} w_j(\mathbf{r}) \left[\epsilon_+ E_+^\perp - \epsilon_- E_-^\perp\right] du_j = 0 \,. \tag{3.64}$$

[4]The value of Δ in the limit is strictly positive.

Inserting the expression for the electric field yields:

$$\epsilon_- \int_{u_j} w_j(\mathbf{r}) \frac{\partial_-}{\partial n} \left[\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right] du_j$$
$$- \epsilon_+ \int_{u_j} w_j(\mathbf{r}) \frac{\partial_+}{\partial n} \left[\sum_{u_i \in \mathcal{T}_n(\Gamma)} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right] du_j = 0\,. \quad (3.65)$$

Being aware of the singularities that are present in the integrand, we are motivated to split the sum over $\mathcal{T}_n(\Gamma)$ into two distinct parts: the domain consisting of the element u_j (which contains the point $\mathbf{r}$) and all other elements $u_i \in \mathcal{T}_n(\Gamma)$ where $i \neq j$. To illustrate the need to make this split, it is useful to take a brief moment to consider the physics involved in this situation. Locally, the discontinuity in the electric field across the interface is due to a layer of polarization charge in the neighborhood of $\mathbf{r}$. This local polarization charge is produced by the unequal response of the two materials under the influence of an external field, $\mathbf{E}_{\text{ext}}$, which is due to the charges on $\mathcal{T}_n(\Gamma) \setminus u_j = \{u_i \in \mathcal{T}_n(\Gamma) | i \neq j\}$. The region $\mathcal{T}_n(\Gamma) \setminus u_j$ excludes this local polarization charge, so the external field is continuous at $\mathbf{r}$ and the directional limits for this quantity are equal:

$$\frac{\partial_\pm}{\partial n} \left[\sum_{\substack{u_i \in \mathcal{T}_n(\Gamma) \\ i \neq j}} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right] = \frac{\partial}{\partial n} \left[\sum_{\substack{u_i \in \mathcal{T}_n(\Gamma) \\ i \neq j}} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right] . \quad (3.66)$$

Extracting the problematic region, u_j, from the sum allows us to write down the continuous external ($\mathbf{E}_{\text{ext}}$) and discontinuous local ($\mathbf{E}_{\text{loc}}$) field terms separately:

$$(\epsilon_- - \epsilon_+) \int_{u_j} w_j(\mathbf{r}) \underbrace{\frac{\partial}{\partial n} \left[\sum_{\substack{u_i \in \mathcal{T}_n(\Gamma) \\ i \neq j}} \sigma_i \int_{u_i} \frac{du_i}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|} \right]}_{E^{\perp}_{\text{ext}}} du_j$$

$$+ \int_{u_j} w_j(\mathbf{r}) \left[\epsilon_- \underbrace{\frac{\partial_-}{\partial n} \int_{u_j} \frac{\sigma_j du_j}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|}}_{E^{\perp}_{\text{loc}}|_-} - \epsilon_+ \underbrace{\frac{\partial_+}{\partial n} \int_{u_j} \frac{\sigma_j du_j}{4\pi\epsilon_0 |\mathbf{r} - \mathbf{r}'|}}_{E^{\perp}_{\text{loc}}|_+} \right] du_j = 0 \,. \quad (3.67)$$

Inserting our choice of weight functions (collocation) 3.55, evaluating the continuous derivative, and integrating out the δ-functions produces:

$$\underbrace{(\epsilon_- - \epsilon_+) \sum_{\substack{u_i \in \mathcal{T}_n(\Gamma) \\ i \neq j}} \sigma_i \int_{u_i} \frac{\hat{\mathbf{n}}(\mathbf{y}_j) \cdot (\mathbf{y}_j - \mathbf{r}') du_i}{4\pi\epsilon_0 |\mathbf{y}_j - \mathbf{r}'|^3}}_{E^{\perp}_{\text{ext}}(\mathbf{y}_i)} + \epsilon_- E^{\perp}_{\text{loc}}(\mathbf{y}_j)\Big|_- - \epsilon_+ E^{\perp}_{\text{loc}}(\mathbf{y}_j)\Big|_+ = 0. \quad (3.68)$$

While one can explicitly evaluate the discontinuous local electric field terms due to the polarization charge through a limiting process [53], in the case of a smooth surface,[5] one can apply Gauss's law over an infinitesimal pillbox [109] about the neighborhood surrounding the centroid $\mathbf{y}_j \in u_j$ to show that:

$$\epsilon_- E^{\perp}_{\text{loc}}(\mathbf{y}_j)\Big|_- - \epsilon_+ E^{\perp}_{\text{loc}}(\mathbf{y}_j)\Big|_+ = -(\epsilon_- + \epsilon_+) \frac{\sigma_j}{2\epsilon_0} \,. \quad (3.69)$$

[5]For a smooth parameterized surface Σ given by the map $\Sigma(u, v) : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$, the partial derivatives $\frac{\partial \Sigma}{\partial u}$ and $\frac{\partial \Sigma}{\partial v}$ exist for all $\mathbf{r} \in \Sigma$. Consequently, there is always a neighborhood about $\mathbf{r}$ for which an local tangent plane exists. This is always true for our choice of shape functions as they are planar polygons.

Finally, we obtain a succinct form for the Neumann boundary value problem in the form of a matrix equation $A\mathbf{x} = \mathbf{b}$, written component-wise as:

$$\frac{1}{2\epsilon_0}\sigma_j = \left(\frac{\epsilon_- - \epsilon_+}{\epsilon_- + \epsilon_+}\right) \sum_{\substack{u_i \in \mathcal{T}_h(\Gamma) \\ i \neq j}} \sigma_i \int_{u_i} \frac{\hat{\mathbf{n}}(\mathbf{y}_j) \cdot (\mathbf{y}_j - \mathbf{r}')du_i}{4\pi\epsilon_0|\mathbf{y}_j - \mathbf{r}'|^3} \tag{3.70}$$

with the explicit identification that:

$$x_i = \sigma_i \,, \tag{3.71}$$

$$b_i = 0 \,, \tag{3.72}$$

and

$$A_{ij} = \frac{1}{\epsilon_0}\left[\frac{1}{2}\delta_{ij} + (1 - \delta_{ij})\left(\frac{\epsilon_+ - \epsilon_-}{\epsilon_- + \epsilon_+}\right)\int_{u_j} \frac{\hat{\mathbf{n}}(\mathbf{y}_i) \cdot (\mathbf{y}_i - \mathbf{r}')du_j}{4\pi|\mathbf{y}_i - \mathbf{r}'|^3}\right] \,, \tag{3.73}$$

where δ_{ij} is the Kronecker delta. This is a discrete version of a Fredholm integral equation of the second kind:

$$\frac{1}{2}\sigma(\mathbf{r}) = \int_\Gamma K_N(\mathbf{r}, \mathbf{r}')\sigma(\mathbf{r}')d\Gamma \,, \tag{3.74}$$

with a kernel given by:

$$K_N(\mathbf{r}, \mathbf{r}') = \frac{\hat{\mathbf{n}}(\mathbf{r}) \cdot (\mathbf{r} - \mathbf{r}')}{4\pi|\mathbf{r} - \mathbf{r}'|^3} \,. \tag{3.75}$$

We note that the factor of $\frac{1}{2}$ is due to our explicit assumption that the boundary is smooth. This is valid for our choice of discretization and use of the collocation method, since we use planar polygonal elements and only evaluate the boundary conditions at the centroids. For a non-smooth surface, this numerical factor may vary as a function of **r** and depends on the local solid angle subtended above and below the surface (see [53]).

Mixed Conditions

In practice, pure Dirichlet or Neumann systems are not commonly encountered, and mixed systems need to be dealt with. However, now that we have specified the matrix equations resulting from pure Dirichlet and Neumann systems, it is not hard to construct the analogous equation for a mixed system. Since we have used the single layer ansatz for both types of systems, there is no difference in the form of the equation, or the solution vector, **x**, which is simply the list of the surface charge densities on each discrete patch of the boundary. Therefore, we only need to specify the matrix elements and the right hand side of this equation, the form of which only depends on whether a specific collocation point is a member of the set of Dirichlet surfaces $\{D\}$ or Neumann surfaces $\{N\}$. The right hand side is given by:

$$b_i = \begin{cases} V(\mathbf{y}_i) & \text{if } \mathbf{y}_i \in \{D\} \\ 0 & \text{if } \mathbf{y}_i \in \{N\} \end{cases}, \tag{3.76}$$

and the matrix elements are:

$$A_{ij} = \begin{cases} A_{ij} = \int\limits_{u_i} \frac{1}{4\pi\epsilon_0 |\mathbf{y}_j - \mathbf{r}'|} du_i & \text{if } \mathbf{y}_i \in \{D\} \\ \frac{1}{\epsilon_0}\left[\frac{1}{2}\delta_{ij} + (1-\delta_{ij})\left(\frac{\epsilon_+ - \epsilon_-}{\epsilon_- + \epsilon_+}\right) \int\limits_{u_j} \frac{\hat{\mathbf{n}}(\mathbf{y}_i)\cdot(\mathbf{y}_i - \mathbf{r}')du_j}{4\pi|\mathbf{y}_i - \mathbf{r}'|^3}\right] & \text{if } \mathbf{y}_i \in \{N\} \end{cases}. \tag{3.77}$$

All that remains is to implement a means of solving a general matrix equation of the form $A\mathbf{x} = \mathbf{b}$. We should note that unlike the matrix equations encountered in finite difference methods, this is a dense linear system. This is due to the long range nature of the Coulomb force which couples each mesh element to all of the other elements in the problem. We also note that because we have chosen collocation as the means of enforcing the boundary conditions, the matrix A is non-symmetric and typically $A_{ij} \neq A_{ji}$. Unfortunately, since a dense non-symmetric linear system is the most general kind, solving the BEM problem for large geometries will require some special techniques.

Chapter 4

Solving Dense Non-symmetric Linear Systems

Now that we have constructed a linear system representing the Laplace boundary value problem, we are left with the task of solving the resulting dense non-symmetric matrix equation. Various direct and iterative methods are available for dealing with problems of this type. To choose an appropriate method, we need to carefully consider how the required memory and computational resources of each algorithm scale with, N, the number of degrees of freedom in the problem.

4.1 Direct Methods

The first and most obvious techniques which come to mind for solving a dense and non-symmetric system are the so-called direct methods. Some of the most well known methods in this category are Gaussian elimination, LU decomposition, and QR factorization [102]. These methods require the explicit computation and storage of the matrix elements of A, which immediately means that they all have minimum memory costs which grow like $\mathcal{O}(N^2)$. An even worse limitation than the memory requirement is that the number of arithmetic operations that are required to compute the solution scales like $\mathcal{O}(N^3)$. In light of the scaling properties of the direct methods and the limitations of modern computer hardware, it is clear that they are generally not practical for solving BEM problems with a dimension much larger than $N = 10^4$. Accurate modeling of the KATRIN spectrometer hardware requires discretizations containing more than $\approx 10^6$ mesh elements, so direct methods are not particularly useful for this purpose. Since extensive literature [102, 64, 218] exists on the topic of direct methods and they are not appropriate for solving the type of large scale BEM problems we are interested in, we will not consider them further.

4.2 Iterative Methods

Another family of techniques for solving linear systems are iterative methods. The basic algorithm of an iterative process relies on being able to compute some measure of the error given a solution estimate. Then a better solution is generated using the data provided by the error. This process is performed repeatedly until some convergence condition is reached. Usually this condition requires that the error be less than some threshold ϵ. A basic skeleton of an iterative method is outlined in algorithm 1. Obviously, there are significant details omitted from algorithm

Algorithm 1 Iterative process to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: Matrix $A$, right hand side $\mathbf{b}$, and initial solution estimate $\mathbf{x}_0$.
1: Compute error estimate $E_0$.
2: while $E_i \geq \epsilon$ do
3:     Generate next solution estimate $\mathbf{x}_{i+1}$ using the previous error estimate $E_i$.
4:     Compute error estimate $E_{i+1}$.
5: end while
Output: The approximate solution $\mathbf{x}_n$, with error $E_n \leq \epsilon$.
```

1, as there are many choices for the manner in which we choose to measure the error and in the way we generate an improved solution estimate. Since we are interested in solving problems with $N \approx 10^6$ or more unknowns, we must confine ourselves to the class of iterative processes which are known as matrix-free methods. Matrix-free methods are those which do not require us to compute or store the matrix A in its entirety. With the exception of the Robin Hood method, we will primarily concern ourselves with a subset of matrix-free iterative techniques, known as Krylov subspace methods.

4.3 Robin Hood

Robin Hood is a novel iterative method that was specifically developed for the solution of Laplace boundary value problems in electrostatics [151, 152, 150, 84]. While it is not in the class of algorithms known as Krylov subspace methods, it is a very important reference against which we can compare other methods, as it has shown itself to be both accurate and scalable to large problems [53]. Being a matrix-free method, the elements of A need not be stored. However, a fast means of obtaining an arbitrary matrix element is generally necessary for its successful implementation. Aside from a means to compute the matrix elements, the Robin Hood method only requires the storage of the current solution estimate, $\mathbf{x}_i$, along with a residual vector, $\mathbf{r}_i$, composed of the error estimates associated with each degree of freedom. This small storage requirement gives Robin Hood optimal memory scaling, proportional to $\mathcal{O}(N)$. An outline of one variant (the Gauss-Seidel limit [53]) of this Robin Hood is given in algorithm 2. For further discussion of this method, its variants, and its relation to other iterative methods such as Gauss-Seidel or Successive-subspace-correction, the reader is referred to [151, 152, 150, 84, 53].

Algorithm 2 Robin Hood algorithm to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: Matrix $A$ and right hand side $\mathbf{b}$. Initially $\mathbf{x}_0 = 0$ and $\mathbf{r}_0 = \mathbf{b}$.
1: while $||\mathbf{r}_i||_\infty \geq \epsilon||\mathbf{b}||_\infty$ do
2:     $r_{i,j} = ||\mathbf{r}_i||_\infty$          ▷ Identify largest element of residual and its index $j$.
3:     $\Delta_j = (b_j - r_{i,j}/A_{j,j})$               ▷ Compute the correction to $x_{i,j}$.
4:     $\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta_j \mathbf{e}_j$              ▷ Update solution approximation.
5:     $\mathbf{z} = A_{*,j}$                           ▷ Calculate the $j$-th column of A.
6:     $\mathbf{r}_{i+1} = \mathbf{r}_i + \Delta_j \mathbf{z}$                        ▷ Update the residual.
7: end while
Output: The approximate solution $\mathbf{x}_n$, with relative residual error $||\mathbf{r}_i||_\infty \leq \epsilon||\mathbf{b}||_\infty$ .
```

The primary disadvantage of the Robin Hood algorithm is that it has an arithmetic scaling that is $\mathcal{O}(N^2)$ [84]. This is due to the fact that the work done during each iteration is proportional to N, while the number of iterations required to reach convergence is also proportional to N. However, the nature of the algorithm is almost embarrassingly parallel. This is because most of the work done during one step of the iterative process consists of many, $\mathcal{O}(N)$, independent calculations. Therefore, it benefits greatly from parallel computing. The adaptation of the Robin Hood algorithm to make use of large CPU clusters and the massively parallel single-instruction-multiple-data (SIMD) architecture, provided by graphics processors (GPUs), has been carried out by [53]. The parallel implementation has shown it can easily handle BEM problems of the scale involved in KATRIN because of its excellent parallel efficiency [53].

4.4 Krylov Methods

Krylov methods are a sub-family of iterative techniques which are in the class of algorithms known as projection processes[1]. Such processes construct a solution by projecting the residual onto an appropriate subspace in order to successively minimize orthogonal components of the error. Specifically, starting with an initial approximation $\mathbf{x}_0 \in \mathbb{R}^N$, a Krylov method will generate an approximation to the solution $\mathbf{x}_n$ given by [157]:

$$\mathbf{x}_n = \mathbf{x}_0 + \mathbf{z}_m \,, \tag{4.1}$$

at the n-th iteration. When no information on the solution is available at the outset, a typical choice for the approximation at the initial step is the zero vector, though this particular choice is not necessary for the success of the method since any choice will suffice (though some may afford faster convergence). The vector, $\mathbf{z}_m$, is drawn from the m-dimensional subspace $\mathcal{S}_m \subset \mathbb{R}^N$ (known as the search space) subject to the condition on the residual:

$$\mathbf{r}_n = \mathbf{b} - A\mathbf{x}_n \perp \mathcal{C}_m \,, \tag{4.2}$$

[1]In fact, the previously mentioned Robin Hood algorithm is also a projection process, though it is not a Krylov method.

where $\mathcal{C}_m$ is the m-dimensional subspace known as the constraint space. A Krylov method is a projection method where $m = n$ and the subspace over which the projection takes place is specifically the Krylov subspace [205]:

$$\mathcal{K}_n(A, \mathbf{r}_0) = \mathrm{span}\{\mathbf{r}_0,\ A\mathbf{r}_0,\ A^2\mathbf{r}_0,\ \ldots, A^{n-1}\mathbf{r}_0\}\ . \tag{4.3}$$

The Krylov subspace is a particularly judicious choice for the search subspace, since it can be augmented recursively through repeated application of the matrix A on the initial data. This choice for the search subspace is motivated by the Cayley-Hamilton theorem. A direct consequence of this theorem is that for an invertible matrix, $A \in \mathbb{R}^{N\times N}$, we can express its inverse as a matrix polynomial with degree no larger than $N-1$ [63, 205]:

$$A^{-1} = c_0 I_N + \sum_{k=1}^{N-1} c_k A^k\ . \tag{4.4}$$

Simple manipulation of the original equation shows us that the initial residual, $\mathbf{r}_0$, is related to the true solution by $\mathbf{x} = \mathbf{x}_0 + A^{-1}\mathbf{r}_0$. Therefore, in light of the Cayley-Hamilton, this implies the solution is given by the series:

$$\mathbf{x} = \mathbf{x}_0 + c_0\mathbf{r}_0 + \sum_{k=1}^{N-1} c_k(A^k\mathbf{r}_0)\ . \tag{4.5}$$

Since the approximate solution at the n-th iteration is drawn from the Krylov subspace, $\mathcal{K}_n(A, \mathbf{r}_0)$, it is easy to see that $\mathbf{x}_n$ is simply the truncated series:

$$\mathbf{x}_n = \mathbf{x}_0 + c_0\mathbf{r}_0 + \sum_{k=1}^{n-1} c_k(A^k\mathbf{r}_0)\ . \tag{4.6}$$

Therefore, the problem of obtaining the best approximation, $\mathbf{x}_n$, amounts to computing the set of coefficients, c_k, which minimize the residual $\mathbf{r}_n = \mathbf{b} - A\mathbf{x}_n$.

4.4.1 GMRES

Probably the most widely known and successful Krylov subspace method is the Generalized Minimum Residual (GMRES) algorithm introduced by Saad et al. [204]. Its success is due to its broad applicability[2] and guaranteed convergence[3].

The GMRES algorithm is a projection process which chooses $\mathcal{C}_n = \mathcal{K}_n(A, \mathbf{r}_0)$ and is in essence a modified Arnoldi iteration [14]. The Arnoldi iteration transforms a general matrix into upper Hessenberg form, H, through a succession of orthogonal similarity transformations [218]. The practical implementation of which is not much more complicated than the familiar Gram-Schmidt orthogonalization process[4]. GMRES incorporates an additional step following the Arnoldi iteration, consisting of a series of Given's rotations which transforms the upper Hessenberg matrix into an upper triangular matrix. This process also conveniently provides the current L_2 norm of the residual error, which allows one to track the progress towards the solution without explicitly evaluating the residual[5]. Once the residual error has converged to an acceptably small value, the approximate solution can be constructed by inverting H, which can easily be accomplished through back-substitution.

For a non-singular system of dimension N, GMRES is guaranteed to converge in no more than N iterations, and in doing so, the residual norm, $\|\mathbf{r}_i\|_2$, at each iteration will produce a strictly monotonically decreasing sequence [158]. However, since any strictly monotonically decreasing sequence is possible (including the worst-case sequence which is essentially constant for all iterations but the very last), this guarantee is not always practically useful, especially when N is extremely large. On the other hand, many systems of interest do in fact converge in k iterations, where $k \ll N$, and this is typically true of the BEM problems we seek to solve. Unfortunately, it is not generally possible to predict beforehand which systems will

[2]Unlike other such methods such as the Conjugate Gradient Method (CG), which requires a symmetric matrix [116], it makes no assumption on matrix structure, hence the term "generalized".

[3]GMRES is guaranteed to converge for non-singular systems.

[4] The process is slightly modified from canonical Gram-Schmidt in order to deal with numerical round-off error in floating point mathematics.

[5]This is advantageous, since evaluating the residual costs an extra matrix-vector product involving the full system matrix A.

converge in an acceptable number of iterations.

The primary disadvantage of the GMRES algorithm is that it must explicitly store the bases of the Krylov subspace. This results in memory usage which grows at each iteration, such that after k iterations the memory usage is $\mathcal{O}(kN)$. This can pose severe difficulties for systems which are slow to converge. In order to accommodate such systems, GMRES is typically used in a restarted fashion, where, after some number, n, iterations the existing set of Krylov subspace basis vectors are discarded. The process then begins again using the current best solution approximation $\tilde{\mathbf{x}}$. Restarted GMRES is typically denoted as GMRES(n) and has been implemented in KEMField as outlined by [204, 205]. The basic process is summarized in algorithm 3. Unfortunately, the restarted algorithm, GMRES(n), no longer has the guaranteed convergence property of the original algorithm and can take much longer to converge to an acceptable residual error or even stagnate completely. Other algorithms with a fixed memory foot-print and better convergence properties than restarted GMRES(n) do exist. Typically, methods such as GCROT [61, 117] and Loose-GMRES (LGMRES) [23] do this by selectively recycling a subset of the basis vectors in the Krylov subspace. However, these algorithms are generally more complicated to implement and so far have not been included in the KEMField software package.

4.4.2 BiCGSTAB

An alternative to GMRES, which does not maintain a full Krylov subspace, is the Biconjugate Gradient Stabilized method (BiCGSTAB). This algorithm is derived from the Lanzcos biorthogonalization procedure with some corrections to aid numerical stability. BiCGSTAB is notable for its small and fixed memory footprint. Since it only needs to carry over a handful of biorthogonal vectors at each iteration, in order to advance the solution, it can avoid the accumulated memory problem of GMRES. Unfortunately, for the problems we are interested in solving, it tends to exhibit somewhat poor convergence. Given its limited use, we will avoid a

Algorithm 3 Restarted GMRES(n) algorithm to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: The matrix $A$, right hand side $\mathbf{b}$, and initial estimate $\mathbf{x}_0$.
 1: $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$    ▷ Compute the initial residual.
 2: $p_0 = \|\mathbf{r}_0\|_2$ and $\mathbf{v}_0 = \mathbf{r}_0 / p_0$    ▷ Normalize and augment Krylov subspace.
 3: $\rho = |p_0|$ and $j = 0$    ▷ Initialize residual error and iteration count.
 4: while $\rho \geq \epsilon \|\mathbf{b}\|_2$ and $j \leq n$ do
 5:     $\mathbf{w} = A\mathbf{v}_j$
 6:     for $i = 0, \ldots, j$ do    ▷ Gram-Schmidt orthogonalization.
 7:         $H_{i,j} = \langle \mathbf{w}, \mathbf{v}_j \rangle$
 8:         $\mathbf{w} = \mathbf{w} - H_{i,j}\mathbf{v}_j$
 9:     end for
10:     $\beta = \|\mathbf{w}\|_2$ and $\mathbf{v}_{j+1} = \mathbf{w}/\beta$.    ▷ Normalize and augment Krylov subspace.
11:     for $i = 0, \ldots, j-1$ do    ▷ Apply Given's rotations on $j$-th column of $H$.
12:         $a = H_{i,j}, b = H_{i+1,j}$
13:         $H_{i,j} = c_i a + s_i b$
14:         $H_{i+1,j} = -s_i a + c_i b$
15:     end for
16:     $c_j = H_{j,j} / \sqrt{H_{j,j}^2 + \beta^2}$    ▷ Compute $j$-th Given's rotation.
17:     $s_j = \beta / \sqrt{H_{j,j}^2 + \beta^2}$
18:     $H_{j,j} = \sqrt{H_{j,j}^2 + \beta^2}$ and $H_{j+1,j} = 0$    ▷ Apply $j$-th Given's rotation to $H$.
19:     $p_j = c_j p_j$ and $p_{j+1} = -s_j p_j$    ▷ Apply $j$-th Given's rotation to $\mathbf{p}$.
20:     $\rho = |p_{j+1}|$ and $j = j + 1$    ▷ Update residual error and iteration count.
21: end while
22: $\mathbf{y} = \underset{\mathbf{y} \in \mathbb{R}^j}{\operatorname{argmin}} \|H\mathbf{y} - \mathbf{p}\|$    ▷ Solve $j \times j$ least-squares minimization.
23: $\widetilde{\mathbf{x}} = \mathbf{x}_0 + \sum_{i=0}^{j} y_i \mathbf{v}_i$    ▷ Compute solution approximation.
24: if $\rho \leq \epsilon \|\mathbf{b}\|_2$ then exit.
25: else
26:     $\mathbf{x}_0 = \widetilde{\mathbf{x}}$. goto 1:    ▷ Restart.
27: end if
Output: The approximate solution $\widetilde{\mathbf{x}}$, with relative residual error $\rho \leq \epsilon \|\mathbf{b}\|_2$.
```

detailed description, except to note that BiCGSTAB does not guarantee convergence for a general non-singular non-symmetric matrix. In addition, the residual norm doesn't necessarily form a monotonically decreasing sequence as the algorithm progresses [45]. An extensive discussion of its properties and development can be found elsewhere (see [223, 205]). A summary of the algorithm as it is implemented in KEMField can be found in algorithm 4.

Algorithm 4 BiCGSTAB algorithm to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: The matrix $A$, right hand side $\mathbf{b}$, and initial estimate $\mathbf{x}_0$.
 1: $\hat{\mathbf{r}} = \mathbf{b} - A\mathbf{x}_0$                      ▷ Compute the initial residual.
 2: $\mathbf{r} = \hat{\mathbf{r}}, \tilde{\mathbf{x}} = \mathbf{x}_0, \mathbf{p} = 0, \mathbf{v} = 0$                      ▷ Initialize values.
 3: $\alpha = 1, \omega = 1, \rho = 1, \hat{\rho} = 1$
 4: while $\|\mathbf{r}\|_2 \geq \epsilon\|\mathbf{b}\|_2$ do
 5:     $\rho = \langle \mathbf{r}, \mathbf{r} \rangle$
 6:     $\beta = (\rho/\hat{\rho})(\alpha/\omega)$
 7:     $\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega\mathbf{v})$
 8:     $\mathbf{v} = A\mathbf{p}$
 9:     $\alpha = \rho/\langle \hat{\mathbf{r}}, \mathbf{v} \rangle$
10:     $\mathbf{s} = \mathbf{r} - \alpha\mathbf{v}$
11:     $\mathbf{t} = A\mathbf{s}$
12:     $\omega = \langle \mathbf{s}, \mathbf{t} \rangle / \langle \mathbf{t}, \mathbf{t} \rangle$
13:     $\tilde{\mathbf{x}} = \tilde{\mathbf{x}} + \alpha\mathbf{p} + \omega\mathbf{s}$                 ▷ Update solution approximation.
14:     $\mathbf{r} = \mathbf{s} - \omega\mathbf{t}$                                ▷ Update residual.
15:     $\hat{\rho} = \rho$
16: end while
Output: The approximate solution $\tilde{\mathbf{x}}$, with relative residual error $\|\mathbf{r}\|_2 \leq \epsilon\|\mathbf{b}\|_2$.
```

Despite the appealing simplicity and fixed memory foot-print of both GMRES(n) and BiGCSTAB, these methods tend to suffer from stagnation when applied to poorly conditioned systems. For such systems, preconditioning is often necessary in order to accelerate convergence, or even to obtain an acceptable solution at all.

4.5 Convergence and Preconditioning

The convergence behavior of the aforementioned GMRES technique in the case of a normal matrix, A, is generally governed in the worst-case sense by the spectrum of A [158]. For a diagonalizable matrix $A = X\Lambda X^{-1}$, where Λ is the diagonal matrix of the eigenvalues λ_k of A and X is the matrix of corresponding eigenvectors, it can be shown that the worst-case rate of convergence is bounded. The bound on the convergence rate (the ratio of the residual norm at the n-th iteration to the original residual norm) is given by [158, 205]:

$$\frac{\|\mathbf{r}_n\|_2}{\|\mathbf{r}_0\|_2} \leq \kappa_2(X) \left\{ \min_{p \in \pi_n} \left[\max_k |p(\lambda_k)| \right] \right\}, \tag{4.7}$$

where $\kappa_2(X) = \|X\|_2\|X^{-1}\|_2$ (i.e. the condition number of the matrix of eigenvectors), π_n, is the set of polynomials with unit value at the origin and whose degree does not exceed n. This min-max problem is very difficult to solve, and unlike better understood symmetric-matrix algorithms, such as the conjugate gradient (CG) method, this expression does not reduce to a simple function of the condition number[6]. Rather, this bound depends on the shape of the eigenvalue distribution, rather than just its extrema [158]. Furthermore, despite the availability of some (generally overly pessimistic) bounds on the convergence rate[7], it is usually not possible to estimate the convergence rate of a linear system under GMRES without a priori knowledge of all the eigenvalues of A. Moreover, the convergence behavior of restarted GMRES(n) [133] or BiCGSTAB is even more difficult to predict. Hence, in practice, it is usually much easier to explore the properties of a particular class of linear systems under various solvers in an empirical manner. Nevertheless, knowing that the above estimate of the worst-case convergence behavior depends on the condition number of the eigenvector matrix X, it is reasonable to suspect that iterative solvers might benefit from some form of preconditioning when dealing with slowly converging systems.

The essential idea of preconditioning is to perform some easily[8] invertible transformation on the linear system of interest which results in a new problem whose eigenvalue distribution has a smaller spectral radius and thus has a smaller condition number. For the moment, we will not explore the exact form of this transformation, since developing effective preconditioning techniques is still an open topic across many disciplines and depends heavily on the type of matrix and the underlying physics describing the BIE we wish to solve. A preconditioning transform can be applied to the system of interest in several different ways, but the technique used in KEMField is called right-preconditioning, as the transformation

[6]The condition number κ of a matrix can be defined in terms of a ratio of its most extreme eigenvalues: $\kappa = \lambda_{\text{max}}/\lambda_{\text{min}}$.

[7]These bounds can be obtained with the assumption that the eigenvalues are bounded away from the origin in the complex plane within a region with a certain spectral radius (defined by the eigenvalue extrema)[98].

[8]We mean 'easy' in the sense that it is less computationally expensive to invert the preconditioning transformation P than the original system matrix A.

is applied on the right-hand side of the system matrix A. Letting P denote the preconditioning matrix, then when using right-preconditioning, we are interested in solving [205]:

$$AP^{-1}(P\mathbf{x}) = \mathbf{b} \, , \tag{4.8}$$

via

$$AP^{-1}\mathbf{z} = \mathbf{b} \tag{4.9}$$

and

$$P\mathbf{x} = \mathbf{z} \, . \tag{4.10}$$

While it is sometimes desirable to have a fixed explicit form for the preconditioner P and its inverse, is is not always possible to obtain or store an effective explicit preconditioner for large problem sizes. Instead, one alternative is to use what is called variable preconditioning, where the inverse action of the preconditioner is obtained by solving equation 4.10 using an iterative method. Applying this preconditioning scheme to GMRES results in what is known as flexible-GMRES (FGMRES) and its restarted variant FGMRES(n) [203]. FGMRES(n) as it is implemented in KEMField is outlined in algorithm 5. Similarly, the right preconditioned version of BiCGSTAB is referred to as BiCGSTAB-P [223] and is outlined in algorithm 6. BiCGSTAB-P can also be used with variable preconditioning, which is known as flexible-BiCGSTAB [45].

4.6 Conclusion

Tackling the Laplace boundary value problems posed by the KATRIN experiment requires an efficient linear equation solver. A wide variety of techniques for solving this type of problem are available, both direct and iterative. After considering the scaling behavior of each of these algorithms (summarized in table 4.1), we conclude that an algorithm which is both iterative and matrix-free is necessary in order to solve the problem with reasonable constraints on time and memory. We note that the scaling of the iterative techniques varies depending on the number of iterations k required to reach convergence. The exact value of k is usually much less than N but can be larger and depends heavily on the condition number of the matrix at hand. Also, since Krylov type iterative methods rely on the evaluation of the matrix-vector product, the scaling of these algorithms is proportional to the cost of this product. If the matrix is highly structured, the arithmetic cost of the matrix-vector product can be as low as $\mathcal{O}(N)$. However, in the naive worst case scenario, the evaluation of the matrix-vector product is $\mathcal{O}(N^2)$. Therefore, in the case of the GMRES algorithm, which is guaranteed to converge in at most $k = N$ iterations, the worst case arithmetic cost to solve the linear equation is $\mathcal{O}(N^3)$ for a completely unstructured matrix. Whereas for the BiCGSTAB algorithm, which has no guarantee of convergence and may never satisfactorily converge, its worst case arithmetic cost is infinite. While the worst case estimates for the arithmetic cost of BiCGSTAB and GMRES are somewhat discouraging, in practice many problems converge much more rapidly. The expected scaling of these methods is indicated in table 4.1.

Name	Type	Memory Scaling	Arithmetic Scaling
QR [102]	Direct	$\mathcal{O}(N^2)$	$\mathcal{O}(N^3)$
LU [102]	Direct	$\mathcal{O}(N^2)$	$\mathcal{O}(N^3)$
Gaussian Elimination [102]	Direct	$\mathcal{O}(N^2)$	$\mathcal{O}(N^3)$
Robin Hood [84]	Direct	$\mathcal{O}(N)$	$\mathcal{O}(N^2)$
BiCGSTAB [223]	Iterative	$\mathcal{O}(kN)$ to $\mathcal{O}(N^2)$	$\mathcal{O}(kN)$ to ∞
GMRES [204]	Iterative	$\mathcal{O}(kN)$ to $\mathcal{O}(N^2)$	$\mathcal{O}(kN)$ to $\mathcal{O}(N^3)$
BiCGSTAB-P [223]	Preconditioned Iterative	$\mathcal{O}(kN)$ to $\mathcal{O}(N^2)$	$\mathcal{O}(kN)$ to ∞
FGMRES [203]	Preconditioned Iterative	$\mathcal{O}(kN)$ to $\mathcal{O}(N^2)$	$\mathcal{O}(kN)$ to $\mathcal{O}(N^3)$

Table 4.1: Table of algorithms used to solve $A\mathbf{x} = \mathbf{b}$ and their scaling properties.

In closing, the two primary algorithms we will use to solve KATRIN's Laplace BEM problem are GMRES and BiGCSTAB. In order to exploit these Krylov subspace based methods and their preconditioned variants, we will need a means to compute the action of the matrix A (or the preconditioner P) on some arbitrary vector $\mathbf{v}$. While this requirement does not preclude the use of straightforward $\mathcal{O}(N^2)$ matrix-vector multiplication (presuming direct storage or calculation of all of the necessary matrix elements), there are much more effective methods available to compute the matrix-vector product for Laplace BEM problems. Despite the fact that A is generally an unstructured matrix, the Fast Multipole Method (FMM) can be used to compute the matrix-vector product with arithmetic cost of $\mathcal{O}(N \ln N)$ or even $\mathcal{O}(N)$. The FMM and its variants will form the basis of the engine we use to solve these large non-symmetric systems arising from the Laplace boundary value problem. The performance of these Krylov techniques in conjunction with the FMM will be explored in subsequent chapters.

Algorithm 5 Restarted FGMRES(n) algorithm to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: The matrix $A$, right hand side $\mathbf{b}$, and initial estimate $\mathbf{x}_0$.
 1: $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$                      ▷ Compute the initial residual.
 2: $p_0 = \|\mathbf{r}_0\|_2$ and $\mathbf{v}_0 = \mathbf{r}_0/p_0$        ▷ Normalize and augment Krylov subspace.
 3: $\rho = |p_0|$ and $j = 0$                 ▷ Initialize residual error and iteration count.
 4: while $\rho \geq \epsilon\|\mathbf{b}\|_2$ and $j \leq n$ do
 5:     $\mathbf{z}_j = P^{-1}\mathbf{v}_j$                  ▷ Apply variable preconditioner.
 6:     $\mathbf{w} = A\mathbf{z}_j$
 7:     for $i = 0, \ldots, j$ do              ▷ Gram-Schmidt orthogonalization.
 8:         $H_{i,j} = \langle \mathbf{w}, \mathbf{v}_j \rangle$
 9:         $\mathbf{w} = \mathbf{w} - H_{i,j}\mathbf{v}_j$
10:     end for
11:     $\beta = \|\mathbf{w}\|_2$ and $\mathbf{v}_{j+1} = \mathbf{w}/\beta$.   ▷ Normalize and augment Krylov subspace.
12:     for $i = 0, \ldots, j-1$ do      ▷ Apply Given's rotations on $j$-th column of $H$.
13:         $a = H_{i,j}, b = H_{i+1,j}$
14:         $H_{i,j} = c_i a + s_i b$
15:         $H_{i+1,j} = -s_i a + c_i b$
16:     end for
17:     $c_j = H_{j,j}/\sqrt{H_{j,j}^2 + \beta^2}$          ▷ Compute $j$-th Given's rotation.
18:     $s_j = \beta/\sqrt{H_{j,j}^2 + \beta^2}$
19:     $H_{j,j} = \sqrt{H_{j,j}^2 + \beta^2}$ and $H_{j+1,j} = 0$    ▷ Apply $j$-th Given's rotation to $H$.
20:     $p_j = c_j p_j$ and $p_{j+1} = -s_j p_j$         ▷ Apply $j$-th Given's rotation to $\mathbf{p}$.
21:     $\rho = |p_{j+1}|$ and $j = j + 1$       ▷ Update residual error and iteration count.
22: end while
23: $\mathbf{y} = \underset{\mathbf{y}\in\mathbb{R}^j}{\operatorname{argmin}}\|H\mathbf{y} - \mathbf{p}\|$          ▷ Solve $j \times j$ least-squares minimization.
24: $\tilde{\mathbf{x}} = \mathbf{x}_0 + \sum_{i=0}^{j} y_i\mathbf{z}_i$             ▷ Compute solution approximation.
25: if $\rho \leq \epsilon\|\mathbf{b}\|_2$ then exit.
26: else
27:     $\mathbf{x}_0 = \tilde{\mathbf{x}}$. goto 1:                                  ▷ Restart.
28: end if
Output: The approximate solution $\tilde{\mathbf{x}}$, with relative residual error $\rho \leq \epsilon\|\mathbf{b}\|_2$.
```

Algorithm 6 BiCGSTAB-P algorithm to solve $A\mathbf{x} = \mathbf{b}$.

```
Input: The matrix $A$, right hand side $\mathbf{b}$, and initial estimate $\mathbf{x}_0$.
1: $\hat{\mathbf{r}} = \mathbf{b} - A\mathbf{x}_0$                                  ▷ Compute the initial residual.
2: $\mathbf{r} = \hat{\mathbf{r}}, \widetilde{\mathbf{x}} = \mathbf{x}_0, \mathbf{p} = 0, \mathbf{v} = 0$                          ▷ Initialize values.
3: $\alpha = 1, \omega = 1, \rho = 1, \hat{\rho} = 1$
4: while $\|\mathbf{r}\|_2 \geq \epsilon\|\mathbf{b}\|_2$ do
5:     $\rho = \langle \mathbf{r}, \mathbf{r} \rangle$
6:     $\beta = (\rho/\hat{\rho})(\alpha/\omega)$
7:     $\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega\mathbf{v})$
8:     $P^{-1}\mathbf{p} = \mathbf{y}$                                  ▷ Apply preconditioner.
9:     $\mathbf{v} = A\mathbf{y}$
10:    $\alpha = \rho/\langle \hat{\mathbf{r}}, \mathbf{v} \rangle$
11:    $\mathbf{s} = \mathbf{r} - \alpha\mathbf{v}$
12:    $P^{-1}\mathbf{s} = \mathbf{z}$                                  ▷ Apply preconditioner.
13:    $\mathbf{t} = A\mathbf{z}$
14:    $P^{-1}\mathbf{t} = \mathbf{w}$                                  ▷ Apply preconditioner.
15:    $\omega = \langle \mathbf{z}, \mathbf{w} \rangle / \langle \mathbf{w}, \mathbf{w} \rangle$
16:    $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}} + \alpha\mathbf{y} + \omega\mathbf{z}$                  ▷ Update solution approximation.
17:    $\mathbf{r} = \mathbf{s} - \omega\mathbf{t}$                                  ▷ Update residual.
18:    $\hat{\rho} = \rho$
19: end while
Output: The approximate solution $\widetilde{\mathbf{x}}$, with relative residual error $\|\mathbf{r}\|_2 \leq \epsilon\|\mathbf{b}\|_2$.
```

Chapter 5

Introduction to the Fast Multipole Method and its Variants

5.1 Motivation and Development

The Fast Multipole Method (FMM) was introduced in the mid-1980s by Rokhlin [202] and Greengard [106, 107]. Initially, it was used as a means to accelerate the calculation of the two-dimensional Laplace boundary value problem but subsequently, as a rapid method for the Coulomb force field calculation in three-dimensional N-body problems. Since that time, it has been extended by numerous authors to cover a very wide range of physical systems and boundary integral equations, such as electrostatics, elastostatics, acoustic scattering, electromagnetic scattering and other problems [160].

The primary motivation behind the original development of the FMM is the long-range $(1/r)$ nature of the Coulomb potential. Since this potential. lacks a cut-off length beyond which its effects can be ignored, BEM and N-body problems governed by it, have a dense set of interactions. In other words, every element or particle is influenced by the individual presence of all the others. For boundary element problems, this results in dense linear systems which become computationally infeasible to solve for large N when using direct methods. Therefore, in order to make use of the boundary element technique for realistic three-dimensional problems, an algorithm with greatly reduced memory and arithmetic scaling is needed.

Since direct methods cannot be used for solving the linear systems arising from the Laplace BEM problem, when N is large, we will need to resort to an iterative approach such as the Krylov techniques summarized in the previous chapter. However, these iterative methods still have a computational cost which scales like $\mathcal{O}(kN^2)$, for k iterations, when using straightforward matrix-vector multiplication. Fortunately, Krylov subspace methods do not require access to the individual elements of the system matrix, A, only a black-box method of evaluating the action of A on an arbitrary vector $\mathbf{v}$. This provides an avenue we may exploit. Naively, the arithmetic cost of a general matrix vector product is $\mathcal{O}(N^2)$, but it is well-known that the matrix-vector products of certain structured matrices (e.g sparse, Fourier, Vandermonde, Toeplitz, etc.) can be evaluated with much less effort [100]. For example, discrete Fourier transforms only require $\mathcal{O}(N \log N)$ operations when using the fast Fourier transform (FFT) algorithm [102]. In order to reduce the complexity of the matrix-vector product involving our system matrix A (which is unstructured), we would like to replace it by an accurate approximation which may be applied in a more structured way. To do this, we split A into near-field (which cannot be accurately approximated) and far-field components:

$$A = A_n + A_f \,. \tag{5.1}$$

The near-field matrix, A_n, is a sparse matrix consisting of an $\mathcal{O}(N)$ number of pre-computed and stored diagonal and near-diagonal terms. Its action may be computed directly. The far-field matrix, A_f, is a diagonal-less dense matrix whose action must be computed in an approximate way without reference to any individual matrix elements. The FMM can be used to evaluate the action of A_f in a manner whose arithmetic costs scale like $\mathcal{O}(N \log N)$ or even $\mathcal{O}(N)$ in certain cases [32]. Such a reduction provides vastly accelerated processing times over the direct approach, and for many problems, it is the only practical approach to obtain a solution.

In order for the FMM to be applicable, it is necessary that the kernel governing the equation be separable or approximately separable. If this is so, then it may be expressed as a series [32]:

$$K(\mathbf{x}, \mathbf{y}) \approx \sum_{k=0}^{\infty} \psi_k(\mathbf{x}) \xi_k(\mathbf{y}) . \tag{5.2}$$

Any set of basis functions $\psi_k(\mathbf{x})$ and $\xi_k(\mathbf{y})$, satisfying equation 5.2 can be used for the expansion. Even a simple Cartesian Taylor series would be sufficient. However, in the case of the Laplace equation, the rotational invariance of the kernel makes it convenient to expand it in spherical coordinates, where the functions $\psi_k(\mathbf{x})$ and $\xi_k(\mathbf{y})$ take the form of the regular and irregular solid harmonics [78], [224]. While in principle, such an expansion is infinite, once an acceptable numerical precision has been specified, the series can be truncated at some maximum degree $k = p$. By expanding the kernel in this way, we can represent the far-field effects of a particular source with a finite set of coefficients. This compresses the field information into a series which is far less computationally expensive to evaluate than using the original source-kernel convolution form of equation 3.39.

Aside from the need to solve large scale Laplace BEM problems, an additional (and in our case, the original) motivation for the use of the fast multipole method is the need for fast field evaluation when tracking charged particles. Once the boundary value problem is solved, particle tracking simulations require evaluating the electric field and potential by summing over all the charge sources on the mesh. Since the arithmetic costs of this direct sum scales proportionally with the number of mesh elements, field evaluation can slow down particle simulations immensely when tracking in complex geometries. In fact, evaluation of the electric field alone is the primary bottleneck in all three-dimensional simulations of the KATRIN experiment.

5.2 Multipole Expansions

To develop the necessary expansion technique, we recall that in the indirect BEM formulation of the Laplace boundary value problem, the matrix equation arises from the application of a single layer potential convolved with the Neumann (eq. 3.75) or Dirichlet (eq. 3.42) kernels. Since the Neumann kernel, $K_N(\mathbf{r}, \mathbf{r}')$, is simply the result of the gradient operator applied to the Dirichlet kernel, it is appropriate to concentrate on the latter:

$$K_D(\mathbf{r}, \mathbf{r}') = \frac{1}{4\pi|\mathbf{r} - \mathbf{r}'|} . \tag{5.3}$$

Apart from the unimportant pre-factor of $1/4\pi$, this is simply the inverse Euclidean distance function, which can be expanded in terms of the spherical harmonics using [78, 129]:

$$\frac{1}{|\mathbf{r} - \mathbf{r}'|} = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \left(\frac{r_{\min}^{l}}{r_{\max}^{l+1}} \right) Y_l^m(\theta, \phi) Y_l^{-m}(\theta', \phi') , \tag{5.4}$$

where $r_{\min} = \min(r, r')$ and $r_{\max} = \max(r, r')$. The above expression can be derived (see [78]) by rewriting the distance formula using the law-of-cosines, then Taylor expanding and collecting terms of like-powers, the result of which is the generating function of the Legendre polynomials P_l. Equation 5.4 then follows from the application of the well-known addition formula of spherical harmonics [129, 73]. In equation 5.4 and throughout this thesis, the spherical harmonics, $Y_n^m(\theta, \phi)$, are defined using the Schmidt-semi-normalized convention following that of [107]:

$$Y_l^m = \sqrt{\frac{(l - |m|)!}{(l + |m|)!}} P_l^{|m|}(\cos\theta) e^{im\phi} , \tag{5.5}$$

where $P_l^{|m|}$ is the associated Legendre polynomial. We note that in this definition the Condon-Shortly phase has not been included and that the following conjugation relation between harmonics with same degree, l, but opposite sign order, m, holds:

$$Y_l^{-m}(\theta, \phi) = \overline{Y_l^m}(\theta, \phi) . \tag{5.6}$$

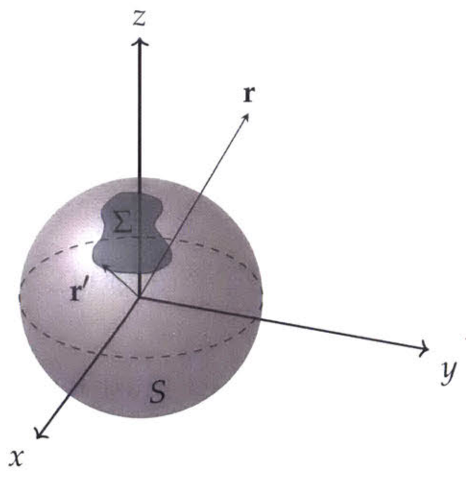


Figure 5-1: Geometry of the multipole expansion.

With the mathematical preliminaries out of the way, we can now consider their application to the Laplace boundary value problem consisting of the set of surfaces Γ. During the construction of the linear system governing this system, we would like to find a faster way to evaluate the far-field contributions due to some surface subset, $\Sigma \subset \Gamma$. We can assume without loss of generality that all source points $\mathbf{r}' \in \Sigma$ are closer to the origin than the evaluation point, $\mathbf{r}$, and that Σ is confined within a sphere, S, with radius, r_0, about the origin of the coordinate system. Hence we have $r_{\min} = r'$ and $r_{\max} = r$, figure 5-1 illustrates this situation. By applying the single-layer ansatz of equation 3.39 to Σ and replacing the Dirichlet kernel with the expansion 5.4 we find:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int_\Sigma \sigma(\mathbf{r}') \left[\sum_{n=0}^{\infty} \sum_{m=-n}^{n} \left(\frac{r'^l}{r^{l+1}} \right) Y_l^m(\theta,\phi) Y_l^{-m}(\theta',\phi') \right] d\Sigma \, . \tag{5.7}$$

This allows us to pull out the $\mathbf{r}$ dependence from under the integral sign yielding:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{Y_l^m(\theta,\phi)}{r^{l+1}} \int_\Sigma r'^l \sigma(\mathbf{r}') Y_l^{-m}(\theta',\phi') d\Sigma \, . \tag{5.8}$$

Therefore, in the region outside of the sphere, S, the potential can be expressed as the sum:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \frac{Q_l^m Y_l^m(\theta,\phi)}{r^{l+1}}, \tag{5.9}$$

where the l-th degree and m-th order multipole moment Q_l^m is given by the integral:

$$Q_l^m = \int_\Sigma r'^l \sigma(\mathbf{r}') \overline{Y_l^m}(\theta',\phi') d\Sigma . \tag{5.10}$$

As mentioned before, in practice, we must must terminate an expansion after some realistic number of terms. We will refer to an expansion which terminates at $l_{max} = p$ as an expansion of degree p. The number of terms in a p-degree expansion is $(p+1)^2$. However. due to the conjugation property (equation 5.6) of the spherical harmonics, the number of independent coefficients that need to be calculated is slightly less and is given by: $(p+1)(p+2)/2$. A p-th degree expansion produces an approximate value of the potential $\widetilde{\Phi}$:

$$\widetilde{\Phi}(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \sum_{l=0}^{p} \sum_{m=-l}^{l} \frac{Q_l^m Y_l^m(\theta,\phi)}{r^{l+1}}, \tag{5.11}$$

which has a truncation error ϵ with respect to the true potential $\Phi(\mathbf{r})$ bounded by [108]:

$$\epsilon = \left|\Phi(\mathbf{r}) - \widetilde{\Phi}(\mathbf{r})\right| \leq \alpha Q_{\text{tot}} \left| \left(\frac{1}{r_0 - r}\right) \left(\frac{r_0}{r}\right)^{p+1} \right| , \tag{5.12}$$

where α is a constant of proportionality which varies depending on the accuracy of the algorithm used to compute the multipole coefficients, and Q_{tot} is the sum of the absolute value of all the charges contained by S. Clearly, the larger the value of p, the lower the truncation error will be, albeit at a larger computational cost.

Now if we were limited to the use of a (remote) multipole expansion of the type described in equation (5.11), we would not have gained a great deal of efficiency over the direct matrix-vector evaluation when solving the BEM problem. Nor would we be offered much of a speed up during particle simulations unless we only need to track charged particles in a region external to all of the charge sources (mesh elements). Since the remote multipole expansion is a Laurent series in r, it converges far away from the origin of the expansion, so is not appropriate for evaluating the field in regions which are bounded by or contain charged sources, such as the KATRIN main spectrometer. Moreover, if we used the remote expansion in order to evaluate the field on collocation points or for tracking particles inside the electrodes of such a structure, we would necessarily have to decompose it into a large number of sub-collections, in order to create overlapping regions where each sub-collection's expansion could separately converge. In this case, rather than having to evaluate the field by summing over a large number of direct evaluations of each electrode, we instead would have to sum over some number, M, multipole expansions in order to calculate the field. Because M would necessarily be proportional to N, evaluating the field at all N collocation points of a BEM problem (evaluating the matrix-vector product) using this approach would still scale like $\mathcal{O}(N^2)$. So while this might be slightly faster than the direct method depending on the distribution of mesh elements, it is unlikely that it would provide a dramatic gain in speed.

However, as an alternative to the multipole expansion, one can also use its Taylor series analog, called the local coefficient expansion. The local coefficient expansion can be made to converge to the potential, Φ, within a sphere, S, with radius, r_0, due to all of the sources outside this sphere. That such an expansion can be constructed is easy to see in light of equation 5.7. If all of the charge sources on Σ are outside of S as in figure 5-2, then $r' > r$, and $r_{\min} = r$ and $r_{\max} = r'$. Once again pulling the $\mathbf{r}$ dependence outside of the integral we find:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \sum_{j=0}^{\infty} \sum_{k=-j}^{j} r^j Y_j^k(\theta,\phi) \int_\Sigma \frac{\sigma(\mathbf{r}') Y_j^{-k}(\theta',\phi')}{r'^{l+1}} d\Sigma \,, \tag{5.13}$$

from which we can see that the local coefficient expansion of the potential is given by:

$$\Phi(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \sum_{j=0}^{\infty} \sum_{k=-j}^{j} r^j L_j^k Y_j^k(\theta,\phi) \,, \tag{5.14}$$

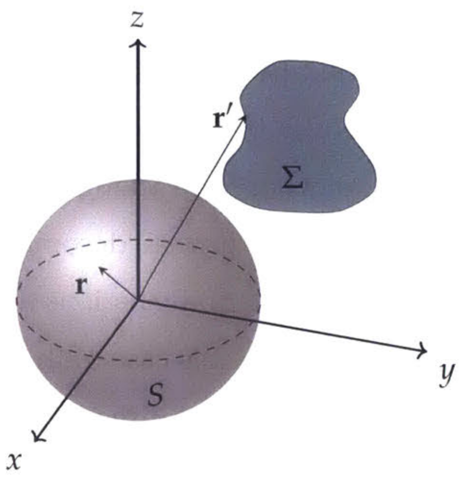


Figure 5-2: Geometry of the local coefficient expansion.

with the coefficients generated from:

$$L_j^k = \int_\Sigma \sigma(\mathbf{r}') \frac{\overline{Y_j^k}(\theta', \phi')}{r'^{j+1}} d\Sigma \,. \tag{5.15}$$

As could be expected, it can be shown the truncated local coefficient expansion has an upper error bound, which in a manner analogous to equation 5.12, decreases as the truncation degree, p, is increased.

Theoretically, if we were able to entirely fill the region of interest with overlapping balls of varying radius, each containing a local coefficient expansion of the charges external to it, we could guarantee a constant $\mathcal{O}(p^2)$ cost for any field evaluation[1]. Unfortunately, *directly* constructing such a set of expansions is not useful for solving the Laplace BEM problem in of itself, since it would require no less than $\mathcal{O}(N^2)$ operations during each matrix vector product. However, as was proven by Rokhlin and Greengard [202, 106, 107], it is possible to construct this set of local coefficient expansions *indirectly* from a collection of remote (multipole) expansions in a manner whose arithmetic costs are much less than the *direct* approach. In order

[1]As an added benefit with respect to the task of tracking charged particles, being able to cover the region of interest with a collection of local coefficient expansions can drastically reduce the number of computationally expensive direct integrations and greatly accelerate the field calculation. This method is not entirely different from forming a field map [54], with the exception that the coefficients of the interpolation are computed directly from the remote sources rather than inferred from local values of the field, and thus much more accurate and memory efficient.

to do so, it is necessary to have a technique to convert between remote and local expansions, as well as translate the expansion origin of a remote or local expansion.

5.3 The Transformation Rules

A straightforward implementation of a local coefficient field map would require the coefficients to be computed through direct integration over the charge sources, in a manner similar to computing the multipole moments. This would be very inefficient, as it would require roughly $\mathcal{O}(Np^2)$ integration operations for each local expansion needed about each of the N collocation points. However, three theorems regarding the translation and transformation of multipole and local coefficients can be used to develop a much better algorithm: the Fast Multipole Method [108]. The first two theorems, regarding the translation of the origin of a multipole or local coefficient expansion, have had a long history, having arisen during the course of development of quantum mechanics and quantum chemistry (see [213] and [58] for further discussion). The third theorem, due to Greengard and Rokhlin [107], allows us to transform a multipole expansion due to localized sources into a local coefficient expansion about another position. Detailed derivations of these theorems can be found in [78] and [224].

The first theorem we will consider describes the transformation of a multipole expansion about one origin, to that of another multipole expansion about a different origin. Visually, this is represented in figure 5-3 and is stated as follows:

Theorem 5.1 *Consider a multipole expansion with coefficients $\{O_n^m\}$ due to charges located within the sphere, D, with radius, a, centered about the origin. This expansion converges for points outside of sphere D. Now consider the point $Q = (\rho, \alpha, \beta) \notin D$. We may form a new multipole expansion, about the point, Q, due to the charges within D, which converges for points outside of the sphere D' which has its center at Q' and radius*

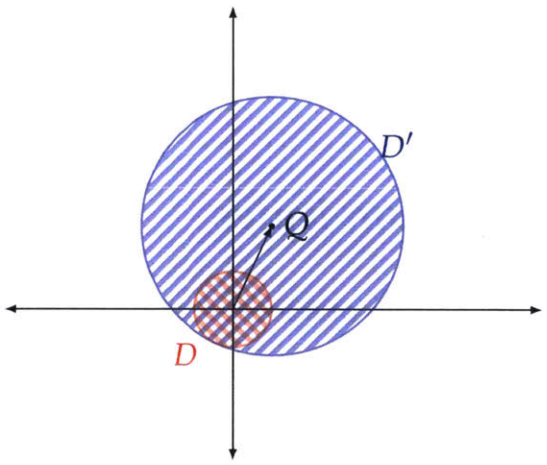


Figure 5-3: Multipole to multipole translation. The red shaded area indicates the region where the original multipole expansion $\{O_n^m\}$ does not converge. The blue shaded area indicates the region where the new multipole expansion $\{M_j^k\}$ does not converge.

$a' = \rho + a$. The coefficients of the new expansion multipole $\{M_j^k\}$ are given by:

$$M_j^k = \sum_{n=0}^{j} \sum_{m=-n}^{m=n} \frac{O_{j-n}^{k-m} i^{|k|-|m|-|k-m|} A_n^m A_{j-n}^{k-m} \rho^n Y_n^{-m}(\alpha, \beta)}{A_j^k}, \tag{5.16}$$

where

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)!(n+m)!}}. \tag{5.17}$$

The second theorem describes the conversion of a local coefficient expansion about one origin into a local coefficient expansion about another origin. Graphically, this is represented in figure 5-4 and stated as follows:

Theorem 5.2 *Consider a local expansion with coefficients $\{O_n^m\}$ due to charges located outside the sphere, D, with radius, a, centered about the origin. This expansion converges for points inside of sphere D. Now consider the point $Q = (\rho, \alpha, \beta) \in D$. We may form a new local coefficient expansion, about the point, Q, due to the charges outside D, which converges for points inside of the sphere D' which has its center at Q and radius $a' = a - \rho$. The coefficients of the new local coefficient expansion $\{L_j^k\}$ are given by:*

$$L_j^k = \sum_{n=j}^{p} \sum_{m=-n}^{m=n} \frac{O_n^m i^{|m|-|m-k|-|k|} A_{n-j}^{m-k} A_j^k \rho^{n-j} Y_{n-j}^{m-k}(\alpha, \beta)}{(-1)^{n+j} A_n^m}, \tag{5.18}$$

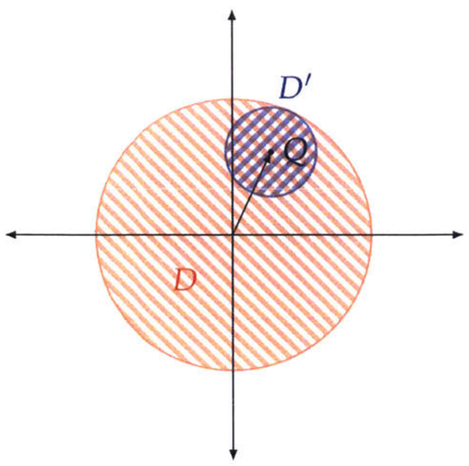


Figure 5-4: Local coefficient to local coefficient translation. The red shaded area indicates the region where the original local expansion $\{O_n^m\}$ converges. The blue shaded area indicates the region where the new local coefficient $\{L_j^k\}$ expansion converges.

where A_n^m is defined by equation 5.17.

Finally, the third theorem describes the conversion of a multipole expansion about one origin into a local coefficient expansion about another origin (see figure 5-5) as follows:

Theorem 5.3 *Consider a multipole expansion with coefficients $\{O_n^m\}$ due to charges located within the sphere, D, with radius, a, centered about the origin. This expansion converges for points outside of sphere D. Now consider the point $Q = (\rho, \alpha, \beta) \notin D$. We may form a local coefficient expansion , about the point, Q, due to the charges within D, which converges for points within the sphere D' which has its center at Q and radius $a' = \rho - a$. The coefficients of the local coefficient expansion $\{L_j^k\}$ are given by:*

$$L_j^k = \sum_{n=0}^{\infty} \sum_{m=-n}^{m=n} \frac{O_n^m i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{j+n}^{m-k}(\alpha, \beta)}{(-1)^n A_{j+n}^{m-k} \rho^{j+n+1}}, \tag{5.19}$$

where A_n^m is defined as before in equation 5.17.

Collectively, these three theorems form the backbone of the Fast Multipole Method.

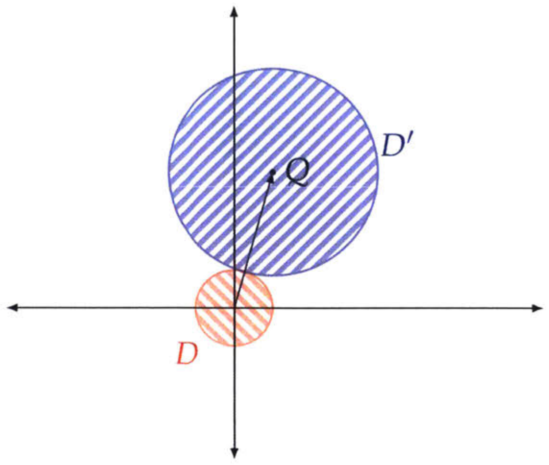
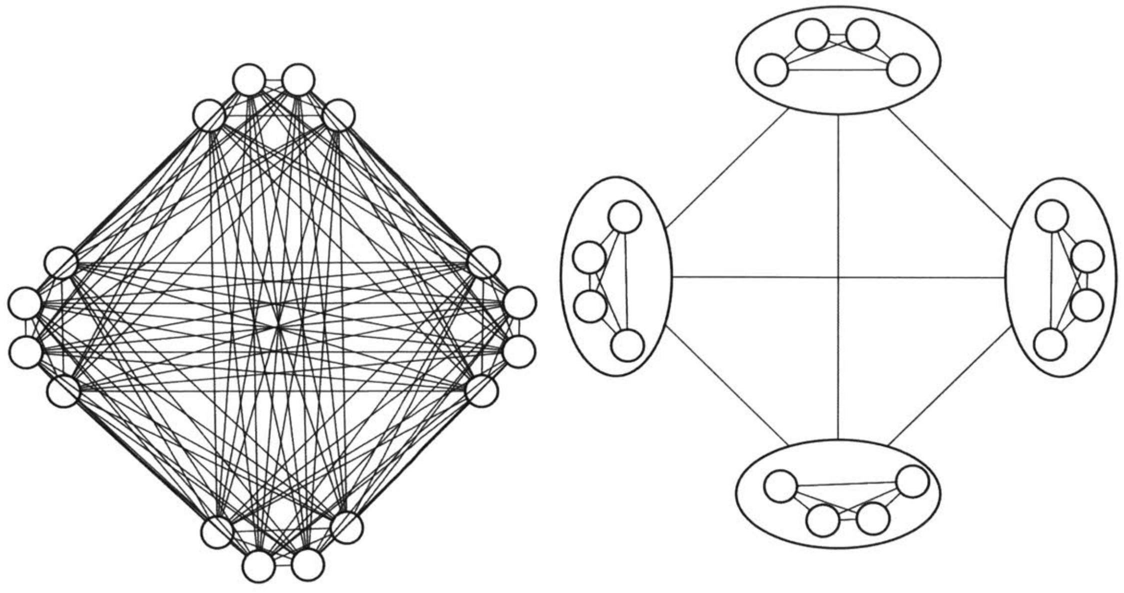


Figure 5-5: Multipole to local coefficient transformation. The red shaded area indicates the region where the original multipole expansion $\{O_n^m\}$ does not converge. The blue shaded area indicates the region where the new local coefficient expansion $\{L_j^k\}$ does converge.

5.4 Canonical Fast Multipole Method Algorithm

The Fast Multipole Method (FMM) has been continuously improved and applied to new problems over the past few decades and many versions exist. However, of primary importance is what we will refer to as the "canonical" algorithm as described in [108, 46]. The canonical fast multipole method makes use of all three of the transformation theorems in order to reduce the computational cost required to evaluate the matrix-vector product as much as possible. The sequence of operations is organized around a tree structure which adaptively subdivides space in order to reduce the number of operations needed. The use of a hierarchical tree to accelerate force calculations during the simulation of N-body problems was pioneered by Appel [11] and expanded upon by Barnes and Hut [25] . Unfortunately, the Barnes-Hut algorithm only makes use of the monopole term in the multipole expansion and as such, it generally lacks sufficient accuracy for many purposes. However, the tree structure of Barnes-Hut is extremely useful and serves as a basis about which the FMM operations can be structured. A very simple graphical argument motivating the use of a tree to accelerate the calculation of Coulomb-like interactions can be seen in figure 5-6. In this figure, the small white circles denote N charge sources,



(a) Direct pairwise interactions.

(b) Far-field interactions mediated through coarse grouping.

Figure 5-6: Using a tree structure to reduce the number of interaction calculations in an N-body problem.

and the lines joining them represent their interactions. In order to calculate the field at each source, we must consider the influence of all its neighbors. If we were to do this by summing over each pairwise interaction, as done in figure 5-6a, the number of arithmetic operations would grow like $\mathcal{O}(N^2)$. However, if we group nearby charges together, we can compute their influence on their far away neighbors as a single entity using a multipole expansion. In figure 5-6b, the near-field interactions (those between members of the same group) are calculated directly, while the far field is treated through the multipole interactions of the coarse groups (represented by the ellipses). In this way, even though only two tree levels have been used, the number of pairwise interactions can be drastically reduced. This type of hierarchical decomposition is the origin of the reduced arithmetic cost of the FMM.

While the details sometimes vary depending on the exact application, the procedure when using the FMM to evaluate a matrix-vector product typically boils down to performing the following sequential processes:

1. Construct the region tree and the associated element lists for each node.

2. Construct the multipole expansion for each mesh element about the center of their respective node, according to equation 5.10.

3. Gather the multipole expansions of each node into that of its parent, proceeding up the tree, using theorem 5.1.

4. Convert the multipole expansion of each node to a local coefficient expansion in its neighboring nodes, using theorem 5.3.

5. Distribute the local coefficient expansion of each node to its children, using theorem 5.2.

6. Evaluate the field at each collocation point due to the local coefficient expansion, using equation 5.14 and directly from sources in nearby nodes.

The first task is the construction of the region tree and associated node-element lists. For a static BEM mesh, the region tree only needs to be generated once at the beginning[2]. Clearly, the example we presented in figure 5-6 is greatly oversimplified, but the general idea is roughly the same in two or three-dimensions. However, rather than an ad-hoc subdivision of space, the hierarchical tree structure used by the FMM is based on an adaptive binary division of space in each dimension, which depends on the local density of mesh elements. When dealing with objects in three-dimensions, the tree-structure takes the form of an octree (called a quad-tree in two-dimensions). The basic node of an octree is a simple cubical volume coupled with a list of the mesh elements contained within it. The cube can be divided along its mid-planes in order to form eight child cubes, as shown in figure 5-7. The mesh elements may then be distributed to their respective child nodes which in turn may be further refined as necessary. The children can be identified by the indices 0 through 7. In order to distinguish between nodes (and their associated cubes and mesh element sets) at different levels of the tree, we will address them with a subscript consisting of the concatenation of their child index and that of their

[2]On the other hand, the multipole moments and subsequent local coefficient expansions must be recomputed any time there is a modification to the value of any of the charge sources on the mesh, so steps 2-6 must be repeated during each matrix-vector product.

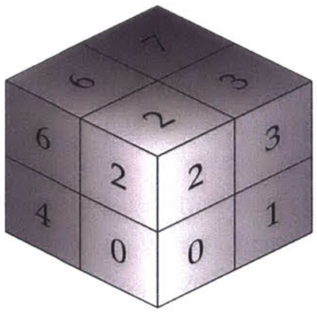


Figure 5-7: Cubical subdivision comprising the basic node structure of an octree. Child cubes are addressed from 0 to 7 and may be subdivided in turn. From this orientation child cube 5 is positioned in the back and is not visible.

parents. We will denote the nodes by $\mathcal{N}$, their associated cubes by $\mathcal{C}$, and mesh element sets by $\mathcal{S}$. For example, the cube associated with the root node would be denoted as $\mathcal{C}_0$ and the address, $\mathcal{S}_{014}$, would represent the mesh element set of the 4-th child of the 1-st child of the root node. In an octree, the lowest level of the tree is simply the cubic bounding volume of the region of interest containing the entire discretization $\mathcal{T}_N(\Gamma)$ of the boundary Γ. Each subsequent level subdivides those child volumes which contain more than some acceptable number, s, of mesh elements. In order to avoid excessive memory usage, node creation is typically terminated after a certain number of levels, or once the node volume is comparable with the size of the mesh elements. The general tree construction routine can be defined recursively, as listed in algorithm 7. Figure 5-8 graphically demonstrates the concept of how such a data structure is applied to a BEM problem. For clarity, we have limited the illustration to two-dimensions by applying a quad-tree with only three levels of subdivision to a plane curve discretized into linear segments. The node addressing scheme is exhibited for the first few levels of the quad-tree.

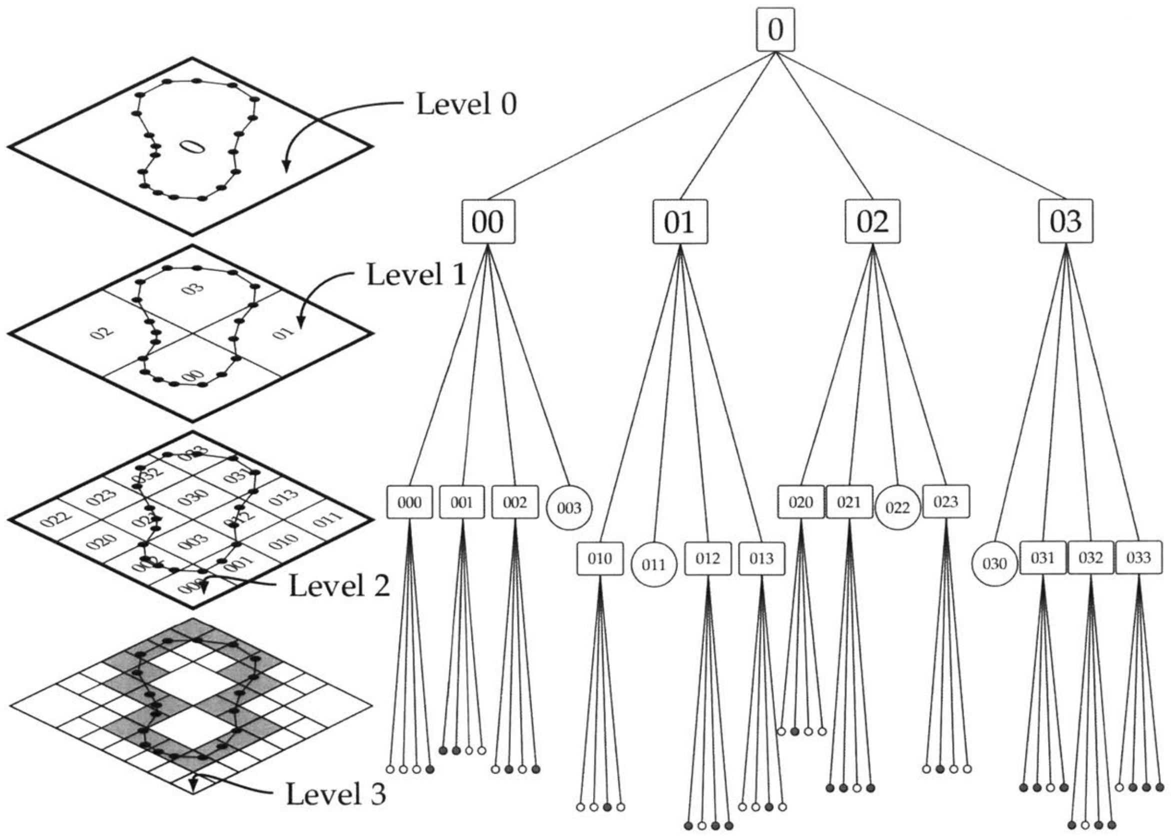
Once the region tree has been constructed, we can then proceed to compute the multipole moment expansions and convert them to local coefficient expansions. To do this, we will need to apply the multipole-to-multipole, multipole-to-local, and local-to-local transformation theorems systematically over the tree. We should note that the root node and its immediate children do not participate in any of the transformation operations since they are too close together. However, in the dia-

Algorithm 7 Recursive construction of region tree

```
Input: Mesh element set $\mathcal{S}_0 \subseteq \mathcal{T}_N(\Gamma)$, and bounding cube $\mathcal{C}_0$ with length $\ell$.
 1: Set tree level $n := 0$, initial address $i = 0$.
 2: procedure: Subdivide($\mathcal{S}_{[i]}$, $\mathcal{C}_{[i]}$, $n$)
 3:   Compute $r_{[i]}$, the size of smallest mesh element in $\mathcal{S}_{[i]}$.
 4:   if $|\mathcal{S}_{[i]}| > s$ and $r_{[i]} < \ell/2^n$ and $n < n_{\max}$ then
 5:     Divide $\mathcal{C}_{[i]}$ into 8 child cubes $\mathcal{C}_{[i]j}$ of length $\ell/2^{n+1}$.
 6:     Sort elements of $\mathcal{S}_{[i]}$ into the 8 subsets $\mathcal{S}_{[i]j}$.
 7:     for $j = 0 \ldots 7$ do
 8:       call Subdivide($\mathcal{S}_{[i]j}$, $\mathcal{C}_{[i]j}$, $n+1$)        ▷ Recurse on the children.
 9:     end for.
10:   else return.
11: return.
Output: The adaptively subdivided octree for the region contained by $\mathcal{C}_0$.
```

grams which follow we have included them in order to make the data flow pattern clearer. In order to define the necessary set of operations, an exact categorization of the tree nodes is needed. The canonical algorithm divides the node collection into three categories [32] with respect to each node. The first category is that of the near-neighbors. For a given cube, its set of near-neighbors are those nodes which are at the same level of tree refinement and which share a face, edge or corner with it. The second category is that of the well-separated nodes. These are the compliment to the set of near-neighbors, being those at the same tree level which do not have any boundary points in common. Finally, the last category is that of the nodes which are part of the interaction-list. These nodes are the children of the near-neighbors of the selected nodes parent. Figure 5-9 demonstrates how each one of these categories are defined with respect to a given node.

At the lowest level of the tree, the calculation of the multipole expansions is very straightforward, as it is simply the evaluation of equation 5.10 for each mesh element about its respective node center. Once the multipole moments are known for each element owned by a leaf node, they can be summed to form the multipole expansion for each leaf node. Then, theorem 5.1 can be applied to collect each child node's multipole expansion into its parents expansion. This collection process is known as the "upward-pass", because the multipole information passes from the



(a) Geometric representation (b) Tree representation.

Figure 5-8: Adaptively refined hierarchical quad-tree construction applied to a boundary curve in two dimensions. Shaded regions represent leaf nodes with non-empty mesh element lists. Rectangles represent composite nodes and circles represent leaf nodes. Node labels at the lowest tree level are not shown for the sake of clarity.

children to their parents. Graphically this process is represented in figure 5-10.

Once a multipole expansion has been formed for each node, its far-field influence can then be converted into local coefficient expansion within other nodes. This processes is carried out using theorem 5.3. Since the error on the multipole expansion (and its local coefficient representation) increases with decreasing distance from the source, it is important that we apply this conversion only to nodes that are far enough away in order to maintain the required accuracy. For a selected node on some tree level, only those nodes which are in the interaction-list participate in the multipole-to-local conversion process. Graphically, the multipole-to-local

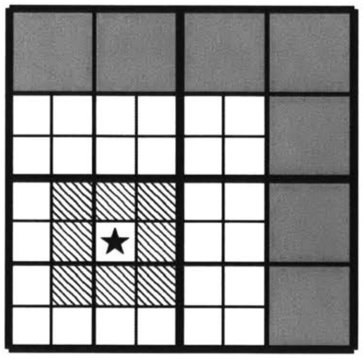
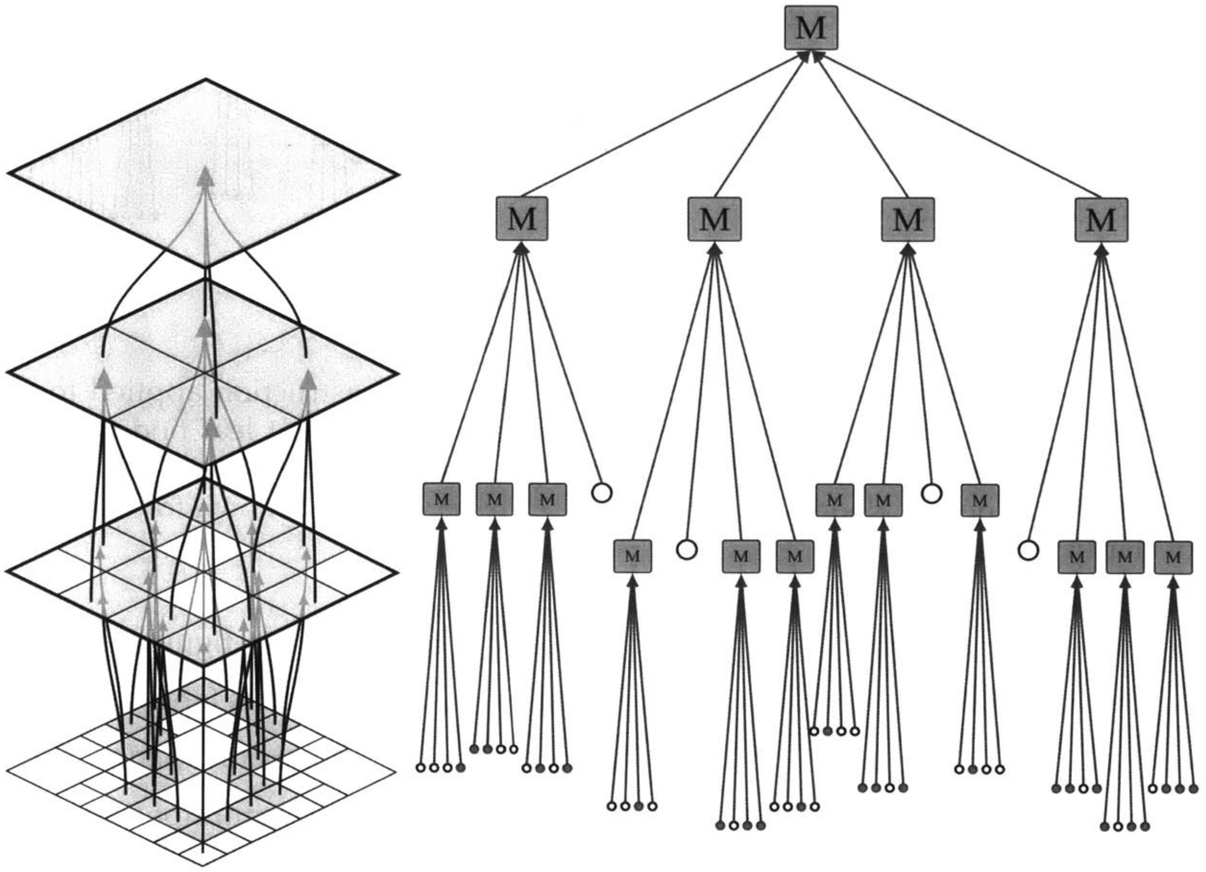


Figure 5-9: Categorizing various nodes according to their relative position with respect to the node of interest (marked with a star). The near-neighbors are shaded with diagonal lines, well-separated nodes are shaded gray, while the nodes which are on the interaction-list are colored white.



(a) Geometric representation (b) Tree representation

Figure 5-10: Upward pass: Compute multipole moments and gather them into larger collections through translation from child to parent.

conversion for a single node is demonstrated in figure 5-11. For the sake of simplicity, figure 5-11, only displays the multipole-to-local conversion operation for a particular node, at a single tree level (level 2).

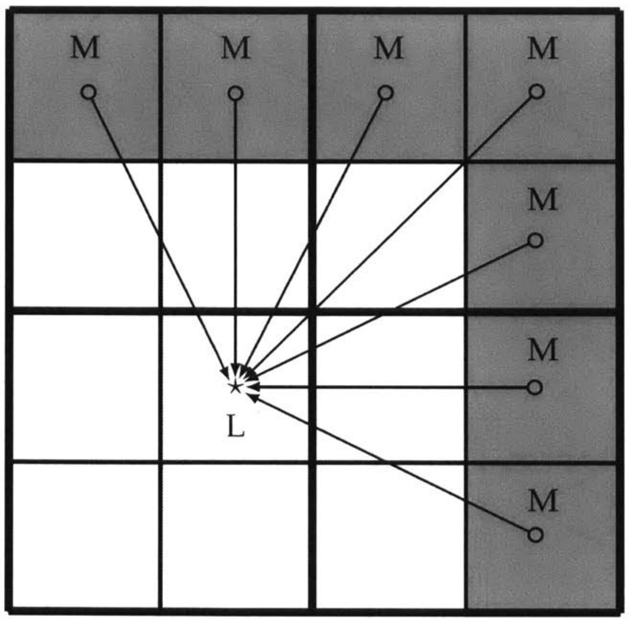
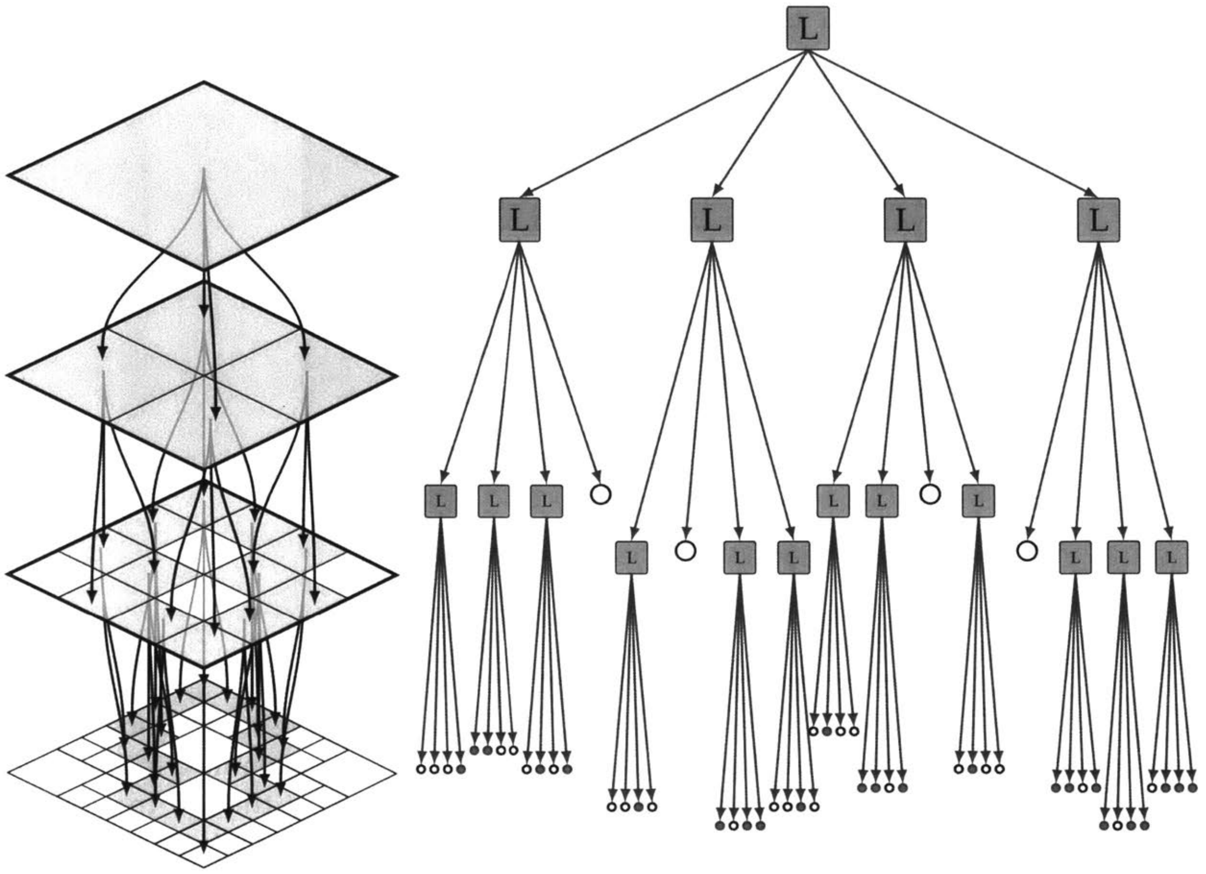


Figure 5-11: Demonstration of the conversion of several multipole expansions into a local coefficient expansion for a particular node. The solid lines denote a multipole-to-local transformation which is carried out between nodes at the 2nd tree level. This conversion is only applied to the shaded nodes (in the interaction-list). The unshaded nodes are too close to participate in the multipole-to-local operation at this level of the tree.

Next, in order to reduce the number of expansions which must be evaluated in order to compute the field, a "downward-pass" is performed. The downward-pass uses theorem 5.2 to re-expand the local coefficient expansion of each parent about the center of each of its child nodes. Figure 5-12 illustrates this operation. The down-converted local coefficient expansions are constructed about each child cube's center from the parent local coefficient expansion and account for all of the charge sources in the well-separated region. This process allows the far-field to be evaluated solely from a single, local-coefficient expansion in each leaf node, rather than requiring us to sum over the local coefficient expansions of all its parents.

Finally, in order to complete the evaluation of the matrix-vector product, the field must be computed at each collocation point. To do this, we first evaluate the local coefficient expansion of the node of interest which contains the collocation point. This provides the contribution from all the charged sources in the far-field,



(a) Geometric representation (b) Tree representation

Figure 5-12: Downward pass: Translation of local coefficient expansions from parent to child nodes.

which are contained in the well-separated nodes and nodes on the interaction-list. We then sum the far-field contribution with the effects of all nearby sources. The nearby sources are those mesh elements which are contained by the node of interest or by any of its near-neighbor nodes (see figure 5-9). The field evaluation from nearby sources must be evaluated directly through either analytic or numerical integration, in order to preserve accuracy.

5.4.1 Complexity/Scaling of the Canonical Algorithm

Given how complicated the fast multipole method is, it is perhaps natural to be somewhat suspicious about how well it can reduce the needed computational effort over the naive approach. However, it can be shown that the number of arithmetic operations for a single FMM-based matrix-vector product is approximately given by [32]:

$$189\left(\frac{N}{s}\right)p^4 + 2Np^2 + 27Ns\,, \tag{5.20}$$

where N is the number of mesh elements, p is the degree of the expansions used, and s is the average number of collocation points (mesh elements) per box at the finest tree level. Typically, the computer run time is dominated by the multipole-to-local transformation term since it is proportional to p^4 and obtaining accurate results demands high values of p. Large p is required because in three-dimensions, the multipole error decreases rather slowly, proportional to $(\sqrt{3})^{-p}$ [32]. Various schemes have been proposed to reduce the cost of the multipole-to-local transformation and increase the accuracy of the FMM when using smaller values of p. One method to do this is to diagonalize the transformation operator through the application of a rotation operation [46], which reduces the number of operations used in the multipole-to-local operation from $\mathcal{O}(p^4)$ to $\mathcal{O}(p^3)$. Another technique, developed by Elliot and Board [77], recognized that the multipole-to-local transformation operator could be represented as a discrete convolution (over the degree and order of the spherical harmonics) and accelerated with the fast Fourier transformation to obtain $\mathcal{O}(p^2 \log p)$ complexity. Unfortunately, because their algorithm requires the normalization terms of the response functions to be factored out, it is numerically unstable for moderate to large values of p. This is precisely where it would otherwise be most advantageous and provide the largest speed-up.

5.5 The Fast Fourier Transform on Multipoles (FFTM) Variant

Ong et al. [180] took a slightly different tack compared to Elliot et al. [77], when applying the FFT to accelerate the multipole-to-local transformation. Their algorithm is known as the Fast Fourier Transform on Multipoles (FFTM). The development of the FFTM relies on the realization that the multipole-to-local conversion operation is in essence a convolution not only over the degree and order of the basis functions, but also over the spatial coordinates. It is well-known that discrete convolution operations can be greatly accelerated through the use of the convolution theorem and the Fast Fourier Transform (FFT) algorithm. Unfortunately, the FFT requires the data to be transformed to be sampled on a uniform grid, which is decidedly not the case for the adaptively generated octree structure of the FMM. However, if we are willing to sacrifice the adaptive hierarchical subdivision of space in favor of a uniform subdivision, we can make use of the convolution theorem to quickly perform the multipole-to-local conversion for all nodes at once. The well-known discrete convolution theorem can be stated as follows [48]:

Theorem 5.4 *Let* $\mathbf{u} = [u_0, u_1, \ldots, u_{\lambda-1}]$ *and* $\mathbf{v} = [v_0, v_1, \ldots, v_{\mu-1}]$ *be two sequences of complex numbers of length* λ *and* μ *respectively. The discrete cyclic convolution (without aliasing) of these two sequences is defined as:*

$$\mathbf{w} = \tilde{\mathbf{u}} \otimes \tilde{\mathbf{v}} = [w_0, w_1, \ldots, w_N] \,, \tag{5.21}$$

with

$$w_n = \sum_{k=0}^{N} \tilde{u}_k \tilde{v}_{n-k} \,, \tag{5.22}$$

and where $\tilde{\mathbf{u}}$ *and* $\tilde{\mathbf{v}}$ *are the augmented (zero-padded) sequences of total length* $N = \lambda + \mu - 1$*, given by:*

$$\tilde{u}_i = \begin{cases} u_i & \text{if } 0 \le i < \lambda \\ 0 & \text{if } \lambda \le i < N \end{cases} \quad \text{and} \quad \tilde{v}_i = \begin{cases} v_i & \text{if } 0 \le i < \mu \\ 0 & \text{if } \mu \le i < N \end{cases} . \tag{5.23}$$

Denoting the discrete Fourier transform of a sequence as $\mathcal{F}(\cdots)$ *and its inverse transform as* $\mathcal{F}^{-1}(\cdots)$*, then the discrete cyclic convolution is given by:*

$$\mathbf{w} = \tilde{\mathbf{u}} \otimes \tilde{\mathbf{v}} = \mathcal{F}^{-1}\left(\mathcal{F}(\tilde{\mathbf{u}}) \odot \mathcal{F}(\tilde{\mathbf{v}})\right) \,, \tag{5.24}$$

where the operator $\odot$ *denotes the point-wise multiplication of the two sequences.*

It can be shown that the arithmetic cost to evaluate equation 5.21 directly using 5.22 scales like $\mathcal{O}(N^2)$, whereas performing the convolution indirectly through use of equation 5.24 and an FFT (such as the Cooley-Tukey algorithm [52]) results in an arithmetic cost proportional to $\mathcal{O}(N \log N)$. Furthermore the discrete convolution theorem can be extended to multidimensional data in a straightforward manner [68]. The efficient evaluation of the discrete convolution by theorem 5.4 motivates the observation of Ong et al. [180] as follows:

Observation 5.1 *Consider a three dimensional grid, indexed by $0 \leq i, j, k < d$, which consists of a set of d^3 cubes of length ℓ and centers $(x_i, y_j, z_k) = \ell(i + \frac{1}{2}, j + \frac{1}{2}, k + \frac{1}{2})$. If we were to form the multipole expansion due to the charges within each cube about its center, and proceed to compute the influence of each multipole expansion on the local coefficient expansion about every other cube center, then using the naive direct implementation of theorem 5.3, would require an arithmetic cost that would scale like $\mathcal{O}(p^4 d^6)$. However, observe that the functional form of this operation is essentially a three dimensional discrete convolution, which we may express as:*

$$L_j^k(x,y,z) \approx \sum_{n=0}^{p} \sum_{m=-n}^{n} \left[\sum_{x' \neq x} \sum_{y' \neq y} \sum_{z' \neq z} M_n^m(x',y',z') \times T_{j,n}^{m,k}(x - x', y - y', z - z') \right] \quad (5.25)$$

with response functions defined as:

$$T_{j,n}^{m,k}(\rho, \alpha, \beta) = \frac{i^{|k-m|-|k|-|m|} A_n^m A_j^k Y_{j+n}^{m-k}(\alpha, \beta)}{(-1)^n A_{j+n}^{m-k} \rho^{j+n+1}}, \quad (5.26)$$

and (ρ, α, β) being the spherical coordinates of the point $(x - x', y - y', z - z')$. Now, let the three dimensional array $\mathbb{M}_n^m$ be the multipole moments associated with each cube in the $d \times d \times d$ grid, and $\mathbb{T}_{j,n}^{m,k}$ be the associated response functions (spanning a grid of size $2d \times 2d \times 2d$). Then the local coefficients associated with each cube in the grid may be computed with an arithmetic cost proportional to $\mathcal{O}(p^4 (d \log d)^3)$ through the use of *theorem 5.4 as:*

$$\mathbb{L}_j^k = \sum_{n=0}^{p} \sum_{m=-n}^{n} \widetilde{\mathbb{M}}_n^m \otimes \widetilde{\mathbb{T}}_{j,n}^{m,k} = \mathcal{F}^{-1}\left(\sum_{n=0}^{p} \sum_{m=-n}^{n} \mathcal{F}(\widetilde{\mathbb{M}}_n^m) \odot \mathcal{F}(\widetilde{\mathbb{T}}_{j,n}^{m,k}) \right), \qquad (5.27)$$

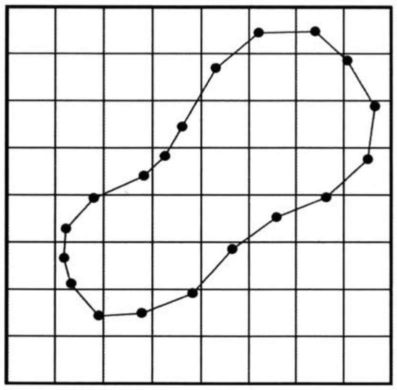
where $\widetilde{\mathbb{M}}_n^m$ and $\widetilde{\mathbb{T}}_{j,n}^{m,k}$ are the augmented (zero-padded) arrays and $\mathcal{F}(\cdots)$ denotes the three-dimensional discrete Fourier transform.

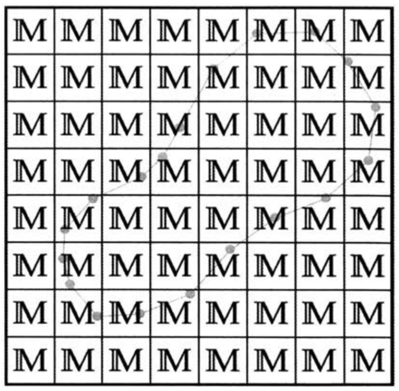
The FFTM algorithm is considerably simpler to implement than the FMM because it does not rely on the multipole-to-multipole and local-to-local translation theorems. Instead, it only requires the following four steps:

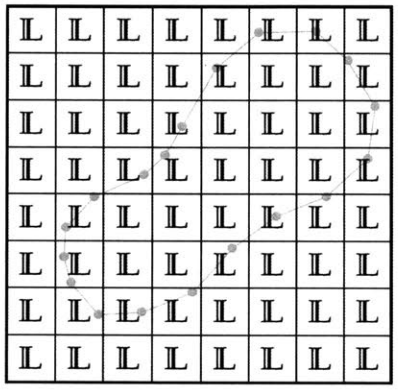
1. Subdivide the region of interest equally in each dimension into smaller cubes.
2. Construct the multipole expansion for each mesh element about the center of their respective cube, according to equation 5.10.
3. Convert the multipole expansion of each node to a local coefficient expansion in all other nodes (excluding its neighbors), using theorem 5.4 and observation 5.1.
4. Evaluate the field at each collocation point due to the local coefficient expansion using equation 5.14 and directly from sources in nearby nodes.

Graphically, the process is shown collectively in figure 5-13. Since the subdivision must be uniform, there is no use for a hierarchical tree structure. Instead, we can simply sort the mesh elements into the appropriate boxes and compute the multipole expansions about their centers. After this operation, we are left with a cubic grid of multipole expansions (figure 5-13b).

Next, in order to perform the multipole-to-local coefficient transformation we apply the convolution theorem. Before doing so, we must compute the response functions, $T_{j,n}^{m,k}$, according to theorem 5.26 and transform them. Additionally, before Fourier transforming the response functions, it is necessary to mask off those which are too close to the origin. This masking off is done by zeroing out all of the response


(a) Region subdivision.


(b) Formation of multipole expansions.


(c) Conversion of multipole expansions into to local coefficients through convolution with response functions.

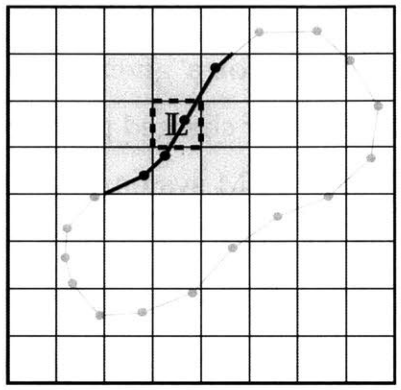

(d) Field evaluation within the dashed box due to local coefficients and nearby mesh elements (bold).

Figure 5-13: FFTM Algorithm.

functions within some number of subdivisions, z, of the origin. Typically, the zero-mask size $z = 1$. It is also important to zero pad the array of multipole expansions and the response functions out to the appropriate length before applying the Fourier transform. Since the discrete Fourier transform assumes that the data is periodic, zero padding is needed in order to avoid data corruption from wrap-around effects.

To compute the convolution, we then perform the point-wise multiplication between the transformed multipole array and the transformed response function array and sum according to theorem 5.4 and observation 5.1. Following that, we then perform the inverse Fourier transformation on the result, to obtain the local coefficients at the center of each subdivision as in figure 5-13c. Finally, we may compute the field at any collocation point through the evaluation of the local coefficient expansion of its respective node and a direct sum over the nearby elements (those which are contained in neighboring cubes within z subdivisions). Figure 5-13d demonstrates the evaluation of the field within the bold dashed box, due to the sum of its local coefficient expansion and direct integration from sources inside its immediate neighbors (within the shaded zero-mask region). For a more concise description of the FFTM algorithm along with some performance metrics, see references [180] and [159].

It is important to point out that while the FFTM is generally more accurate than the FMM for a given maximum expansion degree p (which is extremely advantageous for charged particle tracking), the arithmetic work required by FFTM to evaluate a BEM system matrix-vector product does not have the same $\mathcal{O}(N)$ guarantee as the FMM. This is because it lacks a hierarchical decomposition of space, which allows us to reduce the number of nearby mesh elements about each collocation point until it is roughly equal to some constant, s, which is independent of the total number of mesh elements N. For a uniform subdivision of space, the average number of nearby elements is still proportion to N/d^3, which implies the arithmetic cost of the matrix-vector product still scales like $\mathcal{O}(N^2)$, albeit with a greatly reduced pre-factor. If we were able to increase the number of divisions, d, proportionally with N without limit, then the FFTM algorithm could have scaling properties similar to the FMM. However, because this rapidly increases the space to store the response functions, computer memory limitations ensure d cannot scale without limit. This leads us to conclude that any practical algorithm which scales well for large BEM problems must make use of some sort of hierarchical spatial decomposition in a way similar to the canonical FMM. Therefore, in order to exploit the advantages of each approach we will develop at hybrid algorithm incorporating both the canonical FMM and the FFTM.

Chapter 6

The Hybrid Fast Fourier Multipole Method (HFFMM), a FFTM/FMM Variant

6.1 Introduction and Motivation

The fast multipole algorithm provided by `KEMField` is a novel variation on the algorithm introduced by Greengard and Rokhlin and is a hybrid of both FFTM and FMM. The FMM relies on an octree partitioning of space which provides a very efficient use of memory resources. However, it tends to have a higher cost in performing the multipole-to-local coefficient transformation and field evaluation, because it requires a larger expansion degree, p, to obtain the same accuracy as FFTM. FFTM on the other hand, divides space into a fine but uniform grid. This means that problems with a large number of degrees of freedom, N, tend to require an increasing number of spatial subdivisions, d, to appropriately reduce the required number of near-field integrations. Unfortunately, the value of d tends to be limited due to computer resources because of the amount of memory needed to store the response functions and expansion coefficients scales like d^3. This ultimately limits the number of degrees of freedom we can solve effectively with FFTM.

This is a substantial disadvantage for the purposes of KATRIN, because the main spectrometer requires a high number of mesh elements to represent it accurately. Storing the response functions for a uniform grid with a discretization scale on the order of the smallest features of the main spectrometer would have prohibitive memory requirements, similar to that of a finite difference method. On the other hand, for a grid whose resolution is limited by reasonable memory constraints on the response functions, there would be too many direct evaluations (required in order to maintain accuracy due to nearby mesh elements) to provide a useful acceleration of the field computation. These limitations makes FFTM unappealing by itself, but motivates us to seek a way to combine of the best properties of FFTM and the FMM.

The hybrid algorithm in use by `KEMField` (which we will refer to as the Hybrid Fast Fourier Multipole Method (HFFMM)), couples the accelerated multipole-to-local coefficient conversion of FFTM with the adaptive refinement available to the FMM algorithm by sub-dividing space into a d^3-tree. In the FMM, the number of spatial divisions of any node is always $d = 2$ in each dimension, resulting in an octree. In our algorithm, the number of spatial divisions, $d \geq 2$, is a user selectable parameter. The upward (collection of multipoles into coarser groupings) and downward pass (distribution of local coefficients into finer regions) stages of the hybrid algorithm are functionally the same as that of FMM. The major differences being the number of children each node in the tree contains and the fact that the multipole to local coefficient conversion for neighboring nodes at the same depth in the tree is performed using FFTM. This technique allows us to avoid the high memory usage associated with FFTM when using a fine spatial discretization, but affords us the benefit of FFTM's generally greater speed and accuracy when computing the multipole to local conversion. The disadvantage of this hybrid algorithm is the relatively large number of user controllable parameters, which while allowing greater flexibility, tends to complicate the search for a set of parameters which achieves the best compromise between speed, accuracy, and memory usage.

6.2 Algorithm

The hybrid algorithm proceeds very similarly to the FMM, though the details of each step vary. The main steps are as follows:

1. Preprocess the geometry by computing the centroid and bounding balls of all mesh elements.

2. Construct the region tree according to rules 6.1, 6.2, 6.3, and 6.4, and create the associated mesh element lists for each node. Prune leaf nodes that do not contain any collocation points or mesh elements.

3. Construct the multipole expansion for each mesh element about the center of the cube associated with the node which *owns* it, according to equation 5.10.

4. Gather the multipole expansions of each node into that of its parent, proceeding up the tree, using theorem 5.1.

5. Convert the multipole expansion of each node to a local coefficient expansion in its neighboring nodes, using FFTM from observation 5.1.

6. Distribute the local coefficient expansion of each node to its children, using theorem 5.2.

7. Evaluate the field at each collocation point due to the local coefficient expansion using equation 5.14 and also directly from the mesh elements in the near-field.

The first task of preprocessing the geometry is more a matter of practical implementation than an absolutely necessary step. Preprocessing is mainly intended to simplify the construction of the region tree by making the mesh element and sub-region association simpler to define. The first half of the preprocessing step is to compute all of the centroids of the discretization's mesh elements. This is a simple matter since these are already known, being the collocation points in our BEM implementation. Secondly, we must calculate the minimum bounding sphere of each mesh element in order to know how to place it in the region tree. A spherical bounding volume is convenient because it entails a very small amount of additional information, and it roughly conforms to the region of space where a particular mesh element's multipole expansion is excluded from convergence.

6.2.1 The Tree Structure

Once the mesh has been processed, we can go on to construct the region tree. However, before proceeding further, it is worthwhile to take a moment to clarify our notation and outline the necessary components of the region tree and its nodes. While our primary goal is to solve BEM problems in three-dimensions, there is nothing about the tree construction which is specific to $\mathbb{R}^3$. Therefore, we will consider the general case of D dimensions when describing the tree structure. Throughout the following we will refer to hypercubes as simply cubes, and hyperspheres as balls.

Every node which is a member of the tree can be denoted by $\mathcal{N}_i$. The subscript i is the global node address, which is generated by concatenating every local address of all the node's parents, in order, from the the root node to the node of interest. Since the tree,[1] $\mathcal{N}_0$, is intended to adaptively subdivide space, each node in the tree must be associated with a region. Naturally, the root node is associated with the cube, $\mathcal{C}_0 \subset \mathbb{R}^D$, which represents a bounding cube enclosing the region of interest. The cube, $\mathcal{C}_0$, is centered upon the point, $\mathbf{p}_0 \in \mathbb{R}^D$, and has side length ℓ. When any node in the tree is subdivided, d^D children are produced. For any given node, $\mathcal{N}_i$, we may reference any of its immediate children by appending the child's index, j, using the following notation: $\mathcal{N}_{[i]j}$, where $j = \{0, 1, \ldots, d^D - 1\}$. The child's index j serves to distinguish it amongst its parent's immediate children and is also known as its local address (being the last index of its global address). The region cubes associated with each of these children are produced by dividing their parent's cube into d equal parts along each of the D dimensions, producing a D-dimensional array.

[1]Because every node in the tree is accessible through the root node, we can refer to the complete region tree by referencing the root node, $\mathcal{N}_0$.

Additionally, the spatial location of each child can be related to its local address, j, through the application of row-major ordering to this array. For example, the j-th immediate child of the root node, $\mathcal{N}_{[0]j}$, is associated with the cube, $\mathcal{C}_{[0]j}$, which has a side length, ℓ/d, and is centered upon the point:

$$\mathbf{p}_{[0]j} = \mathbf{p}_0 + \sum_{\kappa=0}^{D-1} \ell \left[\left(\frac{a_\kappa}{d} + \frac{1}{2d} \right) - \frac{1}{2} \right] \hat{\mathbf{e}}_\kappa \; . \tag{6.1}$$

The indices $\{a_\kappa\}$ indicate the spatial location of a node in the child array and are all in the range $\{0, \dots, d-1\}$. The local address, j, can be computed from the spatial indices $\{a_k\}$ using row-major order as:

$$j = \sum_{\kappa=0}^{D-1} a_\kappa d^{D-\kappa-1} \; . \tag{6.2}$$

Similarly, we can invert this relation to obtain the spatial indices of a node from the local address j, through the recursion:

$$a_\kappa = b_{\kappa+1} \bmod d \; , \tag{6.3}$$

$$b_\kappa = (b_{\kappa+1} - a_\kappa)/d \; , \tag{6.4}$$

with the base case:

$$a_{D-1} = j \bmod d \; , \tag{6.5}$$

$$b_{D-1} = (j - a_{D-1})/d \; . \tag{6.6}$$

In addition to a region cube, each node in the tree is also associated with a list of all of the BEM mesh elements it contains/*owns*, a list of all the collocation points it contains, a multipole (remote) expansion, and a local expansion.

Since the HFFMM algorithm requires various sets of operations to be performed over the tree, in addition to the tree structure, we must also specify how to navigate over it. Typically, we are interested in applying a particular action to every node in the tree in a specific order. There are primarily two different ways to visit the nodes in the tree; these are recursive and corecursive order. Recursive order, also known as depth-first, visits all of the children of a particular node before visiting any of its siblings. This visitation pattern can be defined using the notion of a "stack" or rather a last-in-first-out (LIFO) data structure according to algorithm 8. Corecursive order,

Algorithm 8 Apply the operation $\mathcal{A}$ to every node in $\mathcal{N}_{[0]}$ (co)recursively.

```
Input: Root node $\mathcal{N}_{[0]}$, operator $\mathcal{A}$ and empty stack (queue) $S$.
1: $S$.push$\left(\mathcal{N}_{[0]}\right)$                ▷ Push root node onto stack(queue).
2: while $||S|| \neq 0$. do
3:     $\mathcal{N}_{[i]}$ = $S$.pop()    ▷ Remove reference off of top (front) of the stack (queue).
4:     $\mathcal{N}_{[i]} \to \mathcal{A}(\mathcal{N}_{[i]})$                ▷ Apply action $\mathcal{A}$.
5:     if $\mathcal{N}_{[i]}$ has children then
6:         for $j = 0, \ldots, d^D - 1$ do
7:             $S$.push$\left(\mathcal{N}_{[i]j}\right)$   ▷ Add $j$-th child to top (back) of the stack (queue) $S$.
8:         end for
9:     end if
10: end while
Output: Operation $\mathcal{A}$ applied to every node in the tree: $\mathcal{N}_{[i]} \to \mathcal{A}(\mathcal{N}_{[i]})$.
```

also known as breadth-first, visits all of the siblings of a particular node before visiting any of its children, and can be defined using a "queue" or first-in-first-out (FIFO) structure according to algorithm 8 (reading within the parenthesis). It is also sometimes useful to use reverse-(co)recursive visitation, in which case, every node in the tree is visited in depth(breadth)-first way but from the direction of the leaf nodes towards the root node. This can be done in a manner similar to that given in algorithm 8 but with some additional memory usage for temporary references.

As might be expected, the assembly of the hybrid region tree is somewhat more complicated than the standard recursive octree construction in algorithm 7. This is not only because of the differing number of subdivisions, d, but also because the subdivision of any particular node can depend on the state of its neighbors. However, unlike the FMM, there are only two categories of nodes; those which are nearby-neighbors and those which are well-separated. Nearby-neighbors are determined by the zero-mask size, z, and can directly influence each other by directly participating in the multipole-to-local transformation using FFTM. On the other hand, the interaction between well-separated nodes must be mediated through their parents and has to be passed along by the local-to-local transformation through at least one tree level. Rule 6.1 determines whether any two nodes are considered nearby-neighbors.

Rule 6.1 *Assume a zero-mask size, z, and the existence of a node, $\mathcal{N}$, at tree level m. The cube, $\mathcal{C}$, associated with $\mathcal{N}$ has a side length, $\ell_m = \ell / d^m$, and is centered at the point $\mathbf{p}$. Any node $\mathcal{N}'$ belonging to tree level, m, whose cube center, $\mathbf{q}$, satisfies any one of following D inequalities, for $\kappa = 0 \ldots, D-1$:*

$$(\mathbf{p} - \mathbf{q}) \cdot \hat{\mathbf{e}}_\kappa \leq z\ell_m \, , \tag{6.7}$$

is considered a nearby-neighbor of node $\mathcal{N}$. All nodes on the tree level, m, which are not a nearby-neighbor of $\mathcal{N}$ are considered to be well-separated.

Figure 6-1 exhibits the nearby-neighbor and well-separated categories with respect to a particular node for several values z.

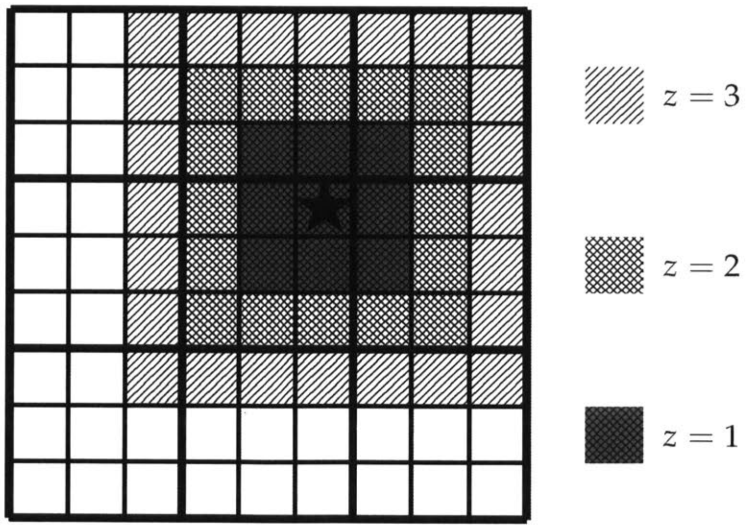


Figure 6-1: Example of the node categories for a two-dimensional tree with two levels ($d = 3$). The node under consideration is marked with a star. Nodes considered to be a member of its set of nearby-neighbors are shaded as indicated for various values of z.

With the nearby-neighbor condition defined, we can move forward with the description of the tree assembly process. To avoid the complexity of three-dimensions, we will illustrate the construction with a very simple two dimensional example. In this example, a square two-dimensional region contains two well-separated electrodes, as shown in figure 6-2. We will use a value of $d = 3$ for the number of divisions to make, limit the depth of the tree to only two levels, $n_{\text{max}} = 2$, and use a zero-mask size of $z = 1$ to define the nearby neighbors.

In order to determine how to proceed with the subdivision of the unrefined region and the subsequent refinement of its children, we will consider the bounding balls of the mesh elements in conjunction with two simple rules. The first rule determines if a particular mesh element should be associated with a certain node and is known as the *insertion condition*. We will use the notation, $B(\mathbf{p}_i, r_i)$, to denote the ball of radius, r_i, centered about the point, $\mathbf{p}_i$, and $C(\mathbf{p}_i, \ell)$ to denote a cube of side length, ℓ, centered about the point $\mathbf{p}_i$.

Rule 6.2 *Consider a member node of the region tree, $\mathcal{N}_i$, at tree level, m. $\mathcal{N}_i$ is structured about a cube, $C(\mathbf{p}_i, \ell_m)$, with $\ell_m = \ell / d^m$. To determine which mesh elements are associated with this node, we construct the spherical assignment region $B_i(\mathbf{p}_i, r_i)$, where $r_i = \eta \frac{\ell_m \sqrt{D}}{2}$, and η is a user defined parameter. The mesh element, u_j, will contribute to the multipole expansion of $\mathcal{N}_i$ if it satisfies the following two conditions:*

$$\mathbf{q}_j \in C(\mathbf{p}_i, \ell_m) \tag{6.8}$$

and

$$B_j(\mathbf{q}_j, r_j) \subseteq B_i(\mathbf{p}_i, r_i) \, , \tag{6.9}$$

where $B_j(\mathbf{q}_j, r_j)$ is the minimum bounding ball enclosing u_j. Furthermore, if the node, $\mathcal{N}_i$, satisfies both 6.8 and 6.9 but has no child node, $\mathcal{N}_{[i]j}$, which also satisfies both of these conditions, then it is said that the node, $\mathcal{N}_i$, owns *the mesh element u_j.*

The second rule is known as the *subdivision condition* and determines when a particular node must be subdivided in order to adaptively refine the region. There are several choices available for the *subdivision condition* which can significantly

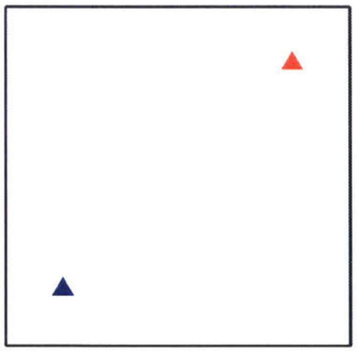


Figure 6-2: An unrefined region containing two localized electrode groupings, denoted by the blue and red triangles.

affect the manner in which the tree is constructed. However, for the time being, we will defer discussion of these other conditions and use the simplest but most aggressive condition according to rule 6.3.

Rule 6.3 *If the node, $\mathcal{N}_i$,* owns *more than one mesh element and it lies at tree level, $m < n_{\max}$, then $\mathcal{N}_i$ will be subdivided into d^D children.*

The result of applying these two rules on our initial region (figure 6-2) is shown in figure (6-3). Once this first pass of refinement has been performed, we then prepare the region for the computation of the expansion coefficients by the application of one more additional rule.

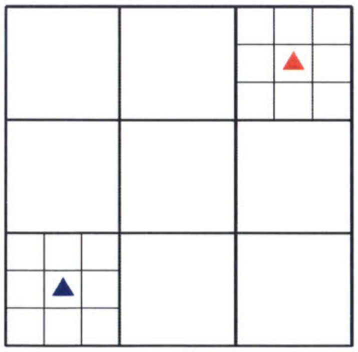


Figure 6-3: After the application of rules (6.2) and (6.3), the region where the field needs to be computed has been subdivided into a tree. At the first level of the tree, the root region is divided into nine child sub-regions. At the second level, the two sub-regions which contain the electrode groupings are further sub-divided into nine children.

Rule 6.4 *Any node, $\mathcal{N}_i$, that lies at tree level, $m < n_{\max}$, and which is also a nearby-neighbor of a node which* owns *an electrode must also be subdivided.*

Rule 6.4 is called the *adjacency-condition* because it causes nearby-neighbors of nodes, which have children containing charged sources, to be subdivided as well. This is necessary in order to apply FFTM because we need a locally uniform grid in order to perform the Fourier transform involved in the multipole-to-local coefficient conversion. Applying this rule results in the final form of the refined region tree, shown in figure (6-4).

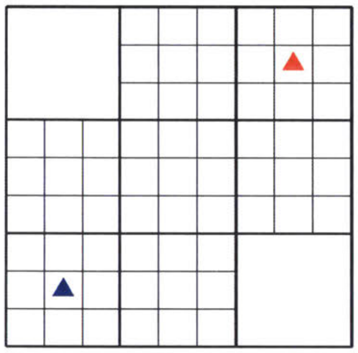


Figure 6-4: The resulting region after the application of rule (6.4).

The final step in the tree construction is the pruning of unused nodes. This step is not strictly necessary, since it is only done for computational efficiency. Unused nodes are those which were subdivided according to the adjacency condition but contain no collocation points where we would need to evaluate the field. Since no field calculation is done within these nodes during the evaluation of the BEM matrix-vector product, we can discard them and avoid calculating their local coefficient expansion. Of course, if we are interested in producing a field map over all space for use in particle tracking, this pruning step is not done.

Once the region tree has been constructed and ownership of each electrode has been delegated to the appropriate nodes, we can compute the multipole moment expansion of each node's region. The first step of this process starts with a recursive visitation of all the nodes in the region tree. During this visitation, we compute the multipole expansion of each owned mesh element about the center of the node

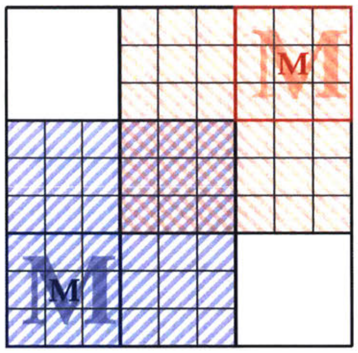


Figure 6-5: The multipole moment labels are color-coded according to the respective electrode group they originate from. The small M indicates the multipole moment of a leaf region due to its contained electrodes. The faded large M indicates the multipole moment of a parent region due to its children. Only regions that have a non-zero multipole moment are labeled. The *cohort* of each region with a colored border is shaded with the same color.

which *owns* it. It is important to note that non-leaf nodes can participate in this process and may have non-zero multipole expansions. This is due to the fact that some mesh elements may be too large to satisfy the *insertion-condition* and be distributed downward to child nodes. For details on how the individual mesh element multipole expansion is calculated, see appendix A for triangular mesh elements, and appendix B for rectangular elements and line segments. To form the multipole expansions of each node's region, we then simply sum over all the multipole expansions of its *owned* mesh elements.

The second step is to progressively gather the multipole moments of each node into the multipole expansion of its parent, using theorem 5.1. This process, starting from the leaf nodes, is performed through a reverse-order recursive visitation over the tree. The multipole-to-multipole translation action continues up the tree until the root node is reached and each region in the tree has been assigned a multipole expansion. Figure 6-5 shows this process graphically.

After all the regions have a multipole expansion, the computation of the local coefficient expansions can begin. To do this, we consider only regions who have a non-zero multipole expansion and consider the influence they have on their *cohort*. We define the nodes that belong to a *cohort* as the union of the node of interest's own children with the children of its nearby-neighbors. Graphically, the members of a cohort are indicated in figure 6-5 by the color-coded shaded regions. In order to compute the contribution that each source (non-zero multipole expansion) region has on the local coefficient expansions of the nodes in its *cohort*, we artificially set the multipole expansions of all regions which are not children of the region of interest to zero. Then, we compute the convolution of this grid of multipole expansions with the response functions using theorem 5.3 and observation 5.1. This action is represented graphically in figure 6-6. It is applied over all nodes in the tree using a corecursive visitation. In this two-dimensional example, we have chosen $d = 3$ with a zero-mask size of $z = 1$. To compute the transformation using FFTM, we copy the multipole expansions of the children of the node of interest into the appropriate position in an array. The array elements corresponding to the node of interest are marked with a dashed box in figure 6-6. This array is structured to conform to the local structure of the region tree about the node of interest. Then, the elements in the array which correspond to the spatial locations of the children of nearby-neighbors of the node of interest are artificially set to zero (these are marked with a zero). To avoid wrap-around artifacts, the array is zero padded (the shaded region) out to a size of $2d(z+1)$. This padding is slightly larger[2] than necessary but this does not significantly affect the number of arithmetic operations needed. The multipole array is then convolved with the response function array (those response functions which have been masked to zero are marked as such), resulting in an array of local coefficient expansions. The resulting local coefficients generated in the shaded region are invalid/irrelevant and are discarded. The useful local coefficients (unshaded region) are then copied out of the array and summed into the local expansions of the *cohort* nodes in the tree. This process is applied to every node which contains at least one child with a non-zero multipole expansion.

After the multipole-to-local conversion has been conducted for each level in the region tree, we can perform the downward distribution of the local coefficients of each parent to its children using theorem 5.2. This action is applied through another corecursive visitation, the final result of which is shown in figure 6-7. After

[2]The minimum allowed size is $2d(z+1) - 1$, but we use $2d(z+1)$.

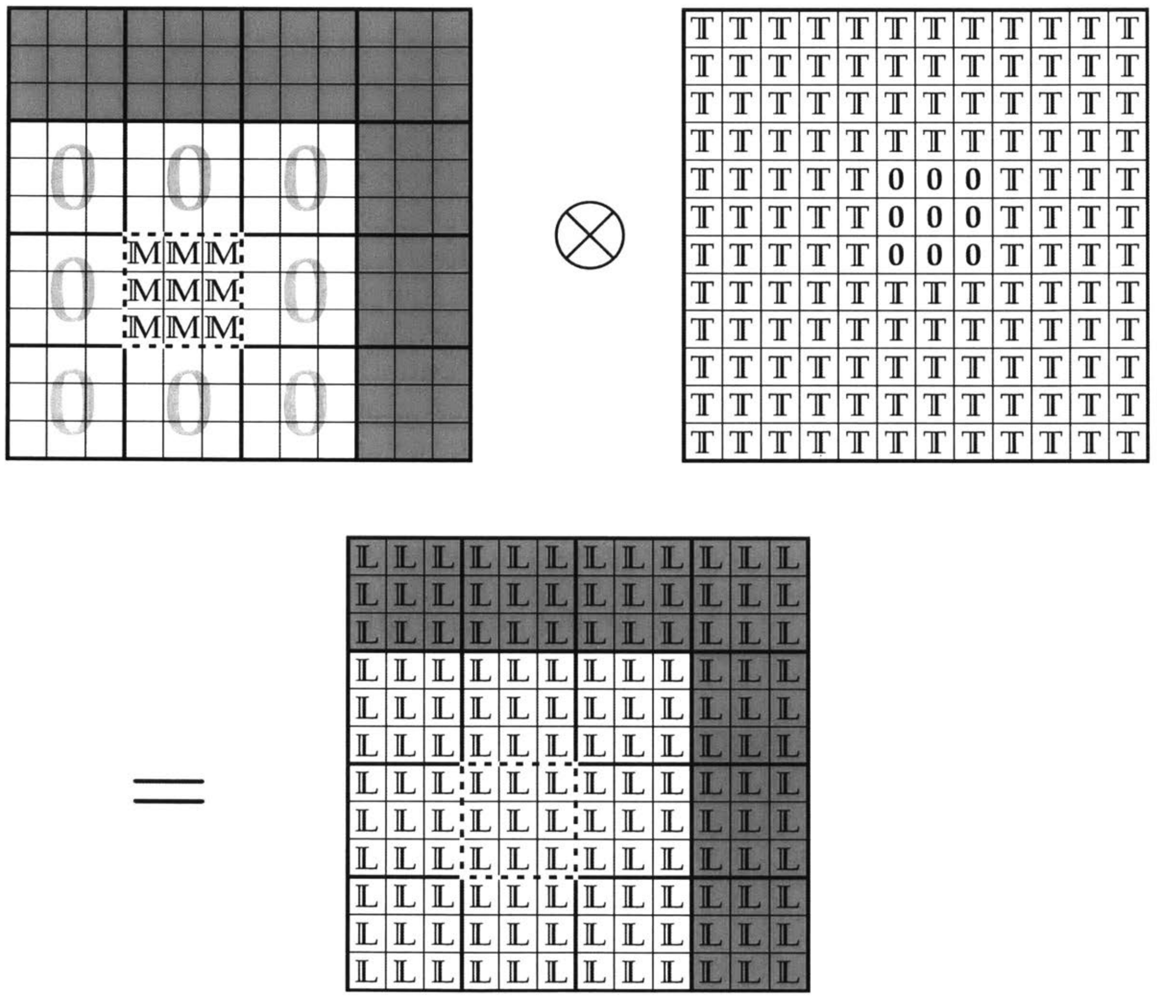


Figure 6-6: Graphical representation of the multipole-to-local transformation process as a convolution, performed using the FFTM.

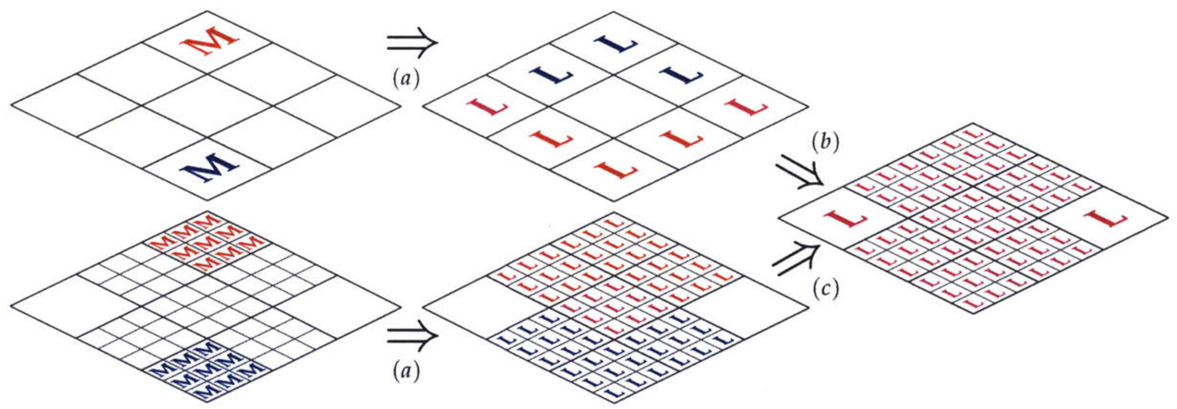


Figure 6-7: The computation of the local coefficient expansions from the multipole expansions proceeds from left to right. First the FFTM (observation 5.1) process is applied separately at each level of the tree to the nodes with non-zero multipole moments as indicated by the arrows marked with (a). Once each node's local coefficient contribution from its nearby-neighbors is computed, then theorem 5.2 is used to propagate the parent's local coefficients down to its children as indicated by the arrow marked with (b). Each parent's contributions are then summed with the existing local coefficient expansions at the finest granularity as indicated by the arrow marked with (c). The local coefficients are color-coded according to the electrode groups by which they have been influenced. Purple indicates they are influenced by both groups of electrodes.

this step all regions in the tree that were not pruned will have a local coefficient expansion. It is important to note that by construction, the mesh elements which are owned by a node or its nearby-neighbors do not contribute to its local coefficient expansion. This is because the expansions of nearby electrodes converge poorly, so we excluded them by using a zero-mask, $z \geq 1$. Therefore, to maintain accuracy when computing the field for a point contained by some region, the influence of the nearby mesh-elements must be evaluated directly.

When evaluating the field, there will be some truncation error due to the multipole expansion. The dominant term in this error tends to originate from the closest source included into the local coefficient expansion, because the truncation error depends on the ratio of the source radius to the distance from the source to evaluation location (see eq. 5.12). The larger and closer the source, the larger the error. Figure 6-8 provides a rough idea of how the truncation error varies spatially in the example region we have used. If we wish to further mitigate the truncation error,

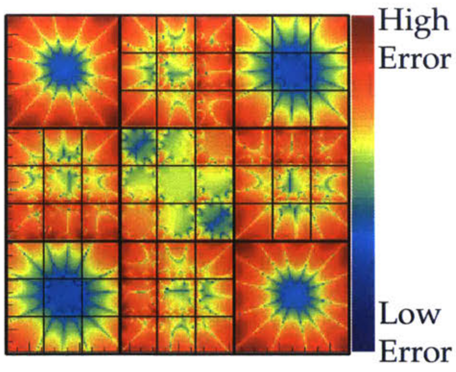


Figure 6-8: Spatial distribution of the absolute difference between the approximate field and true field (arbitrary scale), for a configuration similar to the example geometry.

we have several options. The first is to simply increase the maximum degree of the multipole expansions, p. The next is to increase the number of divisions, d, which decreases the size of the source regions in order to improve the error ratio. The last possibility is to increase the zero-mask size, z, which has the effect of increasing the distance between the source region and the target.

6.3 Scale Independent Transformation Rules

The response functions which govern the application of the multipole-to-multipole, multipole-to-local, and local-to-local coefficient transformations are quite time consuming to compute and would significantly slow down the algorithm if they were to be calculated on an as-needed basis for every transformation. Fortunately, we can perform the majority of the work necessary to compute the response functions upon initialization with little work thereafter. This is because the Laplace equation is scale independent, which implies the response functions can be factored into separate terms containing the radial and angular dependence. Moreover, since the transformations are always applied on a discrete Cartesian grid of nodes with fixed spacing, the angular terms do not change between tree levels and only need to be computed once. The radial terms, which do change for different tree levels, can be accounted for by simply rescaling the source moments and resulting target moments.

To demonstrate this, we will first consider the application of the multipole-to-multipole transformation of theorem 5.1 to the three dimensional grid of d^3 children owned by a node. Assuming that this node lies at some level in the tree where the cube side length is l, then the positions of the child nodes with respect to their parent's center can be determined according to 6.1. Therefore, their displacement in each dimension is simply:

$$\begin{aligned} \Delta x &= l\left[\left(\frac{a_0}{d}+\frac{1}{2d}\right)-\frac{1}{2}\right] \\ \Delta y &= l\left[\left(\frac{a_1}{d}+\frac{1}{2d}\right)-\frac{1}{2}\right] \\ \Delta z &= l\left[\left(\frac{a_2}{d}+\frac{1}{2d}\right)-\frac{1}{2}\right] \end{aligned} \quad .$$

The distance, $\rho(\{a_\kappa\})$, between the expansion center of the child (source) and the parent node's center (target) is given by:

$$\rho(\{a_\kappa\}) = l\sqrt{\sum_{\kappa=0}^{2}\left[\left(\frac{a_\kappa}{d}+\frac{1}{2d}\right)-\frac{1}{2}\right]^2}\,, \tag{6.10}$$

which is clearly dependent on the side length, l, and thus implicitly on the tree level. On the other hand, the angular positions (α, β) of the child node centers (with respect to the parent node's center) are completely independent of the node size, and are given by:

$$\alpha(\{a_\kappa\}) = \arctan\left(\frac{\sqrt{(\Delta x)^2+(\Delta y)^2}}{\Delta z}\right), \tag{6.11}$$

and

$$\beta(\{a_\kappa\}) = \arctan\left(\frac{\Delta y}{\Delta x}\right). \tag{6.12}$$

Now, in order to simplify the process of rescaling the response functions, we would like to rewrite them in such a way that the indices of source moments, (O_n^m), are independent of the target moment, (M_j^k), indices. To do this, we can make the index replacements $j-n \to n'$ and $k-m \to m'$ on equation 5.16, which yields[3]:

$$M_j^k = \sum_{n'=0}^{j} \sum_{m'=-n'}^{n'} \frac{O_{n'}^{m'} i^{|k|-|k-m'|-|m'|} A_{j-n'}^{k-m'} A_{n'}^{m'} \rho^{j-n'} Y_{j-n'}^{m'-k}(\alpha,\beta)}{A_j^k} ,$$

$$M_j^k = \rho^j \sum_{n'=0}^{j} \sum_{m'=-n'}^{n'} \left(\rho^{-n'} O_{n'}^{m'}\right) T_{\mathrm{M2M}}{}_{j,n'}^{k,m'}(\alpha,\beta) . \tag{6.13}$$

Factoring out the dependence on l produces:

$$M_j^k = l^j \left[\sum_{n'=0}^{j} \sum_{m'=-n'}^{n'} \left(l^{-n'} O_{n'}^{m'}\right) \widetilde{T}_{\mathrm{M2M}}{}_{j,n'}^{k,m'}(\alpha,\beta) \right] , \tag{6.14}$$

with the scale independent multipole-to-multipole response functions associated with the expansion at $\{a_\kappa\}$, given by:

$$\widetilde{T}_{\mathrm{M2M}}{}_{j,n'}^{k,m'}(\alpha,\beta) = $$
$$i^{|k|-|k-m'|-|m'|} \frac{A_{j-n'}^{k-m'} A_{n'}^{m'} Y_{j-n'}^{m'-k}(\alpha,\beta)}{A_j^k} \left(\sqrt{\sum_{\kappa=0}^{2} \left[\left(\frac{a_\kappa}{d} + \frac{1}{2d} \right) - \frac{1}{2} \right]^2} \right)^{j-n'} . \tag{6.15}$$

These scale independent functions can be pre-computed once over the full grid (over which each of the indices, a_κ, range independently from $0, 1, \ldots, d-1$) and stored for later reuse. From equation 6.14, it is clear we can use $\widetilde{T}_{\mathrm{M2M}}$ to apply the transformation at any level of the region tree, provided we properly pre-scale the source moments and post-scale the target moments by the appropriate length factors.

The local-to-local transformation can be handled in nearly exactly the same way, except the scale factors have the opposite dependence on the source and target moment indices. Factoring out the scale dependence from the response functions of

[3]Note, we may start the summation from $n' = 0$ since we are free to reorder the summation however we want.

theorem 5.2 yields:

$$L_j^k = l^{-j} \left[\sum_{n=j}^{p} \sum_{m=-n}^{n} (l^n O_n^m) \, \widetilde{T}_{\mathrm{L2L}}{}_{j,n}^{k,m}(\alpha,\beta) \right] , \tag{6.16}$$

with the scale independent response functions for $\{a_\kappa\}$ given by:

$$\widetilde{T}_{\mathrm{L2L}}{}_{j,n}^{k,m}(\alpha,\beta) = \\ i^{|m|-|m-k|-|k|} \frac{A_{n-j}^{m-k} A_j^k Y_{n-j}^{m-k}(\alpha,\beta)}{(-1)^{n+j} A_n^m} \left(\sqrt{\sum_{\kappa=0}^{2} \left[\left(\frac{a_\kappa}{d} + \frac{1}{2d} \right) - \frac{1}{2} \right]^2} \right)^{n-j} . \tag{6.17}$$

Similarly, we may do the same for the multipole-to-local response coefficients of theorem 5.3. Of the three rules, the reuse of the multipole-to-local response functions between tree levels is by far the most critical for good performance. This is because multipole-to-local response functions must be computed over a much larger grid, which has dimensions of $8d^3$. Furthermore, they must be Fourier transformed (an expensive operation) over an even larger grid of dimension $[2d(z+1)]^3$, before they can be used in the convolution operation of observation 5.1. An additional problem is posed by the memory requirements of the multipole-to-local response functions which are proportional to $p^4[2d(z+1)]^3$. Without treating them in a scale independent manner, this memory requirement would be multiplied further by the number of tree levels, $n_{\max}$, which would be especially problematic if we wished to use high order expansions[4]. On account of the large memory requirements of the multipole-to-local response functions, in addition to factoring out the scale dependence, it is also helpful to factor out the normalization coefficients. Factoring out the normalization coefficients reduces the memory growth of the response functions from $\mathcal{O}(p^4)$ to $\mathcal{O}(p^2)$, which can result in significant savings. Performing both the scale and normalization factorization on the multipole-to-local response

[4]For example, for the (not unreasonable) parameters $p = 16$, $d = 8$, and $z = 1$ the memory required to store the unfactorized multipole-to-local response functions approaches 17 GB. This may present an issue for the application of this method to non-scale independent problems such as the Helmholtz equation.

functions of theorem 5.3 produces:

$$L_j^k = i^{-|k|} \imath^{-j} A_j^k \left[\sum_{n=0}^{\infty} \sum_{m=-n}^{m=n} \left((-1)^n i^{-|m|} \imath^{-(n+1)} A_n^m O_n^m \right) \widetilde{T}_{\mathrm{M2L}}{}_{j,n}^{k,m}(\alpha, \beta) \right] , \qquad (6.18)$$

where the scale independent functions are given by:

$$\widetilde{T}_{\mathrm{M2L}}{}_{j,n}^{k,m}(\alpha, \beta) = \begin{cases} 0 & \text{whenever any } |a_\kappa| \leq z, \text{ for } \kappa = 0,1,2 \\ \frac{i^{|k-m|} Y_{j+n}^{m-k}(\alpha,\beta)}{A_{j+n}^{m-k}} \left(\sqrt{\sum_{\kappa=0}^{2} \left[\left(\frac{a_\kappa}{d} + \frac{1}{2d} \right) - \frac{1}{2} \right]^2} \right)^{-(j+n+1)} & \text{otherwise.} \end{cases} \qquad (6.19)$$

These response functions are calculated over a full grid where each of the spatial indices $\{a_\kappa\}$ are allowed to span the range from $-d(z+1)$ to $d(z+1)-1$. Note that those response functions which are within the zero-mask size, z, are forced to zero to avoid the singular region.

6.4 Description Of Parameters and Options

The behavior and performance of the hybrid algorithm is governed by a set of user controlled parameters. The proper set of parameters is heavily geometry dependent. A selection which provides the user acceptable results in terms of speed and accuracy for one particular problem, may be entirely useless for a different problem. In general, to accommodate the hardware and geometry at hand, the best parameters are usually found through trial and error. However, in order to do this, the user needs to have a rough idea about the expected effect each parameter has on the algorithm's performance.

The first parameter to consider is the maximum degree of the multipole/local expansions, p. This parameter controls the number of coefficients in the computed multipole and local coefficient expansions. The storage requirements of the response functions grow in proportion to $\mathcal{O}(p^4)$, while the storage requirements for the resulting region tree which stores the local coefficient grows like $\mathcal{O}(p^2)$. The time for a single field evaluation with no nearby mesh elements also scales like $\mathcal{O}(p^2)$.

Second is the number of spatial divisions, d. This parameter controls the number of subdivisions of each node/sub-region in the tree. Occasionally, it can sometimes be helpful to use a different number of divisions for the root node. This is referred to as the number of top-level divisions, d_t. This feature is primarily used in order to make the workload more granular so that it can be evenly distributed across several computers when the algorithm is run in parallel[5].

The next most important parameter is the zero-mask size, z. In the $z = 1$ case, for a given three-dimensional node, all nodes which share with it a face, edge, or corner with its cubic region are considered neighbors, so it will have 26 neighbors. Similarly, in the $z = 2$ case, following rule 6.1, it will have 124 neighbors. This pattern continues, such that in the general D-dimensional case the number of neighbors of a region is given by $(2z + 1)^D - 1$. The size of the response functions needed for the multipole-to-local transformation grows in proportion to $(2d(z + 1))^D$. Accuracy generally increases with higher values of z, but at the cost of increased memory usage and a larger number of direct evaluations, which tend to slow down the field evaluation.

Also of importance is the maximum tree level, $n_{\max}$, which prevents any node at depth, $n_{\max}$, from being subdivided. This parameter is not strictly necessary, since node subdivision will terminate eventually once the mesh element size approaches the size of the cubic sub-regions associated with each node. However, this limitation is still useful in order to keep the memory usage of the tree from growing beyond machine constraints.

The last parameter is the insertion ratio η. This parameter determines when a mesh element is allowed to be inserted into a node, as given in rule 6.2. It is generally necessary to use $\eta > 1$ in order to prevent mesh-elements from being trapped at high levels in the tree. When $\eta \leq 1$, trapping can happen even for small (relative to node size) mesh-elements if they are located too close to the boundary

[5]Discussion of the parallel implementation will be deferred until the next chapter.

between two nodes. This occurs because even if only a small portion of the mesh element crosses the boundary, it cannot be delegated to a child node and it must remain *owned* by the parent. This sort of trapping can be detrimental to the run time of the program because it causes more direct calls during field evaluation. The use of larger values of η to relax the insertion condition can reduce the number of direct calls needed at the cost of a slight reduction in accuracy. The default value of $\eta = 4/3$ has been determined to strike a satisfactory balance between accuracy and speed for most geometries encountered in practice.

6.5 Estimating the Complexity/Scaling of the Hybrid Variant

Developing an estimate of the computational work involved in the hybrid algorithm is somewhat complicated by the algorithm's dependence on the problem geometry during the tree construction. Since it is a combination of the geometry of the BEM mesh and the insertion and subdivision conditions which together govern the tree construction, the operation count cannot be independent of the mesh element distribution. However, a geometry independent worst-case estimate of the work required in the evaluation of a matrix-vector product can still be useful, but will require some assumptions/modifications to the actual algorithm. The first assumption is that the mesh elements are small enough that they can be treated as points when determining the insertion condition. This has the same effect as setting the parameter $\eta = \infty$, so that the finite size of the mesh elements has no influence on the tree construction. The second assumption is that the mesh element distribution is uniform enough that adaptive subdivision is unnecessary. Thirdly, we also assume that the spatial subdivision can be made fine enough that on average, the number of mesh elements in a leaf node is approximately a constant, s, independent of N. Lastly, for the sake of simplicity, we will assume that the number of top level divisions, d_t, is also d. With these four assumptions, we can then proceed to estimate the number of arithmetic operations involved in the matrix-vector product.

Since we assumed that at the lowest tree level the number of elements in a node is on average s, we can estimate the number of nodes at the lowest tree level as roughly N/s. Furthermore, the reduction factor in the number of nodes at each level above the lowest is d^3 in three-dimensions, so we can expect the number of tree levels to be roughly:

$$n_{\mathrm{max}} = \frac{1}{3} \log_d(N/s) \, . \tag{6.20}$$

Now, we can consider the work involved in each step of the algorithm[6]. The first task is to compute the multipole expansions of each individual mesh element (denoted P2M). Since there are roughly p^2 coefficients in each expansion, we need to perform roughly Np^2 integrations followed by Np^2 addition operations to compute and sum these coefficients into the multipole expansions of the leaf nodes. Therefore, the number of operations for this task is proportional to:

$$K_{\mathrm{P2M}} = 2Np^2 \, . \tag{6.21}$$

Next, we need to gather the multipole expansion of each node into their parent's expansion (denoted M2M). This requires roughly p^4 multiplications and additions for each expansion that must be translated. The multipole-to-multipole operation must be applied to each node with a non-zero multipole expansion at every level in the tree. While the number of nodes at the lowest level of the tree is N/s, the number of nodes at each ascending level of the tree is reduced by a factor of d^3 as the root node is approached. Summing the amount of work needed for the multipole-to-multipole gathering operation over every tree level implies a cost proportional to:

$$K_{\mathrm{M2M}} = p^4 \left(\frac{N}{s}\right) \sum_{k=0}^{n_{\mathrm{max}}} \left(\frac{1}{d^3}\right)^k \leq p^4 \left(\frac{N}{s}\right) \left(\frac{d^3}{d^3-1}\right) \, . \tag{6.22}$$

[6]We ignore the cost of constructing the region tree, which only needs to done once at initialization and not during every matrix-vector product evaluation.

The same line of reasoning and limit also applies to the local-to-local conversion, implying $K_{\text{L2L}} \simeq K_{\text{M2M}}$. Additionally, under the assumption that $d \geq 2$ always, then $\left(\frac{d^3}{d^3-1}\right) \leq 8/7$, and the upper limit for the operation count of the multipole-to-multipole and local-to-local operations (up to a proportionality constant) is given by:

$$K_{\text{M2M}} \leq \frac{8Np^4}{7s} \,. \tag{6.23}$$

Now turning our attention to the cost of the multipole-to-local conversion step (denoted K_{M2L}), we note that for a given division number, d, and zero-mask size, z, the length of the data array for each of the p^4 convolutions we must perform is:

$$\lambda = 2d(z+1) \tag{6.24}$$

in each dimension. Therefore, the number of multiplication/addition operations required to execute the three dimensional Fourier transform via the FFTM for a single node cohort is roughly:

$$p^4 \lambda^3 \log_2 \lambda^3 \,. \tag{6.25}$$

The $\log_2$ is due to the assumption that we are using a radix-2 FFT to perform the Fourier transform, but other radices or mixed-radix algorithms with slightly different costs are possible. Since this operation only needs to be applied to the top $(n_{\text{max}} - 1)$ levels of the tree, the number of nodes is dominated by the next-to-lowest level, which has roughly $N/d^3 s$ nodes. Therefore, the total number of arithmetic operations involved is roughly proportional to:

$$K_{\text{M2L}} = p^4 \lambda^3 \log_2 \lambda^3 \left(\frac{N}{d^3 s}\right) \left[\sum_{k=0}^{n_{\text{max}}-1} \left(\frac{1}{d^3}\right)^k\right] \tag{6.26}$$

$$K_{\text{M2L}} \leq 3p^4 \lambda^3 \log_2 \lambda \left(\frac{N}{d^3 s}\right) \left(\frac{d^3}{d^3-1}\right) \tag{6.27}$$

$$K_{\text{M2L}} \leq \frac{24Np^4\lambda^3 \log_2 \lambda}{7d^3 s} \,. \tag{6.28}$$

Finally, we need to consider the evaluation of the field at the collocation points due to the local coefficient expansions and from near-field pairwise interactions. The cost of evaluating the field due to local coefficients (denoted L2P) at the N collocation points is roughly:

$$\mathcal{K}_{\mathrm{L2P}} = Np^2 \,. \tag{6.29}$$

The near-field cost is proportional to the number of direct pairwise interactions between mesh elements contained by neighboring nodes (denoted $\mathcal{K}_{\mathrm{P2P}}$). For this purpose, we only need to consider the approximately N/s nodes at the lowest level of the tree, since under our assumptions, all the mesh elements are *owned* at the lowest level. Each of these nodes contains approximately s mesh elements and has $[(2z+1)^3 - 1]$ nearby-neighbors also with $\sim s$ mesh elements. Therefore, the total number of pairwise interactions that need to be computed is approximately:

$$\mathcal{K}_{\mathrm{P2P}} = Ns^2(2z+1)^3 \,. \tag{6.30}$$

The sum all of these contributions (weighted by their constants of proportionality[7]):

$$\begin{aligned}\mathcal{K}_{\mathrm{total}} = \alpha_{\mathrm{P2M}}\mathcal{K}_{\mathrm{P2M}} + \alpha_{\mathrm{M2M}}\mathcal{K}_{\mathrm{M2M}} + \alpha_{\mathrm{M2L}}\mathcal{K}_{\mathrm{M2L}} \\ + \alpha_{\mathrm{L2L}}\mathcal{K}_{\mathrm{L2L}} + \alpha_{\mathrm{L2P}}\mathcal{K}_{\mathrm{L2P}} + \alpha_{\mathrm{P2P}}\mathcal{K}_{\mathrm{P2P}} \,,\end{aligned} \tag{6.31}$$

produces a worst-case total arithmetic cost of a matrix-vector product using the HFFMM. While we have made no statement about the constants of proportionality, we expect them to be primarily governed by the hardware and software implementation of the algorithm, and are therefore roughly independent of the number of degrees of freedom, N, and the other user selected parameters.

In light of these work estimates, we can make the following several remarks under the assumption that the proportionality constants are all roughly equal. The first is that in order to roughly balance the workload between each portion of the far-field calculation, a reasonable choice for the number of mesh elements in the

[7] These constants cannot be determined a priori, since they depend on the exact implementation of each calculation.

average leaf node should be around $s = p^2$. Using this choice for s, the work estimate of equation 6.31 reduces to:

$$K_{\text{total}} = \frac{Np^2}{7}\left(37 + 192(z+1)^3 \log_2 \lambda + 7p^2(2z+1)^3\right) . \tag{6.32}$$

From this we can see that like the canonical FMM, the cost of this algorithm scales like $\mathcal{O}(N)$, although the exact prefactor depends heavily on the choice of the parameters p, d and z. Lastly, we remark that we can consider the canonical FMM and FFTM as limiting cases of the hybrid method. In the case where we fix $d = 2$ and $z = 1$, while placing no limits on the number of tree levels (n_{max}) we produce the canonical FMM[8]. Whereas, if we fix $n_{\text{max}} = 1$ but allow d and z to vary freely, the hybrid algorithm becomes the FFTM algorithm.

[8]This is true with the minor caveat that the multipole-to-local calculation is calculated via the convolution theorem in the HFFMM rather than in direct fashion as in the FMM. However, for a division number of $z = 2$ the difference in arithmetic cost between these two methods is negligible.

Chapter 7

Computer Implementation and Performance of the HFFMM Algorithm

7.1 Introduction

The HFFM method has been designed to serve as an optional library included in the KATRIN field solving package `KEMField`. Its purpose is to accelerate the field calculation for complex three-dimensional field problems for both the solution of Laplace BEM problems and fast particle tracking. The `KEMField` main library was written primarily by T.J. Corona [53] in the C++ programming language. It uses object oriented design and template meta-programming techniques so as to allow it to be flexibly extended to cover a wide variety of problems. The fast multipole library is an extension to `KEMField` and also follows these design principles in order to allow maximum flexibility. In addition, through the use of MPI and OpenCL it has been extended to run on both single machines and parallel computing clusters with or without acceleration from graphics processing units (GPUs).

The organization of the fast multipole library mirrors the rest of `KEMField` and is therefore divided into several sub-libraries. These sub-libraries are organized by task. At the base of these sub-libraries is the `core` sub-library, which is responsible for all of the templated and abstract classes associated with operations on tree nodes which do not require a specific form for the tree. The `math` sub-library serves to collect all of the complex (but independent) mathematical operations that are needed (e.g. the rotation of spherical tensors, etc.). The `tree` sub-library organizes the concrete operations which may be applied to tree nodes without explicit reference to the equation being solved (e.g. node subdivision or neighbor finding). The `kernel` sub-library is responsible for all classes which compute spherical harmonic expansions of the equation kernel[1]. The `electrostatics` sub-library is concerned with the specifics of applying the HFFMM to the calculation of electrostatic fields, whereas the `interface` and `utility` sub-libraries localize the interface of the fast multipole library with the rest of `KEMField` and provide user configurable XML bindings respectively.

7.2 Problem Input

The input for a Laplace BEM problem is generated from the KATRIN geometry package KGeoBag. KGeoBag provides both a C++ and a XML interface which allows the user to specify the system geometry, boundary conditions, and parameters controlling the mesh production. Meshing is carried out deterministically depending on the properties of the parent surface, the discretization size scale and a 'power' term which governs how the discretization scale should vary depending on the distance from an edge or corner.

Since the discretization and boundary conditions are constructed and handled externally in KGeoBag, from the perspective of the HFFM method the problem input is simply a collection of independent mesh elements in a global coordinate system. In fact any meshing library may be used as long as it can produce the data (shape and boundary conditions) needed to fill a container of `KEMField` basis functions.

[1]Only three dimensional Laplace is currently implemented here, but it could be extended to cover the Helmholtz equation and other elliptic PDEs.

Since the details of the mesh elements may vary depending on the problem at hand or the library used to create them, and we want to allow for the possibility of extending the HFFMM machinery to other BIE problems, a preprocessing step is necessary. This preprocessing step is needed in order to decouple the tree construction from the geometric details of the mesh, while still allowing for efficient adaptive subdivision of the region of interest. Efficient subdivision demands that the tree construction step have information about the size and placement of the mesh elements. However, it does not depend critically on their shape. Therefore, we can obtain an effective shape-agnostic tree construction algorithm by replacing each of the mesh elements (triangles, rectangles, line segments, etc) with its minimum bounding ball. Calculating the minimum bounding sphere for the current set of available geometric elements in `KEMField` (line segment, rectangle, triangle) is not difficult. However, with a modicum of additional work, a more general algorithm (see [89]) capable of handling any mesh element representable by a convex hull or point cloud can be used. All that the general bounding ball algorithm requires is that the mesh element be able to produce the point cloud of their extrema (i.e. the three vertices of a triangle) which is a trivial proviso for most shapes. An example of this preprocessing step applied to the KATRIN detector region mesh of [53] is shown in figure 7-1.

7.3 Tree Structure

The skeleton on which the HFFM method is built upon is the region tree data structure. This structure is composed of a collection of linked nodes which are designed to encapsulate the minimum amount of information needed in order to navigate over the tree. However, they must also be flexible enough to be able to provide access to the data structures needed to solve an arbitrary BIE. In order to simultaneously accomplish these two goals, the node class `KFMNode` inherits from the class `KFMObjectCollection` which is based around the template class `GenScatterHierarchy` using `KFMObjectHolder`. The details of the template class `GenScatterHierarchy`

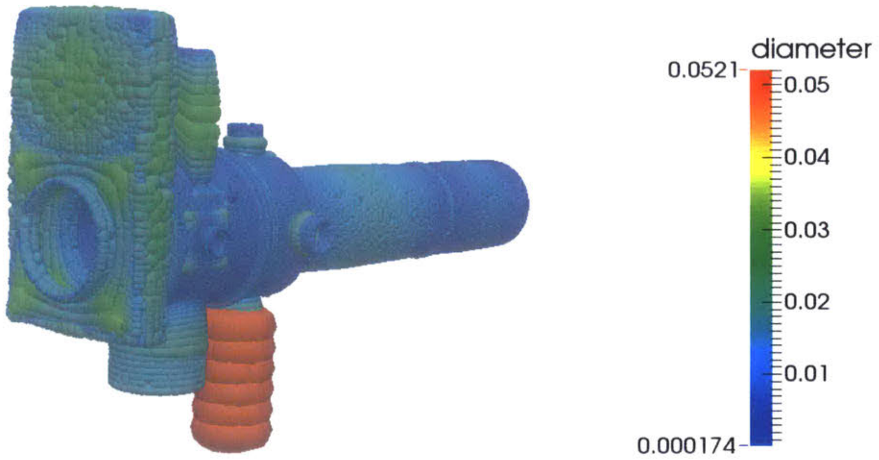


Figure 7-1: Depiction of the size and placement of the bounding balls generated from applying the pre-processing step to the KATRIN detector region mesh.

and its usage can be found in Alexandrescu's classic book [7]. The basic definition of `KFMObjectHolder` and `KFMObjectCollection` can be found in listings 7.1 and 7.2 respectively. For the sake of brevity, throughout this chapter the majority of the implementation details of individual classes will be omitted, leaving only the class structure exposed in the code listings.

```
template<typename T>
class KFMObjectHolder
{
    public:
        KFMObjectHolder():fObject(NULL){;};
        virtual ~KFMObjectHolder(){delete fObject;};
        T* fObject;
};
```

Listing 7.1: KFMObjectHolder.hh

```
template< typename TypeList >
class KFMObjectCollection: public KGenScatterHierarchy<TypeList, KFMObjectHolder >
{
    public:
        KFMObjectCollection(){};
        virtual ~KFMObjectCollection(){};
};
```

Listing 7.2: KFMObjectCollection.hh

To instantiate a concrete node class for a particular BIE, the template class KFMNode of listing 7.3 accepts a typelist specifying the object types which will be constructed and referenced by each node.

```
template<typename ObjectTypeList>
class KFMNode: public KFMObjectCollection<ObjectTypeList>
```

Listing 7.3: KFMNode.hh

For example, the typelist of classes which are needed to construct the node type that is used for solving electrostatic problems in three dimensions with the collocation method is given in listing 7.4.

```
typedef KTYPELIST_9(KFMCubicSpaceTreeProperties<3>,
                    KFMElectrostaticElementContainerBase<3,1>,
                    KFMIdentitySet,
                    KFMIdentitySetList,
                    KFMCollocationPointIdentitySet,
                    KFMCube<3>,
                    KFMNodeFlags<2>,
                    KFMElectrostaticMultipoleSet,
                    KFMElectrostaticLocalCoefficientSet) KFMElectrostaticNodeObjects;

typedef  KFMNode< KFMElectrostaticNodeObjects > KFMElectrostaticNode;
```

Listing 7.4: KFMElectrostaticNode.hh

In this way, a node can be built out of several disparate classes with minimum additional work. Those operators which disregard the node contents altogether (particularly navigation operations) can avoid dependency on every possible concrete node type by accepting a typelist or node type as a template parameter. Data reuse between different nodes is also possible with this object collection scheme, so we can avoid keeping multiple copies of the same object in memory.

It is good practice to ensure that the operations that are to be applied to each node have access to all the information specific to the action at hand, but no more than necessary. This is done so as to avoid introducing spurious dependencies between the various classes and libraries through the typelist present in the node template mechanism. This is made possible through the use of the template class KFMObjectRetriever given in listing 7.5. This template class accepts both the full typelist and a specific data type to be retrieved. The object of interest can then be extracted from the data collection of a node through the use of a static cast which introduces no runtime overhead. This adds a layer of indirection needed to avoid dependencies between the end user (a specific operation) and the full typelist containing all the data types in use with a particular node type. It also ensures that a certain level of encapsulation is maintained. This retrieval mechanism allows all the various tree operations access to the data they need without them being aware of the concrete node type and its full typelist. It also enables each node type to be extended at any time without necessitating the modification of any of the other the operations which act upon it.

```
template< typename ObjectTypeList, typename ObjectType>
class KFMObjectRetriever
{
    public:
        KFMObjectRetriever(){};
        virtual ~KFMObjectRetriever(){};

        static ObjectType* GetNodeObject(KFMNode<ObjectTypeList>* node)
        {
            return static_cast< KFMObjectHolder<ObjectType>* >(node)->fObject;
        }

        static void SetNodeObject(ObjectType* obj_ptr, KFMNode<ObjectTypeList>* node)
        {
            static_cast< KFMObjectHolder<ObjectType>* >(node)->fObject = obj_ptr;
        }
};
```

Listing 7.5: KFMObjectRetriever.hh

7.4 The Actor System

The actors are responsible for applying particular operations to the data contained by the region tree. Since we want to avoid any dependencies between the tree/node structure and the problem data encapsulated by the nodes, the node actors must inherit from an abstract template class which accepts an arbitrary node type as

```
template<typename NodeType >
class KFMNodeActor
{
    public:
        KFMNodeActor(){};
        virtual ~KFMNodeActor(){};
        virtual void ApplyAction( NodeType* node) = 0;
};
```

Listing 7.6: KFMNodeActor.hh

in listing 7.6. Typically, we are interested in applying a set of operations over the entire tree or some subset of nodes which can only be located by visiting the entire tree. This can be done in two main ways: either by visiting each node in a depth-first (recursive) fashion, or in a breadth-first (corecursive) manner, according to algorithm 8. Since this visitation procedure is so common, it is useful to separate out the tree navigation process into two separate actors, `KFMRecursiveActor` and `KFMCorecursiveActor`. These two classes can then be used to traverse the tree and apply any arbitrary operation to the nodes of interest. They also accept options to perform the visitation in reverse order. In addition to these two tree traversal actors, there are two other classes of general purpose actors. These are `KFMCompoundActor`, which can be used to apply a series of operations collectively to each node all at once, and `KFMConditionalActor`, which decides whether to apply a certain action to a particular node depending on some user provided conditional statement.

7.5 Data Structures

It is worthwhile to briefly review the exact contents of the node data structures pertaining to the Laplace BEM problem. These are given in the typelist of listing 7.4. Several of these objects are common to any fast multipole method in three dimensions, regardless of the BIE that is being solved. These are:

1. `KFMCubicSpaceTreeProperties<3>`: This class contains the data needed to construct a d^3-tree. In particular, it manages the number of divisions at the top, d_t, and subsequent tree levels, d, the zero-mask size z, the maximum tree level $n_{\max}$, and the total number of nodes. These quantities must be accessible

through any node in the tree in order for visiting actors to obtain the information necessary to complete their operations. It is templated on the dimension of the problem.

2. `KFMIdentitySet`: This class contains a list of all the mesh elements which are owned by its respective node. This list can then be used to retrieve the relevant mesh elements from their container.

3. `KFMIdentitySetList`: This class is a list of pointers, which reference all objects of type `KFMIdentitySet` that belong to a node's neighbors. Access to the complete list of nearby mesh elements is required in order to perform the local field evaluation. Keeping pointers to neighboring lists is not strictly necessary, but helps to avoid time consuming nearby-neighbor searches over the tree.

4. `KFMCollocationPointIdentitySet`: This class is a list of all the collocation points contained in a node's region of responsibility. Keeping track of the collocation points as separate entities is necessary because the mesh elements are sorted according to the center of their minimum bounding ball, which may not necessarily coincide with the mesh element centroid (the collocation point).

5. `KFMCube<3>`: This class stores the center and side length describing the region cube associated with its node. It is templated on the dimension of the problem.

6. `KFMNodeFlags<2>`: In order to keep track of which nodes need certain operations applied to them, it is useful to provide two boolean flags indicating whether a node contains sources (mesh elements) or targets (collocation points). This class is templated on the number of boolean flags desired for the problem at hand. For electrostatics there are only two flags.

The data structures which are specific to solving electrostatics problems are:

1. `KFMElectrostaticElementContainerBase<3,1>`,

2. `KFMElectrostaticMultipoleSet`,

3. KFMElectrostaticLocalCoefficientSet.

The element container class is templated on the physical dimension of the problem and on the number of degrees of freedom of the basis functions used on the mesh elements. It is provided as a means to retrieve individual mesh elements for various operations. The multipole and local coefficients sets are somewhat self-explanatory and serve to simply store the expansion coefficients pertaining to the spatial region associated with each node.

7.6 Generation of Expansion Coefficients

The generation of the multipole and local coefficient expansion coefficients proceeds according to the rules laid out in the previous chapter. The first step is to visit every node in the tree with the initializer class KFMScalarMomentInitializer which allocates space and zeros out the memory for each expansion. Then, the tree is traversed by the moment distributor, KFMElementScalarMomentDistributor, which collects information about all of the mesh elements and their multipole expansion centers. This data is passed to the KFMElementMomentBatchCalculator which carries out the calculations necessary to obtain the multipole expansions for each mesh element and returns the results to the distributor. The moment distributor then sums up all of the relevant moments into each node's multipole expansion. Following this process, the multipole moments of each region are visited by the KFMScalarMomentRemoteToRemoteConverter actor, which executes the multipole-to-multipole transformation necessary to complete the upward pass. The conversion of the multipole moment expansions into local coefficient expansions is then carried out by the KFMScalarMomentRemoteToLocalConverter. This actor must take care to collect the multipole moment expansions in a way that respects their relative spatial positions and then properly sum the local coefficient contributions into the neighboring nodes. Finally, the downward pass is handled by the KFMScalarMomentLocalToLocalConverter, which adds the contributions from far-field regions into each child node's local coefficient expansion by way of their parent. Each of the aforementioned classes are virtual and can be overloaded with any number of additional techniques to perform the same operations. This allows the end user or developer to adapt the algorithm to exploit any parallelism that might be present in the computer architecture they have available.

7.7 Sparse Matrix Generation and Handling

The generation of the near-field matrix elements is also carried out by a series of actors traversing the region tree. Before this is done, it is assumed that all of the mesh elements have been sorted into nodes and the classes `KFMIdentitySet`, `KFMIdentitySetList` and `KFMCollocationPointIdentitySet` have been initialized and filled. It is important to note that while the number of elements in the sparse near-field matrix is proportional to $\mathcal{O}(N)$, this can still result in a large memory requirement depending on the parameters chosen by the user. On occasion, this memory requirement can exceed the amount of RAM on the machine in use, requiring out-of-core memory access. Figure 7-2 shows an example of a typical sparse matrix generated by the near-field terms in a three-dimensional Laplace BEM problem. When the memory usage is large, it is necessary to store the matrix elements in an efficient way so that they can be easily read back from out-of-core memory on disk storage. To accomplish this, we construct the near-field matrix in two stages. The first stage traverses the region tree and determines the structure of the sparse matrix, while the second stage performs the actual calculation of each matrix element and streams it to disk. The sparse matrix format we use to store the near-field information in is referred to as the block compressed row storage (BCRS), or dense block format [22]. This sparse matrix format is particularly helpful for two reasons. The first is that the row and column indices are readily available, being the indices stored in the `KFMCollocationPointIdentitySet` and the `KFMIdentitySetList` respectively. Secondly, the memory access pattern of the BCRS format allows for rapid evaluation of the sparse matrix-vector product because it avoids indirect addressing as much as possible. Unlike other sparse matrix formats such as coordinate list (COO)

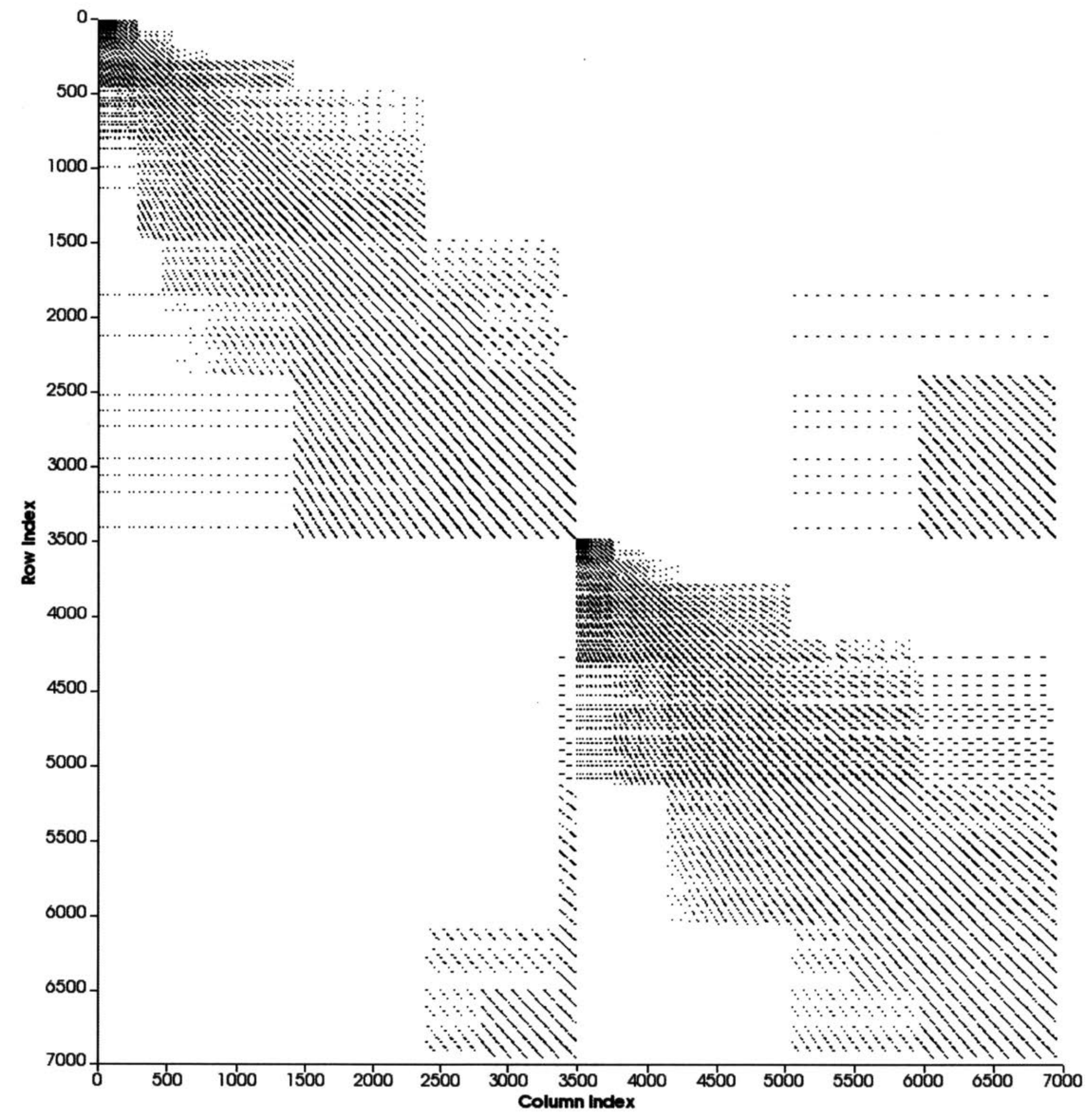


Figure 7-2: Example of the near-field sparse matrix produced by a sphere geometry discretized into approximately 7000 elements with $d = 2$ and $z = 1$. The black regions denote non-zero matrix elements whereas white space denotes zeros.

or dictionary of keys (DOK), each of the individual block is relatively compact and only needs access to a limited number of vector elements rather than needing random access to the entire vector under multiplication [22]. An example of the BCRS compression format is shown in figure 7-3.

The matrix structure determination stage is carried out by the structure generator, `KFMDenseBlockSparseMatrixStructureGenerator`. This class visits each node and examines the lists of mesh elements contained by the nearby-neighbors (which

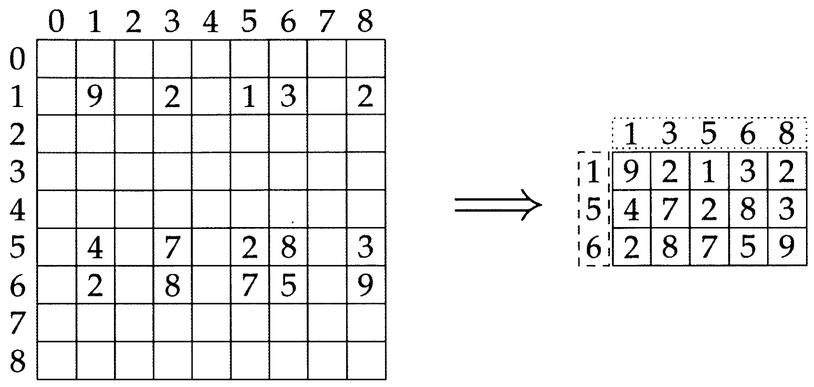


Figure 7-3: An example of the block compressed row storage (BCRS) format applied to a matrix with sparse entries. The row and column indices run along the left and upper edges respectively. Only the non-zero matrix elements along with their row/column coordinates (indicated by the dashed and dotted boxes respectively) are stored.

serve as the column indices of the non-zero matrix entries) and the list of collocation points (which serve as the row indices). From these sets of indices, we can then compute the block size of the matrix elements representing all of the near-field interactions of a particular node. When the matrix elements have to be stored out-of-core, efficient read back requires that the data must be buffered in a certain chunk size. The structure generator determines into which chunk each series of matrix element blocks and their index collections should be grouped. This separation into chunks allows us to approximately fill the buffer size without exceeding it during I/O operations. The information describing which chunk each block and its indices belongs to, as well as the memory offset of each matrix block and indices is written to a structure file. The sparse matrix structure file is generated first so that the total size of the matrix is known, which allows the user to intervene if the parameters that were chosen result in excessive memory usage. This structure file is then used by the `KFMDenseBlockSparseMatrixGenerator` to calculate each matrix element needed while streaming this data continuously to disk.

In order to apply the near-field matrix to a vector, the process is carried out in reverse. The matrix elements are streamed from disk by reading out buffered chunks into memory. Then, the structure file is used to reconstruct the matrix by extracting the matrix elements and indices of each block from the buffered memory chunks. The reconstructed matrix block is then applied to the vector. This is repeated until each block has been multiplied against the vector. Since the

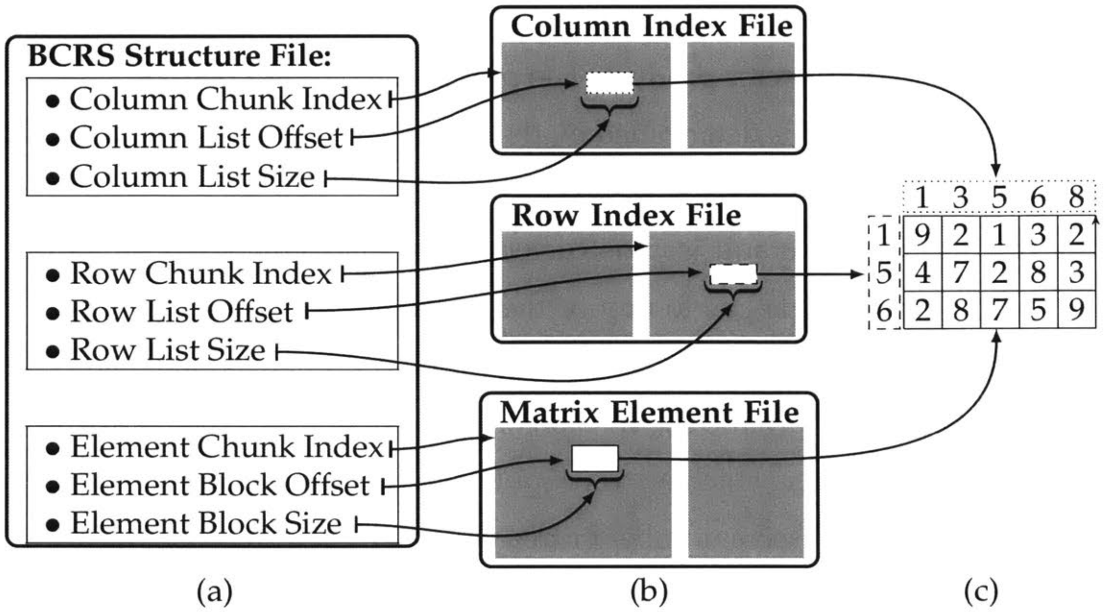


Figure 7-4: The relationship between the files representing the BCRS sparse matrix form. In (a) the structure file stores information describing the shape and location of each block's data. In (b) the matrix element, row, and column indices are stored in their respective files in a compressed format. In (c) a matrix block is reconstructed from decompressed data and ready to be used in the application of the matrix-vector product.

generation of the near-field sparse matrix files can be time consuming, it is helpful to allow the reuse of these files when encountering the same geometry and tree parameters. For this reason, the matrix, index, and structure files are stored with a naming convention which involves the MD5 hash of the relevant tree parameters (d_t, d, z, $n_{\max}$, η) and the meshed geometry in order to facilitate fast look up and matching for reuse.

7.8 Tree Balancing

One important aspect regarding the practical implementation of the HFFMM algorithm, in conjunction with a Krylov subspace solver which needs further discussion, is the need to balance the computational work required by various parts of the matrix-vector product calculation. Specifically, solution progress is most efficiently made by ensuring that the amount of arithmetic work is evenly divided between the calculation of the near-field (sparse matrix) and far-field (HFFMM) portions of the complete matrix-vector product. The proportion of time spent on these two parts of the computation is determined by the structure of the tree subdividing the region of interest.

The region tree by default is constructed according to the rules 6.1 through 6.4. However, minor changes to any of these rules can substantially affect the resultant tree structure. For instance, the subdivision condition as stated according to rule 6.3 causes a very prolific subdivision of region of interest. Overly aggressive subdivision leads to a tree which is overpopulated in comparison to the number of mesh elements. An excessive number of tree nodes will in turn cause the far-field portion of the matrix-vector calculation to dominate the computation time.

In order to modify the construction of the tree in a way which leads to a more equitable distribution of computation, a different subdivision strategy needs to be used. This subdivision strategy is referred to as a *balanced* strategy because it is intended to evenly divide work between the near and far-field terms. The default subdivision strategy of rule 6.3 is termed the *aggressive* strategy. In order to construct a tree which is properly balanced, the decision to subdivide a node is made based on the amount of computation time that would be required to evaluate the field at the collocation points it contains in two simplistic situations. The first situation is that the node remains unsubdivided, requiring a number of near-field evaluations which are roughly equivalent to the evaluation of a matrix-vector product of size $a \times b$. Here, a is the number of mesh elements *owned* by the node, and b is the number of mesh elements *owned* by its nearby neighbors. In the second situation, the node is considered to have been subdivided. In the subdivided case it is assumed that the arithmetic costs necessary are those due to evaluating the HFFMM far-field calculation over this node and its immediate children. In this case, the only remaining near-field evaluations are those due to mesh elements which are too large to be distributed to the child nodes.

Naturally, it is rather difficult to evaluate the cost of these two different scenarios directly from operation counts, since even if a realistic count of the number of operations needed could be obtained, variations in hardware performance (both within a single machine and across multiple machines) would likely render them unreliable. Instead, a more empirical approach to scoring each situation is taken. To determine whether a node subdivision would result in reduced computation time, the two scenarios are scored using a weighted sum of parameters. The parameters come from simple theoretical estimates of the number of operations required, and the weights come directly from a series of timing tests performed during initialization. The two timing tests needed to gauge the cost are based on the most computationally intensive parts of the matrix-vector product: the sparse matrix-vector product and the evaluation of an FFT of the size needed by the multipole-to-local transformation. The score of each scenario is then calculated and the decision to subdivide the node in question is made according to rule 7.1.

Rule 7.1 *Let there be a node, $\mathcal{N}_i$, which lies in a three dimensional tree at level, $m < n_{\max}$. In addition, assume the number of mesh elements* owned *by $\mathcal{N}_i$ (without subdivision) is a, and that the number of mesh elements* owned *by its nearby-neighbors be b. Furthermore, assume that if $\mathcal{N}_i$ were to be subdivided, that the number of mesh elements it would continue to* own *(those not given to its children) would be c. Also, assume that $\{d, p, z\}$ are the parameters chosen to govern the application of the HFFMM, where d is the number of divisions, p is the degree of the multipole expansions, and z is the zero-mask size. Now, let the average time required to evaluate a sparse matrix-vector product consisting of r non-zero elements be α, and let β be the time needed to evaluate a single three dimensional FFT of total size λ^3, where $\lambda = 2d(z+1)$. Then, we can compute two scores for the node, $\mathcal{N}_i$; the subdivided situation score, $f_{\boxplus}$, and the un-subdivided situation score, $f_{\square}$, as follows:*

$$f_{\square} = \alpha\left(\frac{ab}{r}\right) \tag{7.1}$$

and

$$f_{\boxplus} = 2p^4\beta + \alpha\left(\frac{cb}{r}\right) \tag{7.2}$$

If $f_{\boxplus} < f_{\square}$, then the node, $\mathcal{N}_i$, is subdivided, otherwise it is left unchanged.

It should be noted that certain caveats may apply to the use of rule 7.1. The first important caveat is that the weight applied to the cost of the sparse matrix-vector product evaluation, β, may vary substantially depending on whether the sparse-matrix is small enough to fit in RAM, or is too large and needs to be off-loaded to a hard disk. Since the complete size of the sparse matrix cannot be known before the entire region tree is constructed, it is usually best to use the most conservative estimate in order to avoid excessive memory usage. The second caveat is that when the Krylov solver is used in conjunction with a preconditioner (usually an iterative solver coupled with a low-degree HFFMM approximation) the relative proportion of time spent in the near-field evaluation may increase. Typically in this situation the simplest way to bias the scoring back towards a more balanced strategy is to re-weight the cost of the FFT calculation by evaluating the scoring functions with an artificially smaller value of the expansion degree, p. Usually an appropriate choice is to simply compute the weights with the value of p used by the preconditioner.

7.9 Parallel Implementation on Graphics Processing Units

In order to take advantage of the capabilities of modern computer hardware, it is necessary to make use of parallelism whenever possible. Graphics Processing Units (GPUs) provide a tremendous amount of computing power in a relatively small and inexpensive package. However, in order to exploit the capabilities of these devices, we must organize our calculations around the single-instruction-multiple-data (SIMD) paradigm which governs the architecture of GPUs. The organization of most GPUs relies on having a large number of relatively simple (at least in comparison with a general purpose CPU) processors along with several layers of memory. A graphical representation of a generic GPU system is exhibited in figure 7-5.

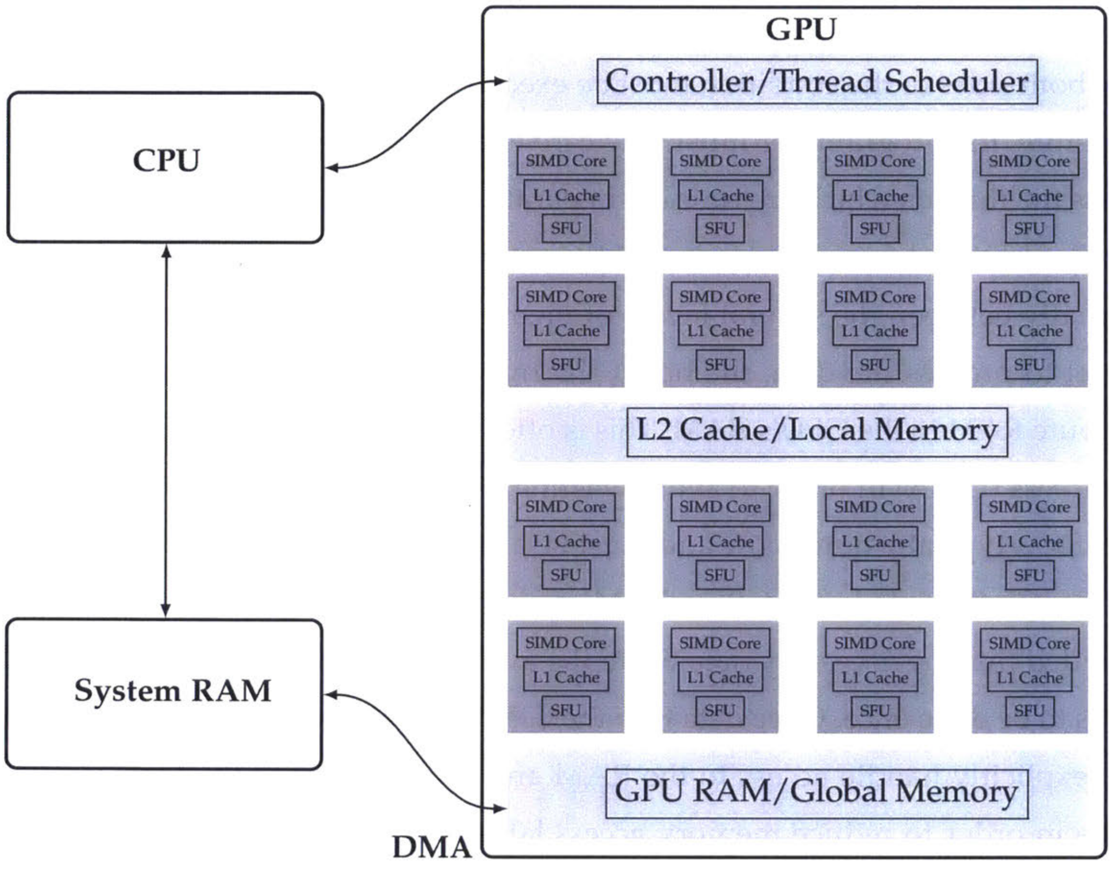


Figure 7-5: Basic architecture of a GPU enabled system.

The array of processors within the GPU each contain a core which executes the SIMD instruction stream along with a small portion of L1 cache memory and a special function unit (SFU). Together, these processors act on data in the local (L2 cache) and global memory (GPU RAM). To leverage the power of this architecture, it is necessary for the programmer to manually manage the movement of data through each layer of memory while maintaining synchronization between each running thread. To do this in an abstract and machine independent way, the OpenCL programming language has been developed by the Khronos group [112]. OpenCL organizes the SIMD computation needed by the program into a series of sequentially executed kernels. Kernels are small programs containing the instructions to be run in parallel on the GPU processors. The GPU processor cores then all execute the same kernel instruction stream in lock step upon different data sets. Since the simple GPU processors lack branch prediction capabilities, if a branch condition is encountered in the instruction stream with different results between threads, then both sides of the condition must be executed sequentially before proceeding. Therefore, the programmer must take care to avoid divergent branch conditions across the threads which are running in parallel.

An additional complication is that the memory management must be handled explicitly between the several layers of memory on the GPU. Before executing a kernel to process the data, the host CPU must transfer memory from the main compute RAM to the GPU's RAM. This is often accomplished through direct memory access (DMA) in order to avoid wasting CPU processing time. This memory transfer is typically quite slow and must be avoided whenever possible. However, in OpenCL the data in the GPU's RAM can be maintained in a persistent state. So provided that the necessary data fits in the available space, the initial transfer only needs to be done once. Once data is in the GPU's global RAM the kernel code must also explicitly handle access to the RAM and transfer data to the local (private) cache in order to reduce memory access latency. Poor memory access patterns can drastically reduce the performance of GPU code, and not all calculations (e.g. sparse matrix-vector multiplication) are amenable to the memory architecture of the GPU.

7.9.1 OpenCL Kernels

OpenCL excels at processing problems with high arithmetic intensity but does poorly when there is a large number of conditional statements. For this reason, only the most computationally intensive portions of the HFFMM were ported to run on the GPU. On account of the algorithm complexity and the amount of branching needed by the tree construction and tree traversal, these subroutines were not ported to the GPU. The primary subroutines ported to run on the GPU include:

- The source-to-multipole moment (P2M) calculation.

- The multipole-to-multipole (M2M) translation calculation.
- The multipole-to-local (M2L) transformation convolution (both the FFT and the point-wise summation).
- The local-to-local (L2L) transformation calculation.

A basic outline of the program flow involving GPU acceleration is shown in figure 7-6.

Of all the ported routines, the source-to-multipole calculation is the easiest to parallelize. This is due to the simple fact that all of the individual mesh elements are completely independent of each other. For this task, each thread is assigned the computation of all of the moments of a single mesh element. Since the memory required to store the multipole expansions of all of the mesh elements typically exceeds that of the GPU's RAM, each pass of the kernel responsible for the source-to-multipole calculation must be staggered with a kernel whose task is to handle the summation of multipole moments into the expansions of the leaf nodes of the tree. During the reduction pass care must be taken to avoid race conditions. To avoid this problem, the number of unique nodes present among the mesh elements in the buffer is determined and each thread is responsible for summing the expansion contributions of a single unique node.

Once the multipole moment expansions are available for all nodes containing sources, the next step is the upward pass. This process is organized by following the region tree traversal in reverse recursive (depth-first) order. The CPU host performs the tree traversal and in this process identifies the relationships between the child and the parent nodes. The node identities are then passed to the kernel running on the GPU, where they are used to look up the appropriate expansion data in the GPU buffer. Once an appropriately large list of node identities is available to the GPU kernel, the multipole-to-multipole expansion transformation process is executed. In a manner similar to the source-to-multipole calculation, a separate reduction pass is needed in order to avoid race conditions in the summation step.

After the multipole expansions of each source region have been calculated,

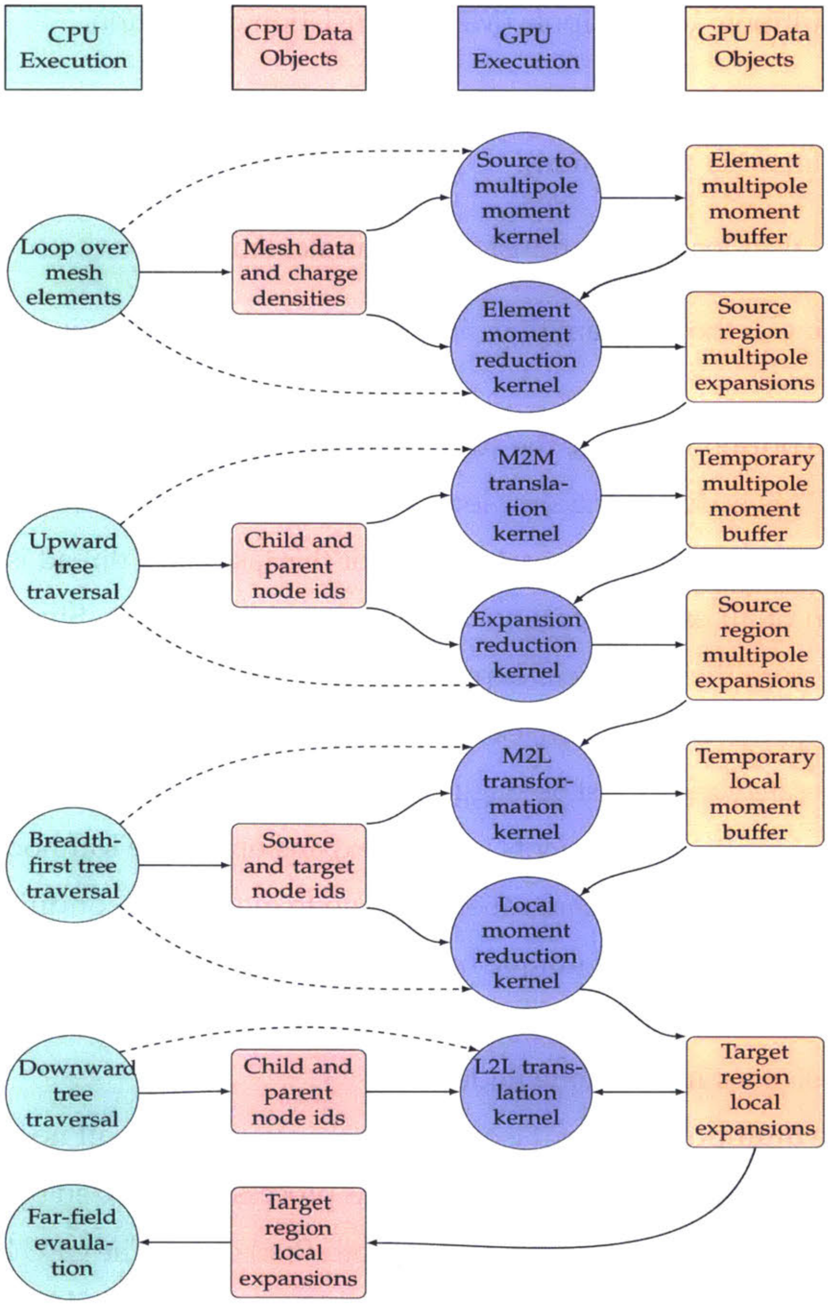

CPU Execution
CPU Data Objects
GPU Execution
GPU Data Objects
Source to multipole moment kernel
Element multipole moment buffer
Loop over mesh elements
Mesh data and charge densities
Element moment reduction kernel
Source region multipole expansions
M2M transla- tion kernel
Temporary multipole moment buffer
Upward tree traversal
Child and parent node ids
Expansion reduction kernel
Source region multipole expansions
M2L transfor- mation kernel
Temporary local moment buffer
Breadth- first tree traversal
Source and target node ids
Local moment reduction kernel
Downward tree traversal
Child and parent node ids
L2L trans- lation kernel
Target region local expansions
Far-field evaula- tion
Target region local expansions

Figure 7-6: Program diagram of the OpenCL accelerated calculation, showing the instruction streams and data objects which exist concurrently. Time flows from the top to the bottom. The left half of the chart consists of data and execution objects existing exclusively on the host (CPU) device, whereas the right half belongs to objects dwelling on the GPU device. The green and blue shaded circles represent control and arithmetic operations belonging to the CPU and GPU respectively. The purple and red shaded rectangles represent the data objects on each device. The solid arrows indicate the flow of data, while the dashed arrows indicate the control hierarchy between different operation subroutines.

the next step is to perform the multipole-to-local coefficient conversion on these expansions. This is also managed by having the CPU host perform a corecursive (breadth-first) traversal of the region tree, which generates a list of associated source and target nodes to participate in the transformation. The multipole-to-local transformation kernel then performs the Fourier transform accelerated convolution for many node collections at once and writes the result into a buffer which can then be reduced by a separate summation kernel.

Following the multipole-to-local conversion, the local-to-local transformation is the last remaining GPU accelerated step. It functions in a way very similar to the multipole-to-multipole transformation step, except that the tree traversal is done in forward recursive order. In addition, no separate summation kernel is necessary to avoid race conditions for the local-to-local transformation, because each child node receives only a single contribution from its parent.

Once the local coefficient expansions are available for all regions containing targets (collocation points) ,the expansion data is passed back from the GPU to the CPU to be used for far-field evaluation. In principle, the far-field evaluation step could also be accelerated by the GPU, but since it is responsible for only a small fraction of the total execution time, this is not considered necessary.

7.10 A Parallel Implementation Using the Message Passing Interface (MPI)

Another paradigm for exploiting parallelism is the multiple-instruction-multiple-data (MIMD) technique, most commonly associated with large computing clusters[2]. This type of parallelism is well suited for algorithms which contains a large number of conditional statements and whose control flow is highly dependent on the data they are processing. On account of this property, MIMD is not well suited to the

[2]It should be noted that the MIMD paradigm is not the same as multiple-program-multiple-data (MPMD), sometimes known as high-throughput computing (HTC), which consists of running many entirely independent processes which do not maintain any type of synchronization or communication.

dense collection of relatively simple processors on a GPU, but it is ideally suited for handling the control of the HFFM method over a cluster of CPU-based machines which may or may not be GPU accelerated.

The Message Passing Interface (MPI) is a standard for the set of libraries used to abstract away the details of synchronizing and communicating between a collection of processes executing disparate tasks directed toward a common result. There is a large variety of implementations of the MPI standard, however, `KEMField` is typically linked against the version distributed by the OpenMPI collaborative [86]. OpenMPI provides the functions needed to construct arrays of processes with specific topologies, maintain synchronization between these processes, and perform synchronous and asynchronous communication between various subsets of running processes.

7.10.1 Static Load Balancing

Some minimal degree of synchronization obviously must be maintained between individual MPI processes so they can cooperate in computing a matrix-vector product for a Krylov subspace solver. This is usually accomplished through the use of a barrier call which forces all processes to reach the same point in the program before they are allowed to proceed. However, processing power is wasted any time a thread is left idle while waiting for others to finish. Therefore, it is important to find a way to balance the work loads between processes to ensure as little time is wasted as possible. For example, it would be extremely inefficient to give one process on a computing cluster 99% of the work load while dividing the remaining 1% over a myriad of other processes.

Since our problem (solving Laplace-BVP) is oriented around a spatial region (containing the meshed boundary of interest), the primary technique for properly apportioning the work load is domain decomposition. In a domain decomposed problem, each separate process is responsible for the work in some spatial subset of the region enclosing the mesh. Unfortunately, determining a domain decomposition which produces an appropriate partitioning of the necessary work among several processes is very difficult. In the ideal scenario, a uniform (spatial) division of the region of interest across all the available compute nodes ought to yield an even distribution of the arithmetic work. However, most real world geometries we wish to consider are highly non-uniform, so such a naive approach produces an unsatisfactory work distribution. This task is further complicated by the fact that many computing clusters contain heterogeneous hardware with varying performance. Given these complications, it generally best to take a more heuristic approach to the domain decomposition problem than to attempt to deduce the optimal partition from theory.

Before attempting to apportion work among several MPI processes, it is necessary to have the HFFM method first construct the region tree. Each process needs a copy of the entire region tree in order to determine how to execute the algorithm operations over its domain and communicate relevant data to other processes. Once each process has a copy of the full tree structure[3], the domain decomposition can take place. This decomposition is done entirely at the top level of the tree. Work is distributed discretely by assigning each process responsibility for the region contained by some set of nodes at the top level. While the domain decomposition process could in principle cross tree levels, restricting it to the top level is helpful in reducing both the amount and the complexity of the needed interprocess communication. However, a consequence of limiting the domain decomposition to the top level of the tree is that it is important to use as large a value as reasonably possible for the number of top level divisions, d_t. Using a large value for d_t provides better granularity in the amount of work that can be apportioned between each process, and in turn, results in better load balancing. This is in fact the reason why the number of top level divisions is provided to the user as a separately controllable parameter.

Immediately following the region tree construction, all of the top level nodes

[3]The tree construction is completely deterministic, so in practice, it is simpler to have each process construct its own copy of the region tree itself rather than having this done by a single process and then passing the needed data between processes.

are uniformly divided between all of the available MPI processes with no regard to their contents. To find a more equitable work distribution, we first have a scoring mechanism assign a work value to each node at the top level of the tree. This scoring mechanism takes into account the estimated amount of arithmetic work needed to process the region contained by each top level node. The score of each top level node is simply the total sum of all the scores of each undivided node (see equation 7.1) and each subdivided node (see equation 7.2) encountered during the recursive visitation of the portion of the region tree on its branch. Once the work score of each top level node is known, we then need to find a way to distribute nodes across the MPI processes in a way which equalizes the work done by each thread. This multi-way partition problem is a variation on the knapsack problem and is known to be NP-complete and quite difficult to solve exactly [144]. However, finding the optimal solution is not necessary for our purpose; an approximate solution will suffice. This greatly simplifies the task and allows us to use a simple simulated annealing approach.

Load balancing through simulated annealing works by stochastically minimizing an energy function. Since the time-to-solution is governed by the slowest thread (i.e. the one which does the most work), an appropriate choice for the energy function to be minimized is the maximum of all the thread work scores. We note that this choice of energy function does not necessarily balance the work done by each MPI thread, but rather attempts to minimize the cost of the worst case work load. This allows a little more flexibility in choosing a solution and yields faster convergence to an acceptable work partition.

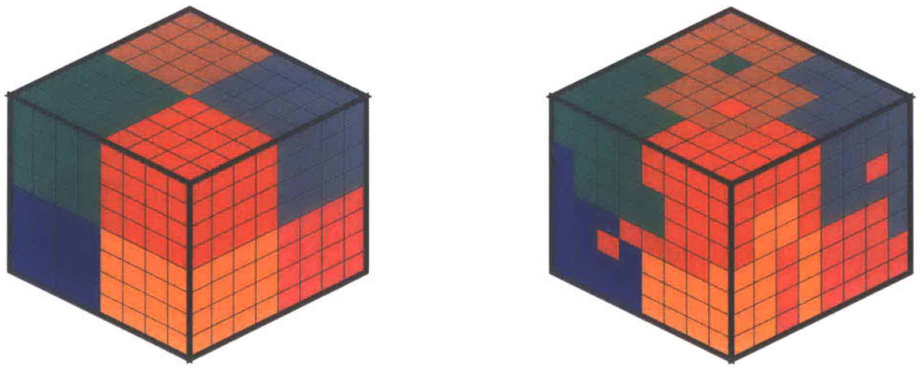
To perform the minimization we start with the initial node-thread distribution and then iteratively permute the partitioning by taking a random node from one process and giving it to another. Upon this permutation, we then compute the energy function of the new state. If the energy is lower than the previous state, we always make the transition. The permutation is disfavored if the new energy is higher than the previous state. However, we do not completely ignore this new state, but rather allow for some non-zero probability to make the transition anyways.

This transition probability is governed by the size of the energy difference and an artificial temperature which is gradually lowered after each iteration. Allowing disfavored transitions allows for a more exhaustive search of the parameter space and yields a greater likelihood of finding the global minimum. The node-thread partitioning method is given in algorithm 9.

Algorithm 9 Assign top level nodes to MPI threads to balance work load.

```
Input: The $d_t^3$ top-level nodes $\{\mathcal{N}_i\}$ and the number of threads $T_{\text{MPI}} < d_t^3$.
 1: Arbitrarily sort top-level nodes into $T_{\text{MPI}}$ sets $S_k$ associated with the $k$-th thread.
 2: Compute the work scores associated with each top-level node: $s_i = s(\mathcal{N}_i)$.
 3: Compute energy function of initial state $E_0 = \max_k \sum_{\mathcal{N}_i \in S_k} s(\mathcal{N}_i)$.
 4: Set initial temperature to $t_0 = E_0/2.0$, $\gamma = 0.95$, and iteration count $n = 0$.
 5: while n < $d_t^3 \times T_{\text{MPI}}$ do
 6:     Randomly choose two sets $S_a$ and $S_b$ with $0 \leq a,\ b < T_{\text{MPI}}$.
 7:     Randomly choose a node $\mathcal{N}_i \in S_a$. Move $\mathcal{N}_i$ from $S_a$ to $S_b$.
 8:     Compute the new energy function of the current state $E_n = \max_k \sum_{\mathcal{N}_i \in S_k} s(\mathcal{N}_i)$
 9:     if $E_n > E_{n-1}$ then
10:         Compute probability for unfavored transition $p = e^{-(E_n - E_{n-1})/t_{n-1}}$.
11:         Randomly generate $x \in [0, 1]$
12:         if $x > p$ then
13:             Restore original state: Move $\mathcal{N}_i$ back to $S_a$ from $S_b$.
14:         end if
15:     end if
16:     Update $t_n = \gamma t_{n-1}$, and $n = n + 1$.
17: end while
Output: The $T_{\text{MPI}}$ updated sets $S_k$ listing the top-level nodes associated with each
    MPI thread.
```

It should be noted that since this is a stochastic algorithm, it causes the MPI threads to diverge during execution. However, since this optimization problem is embarrassingly parallel, the thread divergence can be used to our advantage as a means of exploring the solution space more extensively. The best solution among the candidates generated by each MPI process is taken to be the final node-thread distribution. Figure 7-7 demonstrates graphically the node-to-thread distribution before and after the simulated annealing process has been performed to re-balance the work load.



(a) Initial node-process distribution (b) After node redistribution.

Figure 7-7: Visual comparison of top level node distribution among MPI threads before and after static load balancing has been performed through simulated annealing. Nodes belonging to the same process are shaded the same color. In this example $d_t = 8$ and the number of MPI threads is $T_{\text{MPI}} = 8$.

7.10.2 Task Division

Until now, we have implicitly assumed that while each of the MPI threads may be responsible for processing a different subset of the region tree, it is essentially performing the same task as all the other threads. However, unlike SIMD parallelism, this is not a requirement in the MIMD paradigm. This gives us the freedom to allocate specific tasks to certain MPI threads. We can use this ability to simultaneously evaluate the near-field and far-field portion of the matrix-vector product.

This task-based division of work is advantageous when running on hybrid computing clusters, that is to say, clusters which have GPU accelerators available on their compute nodes. Typically, when running on such a cluster, while calculations are being run on the accelerator device (GPU), the CPU is generally left idle. However, we can make better use of these types of machines when `KEMField` is compiled with both MPI and OpenCL based parallelism enabled. To make use of the available CPU processing power while waiting for results from the accelerator device, we can assign a separate MPI thread to run on it and perform some useful work. In this case, we can assign the CPU the task of evaluating the near-field (sparse matrix) portion of the matrix-vector product.

Having the GPU device evaluate the far-field while the CPU evaluates the near-field effects is a division of work which plays to the advantages of each type of device. The far-field evaluation (requiring the application of the HFFMM) has high arithmetic intensity (requiring the evaluation of many small FFTs) and little divergence among threads. This makes the far-field evaluation well suited to running on GPUs. The near field (sparse matrix) evaluation on the other hand is a task which is generally ill-suited to SIMD type devices such as GPUs. This is for two reasons. The first is that there is a large amount of data dependent branching which causes thread divergence. This is detrimental to performance on SIMD devices. The second reason is that it requires many small and frequent reads from RAM. This also negatively affects performance on devices such as GPUs because of their small local memory cache and the manner in which they access their global memory (RAM) bank. GPUs favor coalesced reads of large chunks of memory which is difficult to adapt to the task of sparse matrix-vector product evaluation without wasteful access of unnecessary data. However, CPUs excel at branch prediction and have much larger local memory caches which favors their use for this task.

Therefore, to maximally exploit the available hardware (specifically hybrid clusters) the `KEMField` implementation of the HFFMM algorithm employs both MPI and OpenCL. Furthermore, we leverage the MIMD ability of MPI to assign work loads which are best suited to each aspect of the hardware.

7.11 Field Map Generation and Calculation

So far, we have extensively discussed the HFFM method as it pertains to solving the Laplace boundary value problem using the BEM, but we have neglected to discuss its use for the fast calculation of the electric potential and field at arbitrary points. Since solving the Laplace BVP with BEM only requires us to evaluate the field and potential at fixed points (collocation points), we are allowed to make certain optimizations (such as using a sparse matrix to evaluate the near field terms) which do not hold for arbitrary locations. However, for charged particle tracking, we must be able to evaluate the potential and field at any point in the region of interest.

For the purpose of creating a field map, the HFFM method proceeds to compute the local coefficient expansions in much the same manner as when used for evaluating the Laplace BVP system matrix-vector product. However, in this case we need to construct local coefficient expansions for all leaf nodes in the region tree, not just those which contain collocation points. Hence, we cannot reduce the work needed by pruning nodes from the tree which do not contain mesh elements. This is because we cannot predict in advance where charged particles may be tracked. Unfortunately, this leads to a much larger memory usage for the local coefficient expansions. While this is unavoidable, if we desire field evaluation which is as fast and accurate as possible, adjusting the parameters $\{d_t, d, z, p, n_{\max}\}$ governing tree construction allows a great deal of user control over the relative importance given to speed, accuracy and memory usage.

Once the local coefficient field map is constructed, the first step in the field calculation is locating the leaf node containing the point of interest, $\mathbf{p}$. This can be done in a simple, recursive manner. At any level of the tree the spatial indexes (a_0, a_1, a_2) of the child node containing $\mathbf{p}$ can be calculated from the center, $\mathbf{c}$, and side length, l, of the cube associated with the parent node according to equation 7.3:

$$\begin{aligned} a_0 &= \left\lfloor \frac{\Delta x}{d} \right\rfloor = \left\lfloor \frac{p_x - (c_x - l/2)}{d} \right\rfloor \\ a_1 &= \left\lfloor \frac{\Delta y}{d} \right\rfloor = \left\lfloor \frac{p_y - (c_y - l/2)}{d} \right\rfloor \\ a_2 &= \left\lfloor \frac{\Delta z}{d} \right\rfloor = \left\lfloor \frac{p_z - (c_z - l/2)}{d} \right\rfloor \end{aligned} . \tag{7.3}$$

During the first step in the recursion, when calculating the containing child node at the top level of the tree, if $d_t \neq d$ then equation 7.3 should be modified accordingly. The recursion terminates once a leaf node (a node with no children) has been reached.

Upon determination of the leaf node containing $\mathbf{p}$, we must have quick access to the list, $\mathcal{L}_n$, (the direct evaluation list) of all the mesh elements which are considered to be in the near-field. These nearby mesh elements must have their contributions evaluated directly, as their contribution is not included in the local coefficient expansion in order to maintain accuracy. This set of mesh elements are those which are either *owned* by the leaf node containing **p**, its nearby-neighbors, or by any of its parents or their respective nearby-neighbors.

Since we want to evaluate the field quickly, we need to be able to construct the direct evaluation list faster or at least as fast as it takes to evaluate the field from the elements on the list. However, in order to construct the direct evaluation list for a leaf node, a fair amount of work is necessary to perform the needed tree traversal. This is because we must visit all of a leaf node's nearby-neighbors and all of the nearby-neighbors of all its parent nodes. This process can be quite expensive when the tree is deep and is unfortunately too slow to perform on an as-needed basis. On the other hand, it can be quite memory intensive to pre-compute and store the direct evaluation list for each leaf node. So as a compromise between these two extremes (as-needed vs. pre-computed), we can instead have each leaf node keep a set of references to the lists (which are stored by the `KFMIdentitySet` object associated with each node) of nearby mesh elements which would be merged to form the direct evaluation list. This set of references is managed by the class `KFMIdentitySetList`. This allows the list of mesh elements in the near-field to be constructed quickly at the time the field evaluation is needed, without navigating the region tree to visit nearby nodes and retrieve their mesh element lists.

We can compute the potential at any point in the region of interest through the use of equation 7.4:

$$\Phi(\mathbf{p}) = \Phi_{\text{near}}(\mathbf{p}) + \Phi_{\text{far}}(\mathbf{r}) = \\ \frac{1}{4\pi\epsilon_0} \sum_{u_i \in \mathcal{L}_n} \int_{u_i} \frac{\sigma_i}{|\mathbf{p} - \mathbf{r}'|} du + \frac{1}{4\pi\epsilon_0} \sum_{j=0}^{p} \sum_{k=-j}^{j} L_j^k Y_j^k(\theta, \phi) r^j \,, \quad (7.4)$$

where $\mathcal{L}_n$ is the set of mesh elements which are considered to be nearby, L_j^k are the local coefficients associated with the leaf node containing **p**, and **r** is the displacement with respect to the local origin (the leaf node's cube center **c**), given by:

$$\mathbf{r} = \mathbf{p} - \mathbf{c}\,. \tag{7.5}$$

The electric field can be evaluated according to equation 3.12 as the negative gradient of the potential, as follows:

$$\mathbf{E}(\mathbf{p}) = -\nabla\Phi(\mathbf{p}) = \mathbf{E}_{\text{near}}(\mathbf{p}) + \mathbf{E}_{\text{far}}(\mathbf{r})\,. \tag{7.6}$$

This gradient can be computed through the use of numerical differentiation; however, this is not necessary. The near field term can be computed directly using equation 7.7:

$$\mathbf{E}_{\text{near}}(\mathbf{p}) = \frac{1}{4\pi\epsilon_0} \sum_{u_i \in \mathcal{L}_n} \int_{u_i} \frac{\sigma_i(\mathbf{p}-\mathbf{r}')}{|\mathbf{p}-\mathbf{r}'|^3} du\,, \tag{7.7}$$

for which analytic methods exist [53]. The far-field term is more quickly and accurately evaluated by first applying the gradient in spherical coordinates and then performing a coordinate transformation. The components of the gradient can be computed according to equations 7.8 through 7.10 as follows (see [159]):

$$E_r = -\frac{\partial\Phi}{\partial r} = -\frac{1}{4\pi\epsilon_0} \sum_{j=1}^{p} \sum_{k=-j}^{j} L_j^k Y_j^k(\theta,\phi) j r^{j-1}\,, \tag{7.8}$$

$$E_\theta = -\frac{1}{r}\frac{\partial\Phi}{\partial\theta} = -\frac{1}{4\pi\epsilon_0} \sum_{j=1}^{p} \sum_{k=-j}^{j} L_j^k \sqrt{\frac{(j-|k|)!}{(j+|k|)!}} \frac{\partial P_j^{|k|}(\cos\theta)}{\partial\theta} e^{ik\phi} r^{j-1}\,, \tag{7.9}$$

$$E_\phi = -\frac{1}{r\sin\theta}\frac{\partial\Phi}{\partial\theta} = -\frac{1}{4\pi\epsilon_0} \sum_{j=1}^{p} \sum_{k=-j}^{j} L_j^k Y_j^k(\theta,\phi) \frac{ik}{\sin\theta} r^{j-1}\,. \tag{7.10}$$

Finally, the far field term of the electric field is given in Cartesian coordinates through the transformation:

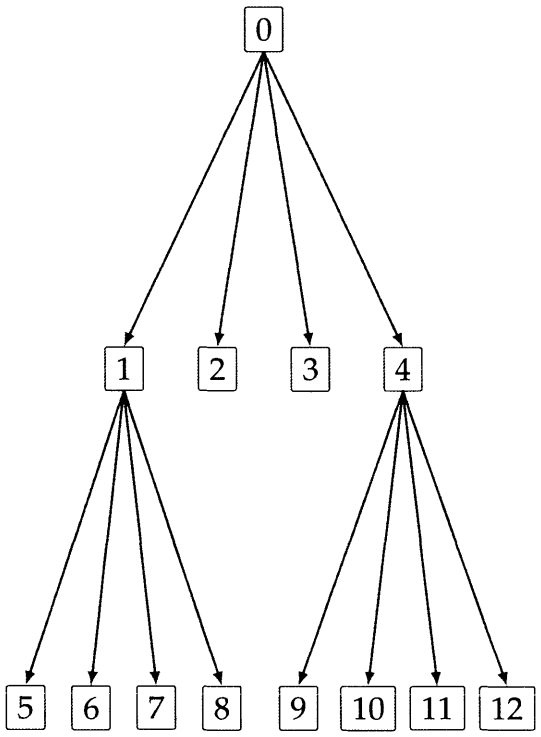
$$\mathbf{E}_{\text{far}}(\mathbf{r}) = \begin{pmatrix} E_x \\ E_y \\ E_z \end{pmatrix} = \begin{bmatrix} \sin\theta\cos\phi & \cos\theta\cos\phi & -\sin\phi \\ \sin\theta\sin\phi & \cos\theta\sin\phi & \cos\phi \\ \cos\theta & -\sin\theta & 0 \end{bmatrix} \begin{pmatrix} E_r \\ E_\theta \\ E_\phi \end{pmatrix}. \tag{7.11}$$

It should be noted that because of the presence of the factor of $(1/\sin\theta)$ in the E_ϕ component, there is some numerical instability encountered when θ is small (near the z-axis). When this case is encountered, we can avoid this problem by evaluating the far-field term of the electric field either through numeric differentiation or by a coordinate transformation. In order to avoid numerical differentiation which is itself sometimes subject to numerical instability, coordinate transformation is the technique implemented in `KEMField`. This is done by first applying a rotation (of $\pi/4$ about the y-axis) on the set of local coefficients, L_j^k according to theorem A.1. Then the electric field is evaluated in the transformed coordinate system, away from the problematic z-axis, and finally, the result is transformed back to the original coordinate system.

7.12 Field Map Persistence

The last aspect of the fast field map implementation which must be considered is the ability to save and reconstruct the field map objects so they may be reused in multiple execution contexts. Maintaining persistence of the region tree, nodes, and their associated data requires that they be serialized so that they can be streamed to disk. This is a non-trivial task, since by its nature, the region tree has a nonlinear memory structure with many cross references between nodes and the data objects they contain. In order to simplify this task, we break the data to be serialized into two parts. The first part manages the structure of the tree, while the second part contains the data objects attached to the tree nodes (such as the local coefficient expansions).

To serialize the tree, we first assign each node a unique number by which we may identify it. Then, to serialize the structure of the tree, we perform a corecursive visitation over the tree and store the identities of the parent and children associated with each node. An example of this simple scheme is shown for a very small tree in figure 7-8. In this example, we have chosen to exhibit the serialized data in a human readable XML format, though in reality, this relational data would be stored on disk in a binary format. We should note that although this storage format does not generate the smallest memory footprint possible, having bidirectional information in the serialized data makes reconstruction of the tree much simpler. To reconstruct the tree, we simply allocate a linear array containing the total number of nodes in the tree, assign unique identifiers to each one, and then re-link the nodes according to the serialized tree structure information.



(a) Graphical representation of a simple region tree.

```
<tree>
  <node>
    <id> 0 </id>
    <parent> </parent>
    <children> 1 2 3 4 </children>
  </node>
  <node>
    <id> 1 </id>
    <parent> 0 </parent>
    <children> 5 6 7 8 </children>
  </node>
  <node>
    <id> 2 </id>
    <parent> 0 </parent>
    <children> </children>
  </node>
                      ⋮
  <node>
    <id> 4 </id>
    <parent> 0 </parent>
    <children> 9 10 11 12 </children>
  </node>
  <node>
    <id> 5 </id>
    <parent> 1 </parent>
    <children> </children>
  </node>
                      ⋮
  <node>
    <id> 12 </id>
    <parent> 4 </parent>
    <children> </children>
  </node>
</tree>
```

(b) Node relations serialized into XML. Vertical dots indicate some omitted data.

Figure 7-8: Example of the serialization of the region-tree node relationships. Output is shown in a human-readable XML format.

Once the marshaling of the tree structure is taken care of, all that remains is to serialize the data objects associated with each node. This task is made simpler by the fact that we don't need any auxiliary data, only that which is required to compute the fields. This consists of the local coefficient expansions (`KFMElectrostaticLocal-`

CoefficientSet) and the lists (KFMIdentitySet) of the mesh elements *owned* by each node. These objects are quite simple in structure, having no external references other than which node they belong to, and are straightforward to serialize. Upon reconstruction, the references to nearby mesh element lists (KFMIdentitySetList) needed for evaluation of the near-field terms can be reconstructed with a small amount of additional work during initialization by repeating the original method of construction.

7.13 Performance of the HFFM Method

There are several aspects that need to be examined in order to evaluate the performance of the HFFM method. Primarily, we are interested in evaluating its speed, accuracy and efficiency in solving the Laplace boundary value problem in both standard and parallel implementations, as well as determining the speed and accuracy of the HFFMM field map as compared to the direct method of integration.

7.13.1 Boundary Value Problem Accuracy

Solving the Laplace boundary value problem with the BEM introduces various errors due to the many approximations that have been made in order to make the problem tractable. The first approximation is the discretization of the original surface into a mesh of linear elements. If there are any curved sections in the original surface, there will necessarily be some error in how we model the surface position with the mesh. The second approximation is that we are treating the charge density on each mesh element as a constant, whereas in reality (even if the mesh model matches the original surface exactly), the charge density will vary within a mesh element. And thirdly, we are treating the fields as approximately constant over each individual mesh element, by only evaluating them at the collocation point (centroid) when determining the residual error with respect to the boundary conditions. Furthermore, we are not attempting to solve the linear system generated by the BEM to absolute convergence[4], but merely to some chosen residual error, ρ. While all of these approximations might seem to conspire to make this method unusable, they are all in essence controllable, and can generally be made small enough so as to be insignificant as long as we use a large enough number of discretizations, N, and solve the linear system to an appropriately small residual error, ρ.

Coupling the HFFMM to the BEM introduces another approximation on top of an already approximate method. Since we are now using a multipole expansion to evaluate the far-field, we would like to understand how the accuracy of the combined HFFMM-BEM algorithm behaves, as the parameters of the HFFMM portion of the algorithm are changed. To do this, we explore the accuracy as a function of the two parameters which have the largest effect, the degree of the expansion, p, and the zero-mask size, z. The test problem we will first consider is that of computing the capacitance of the unit cube. The unit cube capacitance test has the advantage that there is no error introduced by linearizing curved surfaces, and while being quite simple, it does not have a trivial solution. The capacitance is also a global property of the geometry and is therefore sensitive to the global accumulated error of the technique we use to solve the problem. Unfortunately, the value of the capacitance of the unit cube is not available analytically. However, Helsing et. al. [115] have computed the value of the capacitance of the unit cube (in units of $4\pi\epsilon_0$) to be:

$$C_{\text{cube}} = 0.66067815409957\,, \tag{7.12}$$

which is accurate to a relative error of 10^{-13}. This error approaches the machine epsilon of double precision floating point math ($\sim 10^{-15}$). It serves as a sufficiently accurate reference value for our tests, since we will only solve the linear system generated by the unit cube problem to a relative L_2 residual error of $\rho = 10^{-8}$. However, before examining the accuracy as a function of p and z, we must ensure that the geometry has been discretized accurately enough that the error introduced

[4]This is not possible using floating point math in any case.

by the underlying BEM method alone is not the dominant effect. To determine the appropriate number of mesh elements, N, where the error on the BEM approximation becomes smaller than the error due to the multipole expansion, we will repeatedly solve the unit cube problem with a different value of N and expansion degree p until the error on the capacitance approaches the residual error. The result of this test is shown in figure 7-9, from which we can see that once the mesh contains approximately 10^6 elements, the combined error on the capacitance due to the BEM and multipole approximations approaches the allowed residual error. Therefore, we may fix $N = 10^6$ and $\rho = 10^{-8}$ in order to study the error on the capacitance as a function of p and z, the results of which are shown in figure 7-10.

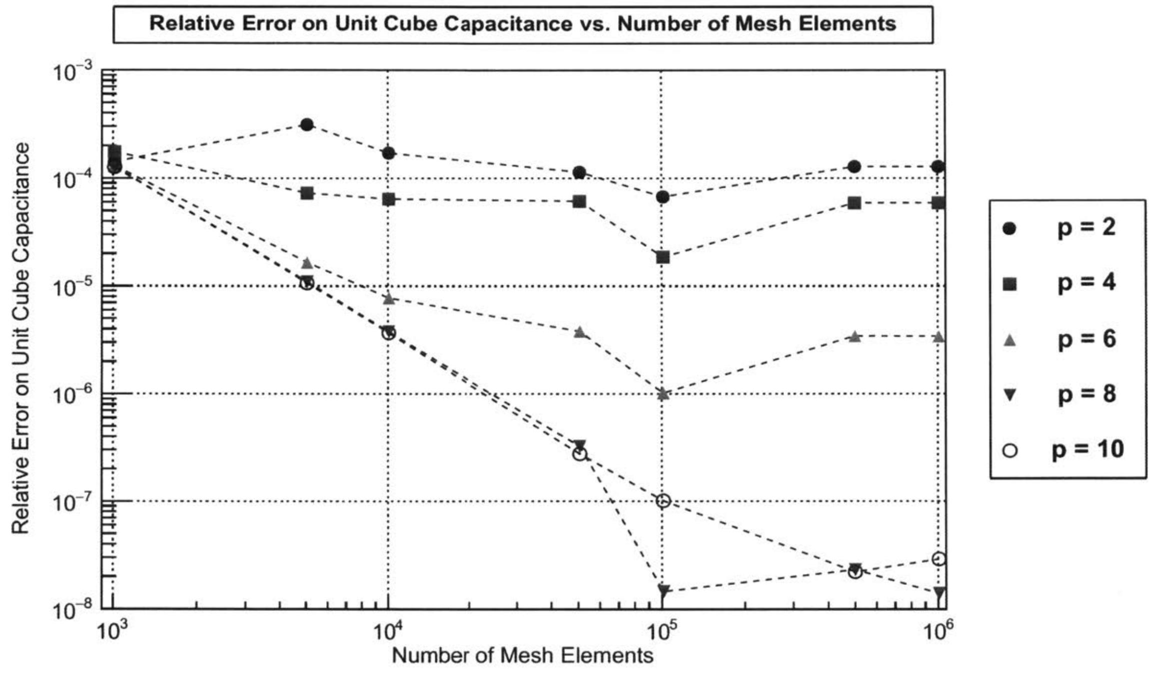


Figure 7-9: Computed error on the unit cube capacitance as a function of the number of mesh elements for various multipole expansion degrees. The linear system was solved to a relative residual error of $\rho = 10^{-8}$ using the combined HFFM method with $z = 1$. The maximum tree level was unconstrained.

From figure 7-10, we can see that as a general rule, the error of the HFFM method decreases as the expansion degree, p, increases. This is a good confirmation that the method works as expected and comes as no surprise since the error on the multipole expansion decreases according to a power law in p (see equation 5.12). In addition, the accuracy is also improved when using a larger value for the zero-mask size, z. This is also expected since this has the effect of increasing the distance between the multipole expansion center and the location where the field is evaluated.

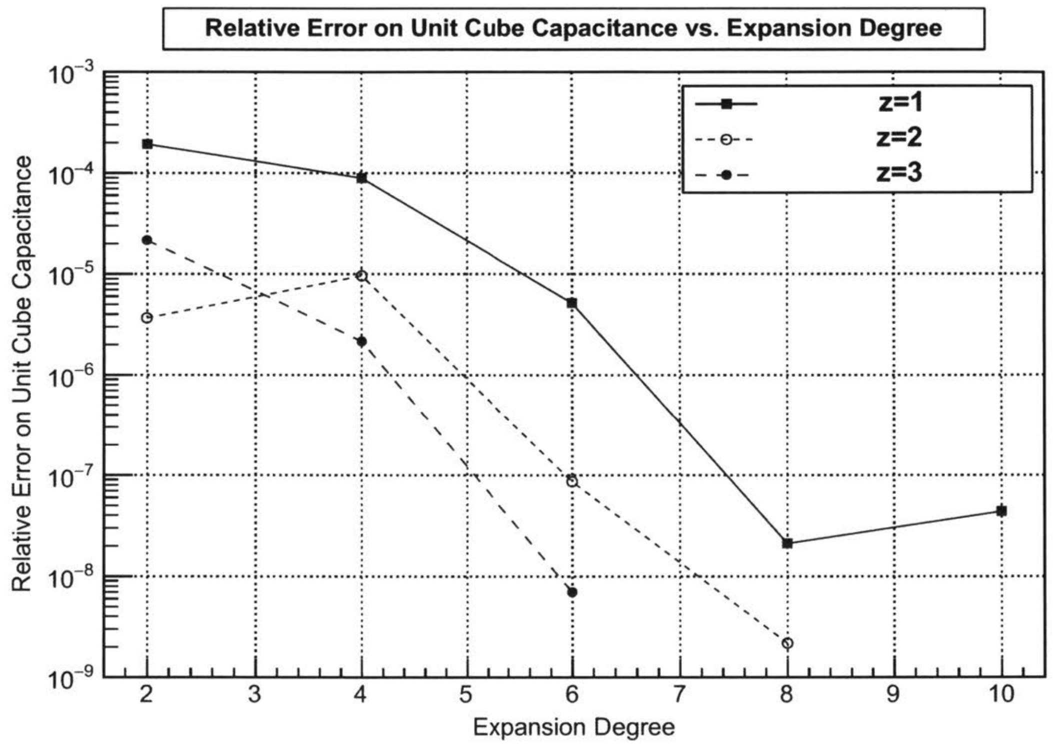


Figure 7-10: Computed error on the capacitance of the unit cube consisting of 10^6 mesh elements as a function of the multipole expansion degree p and zero-mask size z.

7.13.2 Performance with Iterative Krylov Solvers and as a Preconditioner

It is also interesting to explore the convergence behavior of the HFFMM-BEM algorithm when used in conjunction with various Krylov subspace techniques. While there are a multitude of Krylov solvers available, we will consider the two most prevalent: GMRES and BiCGSTAB, along with their preconditioned variants FGMRES and BiCGSTAB-P. We have chosen these methods because of their ability to address the solution of non-symmetric systems generated by the collocation-BEM using only matrix-vector product evaluations as discussed in section 4.4.

To test the performance of the HFFMM, we have solved the Dirichlet problem of the unit cube discretized into 5×10^4 rectangular mesh elements, with its surface set to unit potential. The first Krylov solver we considered for use along with the HFFMM was BiCGSTAB. BiCGSTAB is notable for its very simple implementation and its low memory usage. Figure 7-11 shows the relative residual norm error on the solution as a function of the number of arithmetic operations. As can be seen from figure 7-11, BiCGSTAB can sometimes stagnate and take an exceedingly long time to reach an acceptable level of error. In an attempt to improve the convergence behavior, the BiCGSTAB-P algorithm was also explored in conjunction with a HFFMM based preconditioner using a dipole ($p = 1$) approximation. Unfortunately, this type of preconditioner did not offer any improvement over the non-preconditioned BiCGSTAB algorithm and was in fact substantially worse.

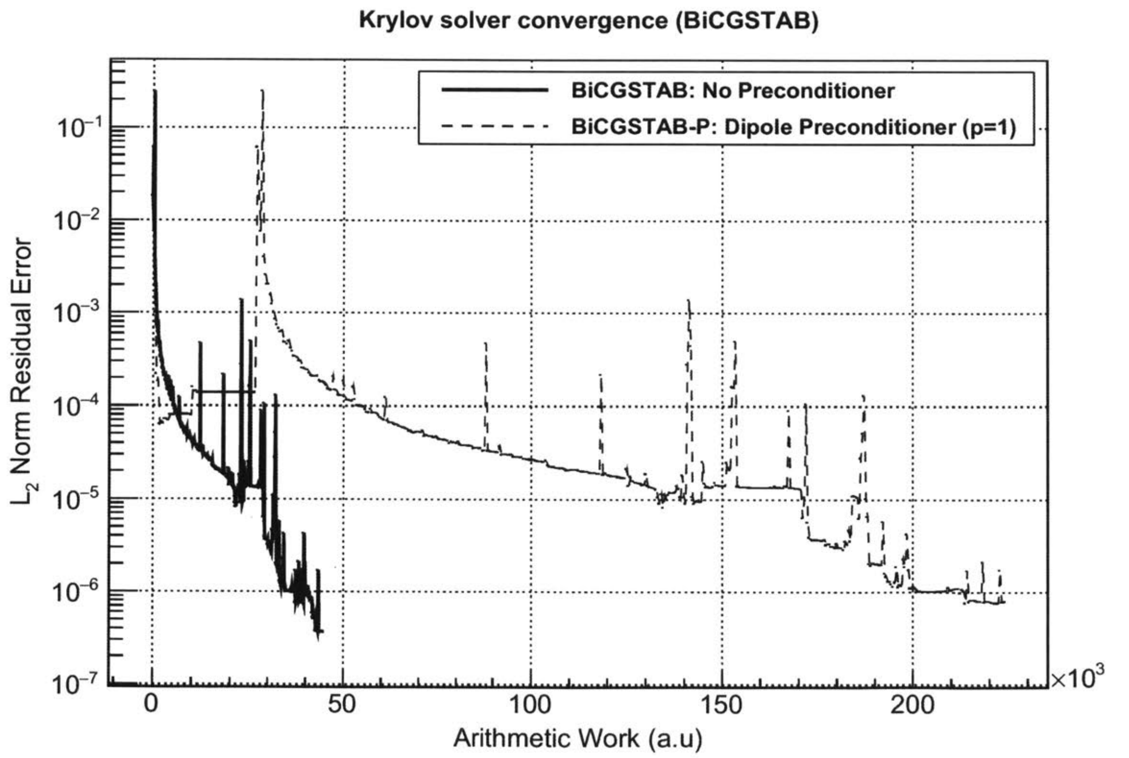


Figure 7-11: Relative L_2 residual norm as a function of the number of arithmetic operations performed when solving the unit cube Dirichlet problem with BiCGSTAB and BiCGSTAB-P.

As an alternative to BiCGSTAB, the better behaved GMRES method was also explored. Unsurprisingly, this method greatly outperforms BiCGSTAB for our test problem. Unfortunately, without preconditioning, the convergence rate of GMRES alone is still fairly slow. However, the FGMRES algorithm coupled with a simple HFFMM-based preconditioner immensely improved performance. Figure 7-12 shows the convergence behavior of GMRES and FGMRES for the unit cube problem with various preconditioners. Out of the several preconditioners tried with FGMRES, the monopole ($p = 0$) and dipole ($p = 1$) preconditioners offered the greatest amount of speed up in the rate of convergence (roughly a factor of 5) for the unit cube test case. It should be noted that for problems which contain Neumann boundary elements, the monopole preconditioner is not sufficient, since although it can approximate the potential it treats the electric field as zero. Therefore, for Neumann and mixed boundary problems the dipole preconditioner is the lowest degree preconditioner that ought to be used.

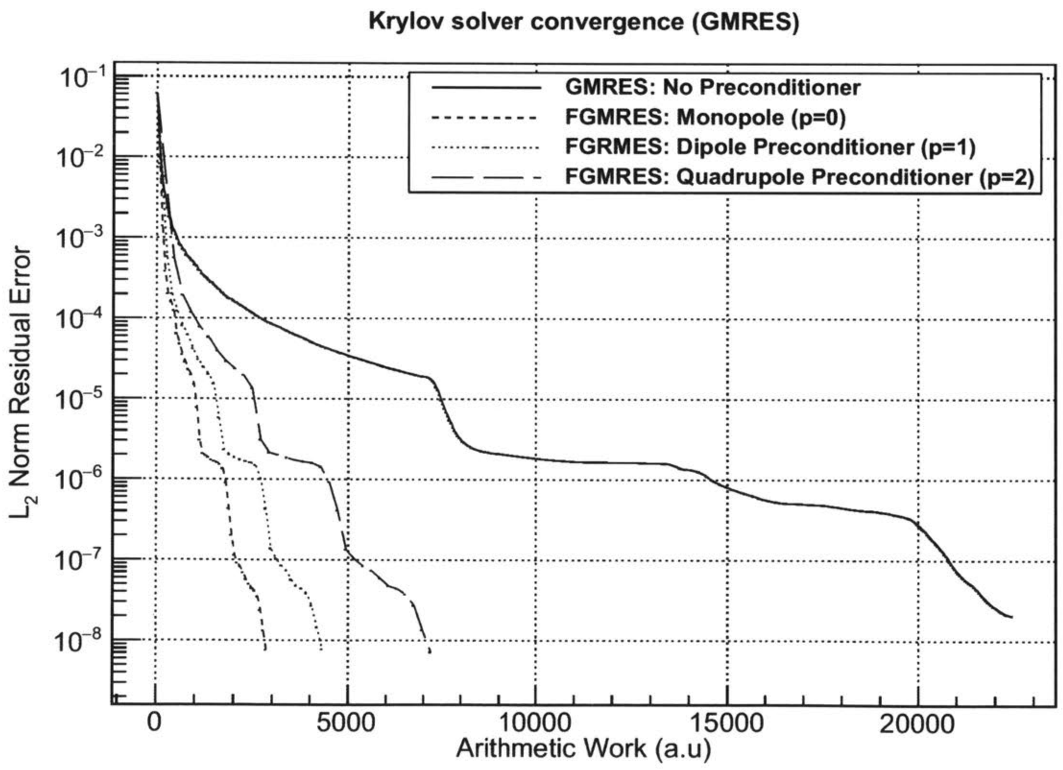


Figure 7-12: Relative L_2 residual norm as a function of the number of arithmetic operations performed when solving the unit cube Dirichlet problem with GMRES and FGMRES.

In order to benchmark the speed of the HFFMM-BEM method, we have chosen to compare it against the Robin-Hood method (which is the current KATRIN standard solver). To do this, we have considered the wallclock time taken to solve the simple problem of computing the unit capacitance of a sphere discretized into triangles. For the purpose of this study, the convergence condition used to terminate the Robin-Hood solver was to ensure that the relative L_∞ norm residual on the solution was less than 10^{-8}. The Krylov based solvers were terminated when the relative L_2 norm on the residual was less than 10^{-8}. For this choice of convergence conditions, the relative accuracy yielded on the final value of the capacitance of the unit sphere by each method was limited solely by the discretization error. The parameters used for the HFFM method were, $p = 8$, $z = 1$, and $d = 4$. Figure 7-13 shows the time taken by each solver to reach the convergence condition as a function of the number of triangles in the mesh [5]. For meshes with more than 10^4 elements, the difference between the quadratic scaling of the Robin Hood method and the linear scaling of the fast multipole method becomes readily apparent.

It is also informative to examine the time to reach convergence as a function of the expansion degree, p, and the zero-mask size, z. To do this, we will use the unit cube geometry discretized into 10^6 elements, which is the same geometry that was used in the accuracy study of figure 7-10. Figure 7-14 shows the wallclock time needed to reach convergence (defined as a relative residual error on the solution of 10^{-8}), as a function of the expansion degree for several values of the zero-mask size.

Since the size of the sparse matrix depends heavily on z, it is not surprising that choosing a larger z causes the time taken to solve the problem to increase substantially. However, while the time as a function of z behaves as expected, what is quite remarkable is that the time required to reach convergence is nearly independent of the expansion degree, p. This is somewhat unexpected, but not altogether surprising once it is noted that this problem was solved using the dipole preconditioner ($p = 1$). Evidently, the majority of the arithmetic work needed

[5]This study was performed on a computer running Debian Linux with an Intel core i7-6700HQ CPU clocked at 2.60GHz, equipped with an NVidia GTX 970M graphics processor and 32GB of RAM.

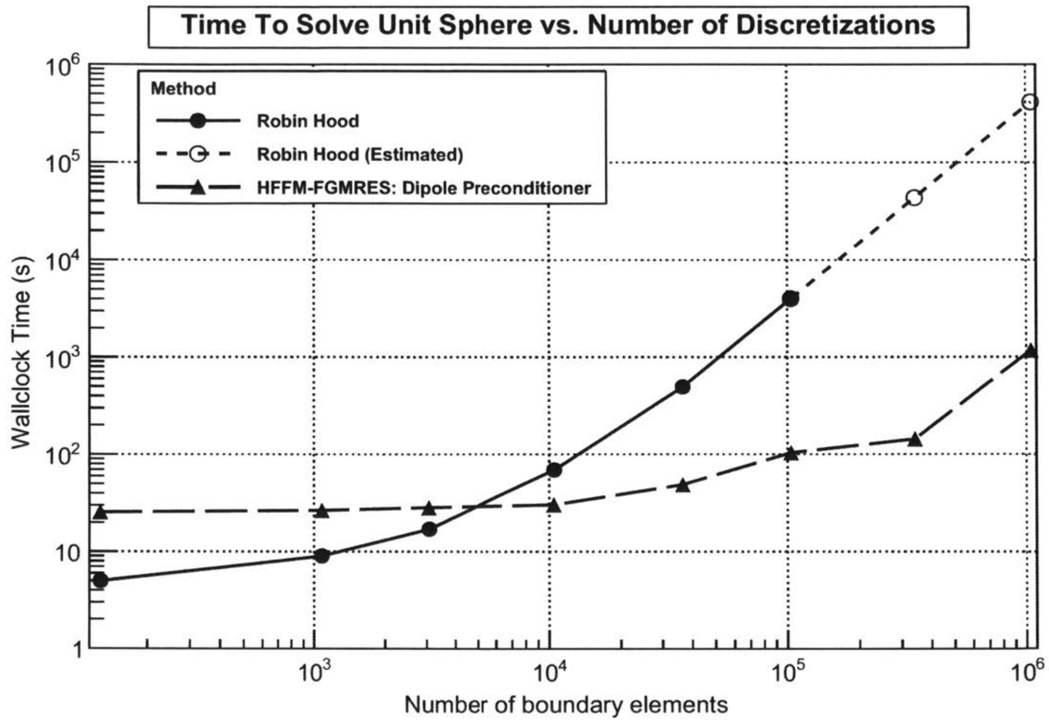


Figure 7-13: Wallclock time to solve the unit sphere capacitance problem as a function of the number of boundary elements for both the Robin Hood method and the dipole preconditioned HFFMM-FGMRES method.

to solve the problem is performed by the preconditioner and the sparse matrix multiplication. However, the preconditioner does not affect the accuracy of the final result, only the choice of p and z used for the full matrix multiplication affects the accuracy. Therefore, whenever memory constraints allow it, it is very advantageous to use a high degree expansion to solve large problems so as to increase the accuracy of the result without much additional cost in time.

7.13.3 Field Map Accuracy and Speed

It is also important to compare the speed and accuracy of the HFFM field map method with the direct integration method for the purpose of field solving at arbitrary locations, which is necessary for particle tracking simulations. To test the accuracy in this use-case, we have considered the unit sphere test problem discretized into approximately 10^4 triangular elements. For this geometry, we have

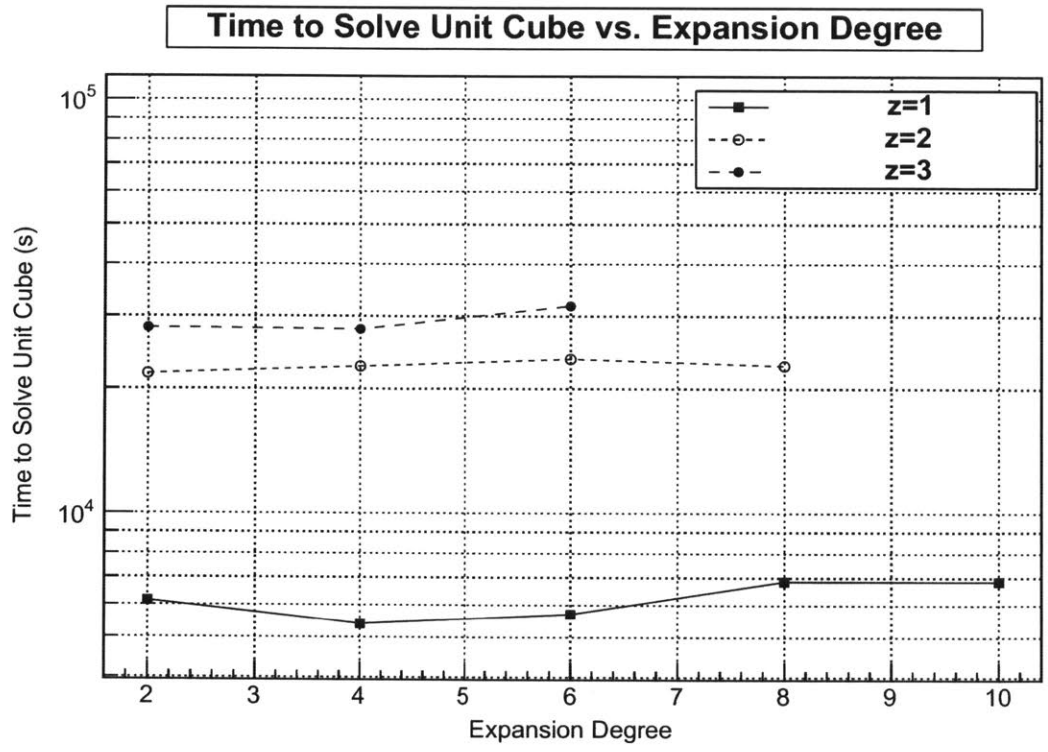


Figure 7-14: Time taken to converge to a relative residual norm error of $\rho = 10^{-8}$ for the unit cube geometry consisting of 10^6 mesh elements as a function of the multipole expansion degree p and zero-mask size z. The solver used was HFFM-FGMRES with a dipole preconditioner.

constructed a field map using $z = 1$ and $d = 4$ and have selected a sample of 10^5 random points uniformly distributed throughout a cubic volume roughly ten times as large as the extent of the BEM mesh. At each point, the electric field and potential were evaluated, using the direct and the HFFM field map (with several choices used for the expansion degree) and compared. The relative difference between the two methods has been calculated and histogrammed. Figure 7-15 shows the distribution of the relative error on the electric potential and figure 7-16 shows the distribution of the L_2 norm error on the electric field as a function of the expansion degree, p, used by the HFFM field map. Note that in these two plots, the histograms have been stacked on top of one another to keep them from overlapping. It is quite apparent that as the expansion degree is increased, the accuracy of the multipole field map improves until it approaches the round off error inherent in the direct calculation. This limit appears to be at a relative accuracy of around 10^{-12} to 10^{-13}

for the electric potential and around 10^{-10} to 10^{-12} for the electric field.

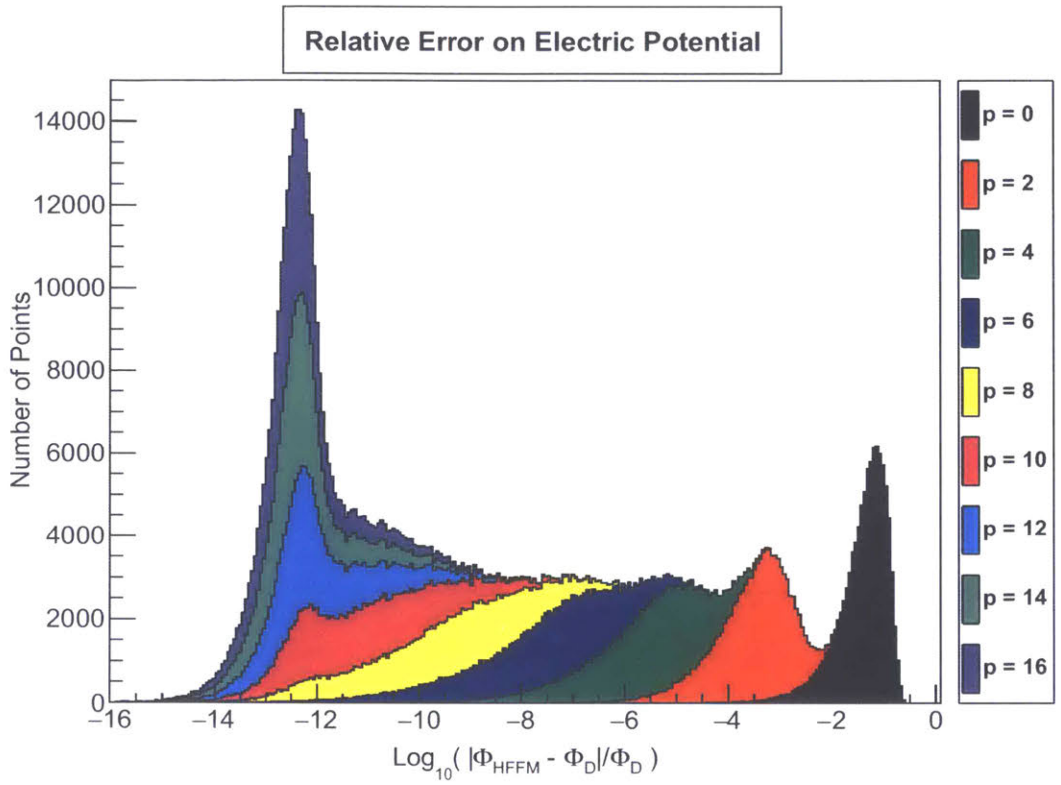


Figure 7-15: Series of stacked histograms of the relative error on the electric potential for various expansion degrees, p.

To justify the use of the HFFM field map over direct integration, we also need to evaluate the speed of each method. For this purpose, the same unit sphere geometry as considered in the previously mentioned accuracy tests was used, but was discretized into a varying number of boundary elements. Figure 7-17 shows the time taken for the potential and field evaluation for each method. It is important to note that the exact time required for a single field/potential evaluation will vary considerably, depending on the BEM geometry at hand and on the parameters selected to generate the field map. However, as seen in figure 7-17, the scaling of the two methods is entirely different, since the HFFMM field map can calculate the field in an amount of time which is roughly independent of the number of BEM elements, while the time taken by the direct method grows in proportion with the number of BEM elements.

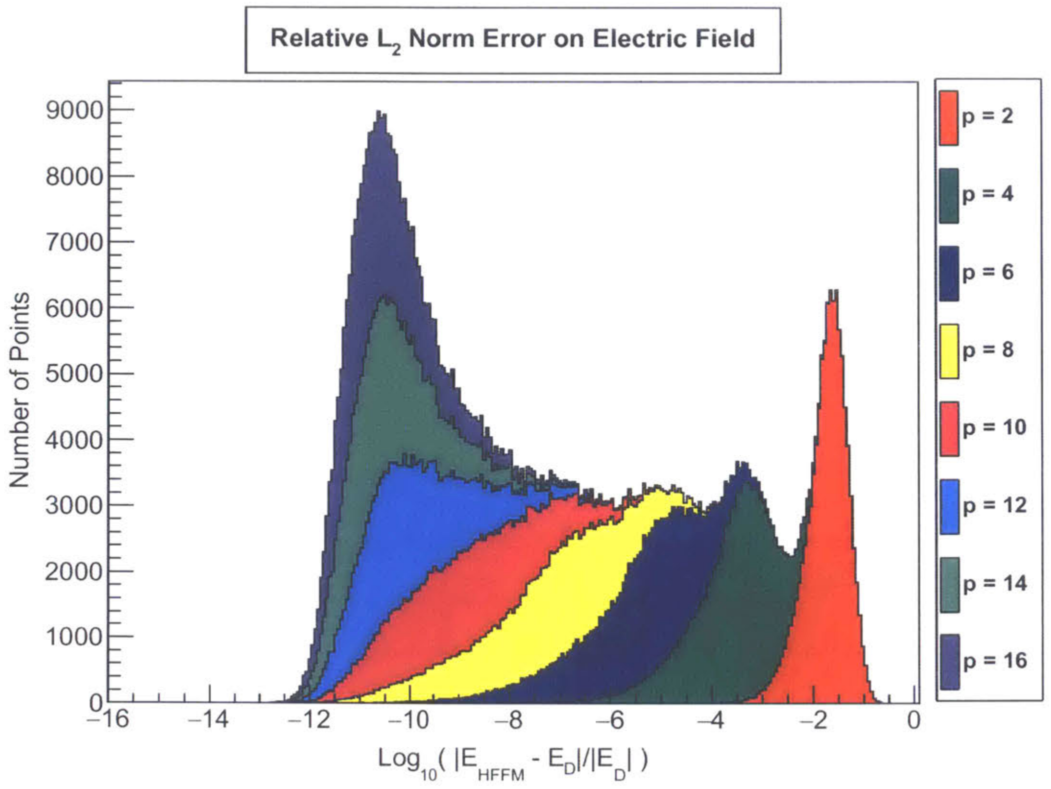


Figure 7-16: Series of stacked histograms of the relative L_2 norm error on the electric field for various expansion degrees, p.

7.13.4 Parallel Efficiency

Since it is not feasible to solve the KATRIN main spectrometer geometry on a single machine, it is useful to estimate the parallel efficiency of the HFFM method when it is distributed across many cores. To test the efficacy of the parallel implementation, a simple geometry with a relatively uniform distribution of mesh elements composed of a grid of tori was used. Figure 7-18 shows the test problem geometry, which was solved using HFFMM-GMRES with a dipole preconditioner until the relative L_2 norm residual on the solution was 10^{-8}. These tests were carried out using MPI+OpenCL on the Babbage cluster at NERSC, which consists of 44 compute nodes, each equipped with two Intel Xeon Phi (Knight' Corner) accelerators. The parameters chosen for the HFFM method are given in table 7.1. Since the workload is distributed by allocating sub-collections of the top-level nodes to each process, it is expected that the number of top-level divisions, d_t, may affect the workload

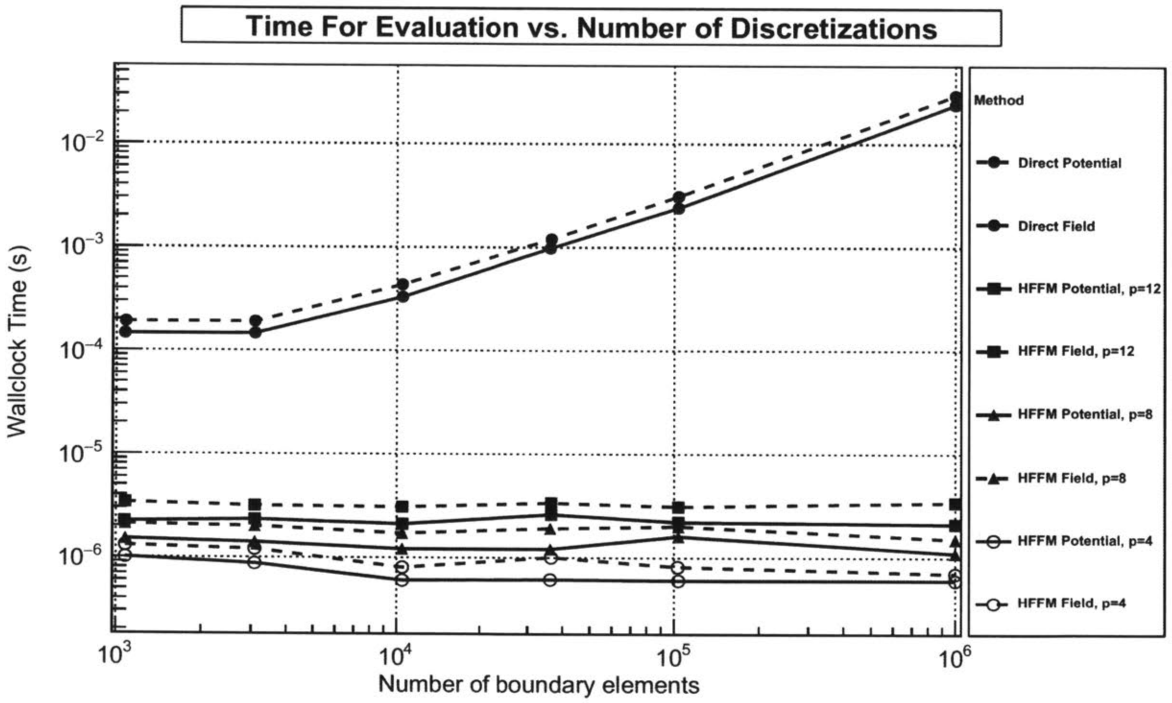


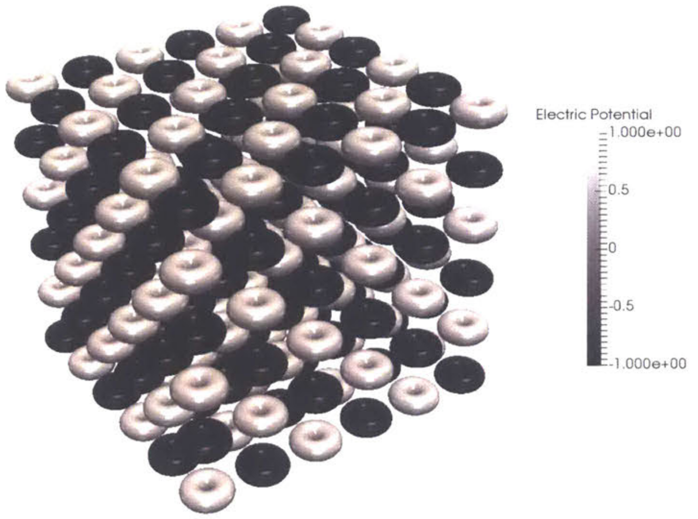
Figure 7-17: Average time taken to evaluate the electric potential and field as a function of the number of boundary elements for various methods. Evaluations of the potential are represented by a solid line, while evaluations of the field are represented by a dashed line.

d_t	d	z	p	η	n_{max}
16, 8, 4	2	1	8	4/3	5

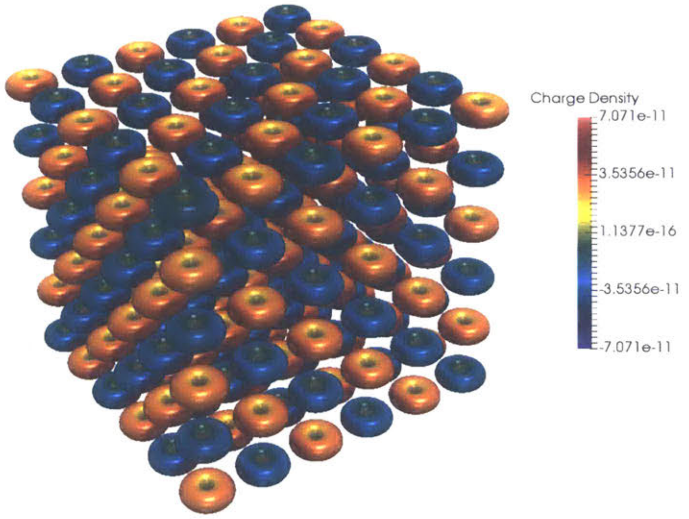
Table 7.1: Parameters used by the HFFM method when solving the test geometry in figure 7-18.

balancing. For this reason, several values of d_t were chosen for this test. Figure 7-19 shows the wallclock time taken to solve the test problem as a function of the number of MPI processes.

When gauging the utility of a parallel version of an algorithm, it is common to measure the factor by which the run time is reduced relative to the single machine implementation. This speed-up factor ($s = T_1/T_N$) is shown in figure 7-20 as function of the binary logarithm of the number of processes. In the case of an ideal implementation of a perfectly parallel problem, it would be expected that a problem which takes time T_1 to be solved by a single processor would take $T_N = T_1/N$ to be solved by N processors. This ideal scenario is rarely, if ever, achieved in



(a) Initial boundary conditions



(b) Resulting charge density

Figure 7-18: The geometry used to test the parallel efficiency of the HFFM method consists of 216 tori held at alternating potentials ($\pm 1V$). The mesh consists of roughly 2×10^6 triangular BEM elements.

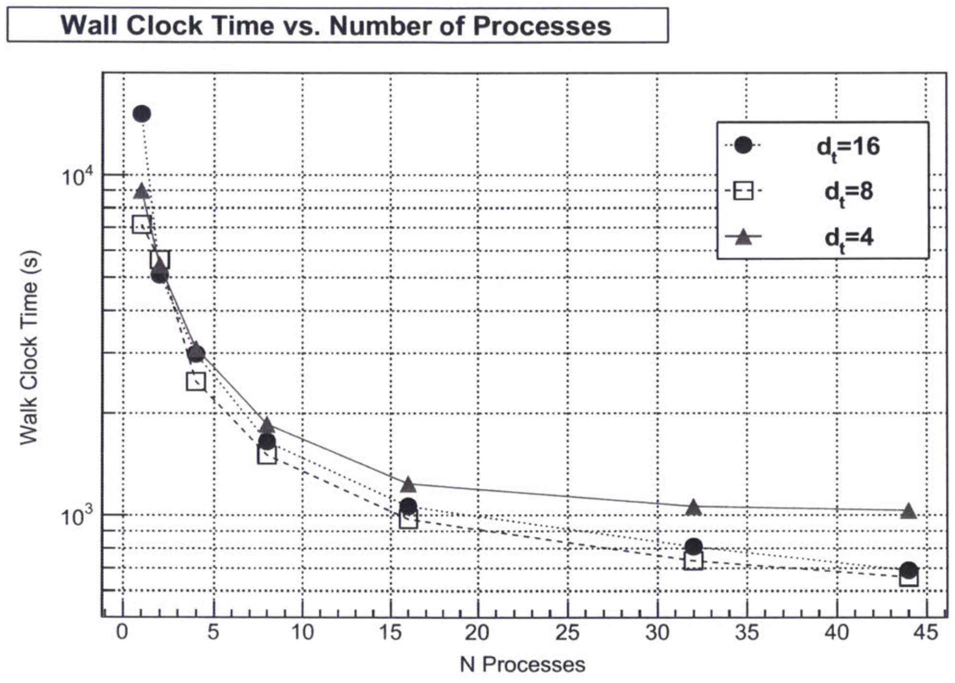


Figure 7-19: Wallclock time to solve the torus test problem with the parallel HFFM method as a function of the number of processes.

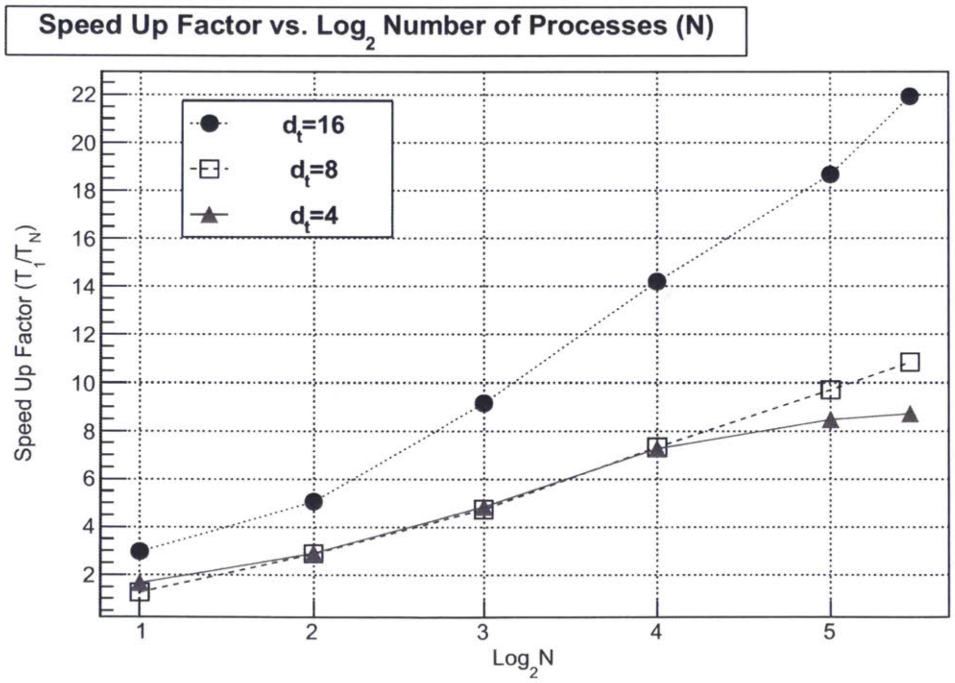


Figure 7-20: Speed up factor of the HFFM method as a function of the binary log of the number of processes (torus test problem).

practice. However, it does motivate a measure of the efficiency, e, of a parallel implementation, which is given as the true speed up factor divided by the ideal speed up factor, $e = s/N$. Figure 7-21 shows the parallel efficiency of the HFFMM algorithm as a function of the binary logarithm of the number of processes. As could be expected of any real program, the parallel efficiency degrades as the number of processes is increased. However, for a moderate number of processes, the parallel efficiency is satisfactory and remains greater than 50% for up to 32 processes when using $d_t = 16$ on our test geometry.

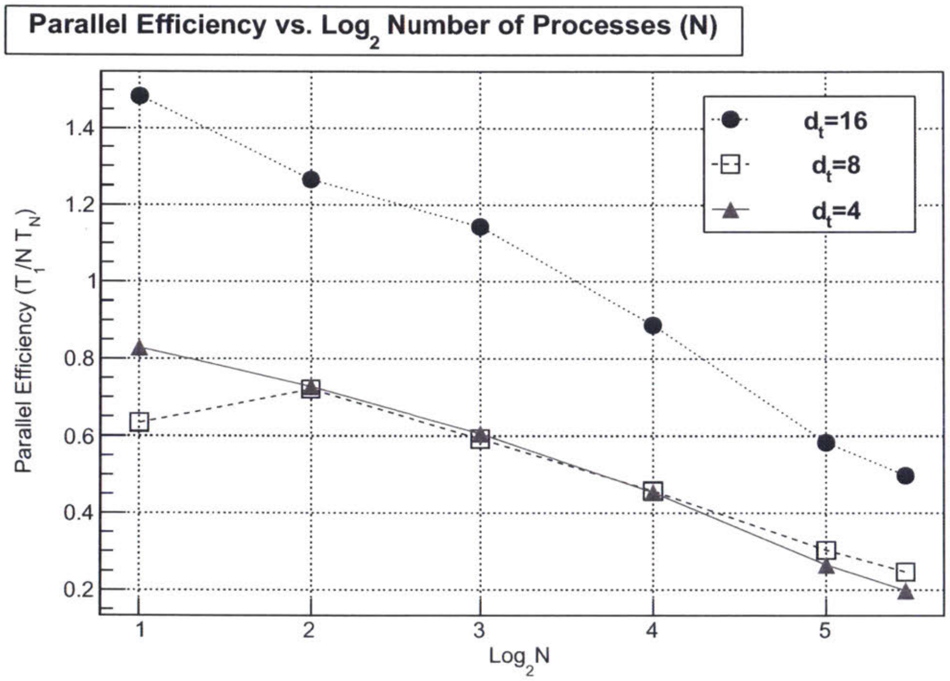


Figure 7-21: Parallel efficiency of the HFFM method as a function of the binary log of the number of processes (torus test problem).

Chapter 8

Measurement of the Spatial Asymmetries in the KATRIN Main Spectrometer's Electrostatic Potential

8.1 Background and Motivation

The electromagnetic design of the KATRIN main spectrometer was intended to produce a nearly axially symmetric analyzing potential, even when in close proximity to the wire electrodes and their support structures. Unfortunately, the azimuthal variation in the potential has been enhanced due to the electrical short circuits between the two layers in the wire arrays [221]. These shorts reduce the effectiveness of the wire arrays in shielding the volume of the spectrometer from the electrostatic influence of the spectrometer wall and the wire array support structures [130]. Since the presence of electrical shorts in the wire arrays was not expected and may not be feasible to repair before tritium data collection is undertaken, it is necessary to measure the size of this potential variation and understand what effect it may have on a neutrino mass measurement.

The purpose of these measurements is two fold. The first is to measure the fine scale azimuthal variation in the analyzing potential at outer radii to ensure it is tolerably small. The second purpose is to validate the three dimensional electric field calculation (specifically the HFFMM) as an accurate method of simulating KATRIN's fields. With a validated field model, we can then confidently use it in a Monte Carlo simulation to estimate what, if any, systematic effect these potential variations may have on the neutrino mass extraction and also determine if pixel-by-pixel knowledge of the transmission function offers any additional resolving power in the eventual neutrino mass analysis.

Ideally, multiple direct measurements of the transmission function itself would provide knowledge of any changes in the analyzing potential over the entire flux tube. However, it is far too time consuming to measure the transmission function over such a large parameter space, which requires both the electron beam position and momentum to be adjusted, as well as the spectrometer's magnetic field and electrode potentials. Therefore, it is not feasible to search for fine scale azimuthal variations in the analyzing potential by direct measurement of the transmission function for anything but a small subset of the flux tube. On the other hand, it was realized during the wire integrity check performed during the first phase of commissioning [125] that the time-of-flight (ToF) spectra can also serve as a sensitive tool for the measurement of the size of the potential variation.

During the SDS1-M12 measurement[1], the transmission function was first sampled at a particular point (north, south, east, west), then the e-gun's energy was set to a value where approximately 50% of the electrons emitted were in transmission while the e-gun's position was scanned azimuthally (with a fixed radial displacement) around the spectrometer. Originally, it was expected that the variation of the rate as a function of the azimuthal angle could be used to map the shift in analyzing potential, since when the beam energy corresponds to a transmission probability of roughly 50%, the change in the rate of electrons at the FPD is roughly a linear function of the analyzing potential that they encounter. Unfortunately, because the shift in the analyzing potential was fairly large, being on the order of the width of the transmission function, there were regions where the rate of electrons reaching

[1]SDS1 was the first commissioning phase of the KATRIN's Spectrometer-Detector-System.

the detector was either maximal or zero, which yields no information about the analyzing potential in that area. Because of this, it was determined that a more satisfactory probe of the shift in potential was the most-probable ToF. Despite being an indirect probe (requiring a full simulation to extract the variation in electric potential from the measured ToF), the ToF is a better method to probe the spectrometer since it is still sensitive to the potential, even when the electrons' energy is above full transmission. This is helpful since the electron gun can be used at a higher energy which is in the region of maximal transmission (where the rate alone would otherwise provide no information about the analyzing potential), making it easier to avoid regions where the rate drops to zero.

Unfortunately, the SDS1 measurement was also complicated by flux tube blockage caused by the spectrometer-to-detector flapper valve which malfunctioned and could not be removed entirely from the beam line. An additional complication to this measurement was posed by the misalignment of detector region magnets which caused a portion of the flux tube to intersect the vacuum chamber wall. The combination of these two effects made it impossible to perform this measurement at nominal magnetic field settings (3-9 G in the analyzing plane) and instead, a low field of 1.5 G had to be used. Thus, while useful for probing the wire integrity, this measurement was not altogether very useful for determining the properties of the main spectrometer in its normal operating range. The misalignment of the magnetic field was also a problem as it introduced a large asymmetry in the field. This made it difficult to perform a purely azimuthal scan of the electron beam without large accompanying changes in the radial position. Additionally, since the wire integrity measurements were only taken at one radial UHV manipulator position and magnetic field setting, it was not possible to determine the radial depth of the azimuthal variations in the analyzing potential. Furthermore, the electron gun manipulator was not instrumented with a read-out of its position data, so the azimuthal position of the electron beam could only be roughly estimated through knowledge of beam-pixel crossings. Hence, it was necessary to revisit this measurement during the second phase of commissioning.

8.2 Experimental Configuration

The basis of examining the main spectrometer and its functionality as a MAC-E filter under various magnetic and electrostatic settings is to probe it with an electron source with consistent and well understood properties. To this end, the commissioning of the main spectrometer was performed through the use of an electron gun provided by the Münster group [220]. This electron gun provides a means of probing the response of the main spectrometer as a function of the incident electron's positions and momentum. The electron gun was placed at the pre-spectrometer entrance of the main spectrometer and was mounted on a UHV manipulator (see figure 8-1) in order to select the electron beam's flux tube position. The beam is generated from a gold-plated optical fiber illuminated with light produced from a UV laser. This produces a nearly mono-energetic photo-emission of single electrons at a rate of several kilohertz. The energy distribution of the produced electrons is roughly Gaussian, with an approximate width of $\sigma = 0.2$ eV [127]. The photo-electrons are then accelerated within the gap of a parallel plate capacitor which is oriented relative to the local magnetic field in such a way as to provide some selectivity over the electron's pitch angle (defined as the angle between the electron's momentum vector and the magnetic field). Unfortunately, the angular selectivity of the electron gun is rather poor at the low voltages that were necessary during this measurement. However, this feature of the electron gun was not critical for these measurements.

Since azimuthal variations in the analyzing potential are expected to be relatively small, it was necessary to ensure that the electron beam passed very close to the wire arrays. For this reason, large electron gun manipulator angles were used. In addition, low magnetic field settings were necessary to expand the flux tube so that its outer limits approach the surface of the spectrometer walls. The azimuthal ToF scans were done at two separate B-field settings; a more sensitive 2 Gauss setting, intended to move the flux tube closer to the wires and maximize the visibility of the ToF variations, and the nominal 3.8 Gauss setting intended for use in an eventual

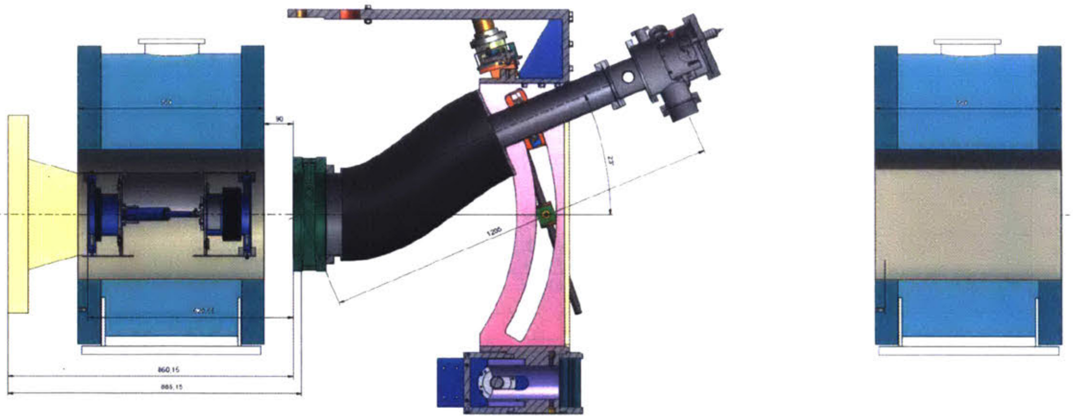


Figure 8-1: Top down view of the Münster group electron gun. The photo-cathode is mounted near the end of the section of gray tube attached to the UHV manipulator. The green cylinders are the pre-spectrometer magnets. The main spectrometer vacuum flange is the yellow trapezoidal object on the left. Image taken from [220].

tritium measurement mode. Fine tuning the magnetic field configuration of the main spectrometer is accomplished through adjustments to the currents in the air coil system (LFCS). These were set so that the magnetic field strength at the mid-plane of the main spectrometer was approximately 2 or 3.8 Gauss in magnitude. A list of the currents used to configure the LFCS coils for each setting is given in table 8.1. The Earth Magnetic Field Compensation System (EMCS) coils were set to the nominal currents of 8.7 A and 50 A for the horizontal and vertical coils respectively. This roughly canceled the effect of the Earth's magnetic field which would otherwise severely distort the flux tube.

To further accentuate the azimuthal variation in the analyzing potential, a greater than standard potential difference (300V instead of the nominal 200V) was placed between the wire arrays and the spectrometer vessel. Ideally, all of the measurements to examine the ToF and transmission function would have been taken at the full nominal potential of 18.6kV. Unfortunately, due to the development of a Penning discharge in the electron gun chamber at high voltages, this was not possible [126]. The occurrence of a Penning discharge is extremely undesirable during operation of the electron gun, because it can cause sputtering damage to the photosensitive

LFCS Coil Number	2 Gauss Current (A)	3.8 Gauss Current (A)
1	5.24	20.92
2	10.48	25.49
3	7.25	20.01
4	18.49	28.23
5	23.94	38.52
6	19.83	27.25
7	7.95	34.16
8	19.30	50.57
9	5.72	10.02
10	29.99	44.36
11	27.29	37.00
12	8.28	20.81
13	39.61	43.14
14 & 15	63.07	50.33

Table 8.1: LFCS coil current summary.

gold layer which is only 40nm thick [124]. Sputtering damage causes changes to the photo-cathode work function and therefore the electron energy distribution. In extreme cases it can disable the gun entirely by removing the gold layer. Therefore, the spectrometer potential was limited to less than 1kV for the majority of the measurements. While not identical to the tritium measurement configuration, these low voltages settings are still comparable because the azimuthal variations we are looking for are primarily a function of the difference between the vessel and wire array potentials, which is essentially the same for both configurations. Table 8.2 shows the different electrostatic configurations used during these measurements. The more positive potential on the steep cone electrodes is necessary to prevent early retardation, which is caused when the incident particles are electrostatically reflected because they have not sufficiently transformed their transverse momentum into longitudinal momentum [110].

Vessel Potential (V)	Wire Array Offset (V)	Steep Cone Offset (V)
-500	-300	+100

Table 8.2: Main spectrometer electrode configuration.

8.3 Measurement Procedure

The collected run data consists of the following two types: time-of-flight (ToF) data and transmission function (TF) measurements. The azimuthal ToF scans were started by first measuring a transmission function to locate the transmission edge. Then the e-gun energy was fixed to a small surplus ($\sim$ 1.25 eV) above the transmission edge, while the UHV manipulator was moved around azimuthally to watch for the changes in the time-of-flight. During these scans, four transmission function measurements were taken in $\sim 90^\circ$ degree intervals.

A transmission function measurement is taken along a particular field line by first fixing the position of the electron gun, and then varying its energy. The location of the transmission edge[2] is usually not known precisely at the start of the run, so typically we perform a fast binary search within $\pm$10eV of the expected energy in order to locate it. Then, once the edge has been found, we scan the electron's gun energy from approximately 1 eV below to 1 eV above the transmission edge in roughly 0.15 eV increments and record the electron rate at the FPD for each energy. Plotting the normalized electron rate (with respect to the maximum rate) as a function of energy yields the transmission function. This transmission function represents the probability that an electron randomly selected from the source ensemble has of being passed through the main spectrometer MAC-E filter as a function of energy. Since it depends on the energy and momentum distribution of the source, a transmission function measurement therefore requires a good understanding of the electron source in order to extract the main spectrometer properties directly from the normalized rate.

One way a transmission function measurement can be augmented is by recording the ToF as a function of energy. This was done in order to help construct a correspondence between the shift in the analyzing potential and the shift in the ToF. This correspondence will be used to obtain a map of the potential variation as a function of the azimuthal angle over a full 360° scan, at a much finer resolution (1°)

[2]We will define as the transmission edge as the energy where the rate of electrons arriving at the FPD from the electron gun is 50% of the maximum rate.

than would be possible by performing many very time consuming transmission function measurements.

Table 8.3 contains a brief description of the run data which is of immediate interest for our purposes.

Collection	Run Numbers	B-field	Surplus energy	Description
(A)	23840-23851	2 Gauss	1.25eV	4 TF + ToF 360° scan, $\Delta\phi = 1^\circ$
(B)	23864-23873	3.8 Gauss	1.25eV	4 TF + ToF 360° scan, $\Delta\phi = 1^\circ$
(C)	23936-23945	3.8 Gauss	1.25eV	4 TF + ToF 360° scan, $\Delta\phi = 1^\circ$, radius varied

Table 8.3: Data summary.

8.4 Data Pre-processing

All the run data is passed through a pre-processing step. This consists of fitting the energy spectrum with a Gaussian centered at the expected energy (the sum of the spectrometer and post-acceleration electrode potentials) and cutting events more than 3σ away from the peak. This provides a rudimentary background rejection which is sufficient for the high-rate e-gun data, since the background rate is less than 1Hz and the E-gun rate is $\sim$ 4.5 kHz. Figure 8-2 shows the typical energy spectrum and marks the cut region. The cut portion of the spectrum is composed primarily of events with an incident electron multiplicity greater than one and comprises less than 1.8% of the total data collection. Additionally, all pixels that do not exhibit a rate in excess of 1kHz at any point during a run are also cut, with the exception of those pixels which are neighbors of a high-rate pixel. This is done to eliminate the majority of the background that is not relevant to the ToF/TF analysis, while keeping pixels which might have had their edge caught by the electron beam.

All of the relevant slow control data during the run has been linearly interpolated for times between measurement points, in order to provide smooth information during sub-runs. The slow control data update rate is typically less frequent than the sub-run time length (about $\sim$ 15 seconds apart vs. 5-10 seconds for each sub-run) so interpolation is necessary in order to estimate the values of slowly changing parameters. The slow control parameters of interest are primarily the

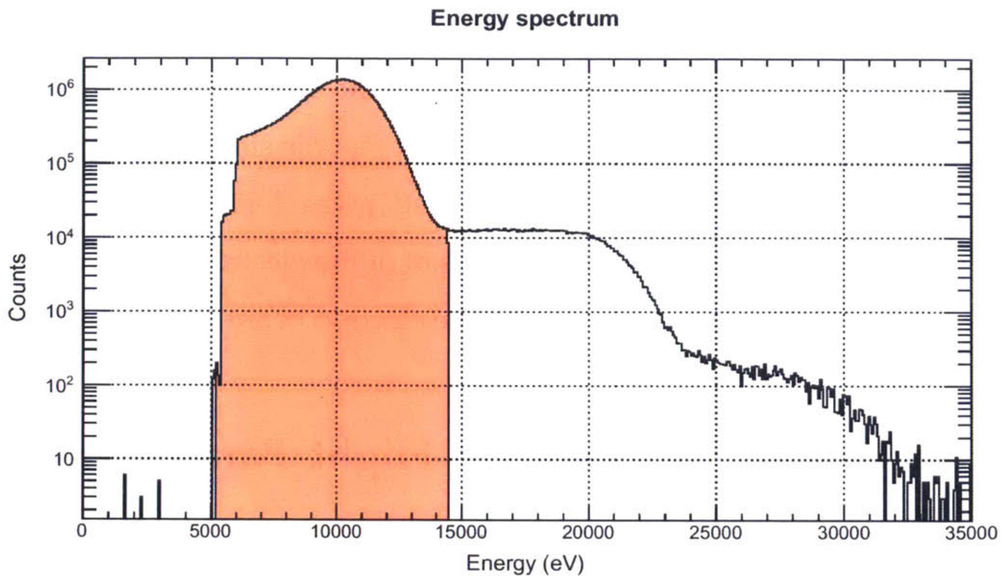


Figure 8-2: The portion of the energy spectrum selected for use in the ToF and transmission function analysis. The shaded region represents the selection of events which passed the energy cuts. This spectrum is from run #23840.

electron gun surplus energy and the position (azimuthal and polar angle) of the UHV manipulator. Since it is important that the polar angle and surplus energy of the electron gun do not change substantially during the course of a run, as a check on data quality, these quantities are plotted as a function of the electron gun's azimuthal angle. Figures 8-3, 8-5, and 8-7 display the deviations of the manipulator's polar angle and the potential difference between the electron gun backplate and inner electrode during the course of run collections (A), (B), and (C) respectively. The potential difference, $\Delta U = (U_G - U_{IE})$, between the electron gun and the spectrometer's inner electrode potential serves as a proxy for the surplus energy, while the polar angle serves as a proxy for the beam's radial coordinate in the flux tube. From these plots, we see there is a small drift in the surplus energy and the polar angle of the electron gun during the first two run collections. For collections (A) and (B), the polar angle drifts less than 0.3%, while the change in the energy is approximately 3.2% of the 1.25eV surplus, which is an acceptable amount of drift. However, for run collection (C), there was an obvious malfunction in the electron gun control during the last run which causes the surplus energy to jump approximately -0.4 eV and the manipulator's polar angle by nearly 0.5°. On account of this malfunction, we have discarded the data for azimuthal angles above $\sim 70°$ from this collection. It should be noted that the sinusoidal trend in the manipulator's polar angle during run collection (C) was deliberately applied. It was intended to compensate for the misalignment of the electron gun system and ensure less radial variation in the position of the electron beam in the flux tube.

8.5 Extraction of the Time of Flight Parameters and Comparison to Monte Carlo

For the run collections in (A), (B), and (C), each 90 degree azimuthal scan has been stitched together and the ToF distribution has been extracted as a function of the electron gun manipulator azimuthal angle. Figures 8-4, 8-6, and 8-8 show the ToF distribution trend for run collections (A), (B) and (C) respectively. Note that the reference pulse arrives at the DAQ system 1.27μs later than the moment an electron is ejected from the gun [128] but that the figures show the raw ToF spectrum which has not had any corrections due to the signal path latency applied. The large 30° gap in each trend centered around 175° is due to several pixels (primarily #106) being disabled because of a bad pre-amp card. The other small gaps are either due to collisions with the main spectrometer walls or a shift in the analyzing potential greater than the surplus energy of the e-gun.

In order to reduce the entire ToF spectrum at each manipulator angle into a single representative number, we have chosen to use the most-probable ToF.

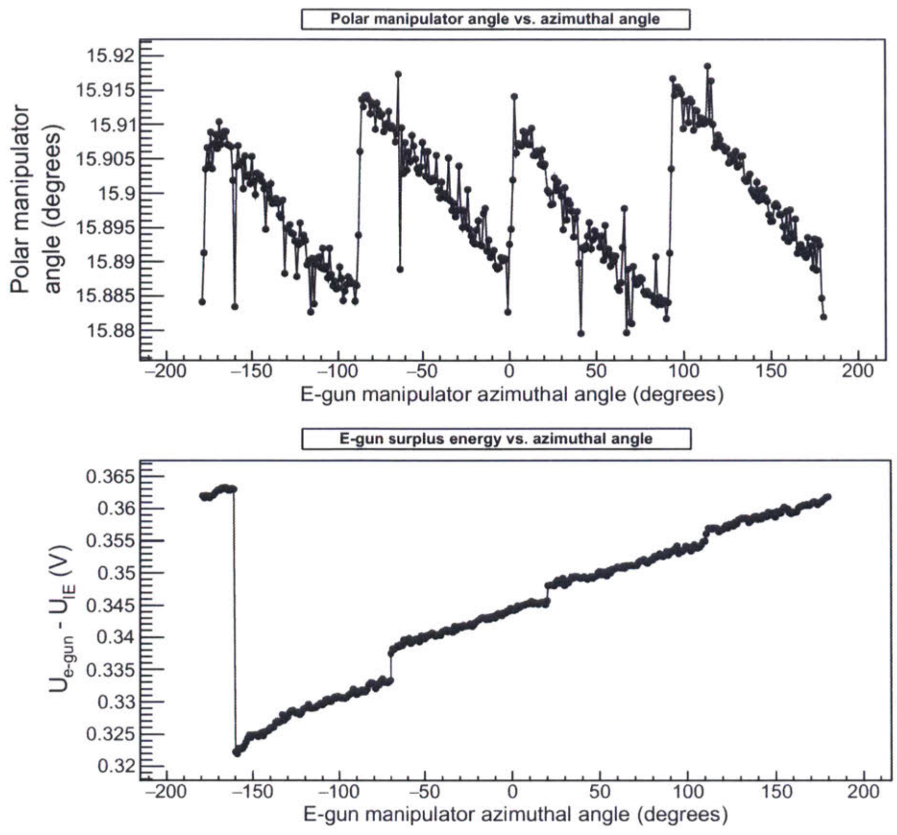


Figure 8-3: Run collection (A).

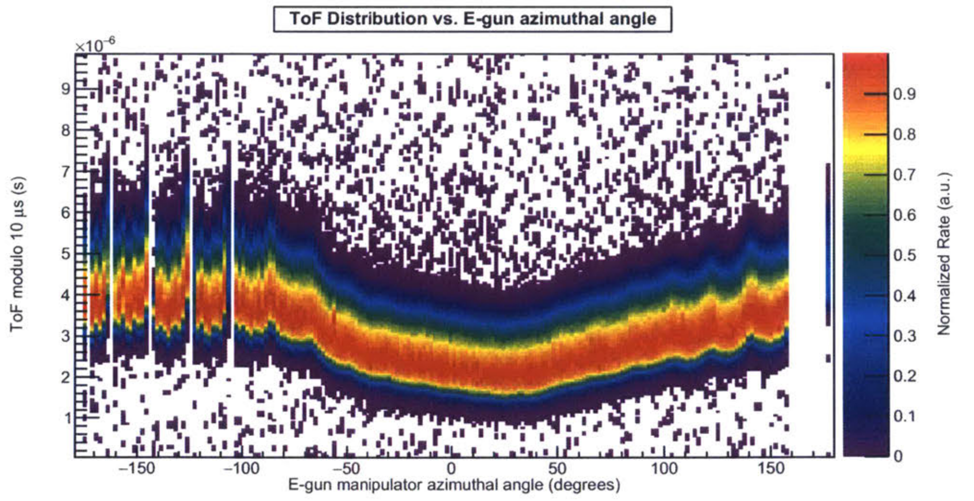
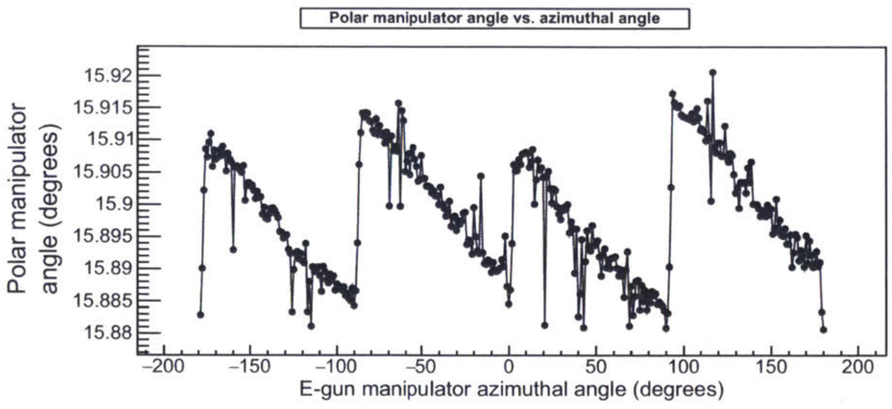


Figure 8-4: The raw ToF distribution as a function of the e-gun's manipulator angle for run collection (A).



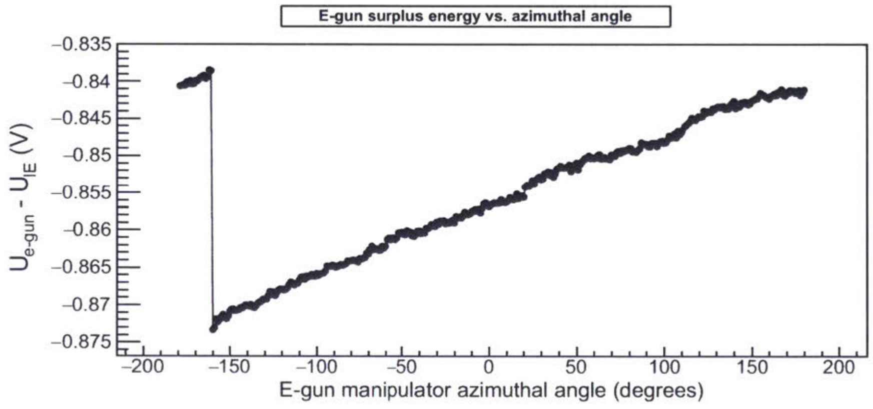


Figure 8-5: Run collection (B).

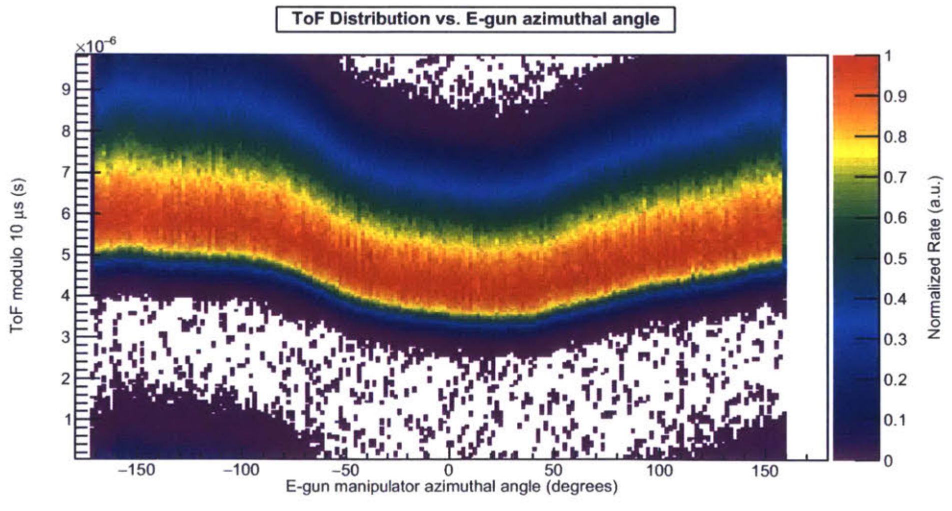


Figure 8-6: The raw ToF distribution as a function of the e-gun's manipulator angle for run collection (B).

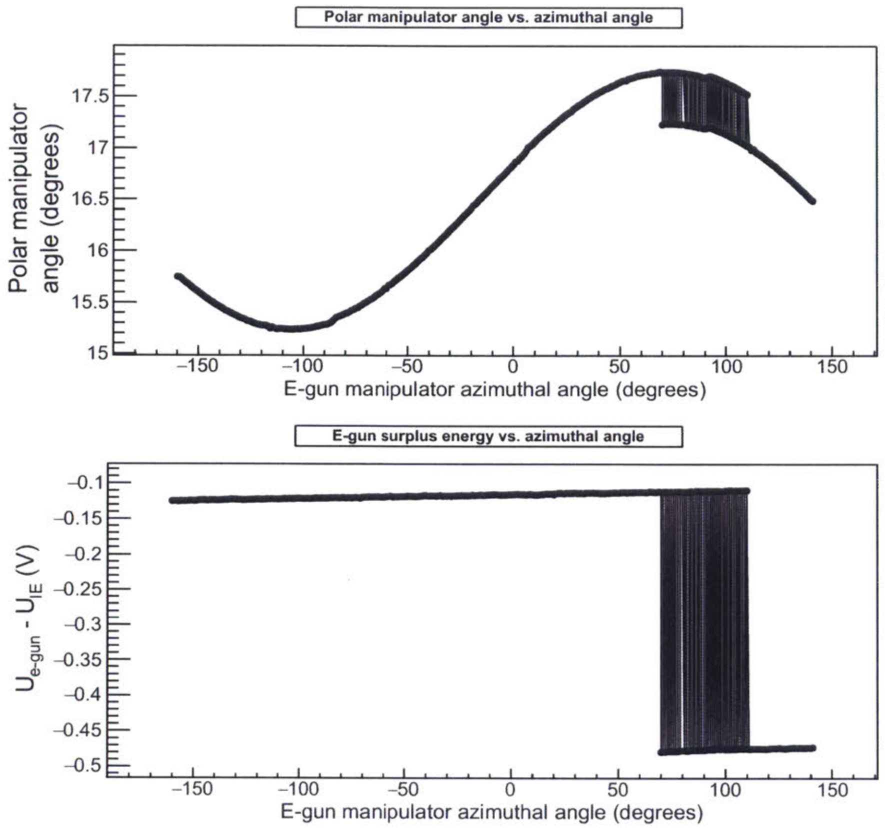


Figure 8-7: Run collection (C).

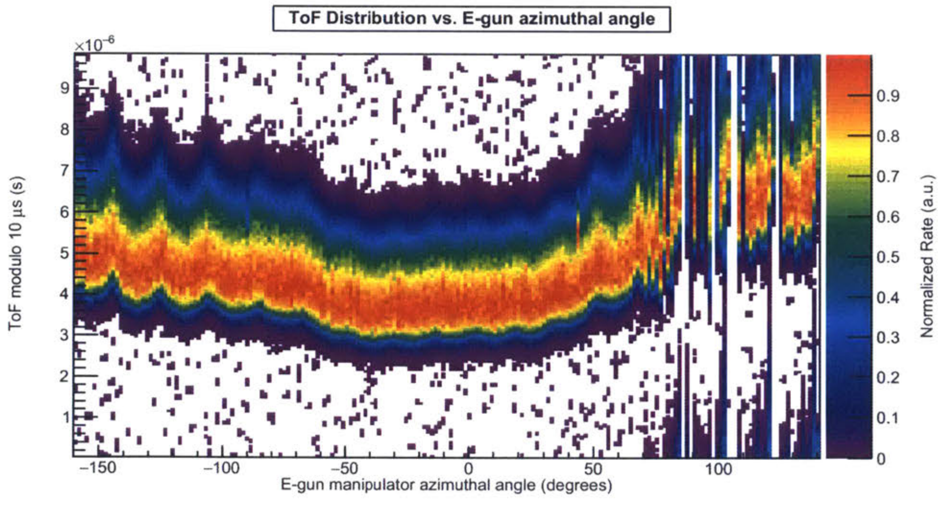


Figure 8-8: The raw ToF distribution as a function of the e-gun's manipulator angle for run collection (C).

The most-probable ToF is defined as the location of the peak of the ToF spectrum as determined by a fit with an asymmetric (two-sided) Gaussian function [137], given by:

$$f(x) = \frac{2}{(r+1)\sqrt{2\pi\sigma^2}} \begin{cases} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) : \text{if } x > \mu \\ \exp(-\frac{(x-\mu)^2}{2r^2\sigma^2}) : \text{otherwise} \end{cases} . \tag{8.1}$$

An asymmetric Gaussian was found to be an empirically acceptable model, since at low surplus energies, the ToF distribution can become quite distorted with respect to the original electron energy distribution. Since the time-of-flight is a periodic function of the laser pulses ($\tau = 10\mu$s), it is necessary to wrap the fit function of equation 8.1. This is because slow electrons from the previous pulse can arrive within the time window of the next pulse. Constructing the wrapped version of the fit function is simple, since for any given PDF $f(x)$, with $x \in (-\infty, \infty)$, we can define its wrapped counterpart $g(y)$, with $y \in [0, \tau)$ as:

$$g(y) = f(y) + \sum_{k=1}^{\infty} f(y + k\tau) + f(y - k\tau) . \tag{8.2}$$

For practical purposes, it is necessary to terminate this sum at some reasonable upper limit of k. Using a value of $k = 5$ has sufficed for our purposes since the ToF distribution width is relatively narrow compared to the laser pulser period, τ, and any contributions outside of this window are negligible. The fit is then performed for all events sorted into 1° azimuthal angular bins. An example of such a fit is shown in figure 8-9. The result from these fits is the most-probable ToF trend for run collections (A) and (B), show in figure 8-10. Angular bins which did not register a rate greater than 200Hz typically do not contain enough events in order to perform a valid fit and were cut from the analysis. The error bars on each of most-probable ToF data points are a uniform 100ns, as this is the FWHM of the timing resolution of the DAQ system. The time resolution of the DAQ system depends on the event energy and the shaping length of the trapezoidal filter. The time resolution has been estimated from [8], using an event energy near the peak (11keV) and the shaping length used during these run collections (1.6μs).

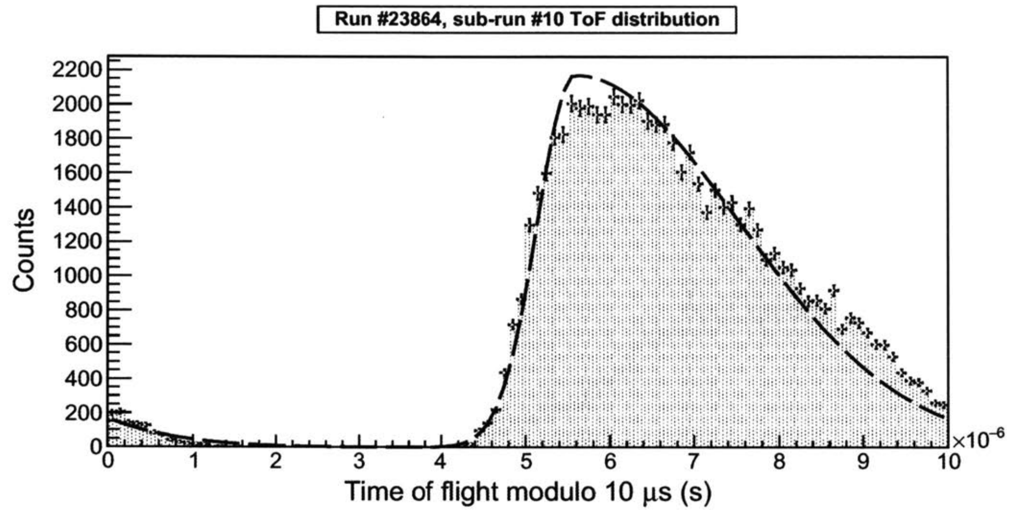
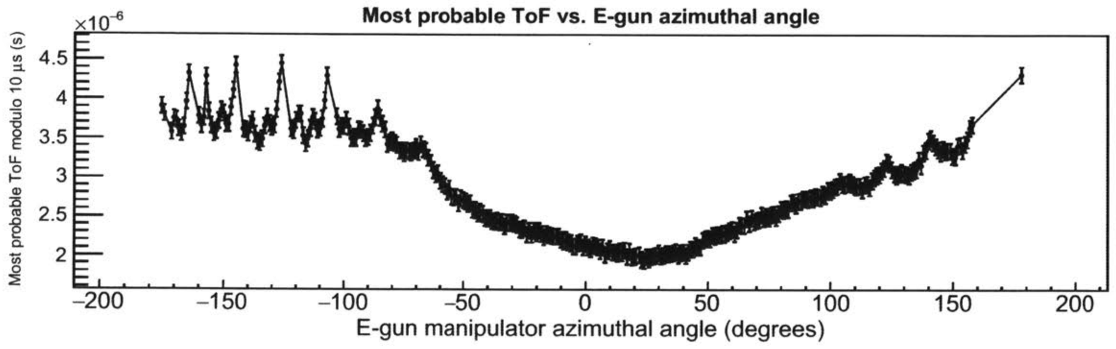
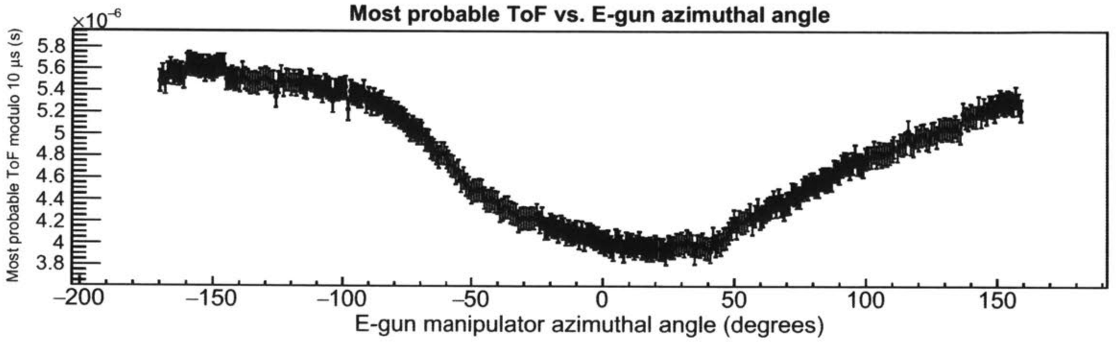


Figure 8-9: Example of the wrapped asymmetric gaussian fit to the ToF distribution. Applied to run #23864, sub-run #10, this corresponds to an approximately 1° slice around $\phi = -150°$ from the ToF distribution trend shown in figure 8-6.

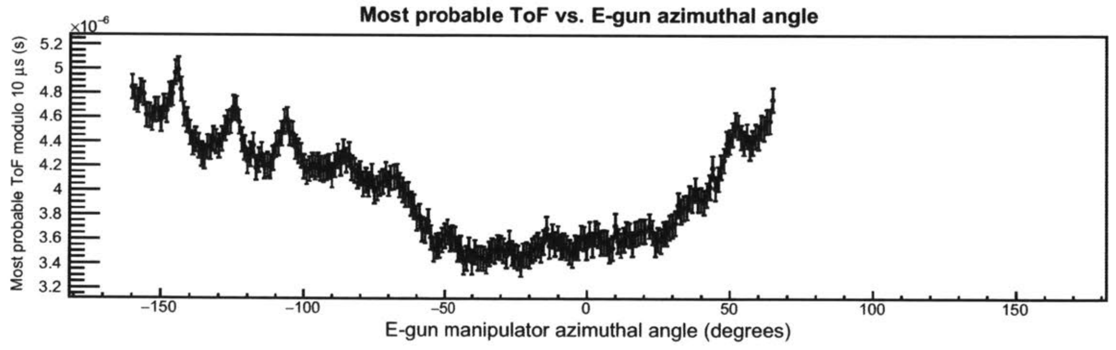
With the most-probable ToF trend as a function of azimuthal angle extracted, we are now in a position to validate the three dimensional electric field calculations. This requires us to make a quantitative comparison of the data we have collected with a global simulation of the spectrometer-detector system. For this purpose, the ToF trend as a function of the azimuthal angle provides a useful test, because unlike the transmission function, it is relatively insensitive to the exact parameters describing the electron gun (particularly the angular distribution of the emitted electrons which cannot be measured with great certainty). The variation in the ToF also has the advantage that it is a relative, rather than absolute, measurement. Furthermore, through the use of the transmission function runs, which allow us to calibrate the ToF as a function of surplus energy, we can convert the most-probable ToF as a function of azimuthal angle into a measurement of the analyzing potential. This enables us to make a direct comparison between the calculated and measured potential values. However, before proceeding further with the data analysis and its comparison to the Monte Carlo simulation, we need to describe the simulation procedure in more detail.



(a) Run Collection (A).



(b) Run Collection (B).



(c) Run Collection (C).

Figure 8-10: Extracted most-probable ToF as a function of the electron gun manipulator azimuthal angle for run collections (A), (B), and (C). Data for run collection (C) above $\phi = 70^\circ$ is corrupt due to a malfunction in the electron gun control and has been excluded. These ToF values have not been corrected for timing signal latency.

8.6 Simulation of a Realistic Three Dimensional Main Spectrometer Model

The first task in simulating the main spectrometer system is the calculation of the electric and magnetic fields. The calculation of the magnetic field from the known current sources is fairly straightforward. However, calculating the electric fields of the main spectrometer requires us to have a detailed model of its geometry. This geometric model was painstakingly assembled by T.J. Corona [53] using the original computer-aided-design (CAD) files and measurements of the vessel hull deformation, and generates roughly 5 million elements once it has been discretized into rectangular and triangular patches. In addition, the main spectrometer model also has 44 parameters describing the voltage settings for each electrically independent element. Figure 8-11 shows each independent module of the main spectrometer and the short circuits between them.

Until recently, the time required to solve for the charge densities of the main spectrometer model precluded any attempt to solve the complete system with all 44 degrees of freedom. Instead, several standard configurations were chosen and solved directly. However, this is not ideal, since typically even while the voltage set points may match the standard configurations exactly, they do not necessarily match the voltage values read back from the hardware. This issue required anyone who wanted to simulate a particular data set to solve the charge densities of each run configuration separately. Unfortunately, this requires an untenable amount of computation time, making this very difficult for anything but small data sets.

However, the use of the parallel-HFFMM algorithm has reduced the time required to solve a single configuration of the main spectrometer system to roughly 3.5 hours [3]. This has made it possible to solve all 44 independent configurations and create a superposition library. To create this library, we set each module to unit potential while holding all the others fixed at zero and solve the resulting linear system until the relative residual norm is less than 10^{-6}. From this library, we

[3]This was done using MPI+OpenCL on 44 nodes of the NERSC Babbage cluster.

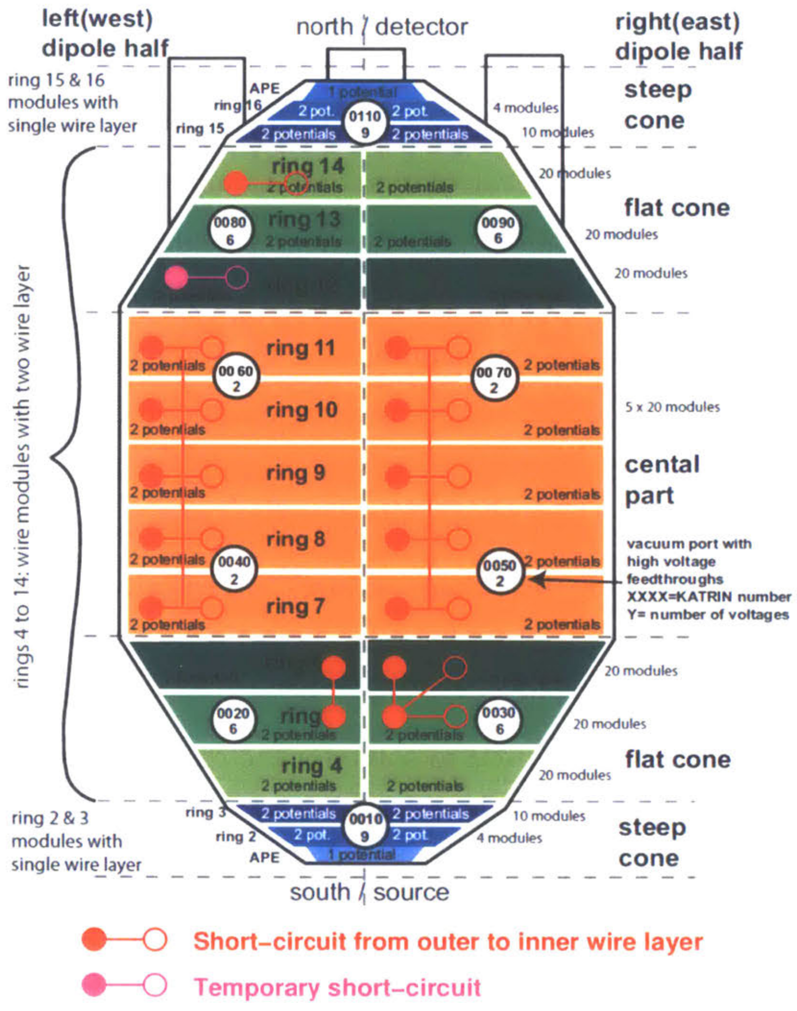


Figure 8-11: Main spectrometer electrode modules. Image taken from [130].

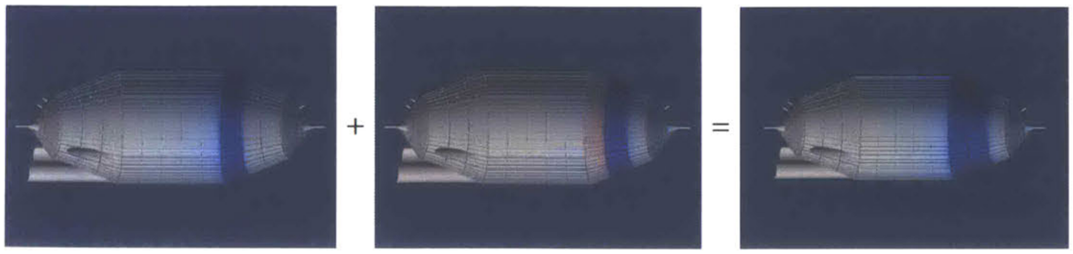


Figure 8-12: Graphical representation of generating a new solution from the scaled superposition of two library solutions.

can then generate the solution of any configuration by simply summing the scaled charge densities. Figure 8-12 demonstrates a simple graphical representation of this procedure. Using this superposition method, we can produce the charge densities of any configuration of the main spectrometer in the time it takes to read the 22GB library from disk.

Once the charge densities are known, we then proceed to construct a fast multipole field map of the spectrometer in order to enable rapid particle tracking. This is also done with the HFFM method using the parameters given in table 8.4 and the aggressive tree division strategy. This choice of parameters produces a field

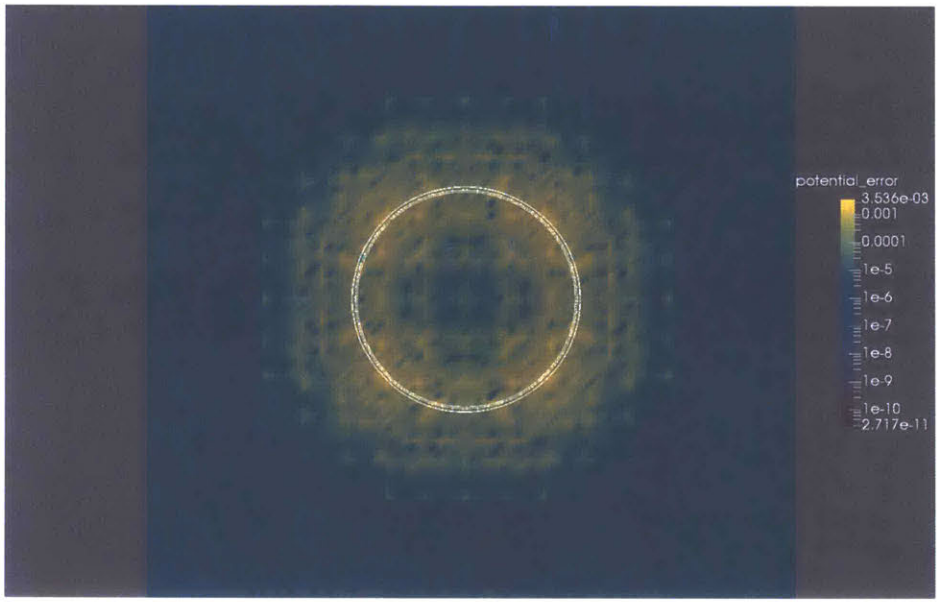
d_t	d	z	p	η	$n_{\max}$
16	2	1	11	4/3	8

Table 8.4: Parameters used to generate the fast multipole field map of the main spectrometer.

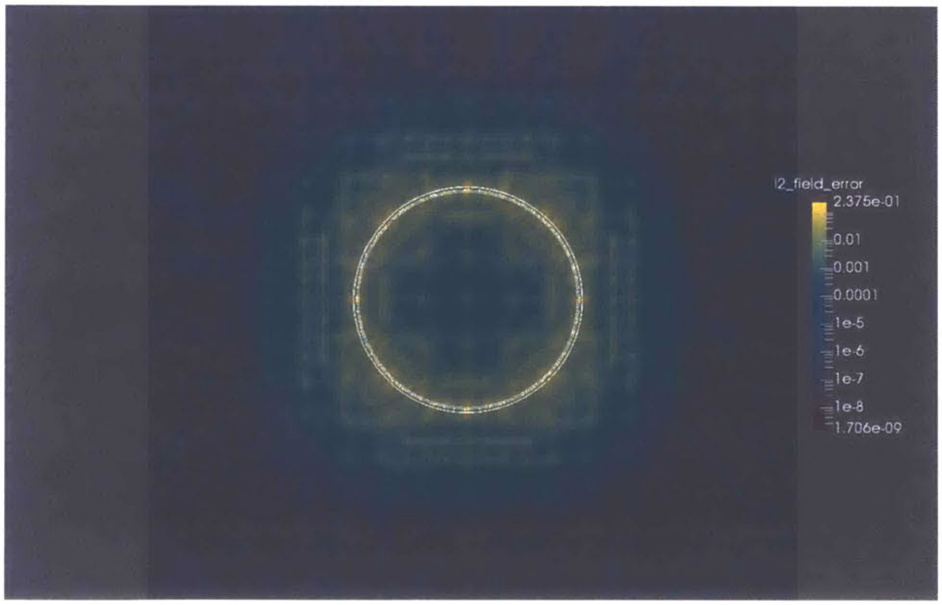
map that requires approximately $13 - 16\mu s$ for each potential/field evaluation, whereas the direct method with GPU acceleration requires roughly 0.1s for each evaluation[4]. This corresponds to a speed up factor of roughly $\sim 6300 - 7500$ over the (GPU-accelerated) direct method. The absolute error between the direct and multipole methods on the potential and field calculations are shown as a function of position over the central plane of the main spectrometer in figure 8-13. In the central region, the error on the potential is generally much less than 3 mV, and the error on field is less than 0.2 V/m.

These errors should not significantly affect the behavior of the simulated particles, but in order to confirm that the field and potential errors are of an acceptable size, we can examine the amount of energy conservation violation as particles traverse the spectrometer model. Figure 8-14 shows a histogram of the energy violation during the course of each track as it passes through the main spectrometer. The mean violation is roughly 0.002 eV, which represents a relative error with respect to the particle's kinetic energy of approximately $\sim 10^{-3}$ to 10^{-6} over the course of its

[4]The speed of the direct and fast multipole methods were evaluated on a computer with 32GB of ram equipped with an Intel core i7-6700HQ processor and a NVidia GTX 970M graphics card.



(a) Potential Error



(b) Field Error

Figure 8-13: The absolute error on the potential and field values generated from the fast multipole field map as compared to the direct method over the X-Y plane through the main spectrometer at $z = 0$.

trajectory. This is small enough to have negligible effect on its motion.

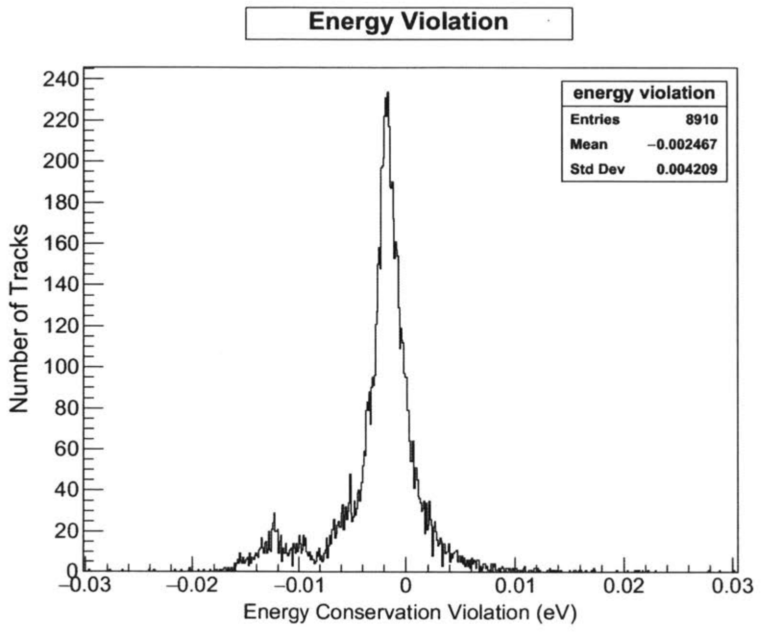


Figure 8-14: The size of energy conservation violation in tracking electrons through the main spectrometer model.

8.7 Particle Tracking and Navigation

In addition to accurate field evaluation, correctly tracking electrons through the spectrometer requires us to solve the equations of motion accurately and efficiently. This demands that we choose a time step for the Runge-Kutta ODE solver which is small enough to prevent the build up of numerical error, but large enough to avoid wasting effort when the error on the variables is already very small. To achieve this, we use the technique of embedded Runge-Kutta solvers, as described in appendix C. This allow us to efficiently estimate the local error on the ODE solution and limit the error in the position and momentum variables to be less than a certain amount. For the purpose of simulating our run data, the maximum allowed local error in the position variable estimated by the embedded Runge-Kutta method during each step was chosen to be 5 nm. The momentum error was not explicitly constrained for these simulations.

Along with solving the equations of motion, particle tracking requires a navigation algorithm, which has the responsibility of determining if the electron path has intersected any known surfaces. The ability to locate intersections of the electron beam with geometric objects is of particular importance for our analysis, since we must properly kill particle tracks which encounter the wire arrays and walls of the spectrometer in order to avoid spurious events at the FPD. The pre-existing Kassiopeia navigation algorithm implemented in the class `KSNavSpace` relies on having the ability to locate the nearest point and normal vector of all surfaces in the geometry. It then uses linear interpolation to compute the particle state between steps and locate intersections [85]. This method can be extremely fast and robust when all of the objects in the geometry model are properly nested and can be described analytically. However, this method was found to be insufficient when faced with the complexity of the three dimensional main spectrometer model. In order to work around this problem, a new navigation algorithm (implemented in the class `KSNavMeshedSpace`) using an octree subdivision of space was developed, based around ideas developed for ray tracing in computer graphics [96, 200].

The alternate navigation algorithm relies on breaking down the elements of the simulation geometry into a mesh in same way as in the BEM. Once a mesh has been specified, we construct an adaptive octree subdivision of space and sort the mesh elements into tree nodes in exactly the same manner in which the fast multipole region tree is constructed in algorithm 7. This adaptive subdivision allows us to quickly locate any nearby mesh elements. With the octree constructed, determining if there has been an intersection with any step of an electron track is relatively simple. First, the ODE solver propagates the solution forward by the appropriate time step. Then, a bounding ball is constructed about this new section of the electron track. All nodes in the octree which intersect the bounding ball are then located and examined to see if they contain any mesh elements. If they do not contain any mesh elements, then no intersection is possible, so the ODE solver proceeds to the next step. However, if the intersected nodes do contain mesh elements, then the interior of the bounding ball must be examined. This requires an appropriately accurate interpolant to represent the electron trajectory between the start and end of each step. Since we have the momentum of the particle available at each step, we can always construct a 3rd order accurate Hermite interpolant regardless of which type of Runge-Kutta integrator is used. However, greater accuracy is sometimes required, so the continuous Runge-Kutta methods of [72] were also implemented in order to enable 5th and 7th order accurate interpolation between the start and end of each step without additional derivative evaluations. The interpolated trajectory is then subdivided into a piecewise linear approximation, so that each segment can then be examined in order to see which octree nodes it intersects. If a segment intersects a node containing mesh elements, then each element must be examined to see if it in turn is intersected. This process iterates over all segments until either an intersection is found or the end of the step is reached. The simplified version of the full technique is summarized in algorithm 10. An additional advantage of this new navigation algorithm is that the particle state at an intersection can be determined much more accurately. This prevents the accumulation of errors during non-physical boundary crossings which only change the state of the simulation (but not the particle). Figure 8-15 demonstrates the use of the `KSNavMeshedSpace` navigation algorithm in tracking particles through the main spectrometer.

8.8 Generation of the Monte Carlo Data Set with Kassiopeia and Comparison to the ToF Data

In order to generate simulated particle tracks that will allow us to compare the measured and calculated field values, we need to make some simplifying assumptions when assembling the Monte Carlo model.

The first and most important simplification is that we will not explicitly model the electron gun. This simplification is needed to make the computational work required tractable. Since the electron gun is physically moved in order to access different positions in the flux tube, modeling the system completely would require

Algorithm 10 Algorithm to locate the intersection of a particle track and a meshed geometry.

```
Input: The geometry mesh $\mathcal{T}_N(\Gamma)$, its corresponding octree $\mathcal{N}_0$, and the next step of the particle trajectory given by the interpolant $\mathbf{x}(t) : \mathbb{R} \to \mathbb{R}^3$, with $t \in [t_0, t_1]$.
1: Construct the bounding ball, $B$, of all points $\mathbf{p} \in \mathbf{x}(t)$ for $t \in [t_0, t_1]$.
2: Find the set of nodes $\{\mathcal{N}_i\}$, with non-empty mesh element lists $\mathcal{L}_i$, such that $B \cap C_i \neq \emptyset$ by recursively visiting the octree, $\mathcal{N}_0$, according to algorithm 8.
3: if $\{\mathcal{N}_i\} = \emptyset$ then
4:     goto end.        ▷ No mesh elements near the bounding ball, no intersection possible.
5: end if
6: Construct the piecewise linear approximation $\tilde{\mathbf{x}}(t)$ specified by the sequence of points $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n\}$, such that $\epsilon > \sup \|\tilde{\mathbf{x}}(t) - \mathbf{x}(t)\|$ for all $t \in [t_0, t_1]$.
7: for $k = 0$ to $k = n - 1$ do
8:     Construct line segment $L_k$ from $\mathbf{p}_k$ to $\mathbf{p}_{k+1}$.
9:     for each $\mathcal{N}_j \in \{\mathcal{N}_i\}$ do
10:        if $L_k \cap C_j \neq \emptyset$ then
11:            for each mesh element $u_m \in \{\mathcal{L}_j\}$ do
12:                if $L_k \cap u_m \neq \emptyset$ then
13:                    Return the intersection point $\mathbf{P} = L_k \cap u_m$.
14:                end if
15:            end for
16:        end if
17:    end for
18: end for
Output: Either no intersection exists, or the first intersection with the mesh, $\mathbf{P}$.
```

us to solve for the charge densities of each of the 360 different configurations of the detector-spectrometer-gun system. To avoid this, we have instead chosen to model the electrons by propagating them forward from the entrance to the main spectrometer with a fixed energy and pitch angle.

This simplified electron gun model also ignores the finite (Gaussian) distributions of electron energy and pitch angle. This is advantageous since it reduces the total number of particles we need to track through the spectrometer model, while still being sensitive to the azimuthal potential variations. This is a reasonable simplification since we can see from the data in figures 8-4 and 8-6 that (at least to first order) the shape and width of the ToF distribution does not change as a function of the azimuthal angle. This implies that whatever the electron gun's true energy

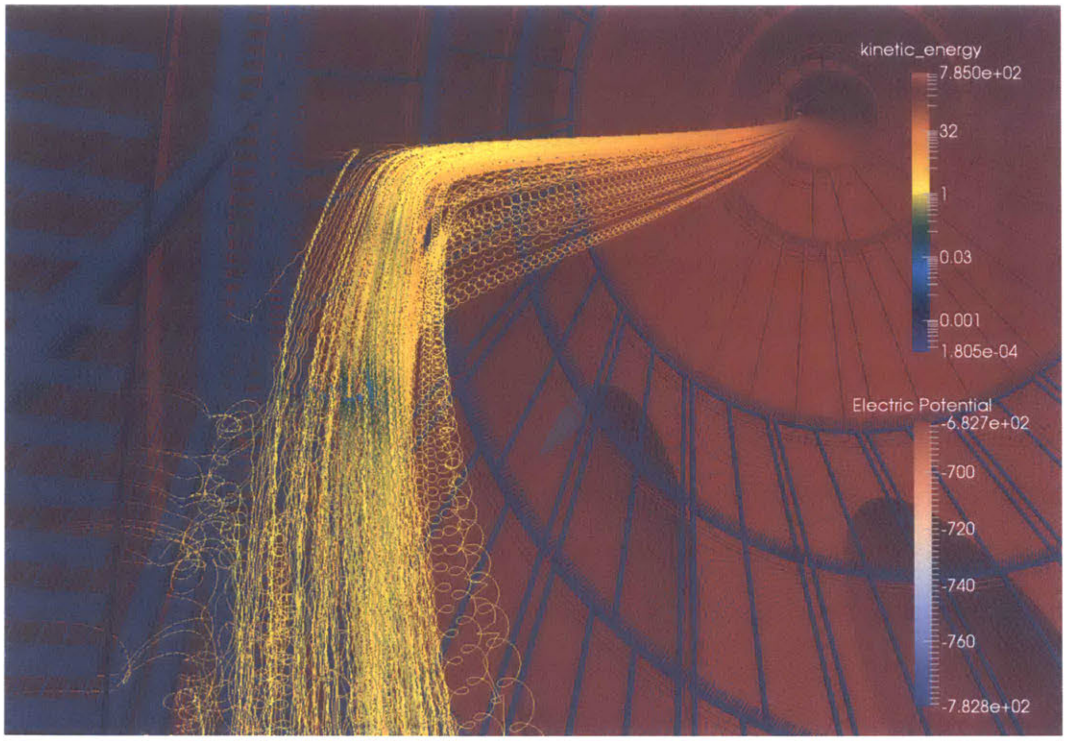


Figure 8-15: Particle tracks through the main spectrometer demonstrating the use of the new navigation algorithm to locate the intersections of electron tracks with the hull and wire arrays.

and pitch angle distributions are, they did not change during the measurement. Therefore, picking a single energy and pitch angle (0°) should still allow us to model the change in the most-probable ToF as a function of angle.

The second simplification is that we assume that the electron's energy at the spectrometer entrance is dictated entirely by the voltage of the electron gun photo-cathode. This reflects our lack of knowledge about the work function of the various surfaces involved, which essentially serve to alter the electron's energy. While we can determine the work function of the electron gun photo-cathode using light of different wavelengths, the work function of the main spectrometer is inaccessible to direct measurement. Fortunately, the effect of the unknown vessel work function is to make a small but relatively uniform change to the absolute potential the particle sees during its flight through the spectrometer. So while this may introduce a small offset to the value ToF, it should leave the size and location of the variations due to spatial fluctuations of the analyzing potential unaffected. To compensate for this, we will allow the simulated ToF trend to float on a constant offset which we will fit out using the per-pixel-ToF data.

Finally, we also need to compensate for our lack of perfect knowledge about the location of the focal plane detector (FPD). In principle, the exact physical position of the FPD can be measured to within a few millimeters using surveying equipment such as a Faro arm [210]. However, of foremost importance for the neutrino mass measurement is not the exact physical position of the FPD but rather its relative position with respect to the center of the flux tube which images the tritium source. This cannot be measured through the use of surveying equipment but only by propagating particles through the flux tube and examining the FPD's response. A variation of this alignment technique has been done previously, but with an asymmetric magnetic field setting intended to propagate particles from the main spectrometer walls onto the FPD [114]. This method can inform us about the alignment of the physical main spectrometer and the FPD. However, is does not necessarily measure the true alignment of the flux tube with the FPD, since by necessity, it does not use the same magnetic field configuration as in a neutrino mass measurement. In order to deal with the FPD placement uncertainty, we will fit out the FPD's relative X-Y position with respect to the simulated flux tube using the hit pattern and per-pixel-ToF data.

With these assumptions out of the way, the first step in the simulation of the most-probable ToF trend is to construct a map of the ToF for all positions on the FPD face in its nominal position. To do this, the field of the main spectrometer and detector system is computed directly from the electrode potential and magnet current settings reported by slow control. Then, we start a large cylindrical grid of electrons at the entrance of the main spectrometer in the central plane of the pre-spectrometer magnet and track them until they either terminate on the FPD or main spectrometer surface. Each simulation consisted of tracking approximately 1.1×10^4 particles. The radius of the starting positions of the particles ranged from 1mm to 33mm, and the azimuthal positions covered 0° to 360° in 1° increments. For

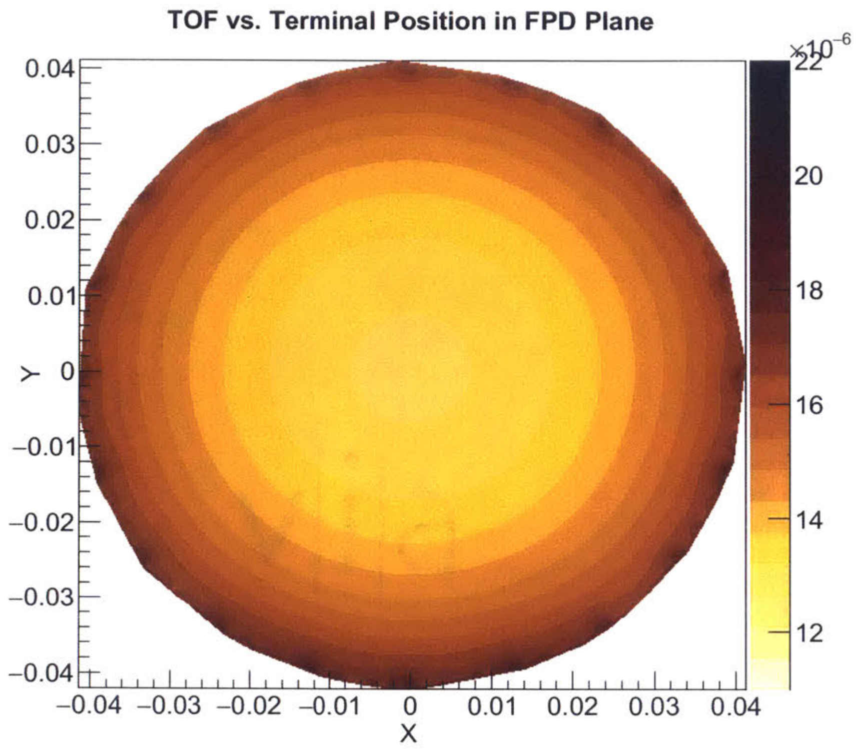


Figure 8-16: The ToF of particles arriving in the detector as a function of position in the FPD plane. Generated from a Kassiopeia simulation of the run conditions of collection (A).

those particles which do successfully reach the surface of the FPD, we can record their simulated ToF as a function of their terminal position, which allows us to reconstruct the expected ToF over the plane coincident with the detector face. The interpolated ToF as a function of position in the FPD plane is shown in figures 8-16, 8-17, and 8-18 for the simulation of the run conditions of collections (A), (B), and (C) respectively. We note that for the run collections with the 2 Gauss magnetic field setting ((A), (C)) the azimuthal variation in the ToF due to the influence of the wire array support structures is readily apparent with a periodicity of approximately 18 degrees, whereas the 3.8 Gauss magnetic field setting of run collection (C) is much less sensitive to this effect. Once we have simulated the ToF as a function of position in the FPD plane, the next step is to reconstruct the path of the electron beam over the FPD face during the course of each run collection. Naively, one would expect

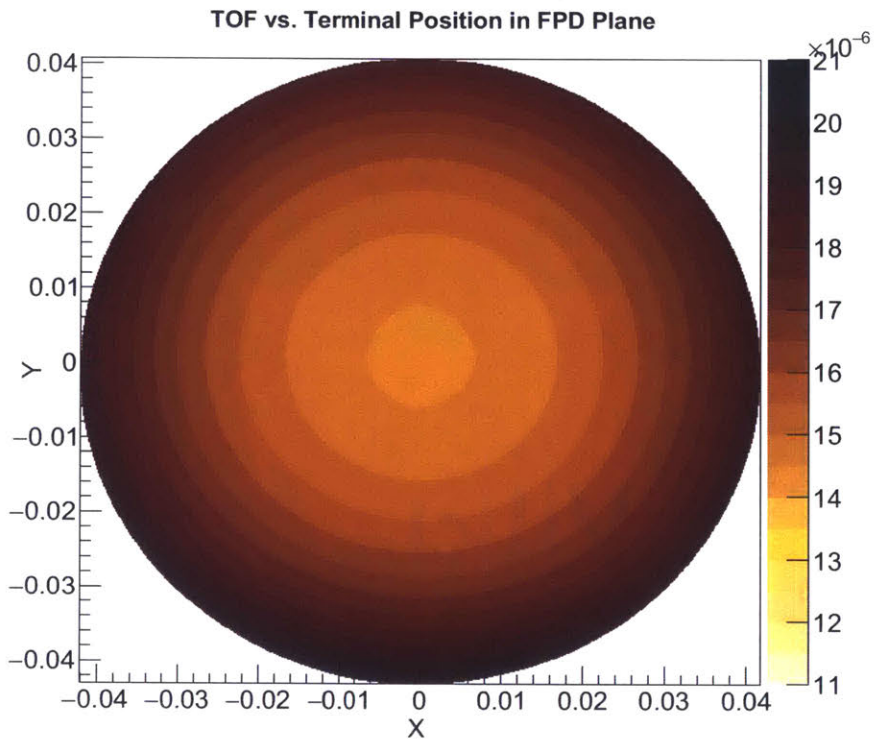


Figure 8-17: The ToF of particles arriving in the detector as a function of position in the FPD plane. Generated from a Kassiopeia simulation of the run conditions of collection (B).

that the electron beam trajectory ought to be circular. However, various factors, such as the imperfect axial symmetry of the magnetic fields and misalignment between the detector, spectrometer, and electron gun systems serves to distort the trajectory from the expected shape. Since we were not able to reproduce the true FPD hit pattern with an (offset) circular trajectory, we have chosen the next simplest model (which has 2 additional parameters), which is that of an oval. The oval model is represented by the following parametric equations as a function of the azimuthal angle ϕ:

$$x = c_x + r\cos(\phi)[1 + \alpha\cos^2(\phi - \phi_0)] \,, \tag{8.3}$$

$$y = c_y + r\sin(\phi)[1 + \alpha\cos^2(\phi - \phi_0)] \,. \tag{8.4}$$

The position of the oval center (c_x, c_y), the nominal radius r, distortion α, and bulge

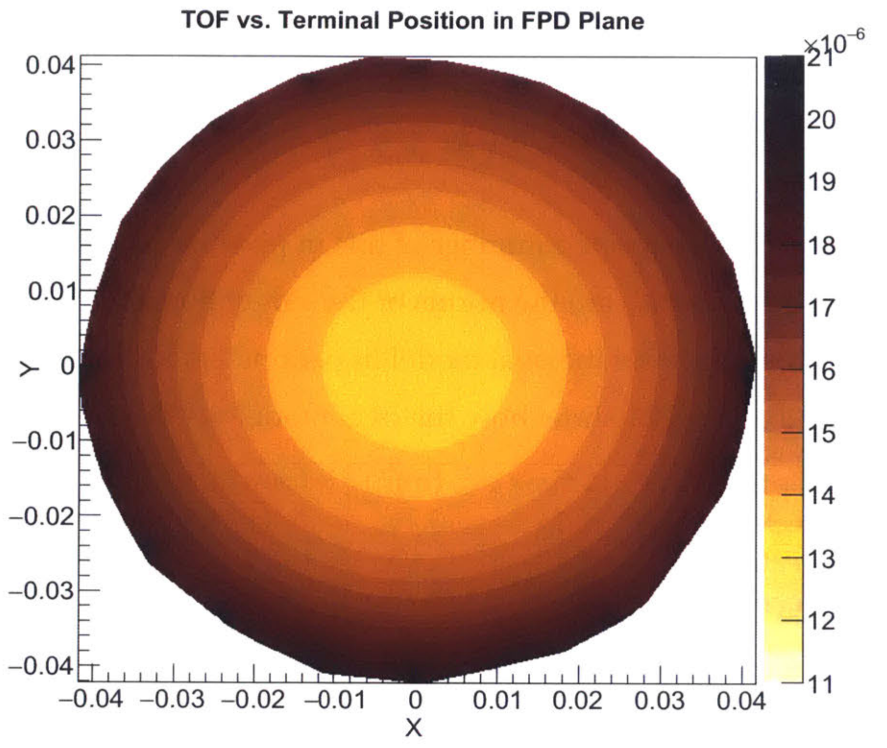


Figure 8-18: The ToF of particles arriving in the detector as a function of position in the FPD plane. Generated from a Kassiopeia simulation of the run conditions of collection (C).

angle ϕ_0 are fit directly from the FPD hit pattern which is completely orthogonal to the ToF data. The oval model fit is a highly non-linear problem, so we rely on a simulated annealing approach (provided by ROOT's Minuit2 package) in order to minimize the objective function. Unfortunately, this method (simulated annealing) is unable to offer any error estimation. However, the error estimating fitter provided by Miniut (MIGRAD) is not able to converge with any reasonable objective function. In any case, MIGRAD's error estimates are probably not meaningful for this fit since the objective function is not of a χ^2 or log-likelihood form. Instead, the objective function is computed using the Bhattacharyya distance measure (BDM) [192] between the oval model's prediction of the FPD hit pattern and the actual FPD hit pattern. The objective function is given by:

$$T_{\text{BDM}} = \left(\sum_{i=0}^{147} \frac{u_i v_i}{N_u N_v} \right)^{1/2} , \tag{8.5}$$

where u_i is the model's expected number of hits in pixel i, v_i is the actual number of pixel hits, and N_u and N_v are the norms of the simulation and data histograms respectively. Table 8.5 shows the oval model fit parameters for each run collection. Figures 8-19, 8-20, and 8-21 show how the oval model fits the hit patterns of run

Description	c_x (mm)	c_y (mm)	r (mm)	α (mm)	ϕ_0
Run Collection (A)	-1.11	-3.16	38.11	1.27	$-104.8°$
Run Collection (B)	-1.20	-2.95	38.15	-1.16	$152.6°$
Run Collection (C)	-0.69	-0.73	39.63	-1.55	$-73.5°$

Table 8.5: Parameters of the oval model fit to the FPD hit pattern for each run collection.

collections (A), (B), and (C) respectively.

For run collection (C), the hit pattern formed from the complete collection (including the corrupted run) was used because although there was some instability in the manipulator position and the surplus energy of the electron gun, this should not severely affect which pixels are hit by the beam.

Once the oval model of the electron beam's path over the FPD's face has been determined from the hit pattern, the next step is to fit out the FPD X-Y displacement and the constant ToF offset. To do this, we first compute the per-pixel-ToF from the data. This is defined as the most-probable ToF constructed from the distribution of all events incident on a single pixel during the run collection. Then, we minimize the χ^2 difference between the per-pixel ToF and the predicted per-pixel-ToF. The χ^2 function is given by the following sum over the set of N_{pixels} pixels hit during the run collection:

$$\chi^2 = \sum_{i=1}^{N_{\text{pixels}}} \left[\frac{(\bar{t}_d(i) - \bar{t}_s(i))^2}{\sigma_d^2(i) + \sigma_s^2(i)} \right] , \tag{8.6}$$

where $\bar{t}_d(i)$ and $\bar{t}_s(i)$ are the mean per-pixel-ToF of the data and simulation respectively. The deviation of the data, $\sigma_d^2(i)$, is taken to be the FWHM of the DAQ timing

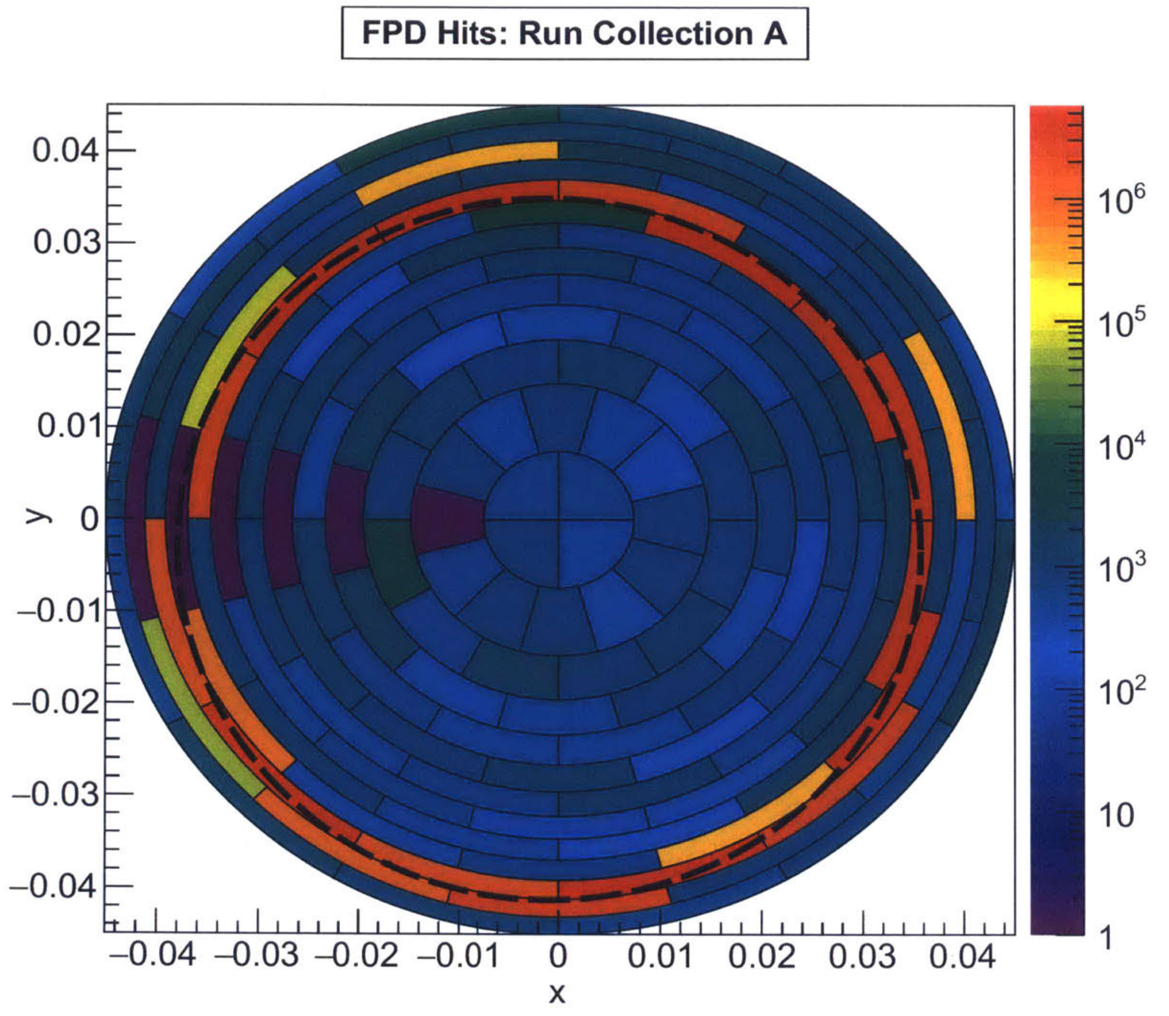


Figure 8-19: The oval model fit to the FPD hit pattern for run collection (A). The black dashed line exhibits the model's electron beam trajectory. The purple pixels, with no events, are all members of the same disabled pre-amp card.

resolution (100ns), while $\sigma_s^2(i)$ is the root-mean-square deviation of the simulation samples on each pixel. The predicted per-pixel-ToF is calculated by sampling the ToF function generated from the Kassiopeia simulation (such as that shown in figures 8-16 to 8-18) at many points located along the oval beam path that has been fit from the hit pattern. We then allow the position of the FPD and thus the corresponding oval shaped beam path model to float in the X-Y plane until the difference between the simulated per-pixel-ToF and the per-pixel ToF is minimized. The result of this minimization for run collections (A), (B), and (C) is shown in

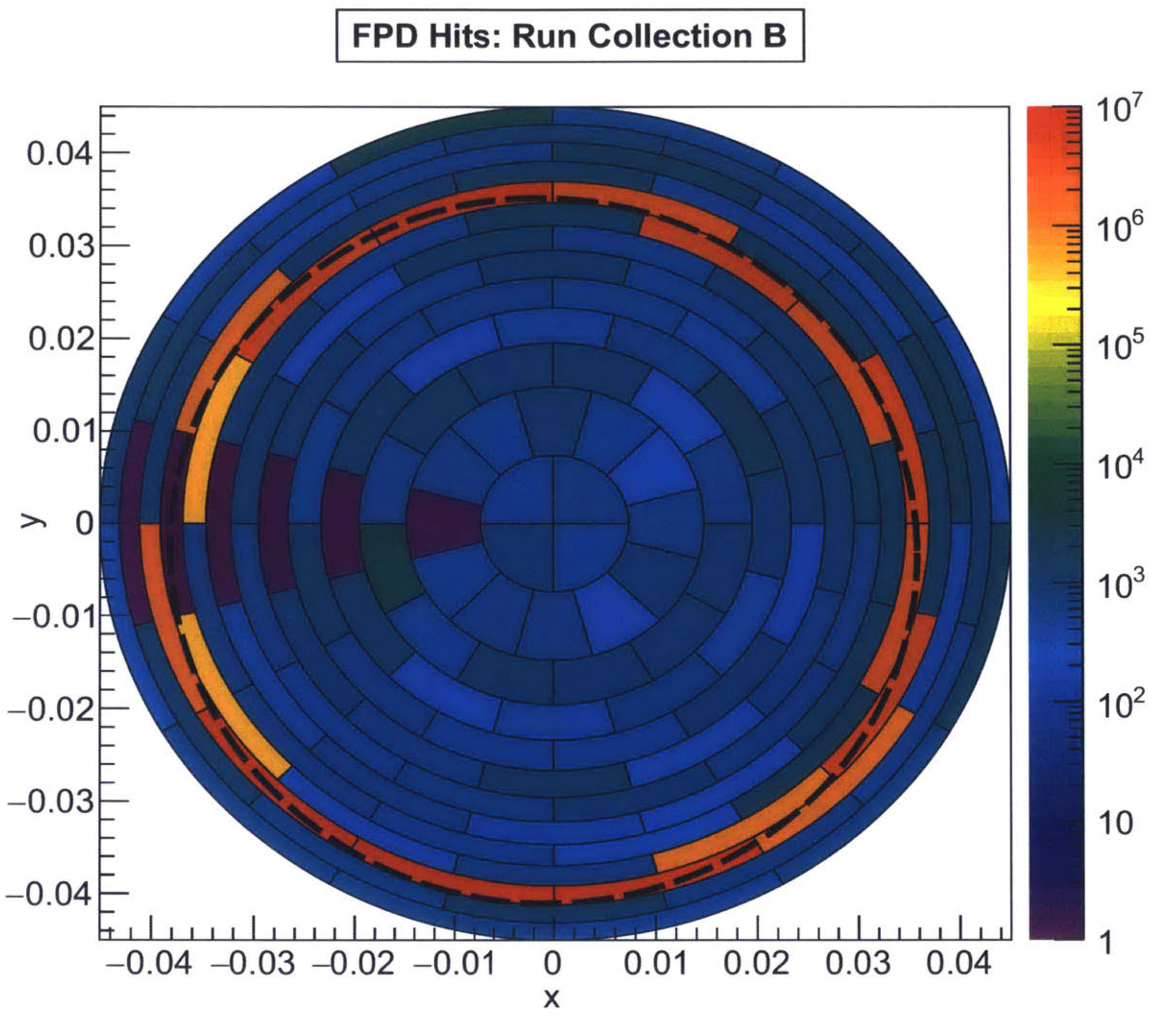


Figure 8-20: The oval model fit to the FPD hit pattern for run collection (B). The black dashed line exhibits the model's electron beam trajectory. The purple pixels, with no events, are all members of the same disabled pre-amp card.

figure 8-22.

Description	Δ_x (mm)	Δ_y (mm)	Δ_t (μs)	Latency corrected Δ_t (μs)
Run Collection (A)	-1.72 ± 0.15	2.61 ± 0.16	-3.35 ± 0.03	-2.08 ± 0.03
Run Collection (B)	-1.90 ± 0.15	2.84 ± 0.18	-3.14 ± 0.03	-1.87 ± 0.03
Run Collection (C)	-1.55 ± 0.19	3.26 ± 0.001	-3.34 ± 0.04	-2.07 ± 0.04
Mean Value	-1.72 ± 0.17	2.90 ± 0.33	-3.28 ± 0.12	-2.01 ± 0.12

Table 8.6: Parameters describing the constant ToF offset and the displacement of the FPD relative to the flux tube center generated from the per-pixel ToF fit. Except for the errors on the mean parameter values, the errors given are those reported from the MIGRAD fitting routine.

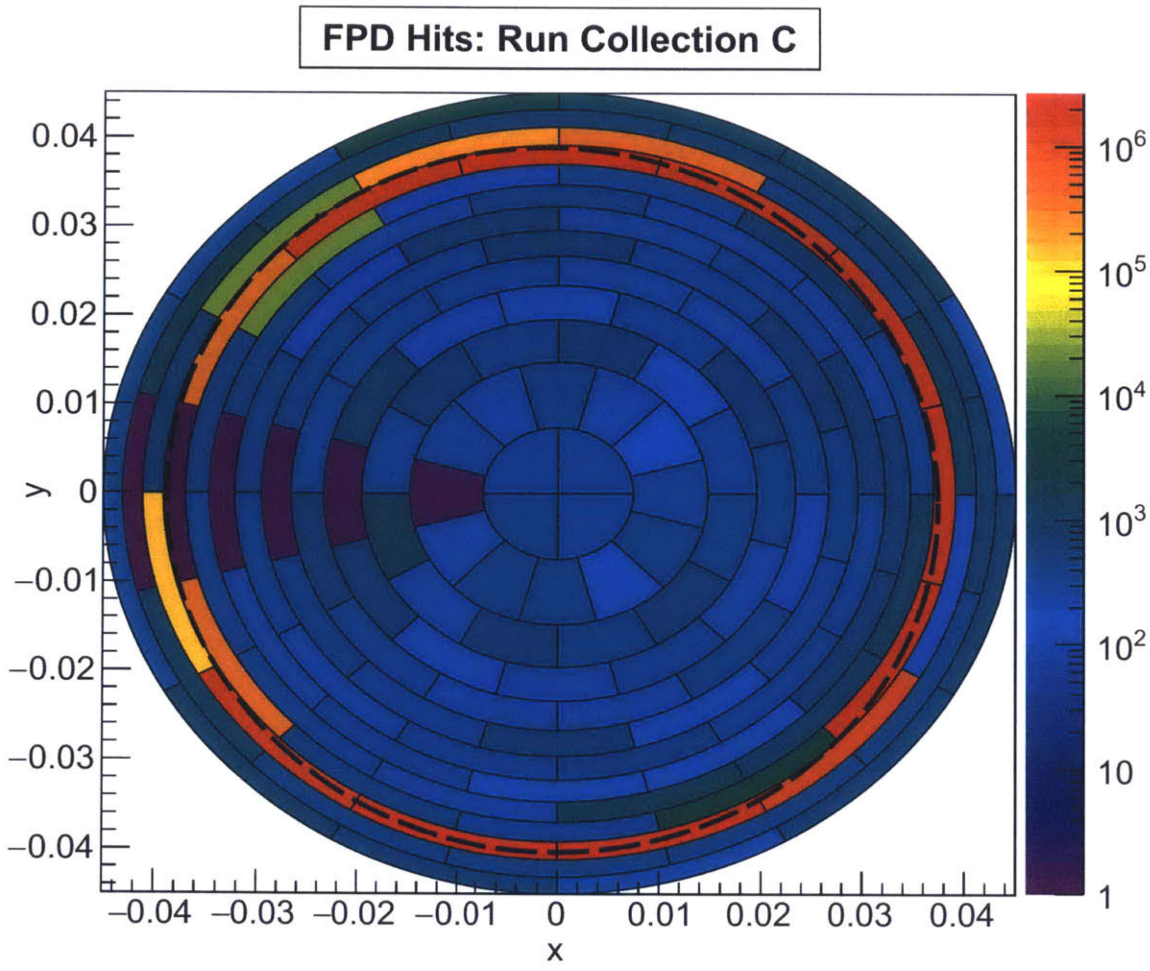
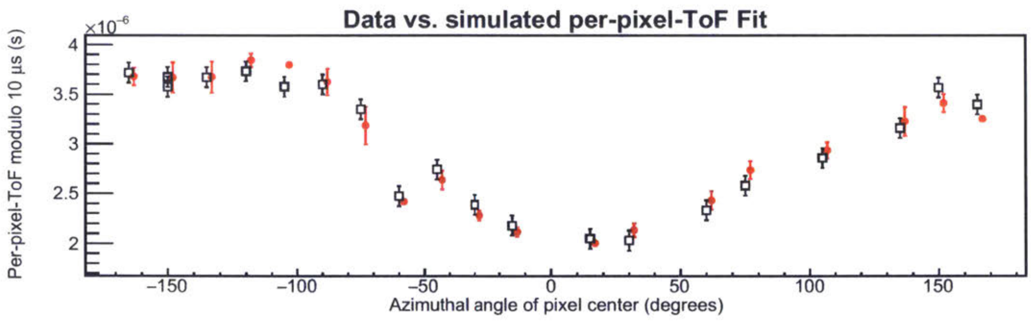


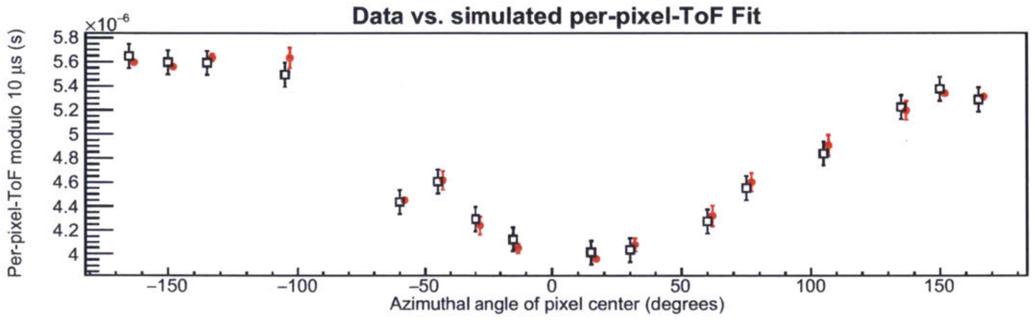
Figure 8-21: The oval model fit to the FPD hit pattern for run collection (C). The black dashed line exhibits the model's electron beam trajectory. The purple pixels, with no events, are all members of the same disabled pre-amp card.

Table 8.6 contains the results of this fit for run collections (A) and (B). The values of the displacement are relative to the nominal position of the FPD used in the simulation and determined from previous alignment measurements [114, 210]. Values of the constant ToF offset which are corrected for the signal latency are listed in the last column.

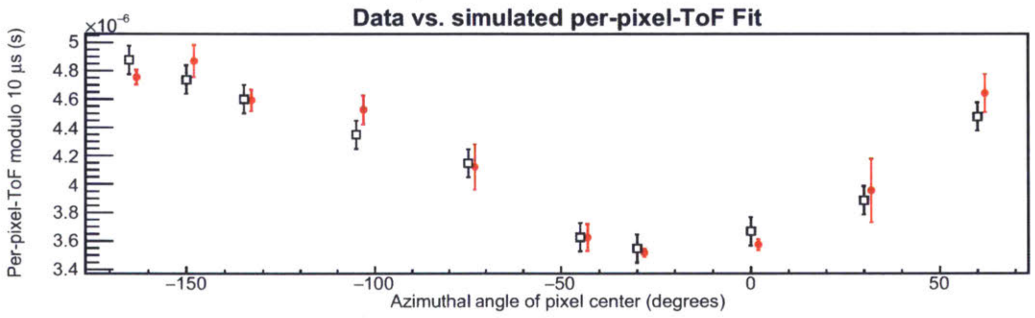
It is worth noting that even for a variety of run conditions and different electron beam paths, the displacement of the FPD with respect to the center of the flux tube as well as the offset between the simulated and measured ToF are both reasonably



(a) Run Collection (A).



(b) Run Collection (B).



(c) Run Collection (C).

Figure 8-22: The per-pixel ToF (black squares) of each run collection plotted with the simulated per-pixel ToF (red dots) after the three parameter fit. The data points are plotted as a function of the azimuthal angle of the center of the corresponding pixel. Each simulated per-pixel ToF point is displaced 2 degrees to the right for clarity.

consistent. It is not immediately clear what reason exists for the anomalously small error on the value of the y-coordinate displacement of run collection (C). However, run collection (C) does not form a complete 360° scan due to the corruption of the last run, so it's possible this had some effect on both the oval model fit and the displacement fit.

With the FPD displacement and constant offset between the data and the simulated ToF determined from the per-pixel fit, we can now examine the behavior of the ToF at sub-pixel resolution by plotting it as a function of the electron gun UHV manipulator's azimuthal angle. To generate a high resolution prediction of the most-probable ToF, we simply sample the simulation generated ToF interpolant along the (corrected for displacement) beam path in 1° increments and apply the constant ToF correction. The predicted ToF and the most-probable ToF trend fit extracted from data are shown overlaid in figures 8-23, 8-25, and 8-27 for run collections (A), (B) and (C) respectively. The normalized residual error between the data and the simulation are shown below each ToF trend.

From figures 8-23, 8-25, and 8-27, we can see there is fairly good agreement between the data and the Monte Carlo. However, a more qualitative comparison is desirable. Unfortunately, due to the non-linear nature of the fitting problem, the reduced χ^2 metric is not appropriate here [10]. Instead, we have histogrammed the normalized residuals in figures 8-24, 8-26, and 8-28. Under the assumption that the simulation model produces an appropriate reproduction of the data and has not been over-fit, we would expect the normalized residuals to form a Gaussian distribution with a mean of zero and a standard deviation of one. Examining the distribution of the normalized residuals for each ToF trend shows that the means are reasonably consistent with zero and that the standard deviations are close to one. However, observing agreement between the shapes of simulated and measured ToF trends is not completely sufficient for us to validate the field calculation. Instead, it is preferable to make a direct comparison between the measured and calculated electric potential in the analyzing region.

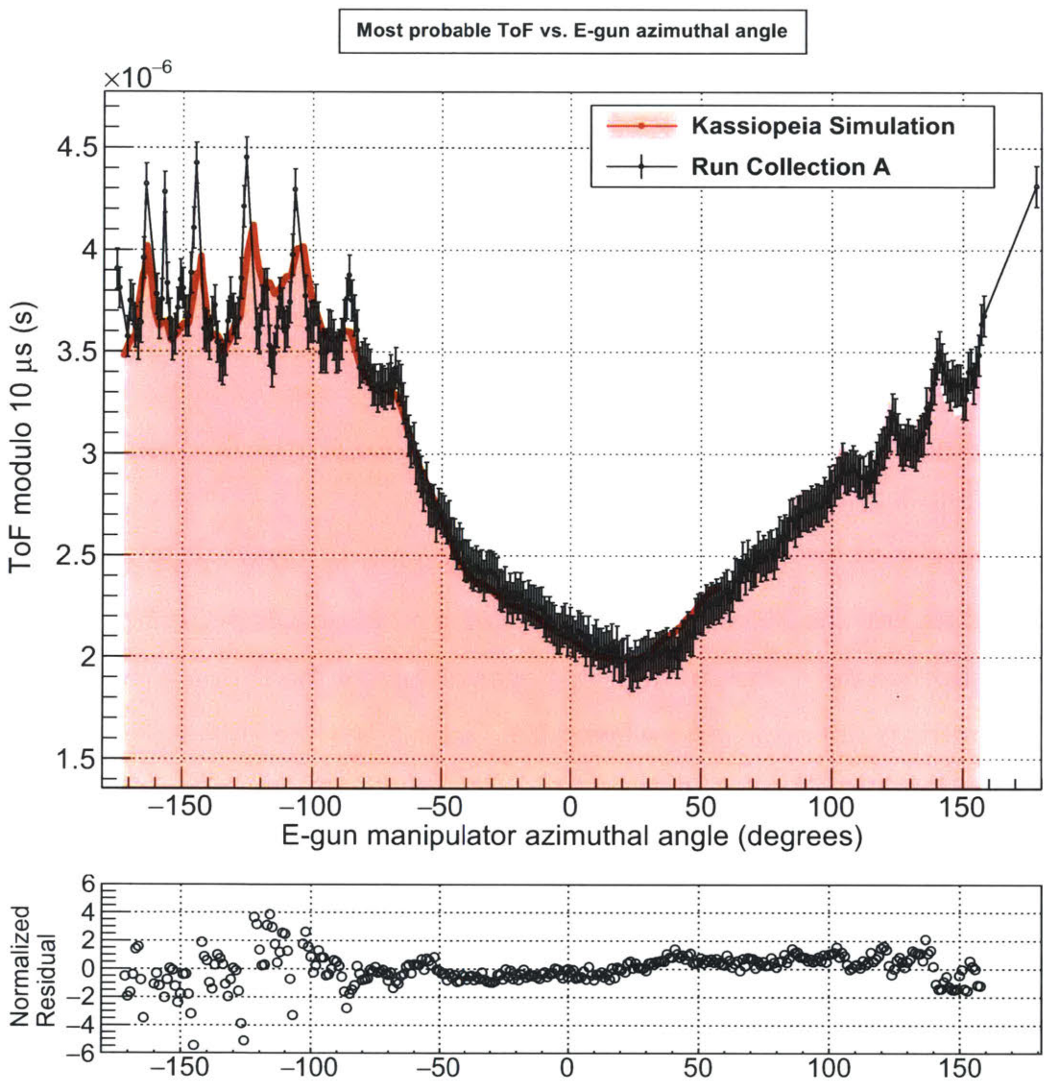


Figure 8-23: The ToF trend as a function of the e-gun's manipulator angle for run collection (A).

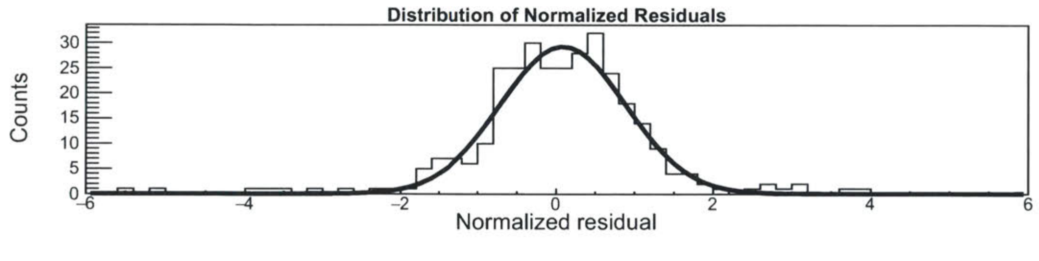


Figure 8-24: Distribution of normalized residuals run collection (A). The black line shows a Gaussian fit to the distribution with $\mu = 0.087$ and $\sigma = 0.79$.

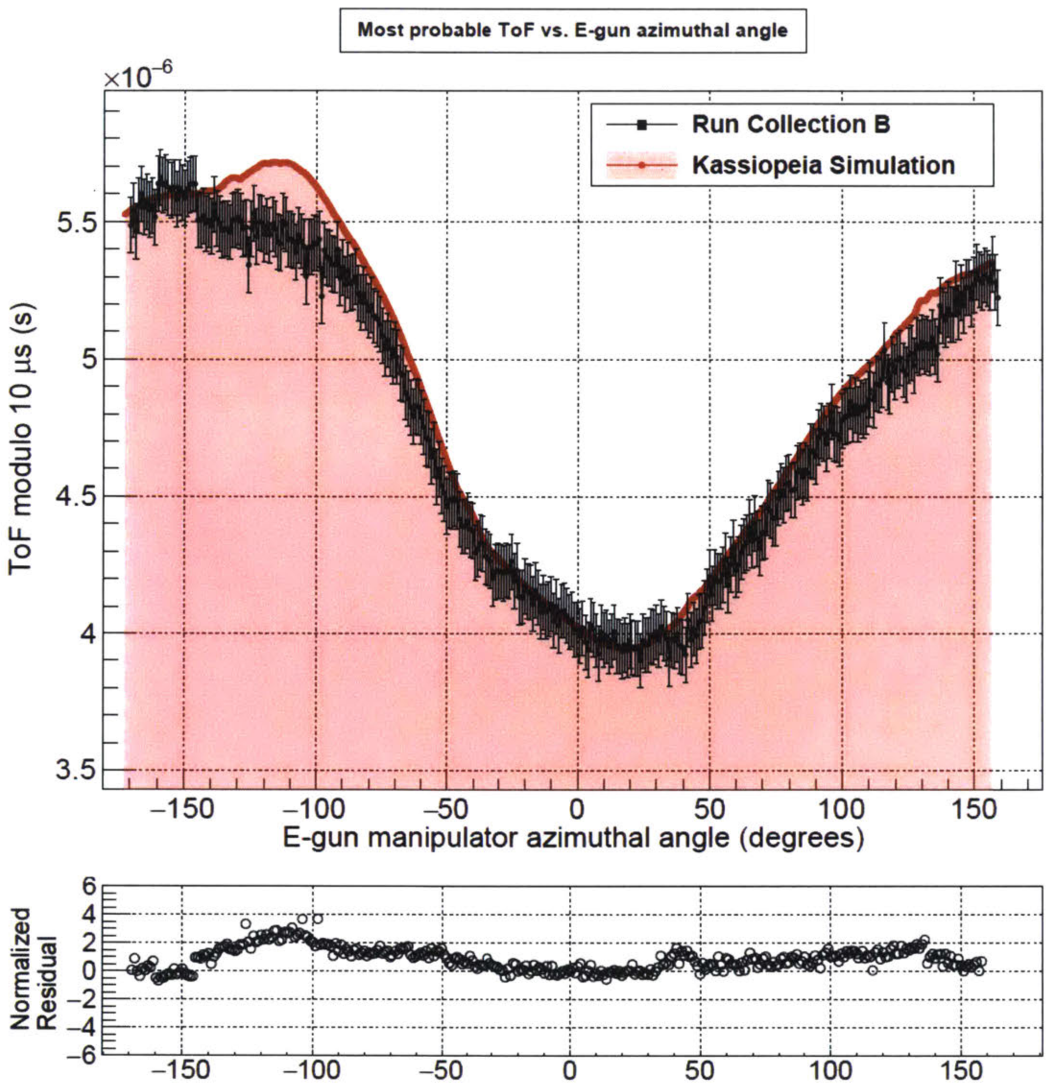


Figure 8-25: The ToF trend as a function of the e-gun's manipulator angle for run collection (B).

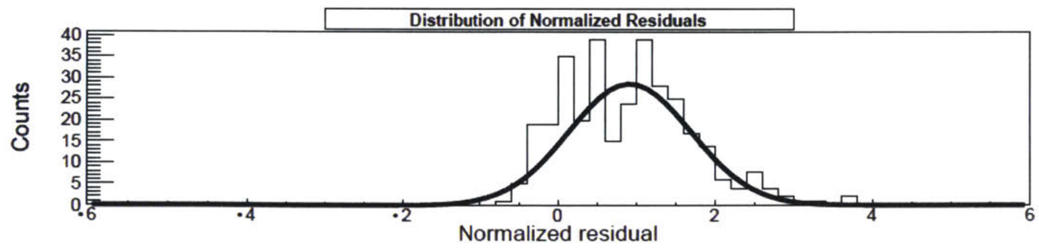


Figure 8-26: Distribution of normalized residuals run collection (B). The black line shows a Gaussian fit to the distribution with $\mu = 0.92$ and $\sigma = 0.78$.

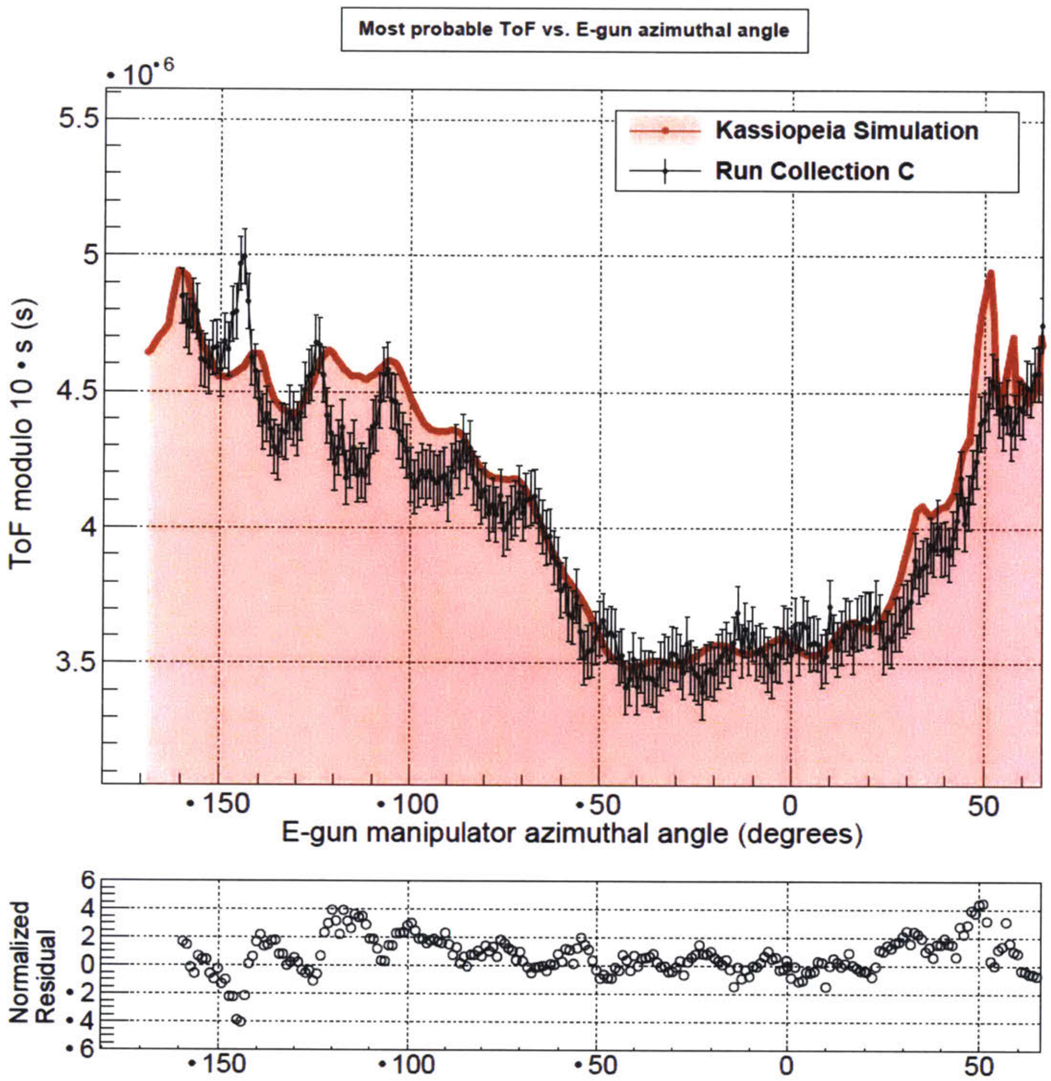


Figure 8-27: The ToF trend as a function of the e-gun's manipulator angle for run collection (C).

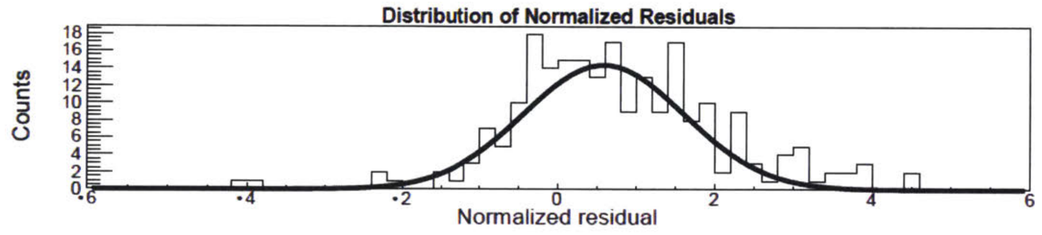


Figure 8-28: Distribution of normalized residuals run collection (C). The black line shows a Gaussian fit to the distribution with $\mu = 0.59$ and $\sigma = 1.02$.

8.9 Extraction of the Electric Potential Variation

In order to better validate the field calculations, we need to extract the actual variation in the analyzing potential from the data in a way that is independent of the Monte Carlo model. To do this, we will make use of the transmission function measurements that were taken at 90° intervals during the azimuthal ToF scans in order to calibrate the ToF response to a shift in surplus energy. Since the energy of the electron gun was fixed during the azimuthal angle scans, any shift in the ToF should primarily be the result of a change in the surplus energy and thus a variation of the potential along the beam path.

In theory, if the position of the electron beam in the flux tube had been perfectly symmetric about the flux tube axis, a single transmission function taken at a point during the azimuthal scan would be sufficient to calibrate the relationship between the ToF and the potential shift. However, because of the misalignment between the electron gun and the flux tube, there is also some radial variation in the electron beam position, which mixes the radial and azimuthal potential inhomogeneities, as well as changing the overall length of the flight path. Unfortunately, due to the small number of transmission function measurements taken during the azimuthal ToF scans, compensating for the radial variation is somewhat difficult and is the dominant error in the resulting extracted potential.

In order determine the analyzing potential from the value of the ToF, we will need to be able to estimate this relation from the transmission function measurements. With a functional form for the relation between the ToF and the analyzing potential, we may then invert it and extrapolate the change in analyzing potential from variations in the ToF. To this end, we will fit the measurements with a function derived from an extremely simplistic model of the underlying physics. The time-of-flight, t, is given by the following integral over the arc length:

$$t = \int_0^{\ell_f} \frac{d\ell}{v(\ell)} \,. \tag{8.7}$$

In order to compute this integral exactly, we would need to know the particles exact trajectory and velocity along its path. Therefore, computing the exact value of the time of flight in a realistic model of the main spectrometer is generally only possible by numerically solving the equations of motion. Such a degree of complexity is not necessary to produce a simple model for the fit function, but we can gain some insight from examining simulated particle trajectories. Figure 8-29 shows

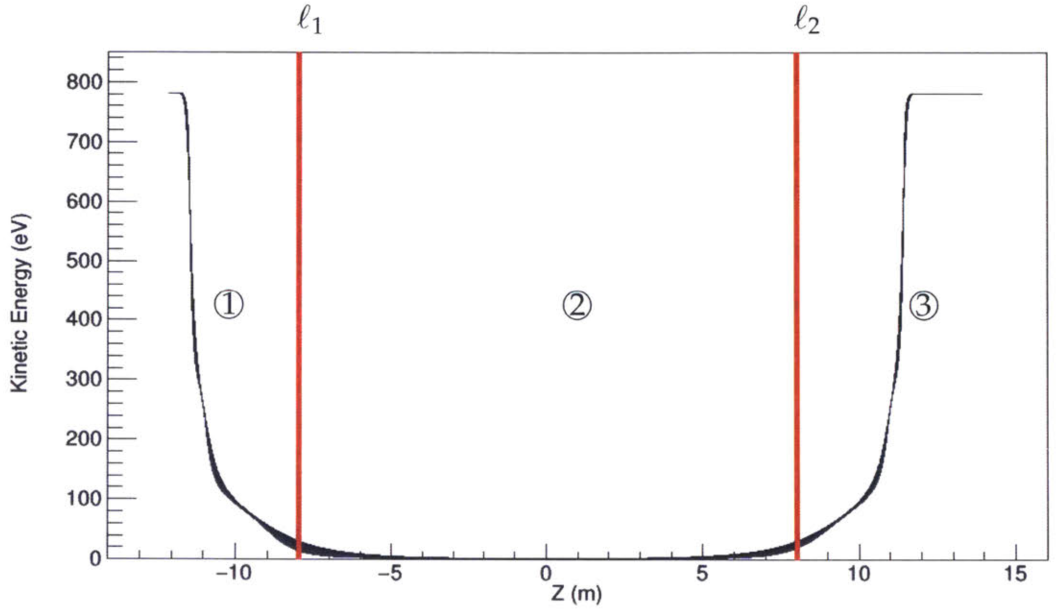


Figure 8-29: Behavior of the kinetic energy of a particle as it traverses the main spectrometer MAC-E filter. The electric and magnetic fields were configured according to the run conditions of collection (B).

the expected behavior of an electron's kinetic energy as it passes through the main spectrometer configured as in run collection (B). What is notable from this graph is that we can divide the trajectory into three regions: the analyzing region (2) where the kinetic energy is small and essentially constant, and the entry and exit regions (1) and (3), where the kinetic energy of the electron is large and it is undergoing a large amount of acceleration. With this information it is reasonable to assume that to a large degree, the ToF only depends on the particle and spectrometer properties in region (2), and that the contributions of the entry and exit regions are small and essentially constant. Furthermore, we will also assume that the particle has no transverse momentum and that the trajectory is a straight line. This ignores not only the gyro-motion but also the fact that the particles of interest (since their starting positions are off-axis) follow the curved path of the magnetic field lines. Therefore, we replace the integral of 8.7 with the exceedingly simple approximation:

$$t \approx \int_{\ell_1}^{\ell_2} \frac{d\ell}{v(\ell)} + \gamma \approx \frac{L}{\langle v \rangle} + \gamma \,, \tag{8.8}$$

where, $L = \ell_2 - \ell_1$, is the length of the electron trajectory in the analyzing region, γ is an arbitrary offset due to the particle's travel time outside of the analyzing region, and the average velocity is:

$$\langle v \rangle = \sqrt{\frac{2\langle E_k \rangle}{m}} = \sqrt{2(E_{\text{tot}} - q\langle U \rangle)/m} \,. \tag{8.9}$$

The electron's total energy is defined by the electron gun potential U_{G}:

$$E_{\text{tot}} = qU_{\text{G}} \,, \tag{8.10}$$

which is related to the inner electrode potential by the surplus energy, $q\Delta$:

$$q\Delta = qU_{\text{G}} - qU_{\text{IE}} \,. \tag{8.11}$$

Throughout the spectrometer, the potential in the analyzing region is slightly more positive than the inner electrode potential by a spatially dependent amount, $\beta(r, \phi)$, which represents the radial and azimuthal inhomogeneity:

$$U(r, \phi) = U_{\text{IE}} + \beta(r, \phi) \,. \tag{8.12}$$

Combining equations 8.12 and 8.11 with 8.8 yields a three parameter model for the ToF as a function of surplus energy (at a fixed starting position). This model is given by:

$$t(q\Delta)|_{r,\phi} = \frac{L}{\sqrt{\frac{2q}{m}\left(U_G - \langle U \rangle\right)}} + \gamma \tag{8.13}$$

$$= \frac{L}{\sqrt{\frac{2|q|}{m}\left(\Delta + \langle \beta \rangle\right)}} + \gamma\,, \tag{8.14}$$

where L, γ, and $\langle \beta \rangle$ are the parameters to be fit from the ToF data.

In order to reconstruct the true ToF as a function of energy, we must first apply a cycle correction to the raw data which is only known up to some multiple of the pulser period of $\tau = 10\mu s$. To perform this cycle correction, we make the assumption that the ToF as a function of surplus energy is a monotonically decreasing function. Therefore, if we compare two adjacent data points (t_1, E_1) and (t_2, E_2), if $E_1 < E_2$ then we expect to have $t_1 > t_2$. However, if this is not the case, and instead, $t_1 < t_2$, then there are two possible reasons which may cause this to happen. The first reason is measurement error, while the second is that the change in the ToF between the two energies has been great enough that at least one pulser period has been crossed. If the reason is measurement error, then the change in ToF should not be very large, and the slope should be less than some parameter κ[5]:

$$m = \frac{t_2 - t_1}{E_2 - E_1} < \kappa\,. \tag{8.15}$$

If however, the slope, m, is larger than this limit, then the ToF increase must be due to a pulser period crossing. In this case, we need to modify all values of the ToF for energies, $E_1 < E_2$, by making the correction $t \longrightarrow t + \tau$. To ensure that the ToF is reconstructed properly, we iterate over all the data points in order to apply this correction as many times as necessary. The reconstructed ToF curve is then fit with the model of 8.14. An example of the data and resulting model fit is shown in figure 8-30 for run #23938. We note that value of the parameter γ is negative, which is obviously non-physical. However, this is not the true value and is of no significance

[5]Empirically, the appropriate value of κ was found to be approximately $3\mu s/(0.16\text{eV})$.

because our reconstruction of the ToF vs. energy curve is only unique up to some constant multiple of the pulser period. Applying this model to each transmission

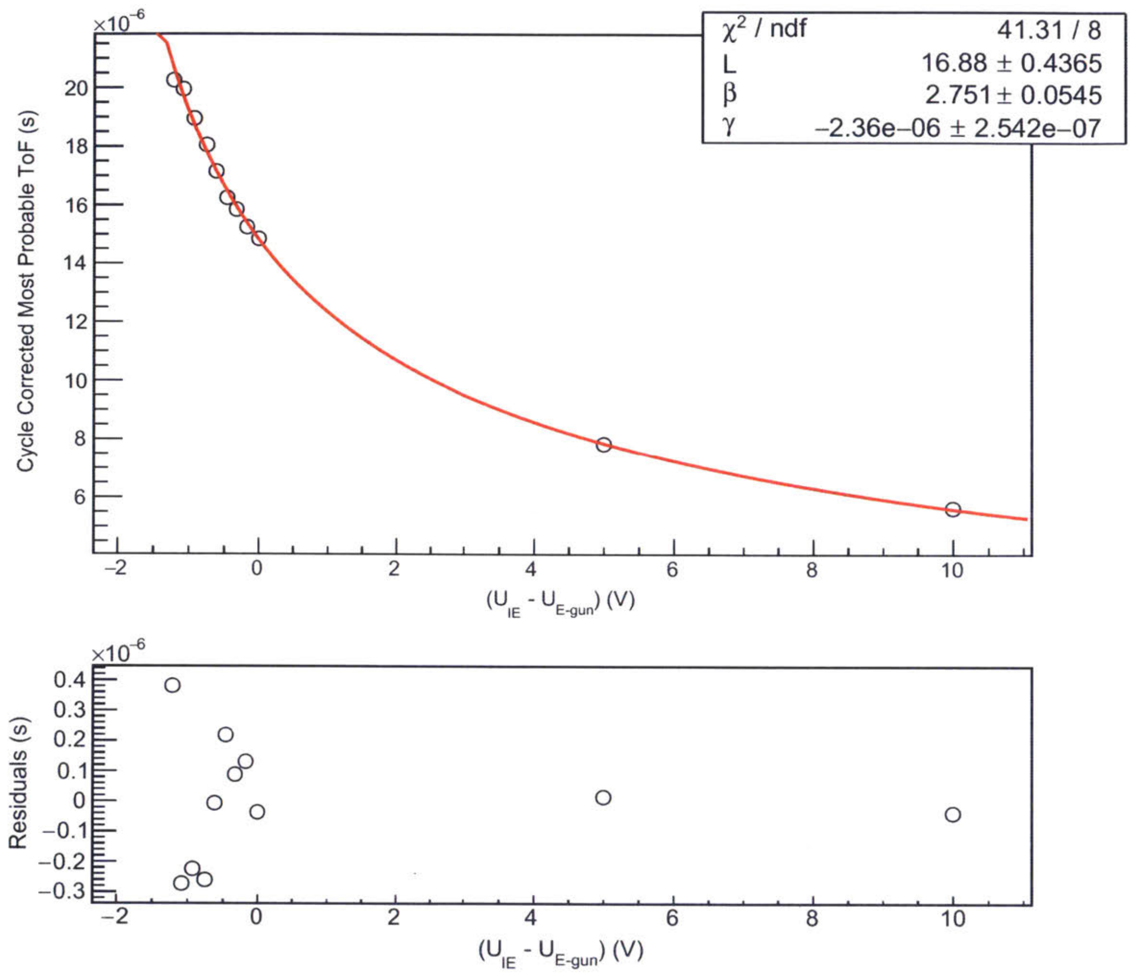


Figure 8-30: The ToF as a function of surplus energy. The red line is the fit according to the model given in equation 8.14. The black circles are the data points of the most-probable ToF at each measured energy. The error on the ToF is taken to be the DAQ resolution of 100ns.

function measurement in collections (A), (B), and (C), yields the parameter values given in table 8.7. The fits are shown in appendix D for visual comparison with the data.

Now in order to proceed with the extrapolation of the mean shift of the electric potential in the analyzing region, $\langle \beta(\phi, r) \rangle$, from the measured ToF recorded during the azimuthal scans, we simply need to invert the relation of equation 8.14, which

Run Collection	Description	L (m)	$\langle\beta\rangle$ (V)	γ (μs)	Angle ϕ
(A)	Run # 23839	16.88 ± 0.44	2.75 ± 0.05	-2.36 ± 0.25	$-160.0°$
	Run # 23844	14.71 ± 0.17	2.50 ± 0.01	-1.44 ± 0.15	$-70.0°$
	Run # 23847	14.92 ± 0.14	2.84 ± 0.01	-1.96 ± 0.14	$20.0°$
	Run # 23850	13.97 ± 0.13	2.58 ± 0.01	-0.99 ± 0.13	$110.0°$
	Mean Value	15.12 ± 1.24	2.67 ± 0.16	1.69 ± 0.60	NA
(B)	Run # 23863	13.23 ± 0.10	2.74 ± 0.01	-0.66 ± 0.12	$-160.0°$
	Run # 23866	12.75 ± 0.11	2.75 ± 0.01	-0.77 ± 0.12	$-70.0°$
	Run # 23869	13.74 ± 0.14	2.99 ± 0.01	-1.62 ± 0.14	$20.0°$
	Run # 23872	12.88 ± 0.12	2.77 ± 0.01	-0.89 ± 0.13	$110.0°$
	Mean Value	13.15 ± 0.44	2.81 ± 0.12	-0.99 ± 0.43	NA
(C)	Run # 23933	18.21 ± 0.33	2.87 ± 0.03	-3.24 ± 0.21	$-160.0°$
	Run # 23938	14.60 ± 0.16	2.59 ± 0.01	-1.51 ± 0.15	$-70.0°$
	Run # 23941	15.26 ± 0.18	2.71 ± 0.01	-2.00 ± 0.16	$20.0°$
	Mean Value	16.14 ± 1.86	2.72 ± 0.14	-2.25 ± 0.89	NA

Table 8.7: Parameters of the ToF as a function of surplus energy for the model of equation 8.14, fit to various data sets.

yields:

$$\langle\beta(\phi,r)\rangle = \frac{L^2}{\frac{2|q|}{m}\left(t(q\Delta)|_{r,\phi} - \gamma\right)^2} - \Delta\,. \tag{8.16}$$

The value of the ToF and surplus energy are known directly from recorded data as a function of the electron gun manipulator's azimuthal angle, ϕ. However, we note that the parameters, γ and L, do change as a function of angle, so there is some ambiguity in the choice we should make for their values when reconstructing $\langle\beta\rangle$. Ideally, we would have a fine enough sampling to interpolate γ and L for positions in between the transmission function measurements. However, with only four sample locations, this is not enough to enable a smooth and reliable interpolant. Instead, we will simply use the mean values of $\bar{L}$ and $\bar{\gamma}$ of each run collection to extract $\langle\beta\rangle$ from equation 8.16. This choice will naturally introduce some slowly varying bias into the resulting values of $\langle\beta\rangle$. However, unlike an interpolant, which is designed to match the known values of $\langle\beta\rangle$ exactly at the measured locations, this choice does allow us to form a crude estimate of the error. We can estimate the error by examining the difference between the ToF-extracted value of $\langle\beta\rangle$ and the value determined by the transmission function measurement. The error, σ_b, due to this model based bias is given for each run collection in table 8.8.

Description	σ_b (V)
Run Collection (A)	0.48
Run Collection (B)	0.11
Run Collection (C)	0.47

Table 8.8: The expected error in the reconstructed value of $\langle\beta\rangle$ for each run collection, due to the use of the mean values of $\bar{L}$ and $\bar{\gamma}$ in the model.

It is important to note that while the error, σ_b, is not a Gaussian random error, it is still independent of the error due to the ToF jitter. Therefore, in order to estimate the error on the extracted value of $\langle\beta\rangle$, we will simply sum in quadrature the uncertainty due to the ToF variance, σ_t, and the bias σ_b:

$$\sigma_\beta = \sqrt{\sigma_t^2 + \sigma_b^2} \, . \tag{8.17}$$

The extracted values of $\langle\beta\rangle$ are shown in figures 8-31, 8-33, and 8-35 for run collections (A), (B), and (C) respectively. The shaded region in these figures represents the error band about the extracted value of $\langle\beta\rangle$, and the points marked with a star are the analyzing potential values extracted directly from the transmission function measurements.

The value, $\langle\beta\rangle$, is the time-weighted average of the shift of analyzing potential with respect to the inner electrode within the analyzing region, given by:

$$\langle\beta\rangle = \left[\frac{1}{t_2 - t_1}\int_{t_1}^{t_2} U(\mathbf{x}(t))dt\right] - U_{\mathrm{IE}} \, , \tag{8.18}$$

where t_1 and t_2 are the times of the entry and exit of the electron through the analyzing region. In order to compare the value of $\langle\beta\rangle$ extracted from the ToF data to that computed from the field model, we need to examine a large collection of simulated electron tracks through the main spectrometer. Using a Runge-Kutta method to solve the equations of motion of a particle through the spectrometer produces a track which consists of some number, N, of time-ordered sample points along its path through phase space: $\{t_i, \mathbf{x}_i, \mathbf{p}_i\}$. From these sample points, we can compute the discrete approximation to equation 8.18 as follows:

$$\langle\beta\rangle = \left[\frac{1}{t_b - t_a}\sum_{i=a}^{b} U(\mathbf{x}_i)(t_{i+1} - t_i)\right] - U_{\text{IE}}\,, \tag{8.19}$$

where the step indices a and b are those associated with the particle entering and exiting the analyzing region respectively. In order to determine a and b, we rely on the mean value of the length, $\bar{L}$, extracted from the data. First, we compute the cumulative arc length of the particle's trajectory for each sample point $\ell_i = \ell(\mathbf{x}_i)$. Then, we locate the sample location, $\mathbf{x}_j$, which is closest to the mid-plane of the spectrometer $z = 0$. Any sample point, $\mathbf{x}_k$, associated with an arc-length, ℓ_k, that satisfies:

$$|\ell_k - \ell_j| < \bar{L}/2\,, \tag{8.20}$$

is within the analyzing region. Therefore, the values of the first and last indices in the sum, a and b, can be found by determining which two indices satisfy the following conditions:

$$\ell_j - \ell_a > -\bar{L}/2 \quad\text{and}\quad \ell_j - \ell_{a-1} < -\bar{L}/2\,, \tag{8.21}$$

$$\ell_j - \ell_b < \bar{L}/2 \quad\text{and}\quad \ell_j - \ell_{b+1} > \bar{L}/2\,. \tag{8.22}$$

Using these definitions and equation 8.19, we can compute $\langle\beta\rangle$ for every simulated particle track which impinges on the FPD, allowing us to construct $\langle\beta\rangle$ as a function over the FPD plane in a manner similar to that of the ToF interpolation. Then, to produce $\langle\beta\rangle$ as a function of the manipulator angle, ϕ, we sample the simulation generated interpolant along the oval beam path fit from the FPD hit data. We note that since we are using the same constant value of $\bar{L}$ to define the analyzing region, the simulated value of $\langle\beta\rangle$ should be biased in a manner similar to the extracted value. The simulated value of $\langle\beta\rangle$ at each measured angle is shown as a blue dashed line along with the data (black) for comparison in figures 8-31, 8-33, and 8-35. Generally we are not interested in the absolute value of the potential, but only the change as a function of azimuthal angle. Therefore, in order to compare the variation in $\langle\beta\rangle$ as a function of angle to the simulation, $\langle\beta\rangle$ has been uniformly shifted by an amount, s_β and plotted as the red line. The value of s_β was chosen to eliminate the average difference between the simulated value and the extracted value of $\langle\beta\rangle$ over all angles measured. This value is taken to be:

$$s_\beta = \frac{-1}{N_\phi}\sum_{i=1}^{N_\phi}(\langle\beta(\phi_i)\rangle_s - \langle\beta(\phi_i)\rangle_d) \ . \tag{8.23}$$

This shift represents our lack of knowledge about the true kinetic energy of the electron beam due to the non-zero spread in the initial photo-electron energy distribution, as well as the work function of the photo-cathode and main spectrometer electrode surfaces. The work function of the photo-cathode changes the starting energy of the electron from its expected value, while the work function of the main spectrometer modifies the potential seen by the electron beam during its flight. The value for s_β for each run collection is given in table 8.9.

Description	s_β (V)	σ_β
Run Collection (A)	-0.18	0.061
Run Collection (B)	-0.07	0.033
Run Collection (C)	-0.21	0.073

Table 8.9: The values of the mean and RMS spread of the residuals between simulated and extracted value of $\langle\beta\rangle$.

The residuals between the calculated (unshifted) value of $\langle\beta\rangle$ and the value of $\langle\beta\rangle$ extrapolated from the ToF measurement are shown in figures 8-32, 8-34, and 8-36. From the distribution of the residuals, we can see that the absolute difference between the calculated and measured potential is less than $\sim 0.2V$ and that the RMS spread is between 0.03 and 0.07 V. The values of the shift and RMS spread of the residuals are given in table 8.9.

In conclusion, we have made a measurement of the mean potential along the electron's flight path within the analyzing region of the KATRIN main spectrometer for several electric and magnetic configurations. In the process of reconciling the FPD hit pattern with the ToF variation as a function of angle, we have determined that there is a small (order of millimeters) displacement of the FPD relative to the center of the flux tube. The measurement also shows that the azimuthal variations in the ToF for the 3.8 Gauss field setting (which is one of the lowest magnetic field settings considered for the upcoming neutrino mass measurement) are primarily due to the flux tube misalignment relative to the main spectrometer, and (at least for all but the outermost two pixel rings) not due to the influence of the electrical short circuits. Furthermore, we have computed the expected value of the electric potential utilizing the HFFMM algorithm for field computation and made a direct comparison of the measured potential with the calculated value. This comparison has demonstrated that the field calculations have an absolute error on the order of several tenths of a volt and a relative error[6] of roughly 0.01-0.03%.

[6]Taking the relative error to be the mean value of the residual, s_β, divided by the inner electrode voltage, we have, in the worst case scenario: $s_\beta / U_{\mathrm{IE}} = 0.21/782.0 = 2.6 \times 10^{-4}$. However, assuming that a constant shift in the potential is irrelevant, then the relative error can be assumed to be dependent on the spread in the residual, which in the worst case is $\sigma_\beta / U_{\mathrm{IE}} = 0.073/782.0 = 9.3 \times 10^{-5}$.

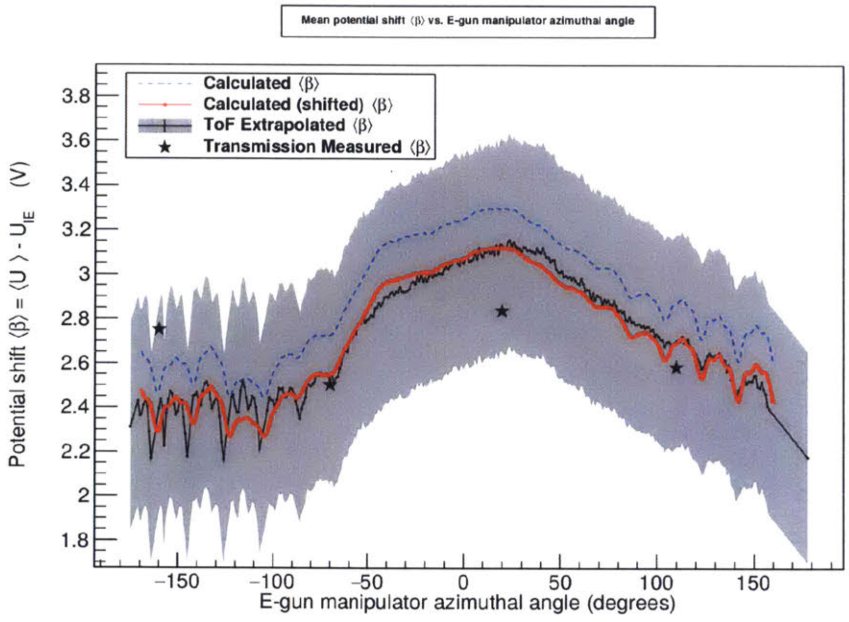


Figure 8-31: The mean analyzing potential variation for run collection (A). The ToF extrapolated data is black with a gray error band. The red line is the simulated $\langle \beta \rangle$, while the stars mark the locations of $\langle \beta \rangle$ measured from the transmission function runs.

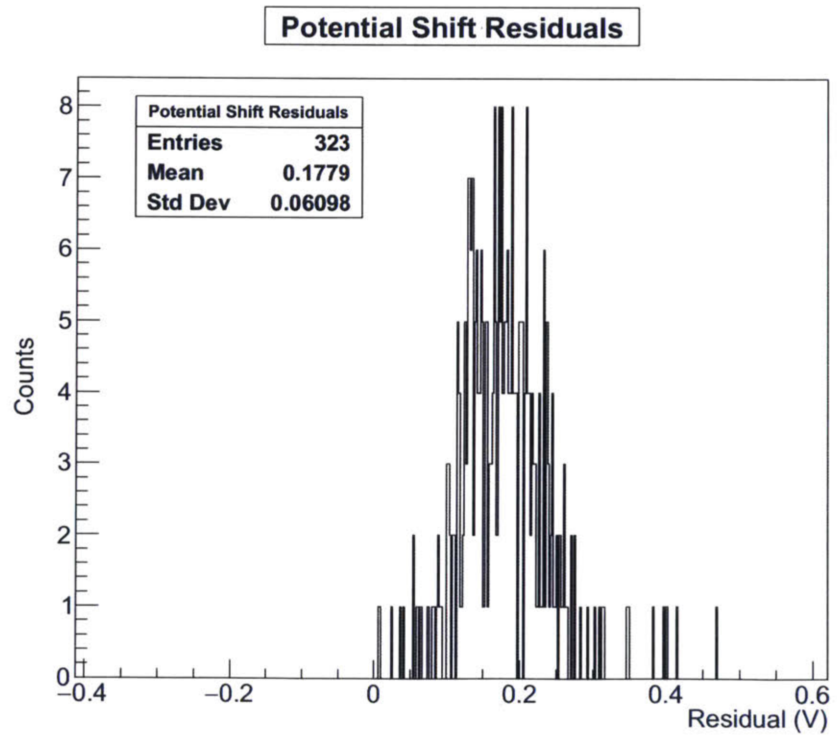


Figure 8-32: The residual error on the analyzing potential for run collection (A).

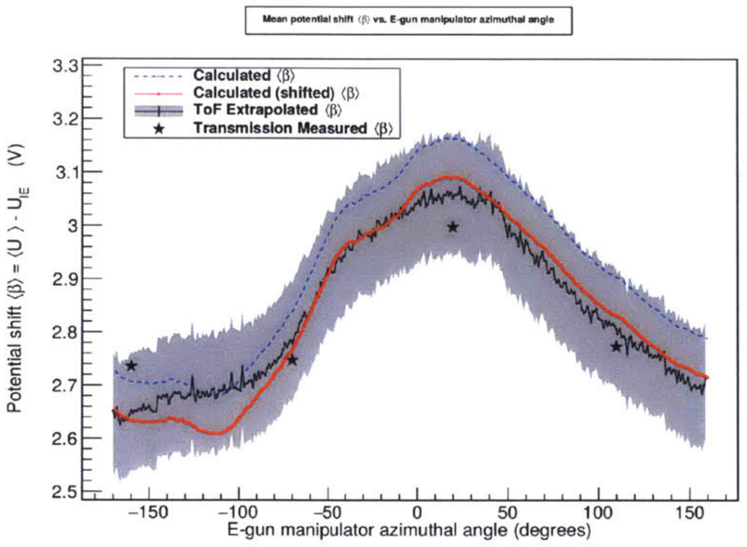


Figure 8-33: The mean analyzing potential variation for run collection (B). The ToF extrapolated data is black with a gray error band. The red line is the simulated $\langle \beta \rangle$, while the stars mark the locations of $\langle \beta \rangle$ measured from the transmission function runs.

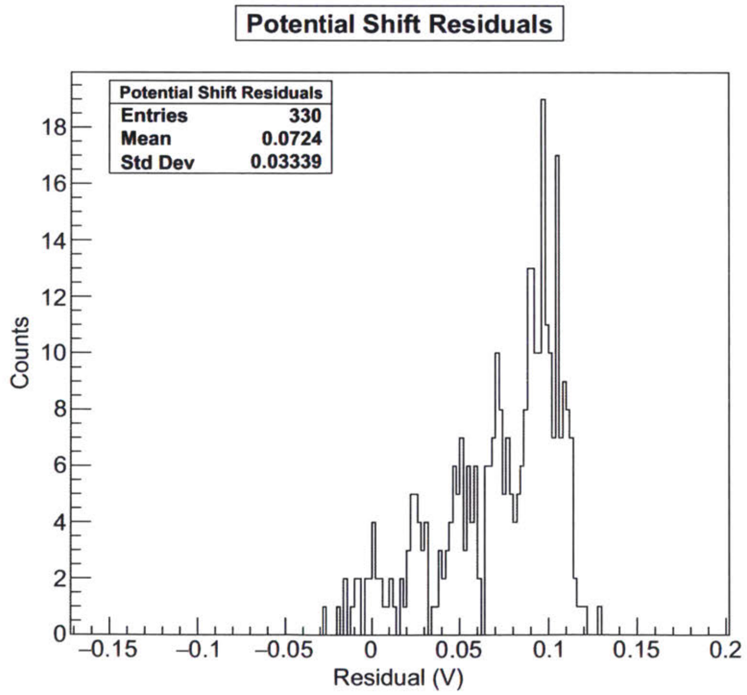


Figure 8-34: The residual error on the analyzing potential for run collection (B).

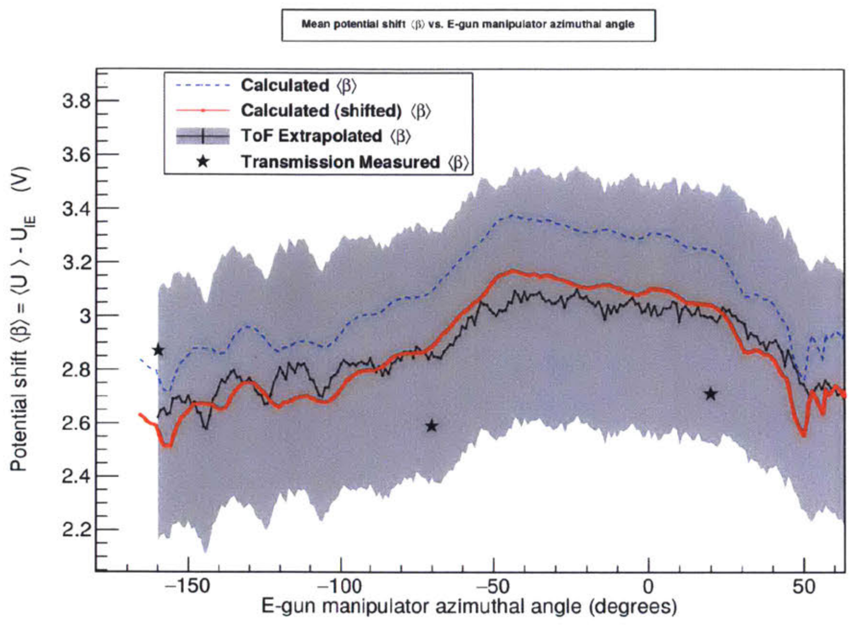


Figure 8-35: The mean analyzing potential variation for run collection (C). The ToF extrapolated data is black with a gray error band. The red line is the simulated $\langle \beta \rangle$, while the stars mark the locations of $\langle \beta \rangle$ measured from the transmission function runs.

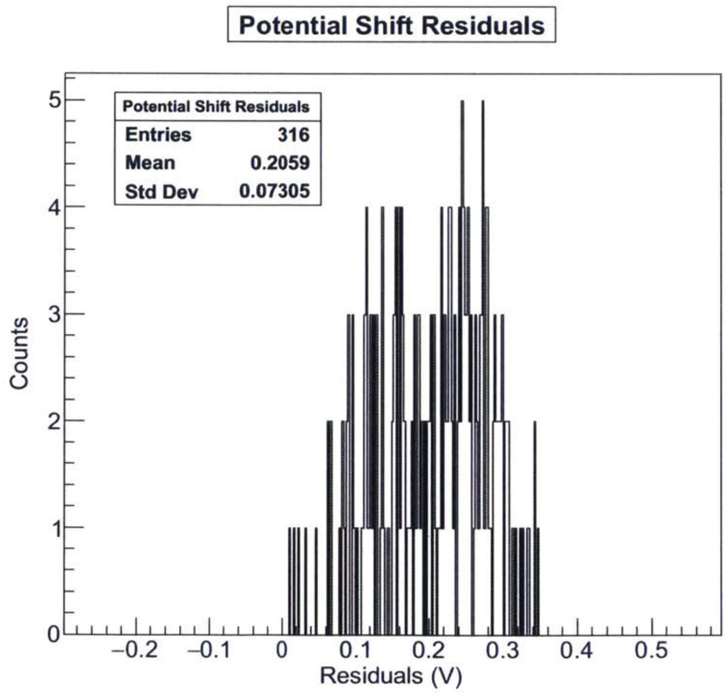


Figure 8-36: The residual error on the analyzing potential for run collection (C).

Chapter 9

Constraining the Systematic Effects of the Transmission Function Model on the KATRIN Neutrino Mass Measurement

9.1 Motivation and Observables

In order to extract a neutrino mass value from the count rate data of the KATRIN experiment, it will be necessary to fit the data with a model that makes a realistic representation of the true system. Constructing such a model requires us to make some assumptions about the behavior of each component of the experiment. These assumptions can be made directly through measurement, inference from simulation, or some combination of the two. However, whatever these assumptions are, there will inevitably be some uncertainty associated with them. The deviation between the true value of each parameter and the value assumed when constructing the experimental model will induce a systematic shift on the extracted value of the neutrino mass. These shifts are termed systematic errors and must be well constrained so their influence does not exceed the purely statistical error inherent in stochastic processes such as radioactive decay.

One important assumption which goes into the experimental model of the KATRIN experiment is how we represent the transmission function of the main spectrometer. In particular, we are interested in knowing how well we must constrain the spatial inhomogeneities of the transmission function, so that the systematic error induced by this effect is limited. The spatial inhomogeneities of the main spectrometer transmission function arise primarily from the variation of the electrical potential and magnetic field in the analyzing region. Therefore, constraining the uncertainty in the transmission function variation requires a detailed understanding of the electric and magnetic fields over the analyzing plane. However, for the purpose of this study, we will mainly concern ourselves with the variation in the electric potential and assume that any magnetic field inhomogeneities which may exist are purely axially symmetric.

The observables in the KATRIN experiment are rates, or rather, counts of electron events at the FPD as a function of the spectrometer fields and source configuration. The number of events seen by the i-th pixel of the FPD while the spectrometer is fixed at potential, U_j, is given by:

$$N_{ij} = t_j B_i + t_j A_i \int \left[\frac{d\Gamma(E, m_{\nu\beta}, E_0)}{dE} \times R_i(qU_j, E) \right] dE \,, \tag{9.1}$$

where t_j is the measurement time spent on potential U_j, E is the electron kinetic energy, B_i is the background rate for the pixel, A_i is a normalization constant representing the strength of the source imaged by the pixel, $d\Gamma/dE$ is the differential decay spectrum of equation 2.1, and R_i is the response function associated with the pixel.

Hidden in the response function, R_i, is all of the dependence on the source and spectrometer properties imaged by one pixel. R_i is given by a weighted, nested convolution of the transmission function with the inelastic scattering energy loss function $f(\epsilon)$, as follows:

$$R_i(qU_j, E) = P_0 R_{i,0}(qU_j, E) + P_1 R_{i,1}(qU_j, E) + P_2 R_{i,2}(qU_j, E) + \cdots \ , \tag{9.2}$$

where the n-th discrete scattering responses are:

$$\begin{aligned} R_{i,0}(qU_j, E) &= \int T_i^{\text{eff}}(qU_j, E - \epsilon)\delta(\epsilon) d\epsilon \ , \\ R_{i,1}(qU_j, E) &= \int R_{i,0}(qU_j, E - \epsilon) f(\epsilon) d\epsilon \ , \\ R_{i,2}(qU_j, E) &= \int R_{i,1}(qU_j, E - \epsilon) f(\epsilon) d\epsilon \ , \\ &\vdots \end{aligned} \tag{9.3}$$

and the weights, P_n, are the average probability of an electron undergoing n inelastic scattering events as it transits the source. The true picture is of course somewhat more complicated, due to the non-uniform gas density of the source, as well as the scattering probability's dependence on pitch angle, but these details are beyond our scope and are discussed in detail in [120, 134, 110]. The energy loss function, $f(\epsilon)$, must be determined through a dedicated measurement phase at KATRIN in order to meet the demanding goals of the design report [51]. It has, however, been measured previously for T_2 at the Troitsk experiment, so for the purpose of this study, we use the energy loss model of Aseev et al.[16], which is shown in figure 9-1.

In order to compute the response function of equation 9.3, the KATRIN experiment also needs to resolve the effective transmission function, $T_i^{\text{eff}}(qU_j, E)$, for each pixel of the main spectrometer. For an isotropic source along a single field line, the transmission function of the spectrometer is well approximated by the analytic

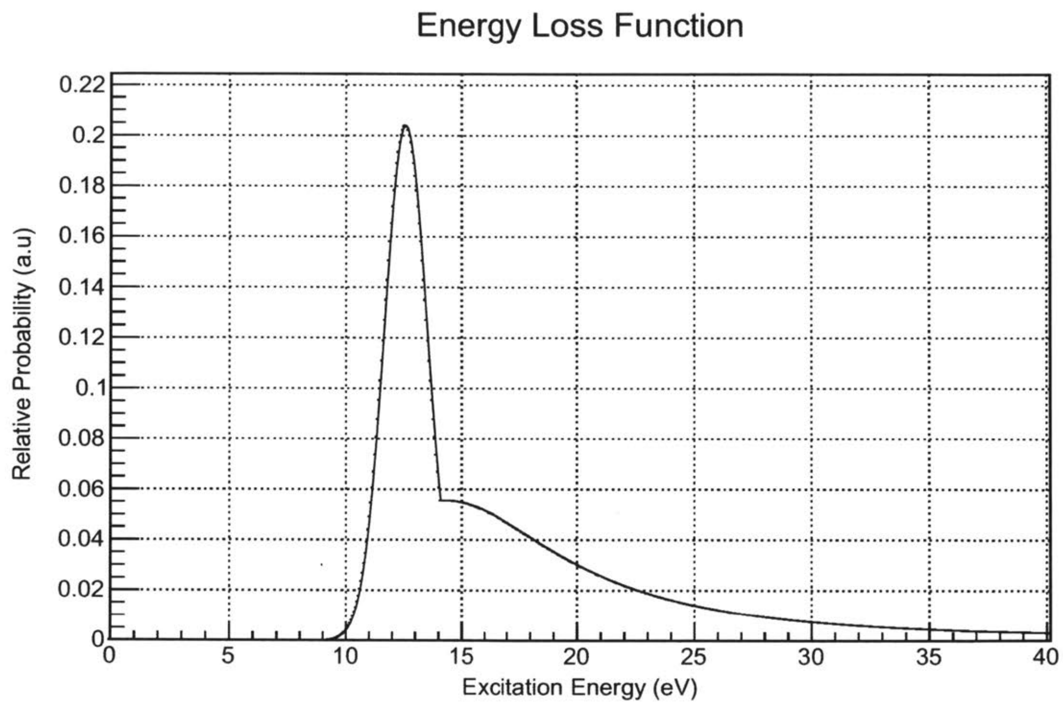


Figure 9-1: The energy loss function, $f(\epsilon)$, due to inelastic scattering in the T_2 source [16].

form of equation 2.9, with relativistic corrections, it becomes [110]:

$$T(E,qU) = \begin{cases} 0 & \text{if } E - qU \leq 0\,, \\ \dfrac{1-\sqrt{1-\frac{E-qU}{E}\cdot\frac{B_S}{B_a}\cdot\frac{(\gamma_a+1)}{(\gamma_S+1)}}}{1-\sqrt{1-\frac{\Delta E}{E}\cdot\frac{B_S}{B_a}}} & \text{if } 0 \leq E - qU \leq \Delta E\,, \\ 1 & \text{if } E - qU > \Delta E\,. \end{cases} \tag{9.4}$$

Further corrections also must be made for synchrotron radiation losses, Doppler spreading, and inhomogeneities in the source region magnetic field. However, for the sake of simplicity, we will not include these other effects in this study.

It should be stressed that the effective transmission function imaged by one pixel, T_i^{eff}, is not equivalent to the single-field line approximation of equation 9.4. This is due to the spatial variation of the analyzing potential and the magnetic field across the spectrometer. This modifies the width and the location of the edge of the transmission function as a function of spatial position. If no effort were made to resolve the spatial location of an electron event at the FPD, this effect would severely degrade the energy resolution of the spectrometer. Of course, the designed segmentation of the FPD provides a means of compensating for this effect, since each pixel is exposed to a small portion of the flux tube over which these variations are much less significant. Nevertheless, since the pixels of the FPD do have a finite extent, there is still some irreducible smearing of the transmission function as viewed by each pixel due to the field inhomogeneities.

Under the adiabatic approximation, the effective transmission function for each pixel can be computed as a flux-source-density weighted average of the analytic transmission function (eq. 9.4) integrated over the pixel area, Δ_i. For the i-th pixel, this is given by:

$$T_i^{\text{eff}}(qU,E) = \frac{\int\limits_{\Delta_i} T(qU,E,r,\phi)\rho(r,\phi)\mathbf{B}(r,\phi)\cdot d\mathbf{A}}{\int\limits_{\Delta_i} \rho(r,\phi)\mathbf{B}(r,\phi)\cdot d\mathbf{A}}\,, \tag{9.5}$$

where $\rho(r,\phi)$ is the column density of the tritium source imaged by the detector, $\mathbf{B}(r,\phi)$ is the magnetic field at the detector face, and $T(qU,E,r,\phi)$ is the transmission function (eq. 9.4) associated with the single-field line specified by the coordinates (r,ϕ) at the FPD face. To very good approximation, both $\rho(r,\phi)$ and $\mathbf{B}(r,\phi)$ are essentially constant over the area of one pixel, so equation 9.5 reduces to a simple area weighted average:

$$T_i^{\text{eff}}(qU,E) = \frac{1}{|\Delta_i|}\int\limits_{r_{i,1}}^{r_{i,2}}\int\limits_{\phi_{i,1}}^{\phi_{i,2}} T(qU,E,r,\phi)r\,dr\,d\phi\,. \tag{9.6}$$

The evaluation of the above expression requires us to know the single-field line transmission function (of equation 2.9) and consequently, the analyzing potential, $U_A(r,\phi)$, and analyzing magnetic field strength, $B_A(r,\phi)$, imaged onto the FPD everywhere within one pixel. Clearly, it is not realistic to measure these quantities directly everywhere. However, the transmission function broadening can be explored indirectly through the use of the three-dimensional electric field model (validated though the measurements described in the last chapter) in conjunction with a Monte Carlo particle tracking simulation provided by the `Kassiopeia` package.

We will use this Monte Carlo simulation in order to answer several questions about the transmission function exposed to each pixel and the experiment as a whole. These are as follows:

1. What is the distribution of the analyzing potential viewed by each pixel?
2. How is the transmission function modified by the finite extent of each pixel?
3. What is the systematic error induced by treating only the FPD rings distinctly, or ignoring the pixel-by-pixel variation altogether?
4. Is the transmission function variation among pixels within one ring small enough that it can be ignored? In other words, can the spectrometer be modeled as completely axially symmetric?
5. How well does the analyzing potential need to be measured so as to reconstruct the spatial variation of the transmission function? To what level can the systematic error, due to the transmission function spatial inhomogeneities, be limited by measurement?

9.2 Calculation of the Pixel-wise Effective Transmission Functions

In order to generate a reliable model of the transmission function seen by each pixel, we need to generate a large sample of electron tracks passing through the main spectrometer. A large sample is needed so that statistical fluctuations do not overwhelm the effects we are seeking. Somewhat similar studies have been done previously, using an axially symmetric model of the main spectrometer electric and magnetic fields [70, 110]. However, until the development and implementation of the fast multipole technique described in this thesis, using a fully three dimensional model of the electric field was not computationally feasible. This was due to the slowness of the direct evaluation. Since the direct method requires $\sim 0.1 - 1$ seconds to evaluate the electric field and several thousand evaluations ($> 10^4$) are needed for each particle track, a simulation of $\sim 10^6$ particles would require several CPU-centuries to complete. The HFFM method reduces this time by roughly a factor of 6000-7000 and makes the simulation of large particle sets viable.

To generate the events used to study the pixel-dependence of the transmission function, the fully three dimensional BEM model of the main spectrometer as described in [53] was used in conjunction with the HFFM method to compute the electric fields. The BEM mesh was also used for navigation in order to resolve particle collisions with the spectrometer walls. The Laplace BVP was solved using the super-position solver (described in section 8.6) with the electrode potential values as described in table 9.1. This potential setting was chosen from several standard

Vessel Potential (V)	Wire Array Offset (V)	Steep Cone Offset (V)
-18400	-200	+100

Table 9.1: Main spectrometer electrode configuration. Here the hull is set to -18.4kV, the inner electrode to -18.6kV, and the steep cone electrodes to -18.5kV. Both wire layers (where they exist) are set to the same potential.

configurations used during the SDS series of commissioning measurements because it reflects a worst-case "shorted" configuration of the main spectrometer. If the electrical short circuits between the wire layers are repaired before tritium data taking starts, the uniformity and axial symmetry of the electrical potential viewed by the detector should be improved. However, since this may not be the case, it is important to understand what effect this deviation from the original design may have on the neutrino mass extraction. Once the charge densities were obtained, a HFFM field map was constructed according to the parameters of table 9.2. Tracking was performed using the adiabatic approximation with an 8-th order embedded Runge-Kutta method. The estimated local error on the position and momentum of the particle were limited to 50 nm and 0.018 eV/c respectively, so as to provide a reasonable trade off between the speed and accuracy of the simulation.

d_t	d	z	p	η	$n_{\max}$
16	3	1	13	4/3	5

Table 9.2: Parameters used by the HFFM method to map the field of the three-dimensional main spectrometer for particle tracking studies.

Since for the purpose of this study we are primarily concerned with the azimuthal inhomogeneities of the electric potential, the magnetic field of the main spectrometer was treated using the axial symmetric approximation. The magnet positions of the full experiment beam-line model and optimized current values, as laid out in table B.7 of [110], were used to model the magnetic field using the 3 Gauss settings. The optimized settings for a higher (6 Gauss) analyzing magnetic field were also modeled using the same current and position values for the magnets, with the obvious exception of the LFCS coils, the currents of which are given in table 2 of [179]. We will refer to these magnetic field configurations as the 3 Gauss and 6 Gauss settings respectively. The EMCS was configured so as to perfectly compensate the transverse components of the Earth's magnetic field.

A total of 3.2×10^6 electrons were tracked through the spectrometer for each magnetic field setting, starting from the mid-plane of the first pre-spectrometer magnet (z=-12.1) and ending at the detector plane (z=13.93). Their starting positions were uniformly distributed over the flux tube, up to a radius of 0.04 m. Their starting momentum was isotropically distributed up to 70° with respect to the magnetic field direction. This was done in order to capture all particle behavior up to the cut-off angle (which corresponds to 66° in the pre-spectrometer magnet).

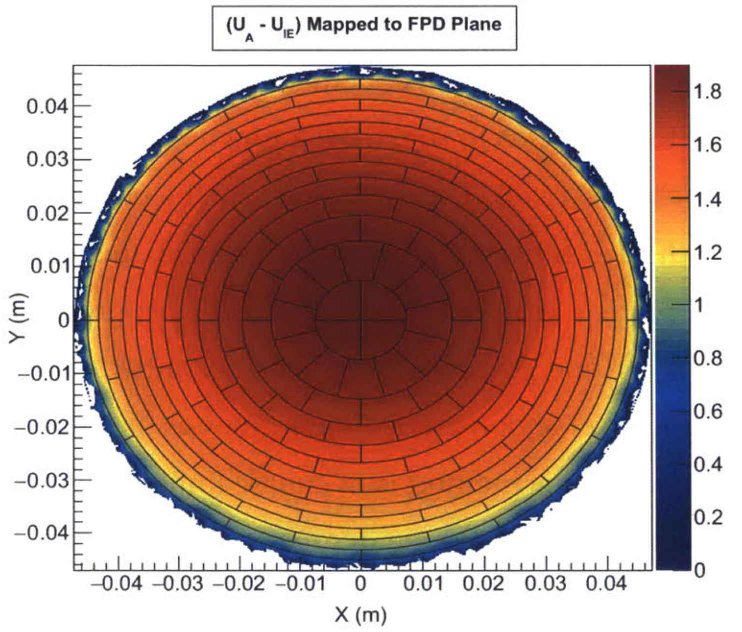
9.3 Description of Transmission Function Model

Using the Monte Carlo data set, we can construct a model of the transmission function imaged by each pixel of the FPD in two ways.

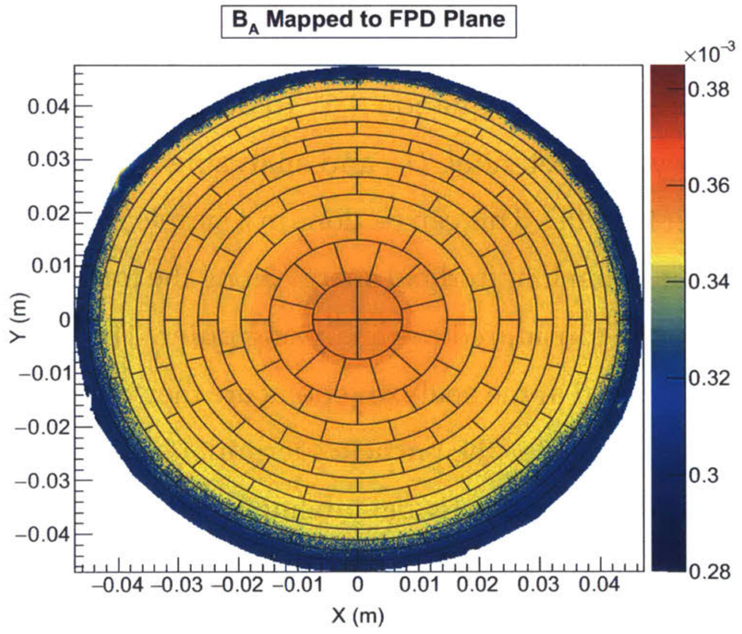
The first method is quite straightforward. We simply count the number of particles in each energy bin that either reach the detector or are electrostatically reflected in the volume of the spectrometer[1]. Clearly, determining the pixel associated with a particle that has reached the detector face is a simple matter. However, to compute the transmission probability, we also need to make a pixel assignment for those particles which were reflected and never actually reached the detector. To do this, we use the transmitted particles to construct a function which maps the starting plane location to each FPD pixel. Then, this map can be used to look up the FPD pixel associated with each reflected particle. In essence, this map represents the pixel each reflected particle would have hit, if it had been provided with enough energy. Constructing the energy-binned ratio of transmitted to reflected particles emitted from the same pixel region immediately gives us the effective transmission function. The accuracy of this method is of course limited by the statistics of the Monte Carlo simulation, as well as the degree to which energy conservation is violated by the finite numerical precision with which the equations of motion are solved. It best serves as a means by which we can evaluate the self-consistency of the Monte Carlo simulation with the second method.

The second method is to perform the calculation of the area-averaged transmission function directly from equation 9.6. To do this, we must first construct the value of the analyzing potential, U_a, and analyzing magnetic field, B_a, that is mapped to the face of the FPD. This can be done in a simple manner by determining the location of the minimum longitudinal energy along the track of each transmitted particle. This location is considered the analyzing point. The value of the electric potential and magnetic field at the analyzing point are then taken to be the values of $U_a(r, \phi)$ and $B_a(r, \phi)$ mapped to the particle's terminal location, (r, ϕ) on the FPD. Figure 9-2a shows the analyzing potential inhomogeneity, $(U_A - U_{IE})$, of the main spectrometer mapped to the FPD face for the 3 Gauss magnetic field configuration. Figure 9-2b shows the corresponding analyzing magnetic field inhomogeneity. The potential and magnetic field inhomogeneities for the 6 Gauss setting are shown in figure 9-3.

[1]Particles which are magnetically reflected at the pitch magnet only affect the overall normalization and are ignored for the purpose of computing the transmission probability.

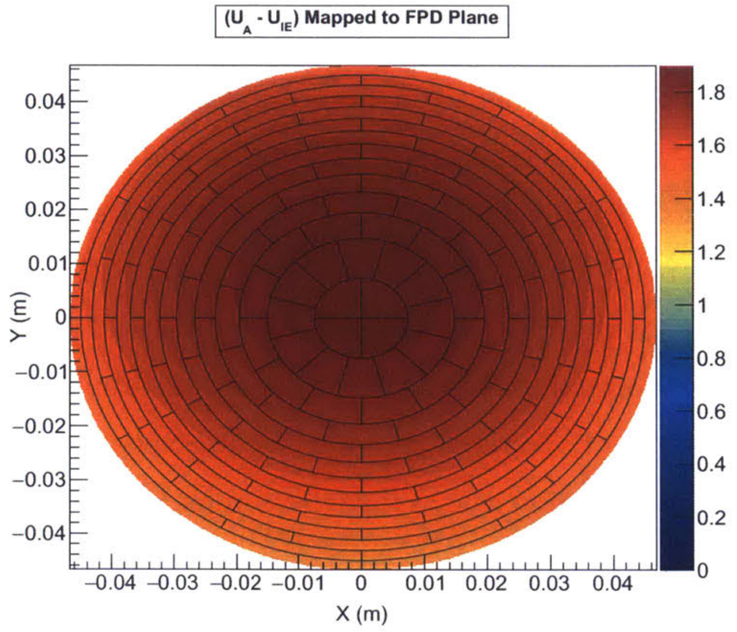


(a) Analyzing potential inhomogeneity, $(U_A - U_{IE})$. The color axis units are volts (V).

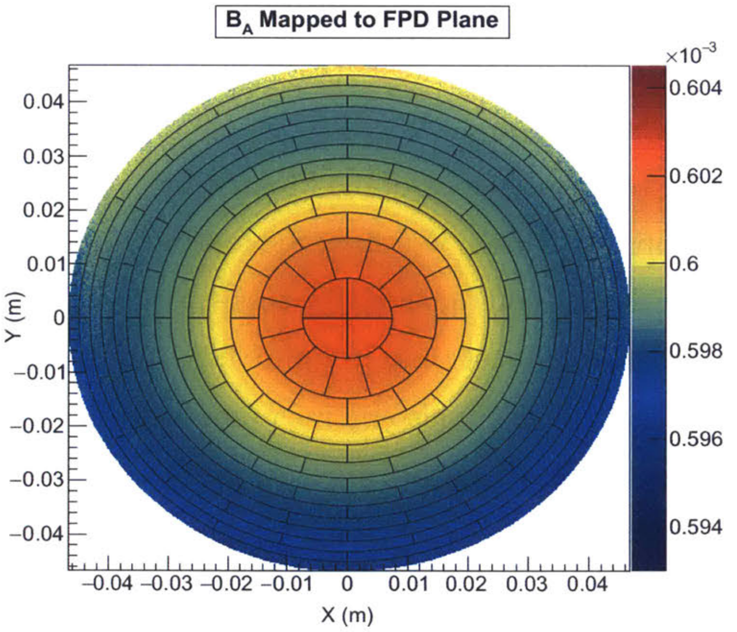


(b) Analyzing magnetic field, $|B_A|$. The color axis units are Tesla (T).

Figure 9-2: The analyzing potential and magnetic inhomogeneities mapped by particle tracks to the FPD for the 3 Gauss field setting.



(a) Analyzing potential inhomogeneity, $(U_A - U_{IE})$. The color axis units are volts (V).



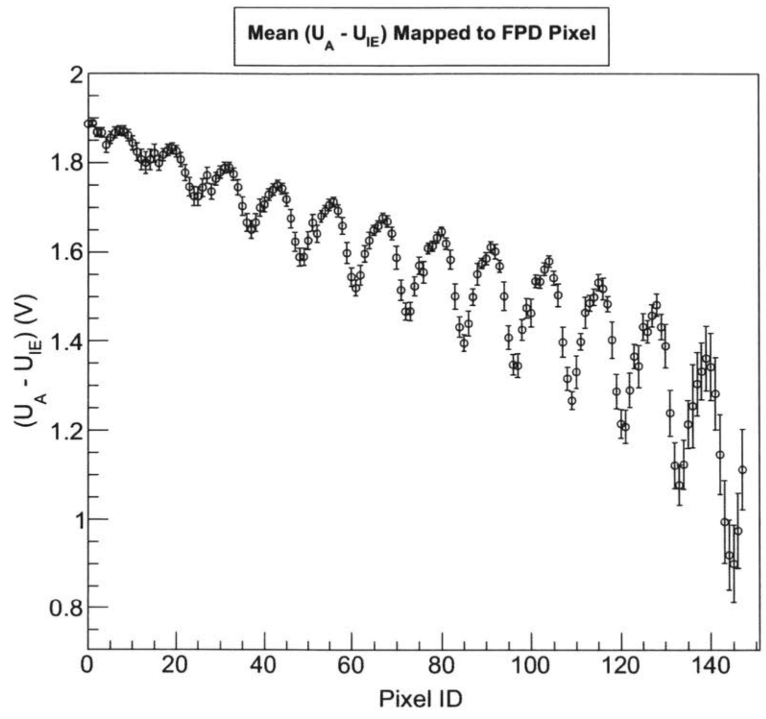
(b) Analyzing magnetic field, $|B_A|$. The color axis units are Tesla (T).

Figure 9-3: The analyzing potential and magnetic inhomogeneities mapped by particle tracks to the FPD for the 6 Gauss field setting.

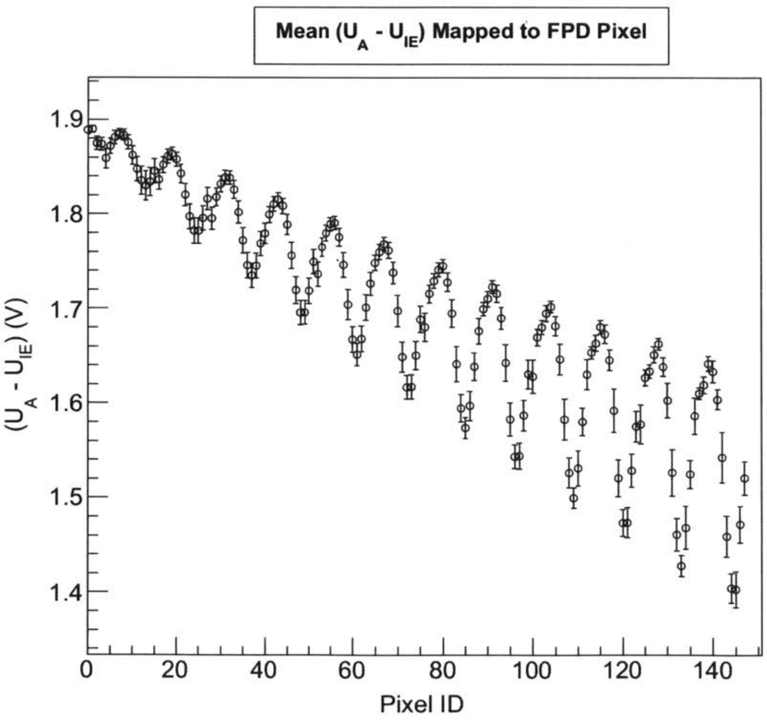
From figures 9-2 and 9-3, it is readily apparent that there is a vertical asymmetry in the analyzing potential. To see how this affects the analyzing potentials imaged to pixels within one ring, it is helpful to plot the mean analyzing potential inhomogeneity for each pixel as a function of the pixel ID. The mean potential inhomogeneity for each pixel is shown in figures 9-4a and 9-4b for the 3 Gauss and 6 Gauss field settings respectively. The error bars represent the RMS variation of the potential inhomogeneity within each pixel. The variation within each ring (visible as large oscillations) is due primarily to the overall vertical asymmetry and is a result of the vessel deformation. The RMS spread within individual pixels is mostly due to their radial extent. Some of the outermost pixels also suffer from some additional azimuthal variation due to the wire combs, but this is a sub-dominant effect.

Once we have constructed the maps of the analyzing potential and magnetic field, we can then proceed with the calculation of the effective transmission function for each pixel. To do this, we compute equation 9.6 numerically using 32×32 Gauss-Legendre quadrature over the area of each pixel in uniform 3meV steps from 4 volts below to 2 volts above U_{IE}.

As a check of the self-consistency of the Monte-Carlo with the area-weighted transmission function calculation, we show the results of both methods by computing the effective transmission function (3 Gauss setting) for the inner-most ring (bullseye) and for the outer-most ring, in figure 9-5. The Monte Carlo result is primarily limited by statistics, but also by the numerical accuracy of the particle tracking which causes small energy violations on the order of 0.05 eV between the start and end of a track. It should be noted that the accuracy of the particle tracking is mainly limited by the choice of step size for the numerical integrator, and not by the field calculation. Therefore, the energy violation could in principle be made negligible, but at the cost of more integration steps, resulting in a slower run time and worse statistics. However, in light of the comparison between the two methods, it appears that the results are consistent to within the Monte Carlo error. Therefore, it appears appropriate to use the simpler method of equation 9.6 to calculate the



(a) 3 Gauss setting.



(b) 6 Gauss setting.

Figure 9-4: The mean potential inhomogeneity for each pixel of the FPD, labeled by pixel ID. Variation within rings is clearly evident.

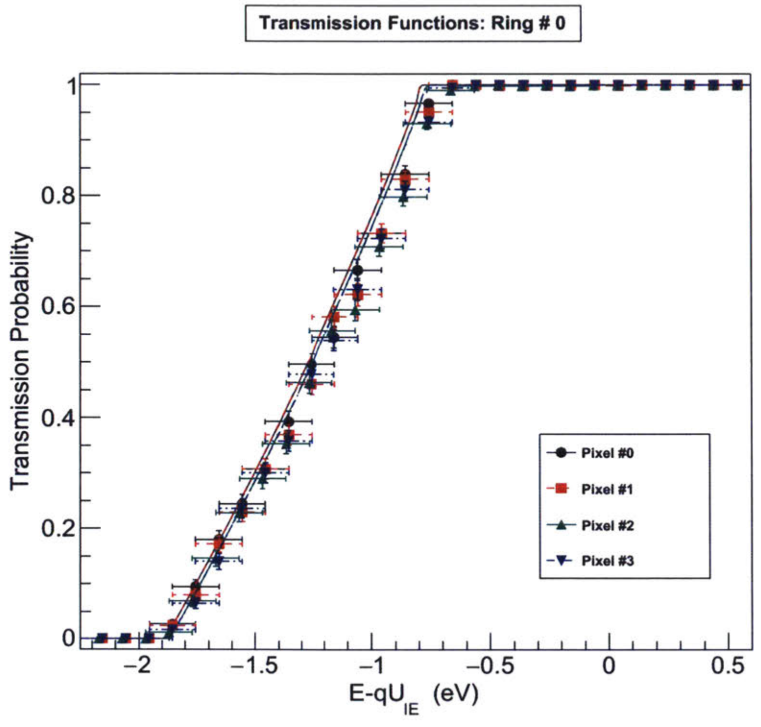
effective transmission function of each pixel.

It is also informative to calculate the effective transmission function for the entire detector as a whole, as well as for each ring. Comparing these with the pixel resolved approach demonstrates the effectiveness of the detector pixelation in improving the energy resolution of the spectrometer. Figure 9-6 shows the effective transmission function (for the 3 Gauss setting) for the entire detector and resolved into individual rings and pixels. From this figure, we can see that the radial and azimuthal inhomogeneities severely degrade the energy resolution of the spectrometer if no segmentation or only ring based segmentation were to be used. However, nearly the full $\sim$ 1 eV resolution is recovered when using individual pixel-by-pixel segmentation.

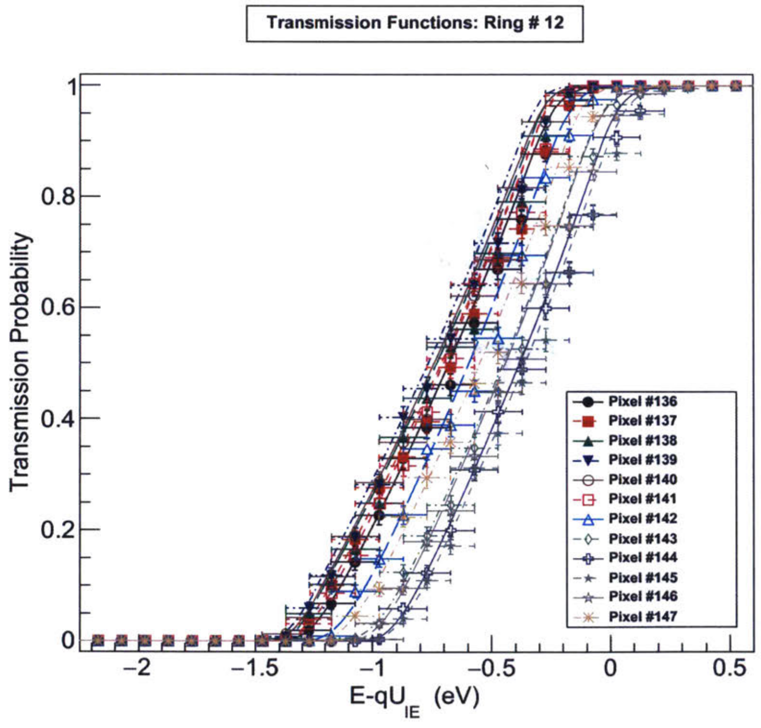
9.4 Method of Ensemble Tests

Now that we are able to calculate the effective transmission function for each pixel, we need to evaluate the effect that our choice of transmission function model has on the extracted neutrino mass. To do this, we have chosen to use the method ensemble tests [51, 110, 140] because of its ease of implementation and straightforward interpretation. This is done by constructing pseudo-data from a number of toy experiments and fitting the pseudo-data with a model of our choosing (not necessarily the same as the model used to generate the pseudo-data). To generate a toy experiment, we start by choosing a set of parameters representing the KATRIN experiment which determine the rate of electrons leaving the source and reaching the detector according to our understanding of equation 9.1. After computing the expected number of counts, N_{ij}, for each detector pixel, i, and potential setting, U_j, we then draw a random value, $\widetilde{N}_{ij}$, from a Gaussian distribution[2] with a mean $\mu = N_{ij}$ and width $\sigma_{ij} = \sqrt{N_{ij}}$. This set of randomized counts $\{\widetilde{N}_{ij}\}$ is the pseudo-data, which

[2]Radioactive decay is of course a Poisson process. Therefore, the spread on the measured number of counts, N, collected by the KATRIN detector is expected to follow a Poisson distribution. However, when the mean number of events is large enough, this reduces to a Gaussian distribution with a mean $\mu = N$ and width $\sigma = \sqrt{N}$. This is the method adopted in the KATRIN design report [51].



(a) Pixels of the inner bullseye.



(b) Pixels of outer ring (12).

Figure 9-5: Comparison of the two methods to compute the effective transmission function for the pixels in the bullseye and outer ring for the 3 Gauss setting. The lines are the effective transmission computed by equation 9.6, while the marked points were computed from the Monte-Carlo ratio of counts.

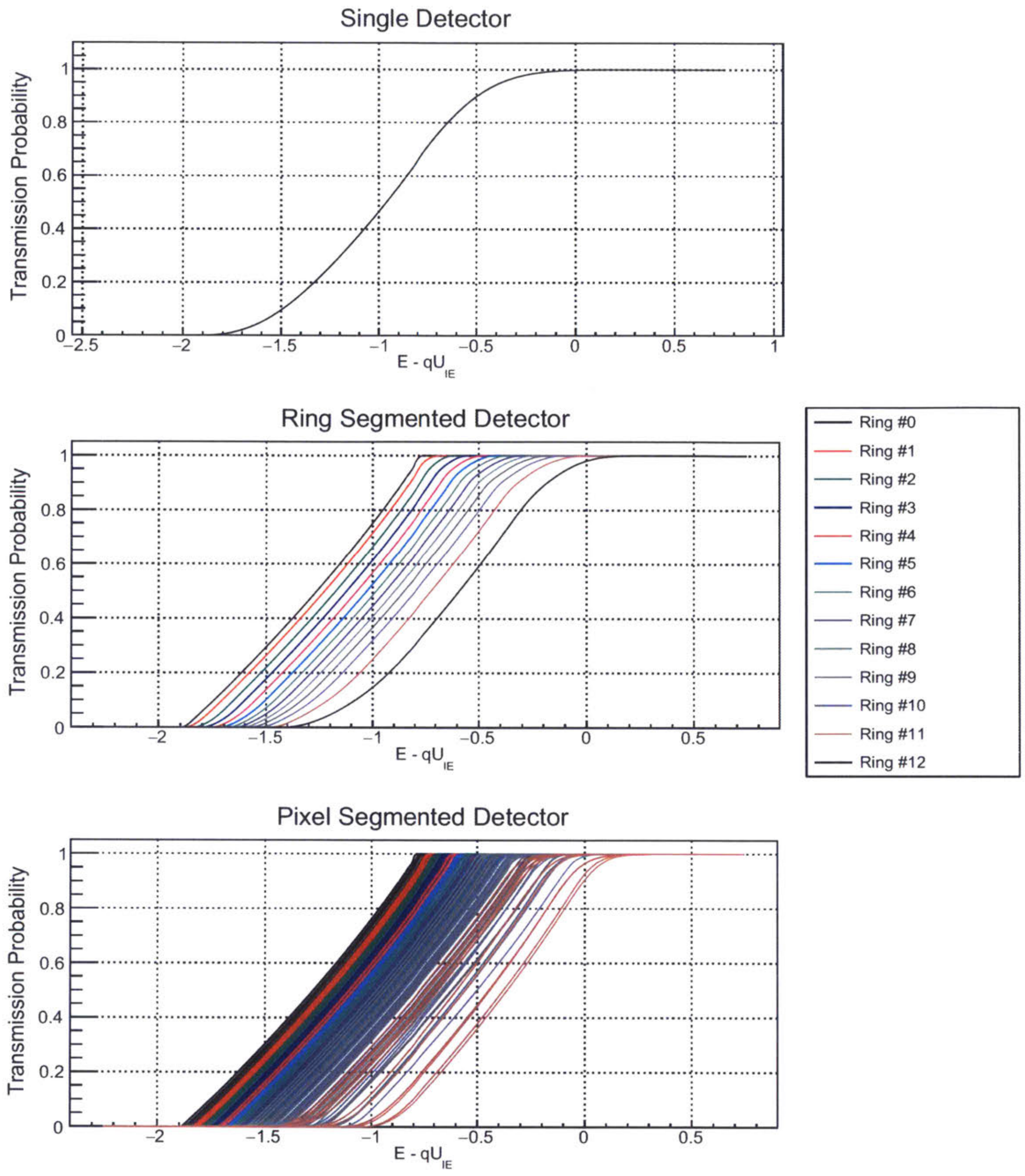


Figure 9-6: The effective transmission function of the FPD under various segmentation regimes for the 3 Gauss field setting.

represents a hypothetically possible measurement, based on our understanding of the experiment. The parameters used to describe the spectrum-response model when generating the pseudo-data, $\{\widetilde{N}_{ij}\}$, are the Monte Carlo "truth" values which we intend to extract. Repeating the pseudo-data generation and subsequent fit many times builds up a distribution of possible measurement values. The spread in the distribution of fit values obtained from the pseudo-data provides an estimate of the statistical power of a measurement, while any deviation of the mean away from the true value (used to construct the pseudo-data) represents a systematic error associated with the model used when fitting the pseudo-data.

To extract the parameters of interest, (E_0, m_{ν_β}), along with the so-called nuisance parameters[3] (A, B) from the randomized pseudo-data, we need to construct an objective function. A simple choice is the χ^2 metric which is given by:

$$\chi^2 = \sum_i^{M_p} \sum_j^{M_u} \left(\frac{\widetilde{N}_{ij} - N_{ij}(A, B, E_0, m^2_{\nu_\beta})}{\sigma_{ij}} \right)^2 . \tag{9.7}$$

This summation is rather expensive to compute, especially when we consider that for the standard measurement program the number of potentials is $M_u \sim 40$, and the number of pixels is $M_p = 148$. This results in a total of 5920 terms in the sum. Of course, this is not a particularly large number. However, it is important to remember that the objective function must be evaluated many times (hundreds) in order to explore the parameter space, and that the evaluation of each term in the series must be computed using high order numerical integration[4].

Evaluating equation 9.1 requires a model of the differential β-decay spectrum for molecular tritium. However, the differential spectrum was calculated using the simple spectrum of equation 1.31 instead of the full model of equation 2.1, so as to avoid the full summation over all the possible molecular final states. Since the theoretical discrete final state model of Saenz et al. [206] consists of 155 discrete terms, this implies that a single evaluation of the fully pixelated objective function would require over a million evaluations of the simple differential β-decay spectrum of equation 1.31. Therefore, this feature of the differential spectrum was not modeled. Although it will be important to include the final states in the final

[3]For simplicity we treat the background rate, B, and source activity, A, as uniform over all pixels. Though in principle these parameters could vary pixel-to-pixel.

[4]The integration method we chose to evaluate equation 9.1 is 256^{th}-order Gauss-Legendre quadrature.

neutrino mass extraction, since their effect is independent of the response function, the influence of the final states is not expected to modify systematic effects caused by the transmission function model.

In addition to the differential spectrum model, we also need a model for the response of each pixel. As mentioned before, this requires the energy loss function $f(\epsilon)$, the scattering probabilities, P_n, and the effective transmission function of each pixel (as calculated in the previous section). Since the purpose of this study is to investigate the pixel-wise effective transmission function, the energy loss function and scattering probabilities were assumed to be known completely and were calculated using the model of Aseev et al. [16] using the scattering probabilities of table 9.3 [140]. The convolution of equation 9.3 was calculated for up to four scatterings, by discretely sampling the energy loss function and effective transmission function every 3meV. The discrete samples were then convolved in an accelerated way through the use of the convolution theorem and a FFT. Evaluation of the response function during the integration of equation 9.1 was done by linearly interpolating between the set of discrete sample points. The resulting response function for all 148 pixels, colored by their respective ring, is shown in figure 9-7. The individual pixel response functions are difficult to resolve by eye on such a large energy scale, but the spreading is readily apparent.

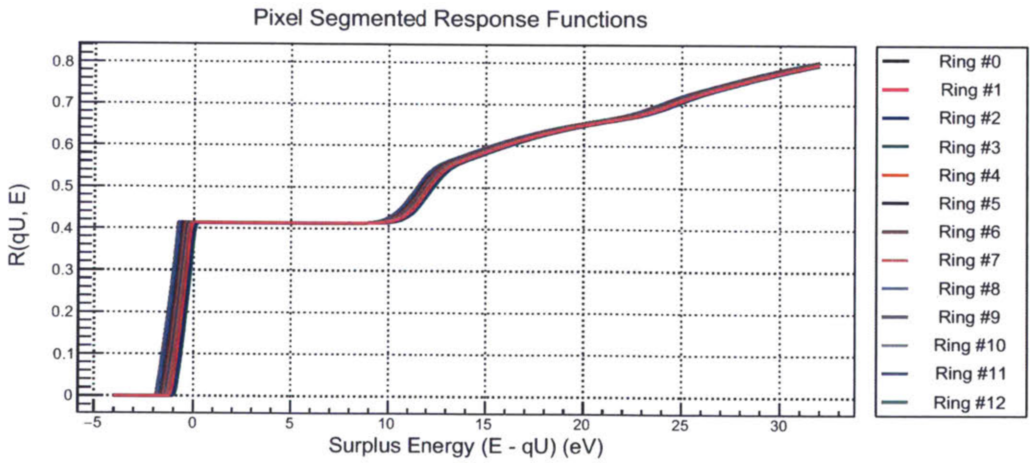


Figure 9-7: The pixel segmented response functions.

The other parameters describing the KATRIN model, along with the scattering probabilities that were used for performing the ensemble tests, are given in table 9.3.

Name	Value
End-point, E_0	18575 eV
Neutrino mass squared, $m^2_{\nu_\beta}$	0
Detector efficiency, ϵ_{FPD}	0.9
Background rate, B_r	0.01 Hz
Energy loss model	Aseev et al. [16]
Scattering probabilities [140]	$P_0 = 0.413$, $P_1 = 0.293$, $P_2 = 0.167$ $P_3 = 0.079$, $P_4 = 0.032$
Column density, ρ	5×10^{17} cm^{-2}
T_2 fraction f_{T_2}	0.95
Analyzing plane area, A_p	63.6 m^2
Source magnetic field, B_s	3.6 T
Pinch magnetic field, $B_{\max}$	6.0
Measurement Range[5]	$E_0 - 30$ to $E_0 + 5$ eV
Measurement Time Distribution	3 years, design report [51]

Table 9.3: KATRIN parameters used for the ensemble tests.

9.4.1 Evaluation of Transmission Model Induced Systematic Shifts

In order to evaluate how a particular choice of transmission function model may affect the value of the extracted neutrino mass, we first need to decide upon a Monte-Carlo "truth". For this, we will use the fully pixelated set of effective transmission functions obtained from calculating equation 9.6 using the particle tracking simulation data. Then, we will fit the pseudo-data generated from the "truth" model using values of N_{ij} calculated with other representations of the spectrometer transmission function.

The alternative transmission function models we will consider are somewhat naive, but represent simple assumptions based on what we can obtain (the analyzing potential along a single field line) with limited electron gun measurements and no further assumptions. These alternative models are as follows:

[5]The measurement voltages were adjusted upwards by 1.7 V to compensate for $U_A \neq U_{IE}$.

1. An ideal single field line transmission function (eq 9.4) based on the value of U_A and B_A evaluated at the central radius of each ring at $x = 0$. This assumes perfect axial symmetry and generally accounts for the overall radial inhomogeneity but ignores the radial inhomogeneity within one ring.

2. An ideal single field line transmission function (eq 9.4) based on the value of U_A and B_A evaluated at the center of each pixel. This makes no assumption about symmetry but ignores the inhomogeneity within a single pixel.

3. A smeared, radially averaged (at $x = 0$) transmission function (eq 9.6, but with $\Delta\phi = 0$) for each ring. This assumes perfect axial symmetry and accounts for the radial inhomogeneity within one ring but ignores azimuthal variations.

Each ensemble test consists of 10^4 pseudo-data collections and a subsequent fit. The results of a single test are shown in figure 9-8 (test of the 3 Gauss setting, model #3), which demonstrates the distribution generated from the fitted values of $m^2_{\nu_\beta}$. The width of the distribution represents the statistical sensitivity, while the deviation from zero represents the systematic shift induced by our choice of transmission function model (in this case, model # 3). The results of the ensemble tests for models #1-3 are shown for both the 3 Gauss and the 6 Gauss field settings in table 9.4.

From table 9.4, it is evident that the effect of making an axially symmetric approximation when modeling the main spectrometer transmission function is smaller for the 6 Gauss field setting than for the 3 Gauss setting. This comes as no surprise since the potential inhomogeneity is much reduced in this case. It is also clear that the models which ignore inter-pixel/ring variation of the transmission function are not satisfactory at all. However, the use of the simple axially-symmetric, radially-smeared model of the transmission function (model #3) still results in a systematic shift of -6.3×10^{-3} eV^2. Since the designed limit on the quadrature summed systematic errors of the KATRIN $m^2_{\nu_\beta}$ measurement is 17×10^{-3} eV^2, this represents an additional 6.6% contribution to the total systematic error which was not envisioned in the design document [51]. It should be pointed out that this is

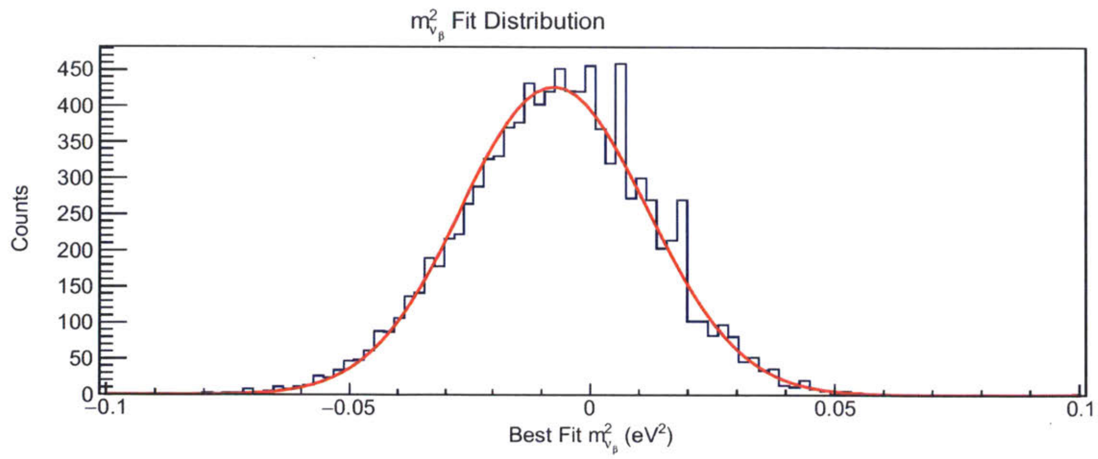


Figure 9-8: An example of the results from an ensemble test for the 3 Gauss field setting. Pseudo-data was generated using the fully pixelated effective transmission function, while the fit was performed using transmission function model #3.

Transmission function model	3 Gauss: σ_{stat} on $m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	3 Gauss: syst. $\Delta m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	6 Gauss: σ_{stat} on $m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	6 Gauss: syst. $\Delta m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)
Model # 1	22	-67	19	-22
Model # 2	21	-49	19	-11
Model # 3	19	-7.4	18	-6.3

Table 9.4: Table of systematic shifts on the $m^2_{\nu_\beta}$ induced by transmission function model choice.

by no means is an insignificant effect and is comparable with some of the largest identified systematic effects estimated in the design report [51].

The statistical errors are also slightly increased, but are not as detrimental to the measurement as the systematic shifts. Therefore, to obtain the best neutrino mass sensitivity possible, it is necessary to include the azimuthal variation in the transmission function model. However, from this study two additional questions arise: To what level of detail do we need to know the azimuthal variation of the transmission function, and how well does this additional knowledge improve our limits on the systematic error?

9.5 Measurement Strategy for the Full Reconstruction of the Transmission Function

The particle tracking studies have shown that there is significant spatial variation in the transmission function and that this variation is not purely axially symmetric. Furthermore, from the method of ensemble tests, it is clear that several naive models of the transmission function result in significant systematic errors in the neutrino mass extraction. Since reliance on the field calculations and simulation alone is unacceptable as a means of constraining the analyzing potential (due to unknown system misalignment or deviations in the model geometry), it is necessary to explore a method by which we may reconstruct the transmission function over the entire spatial extent of the flux tube from measurement alone. However, it is not immediately clear where physically and with what resolution the analyzing potential ought to be measured in order reconstruct it. In order to determine this, we will resort to sampling theory.

When dealing with the reconstruction of functions in polar coordinates, the immediate method which springs to mind is that of Fourier-Bessel expansion [211]. However, this method is unappealing for the reason that the function to be reconstructed must be sampled in the radial coordinate at the locations of the zeros of each of the n Bessel functions used in the expansion. This method is satisfactory, and indeed, ideal, for an easily sampled function, where control over the exact location of each sample point can be made. However, it is not particularly helpful when reconstructing the analyzing potential everywhere. This is because it is difficult to control the exact position of the electron beam, and precise knowledge of its location can only be obtained at special points (such as the intersection of two or three pixels). Therefore, as an alternative to Fourier-Bessel series expansion, we will make use of Lagrange interpolation in polar coordinates. Fortunately, this method does not require uniformly spaced or otherwise specifically positioned samples in order to reconstruct the function.

Marvasti [168] presented a theorem on the number and the locations of samples

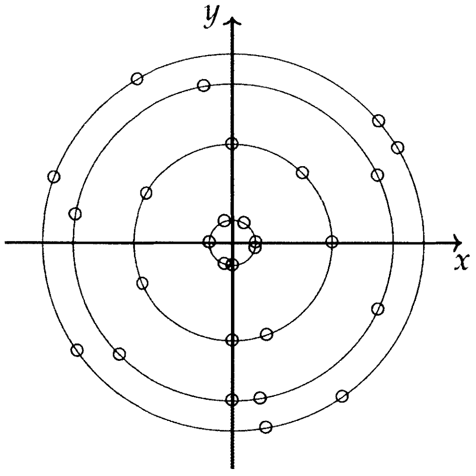


Figure 9-9: The non-uniform polar sampling strategy of theorem 9.1.

needed to reconstruct a two-dimensional band-limited function in polar coordinates using complex functions. Margolis subsequently adapted this theorem so it can be expressed without complex numbers [167]. There are two versions of this theorem, one which relates to samples taken on non-uniform angular intervals along unevenly spaced rings, and a second, in which the samples are taken with arbitrary spacing, along lines which pass through the origin at non-uniform angular intervals. Although either technique can be used to reconstruct band-limited functions, for the sake of simplicity, we will only consider the first version of this theorem, which is stated nearly verbatim, as follows (see [167] for the original statement and further details):

Theorem 9.1 *Let $\{r_n; n = 0,1,2,\ldots\}$ be a sampling sequence of real numbers with average density greater than R/π, where each number corresponds to a circle with radius, r_n, centered at the origin. Let $\{\theta_{nm}; n = 0,1,2,\ldots, m = 0,1,2\ldots, N-1\}$ be a set of real numbers, which defines nonuniform samples on the circle r_n, where $N \geq 2K+1$. If $\{r_n\}$ satisfies:*

$$\begin{aligned} \left|r_n - n\frac{\pi}{R}\right| &< L < \infty\,, \\ |r_n - r_k| &> \delta > 0\,, \quad n \neq k\,. \end{aligned} \tag{9.8}$$

Then, any function $f(r,\theta)$ band-limited to the circular disk of radius, R, and angularly *band-limited to K can be perfectly reconstructed from the set of nonuniform samples,* $\{\tilde{f}(r_n, \theta_{nm})\}$ *by:*

$$f(r,\theta) = \sum_{n=0}^{\infty} \sum_{N-1} \tilde{f}(r_n, \theta_{nm}) \Phi_{nm}(\theta) \frac{G(r)}{G'(r_n)(r - r_n)} , \tag{9.9}$$

where

$$G(r) = (r - r_0) \prod_{n=1}^{\infty} \left(1 - \frac{r}{r_n}\right) , \tag{9.10}$$

and

$$\Phi_{nm}(\theta) = \begin{cases} \prod\limits_{\substack{q=0 \\ q \neq m}}^{\infty} \frac{\sin((\theta - \theta_{nq})/2)}{\sin((\theta_{nm} - \theta_{nq})/2)} & N \text{ odd} , \\ \cos\left(\frac{\theta - \theta_m}{2}\right) \prod\limits_{\substack{q=0 \\ q \neq m}}^{\infty} \frac{\sin((\theta - \theta_{nq})/2)}{\sin((\theta_{nm} - \theta_{nq})/2)} & N \text{ even} . \end{cases} \tag{9.11}$$

An example of this sampling scheme is shown in figure 9-9 and demonstrates that uniform spacing between the samples is not necessary to reconstruct a band-limited function, as long as the density is sufficient to satisfy the Nyquist criterion. Using this theorem, we can then determine what number of measurement points would be needed to sufficiently determine the analyzing potential throughout the flux tube cross section so as to limit the systematic shift induced by our limited knowledge of the transmission function. To do this, we will first generate several sampling schemes and examine the error between the calculated and reconstructed analyzing potential. Then, we will consider the systematic shift induced on the neutrino mass measurement caused by using a transmission function model based on the reconstructed analyzing potential. We note that the same arguments also hold for the analyzing magnetic field. However, for the purpose of this study, we will limit ourselves to reconstructing the potential only and treat the magnetic field as if it were known perfectly. Before proceeding, we should make note that this interpolation method is by no means the only way to reconstruct the analyzing potential everywhere. However, it is advantageous because of its loose requirements on the sampling positions and its ideal convergence properties when applied to band-limited functions. In addition, it is also probably not realistic to completely

Name	Number of measurement points	Ring locations	Azimuthal locations (°) (approximate)
4×4	16	$1, 5, 9, 13$	$0, 90, 180, 270$
5×5	25	$0, 2, 6, 10, 13$	$0, 72, 144, 216, 288$
6×6	36	$0, 2, 6, 9, 11, 13$	$0, 45, 90, 135, 180, 225, 270, 315$

Table 9.5: Table of sampling schemes to reconstruct the analyzing potential (and transmission function) throughout the flux tube.

ignore the field calculation and rely solely on measurements when developing a full model of the transmission function. However, this self-imposed limitation is effective for exploring a worst-case scenario where this extra information is not relied upon.

Since measurement of the transmission function (and analyzing potential) is relatively time consuming, it is not realistic to measure it for every pixel. Instead, we need to make a judicious choice for the total number and location of each of the measurements. For this purpose, we will consider three simplistic sampling schemes. These are outlined in table 9.5. The radial locations are assumed to be at the outer radius of each specified ring, and the azimuthal locations are given by the listed values. These locations have been chosen somewhat arbitrarily, and it is expected that the sampling locations of an actual measurement would deviate. This should not generally be an issue given the flexibility of theorem 9.1, as long as the full radius of the flux tube (FPD) is covered and the number of sample points is the same. However, while there is no strict requirement on the exact azimuthal location of each measurement, it is important that the samples along a single ring lie at roughly the same radius. This should not be too difficult a requirement provided that the measurement locations are close to the pixel's radial edge, where boundary crossings can be used to position the beam just inside each pixel.

The results of interpolating the analyzing potential using each of the sampling schemes in table 9.5 are shown in figure 9-10. It should be noted that this evaluation of the sampling methods neglects measurement error on the analyzing potential. However, as this can generally be expected to be roughly 10-30mV or less [110], it is not a significant source of error in comparison to that which is inherent in the interpolation technique itself. From figure 9-10, we can conclude that in general, the 6 Gauss field setting is more easily reconstructed and needs fewer sampling points than the 3 Gauss field setting. This comes as no surprise since for the 6 Gauss field setting, the flux tube is further away from the spectrometer walls and exhibits much less azimuthal variation. In general, for both field settings, the variation in the analyzing potential can be resolved to within roughly 50 mV for the 6×6 sampling scheme. However, for the 3 Gauss field setting, the outermost ring cannot be properly resolved and exhibits large ~ 0.3V errors. This is due to the influence of the wire comb support structures. Fortunately, this effect dies off within the outermost ring, leaving the interior of the flux tube unaffected. If desired, the potential variation in the outermost ring could be resolved properly, but this would require a large number of azimuthal sampling positions ($\gtrsim 20$), which may not be realistic.

Now that we have directly explored our ability to infer the analyzing potential between measurement points, what remains is to determine what systematic effect each sampling scheme has on the neutrino mass measurement. To do this, we repeat the procedure of the previous section, generating a set of pseudo-data using the Monte-Carlo calculated effective transmission functions, and then fitting it with an alternate transmission function model. However, instead of the previously mentioned transmission function models, we will construct the fit models by using the sample-reconstructed analyzing potential when computing the effective transmission function of each pixel. The results of these ensemble tests for each of the sampling schemes listed in table 9.5, and for both the 3 and 6 Gauss field settings, are given in table 9.6.

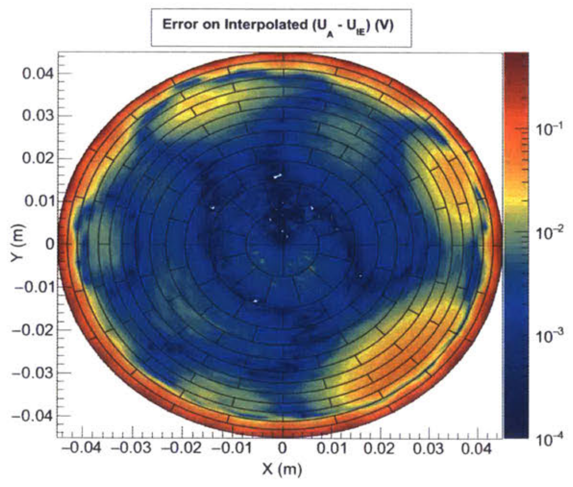
From these results, we can see that for the 3 Gauss field setting, more detailed sampling schemes generally reduce the systematic error on the neutrino mass. When considering the 3 Gauss field setting, using the most detailed measurement scheme (6×6) to reconstruct the analyzing potential yields a reasonably acceptable level of systematic error of 2.7×10^{-3} eV^2. This represents roughly 1.25% of the full systematic error budget of 17×10^{-3} eV^2. On the other hand, the 6 Gauss field

Transmission function model	3 Gauss: σ_{stat} on $m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	3 Gauss: syst. $\Delta m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	6 Gauss: σ_{stat} on $m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)	6 Gauss: syst. $\Delta m^2_{\nu_\beta}$ $\times 10^{-3}$ (eV^2)
Reconstructed, 4×4	18	3.6	18	5.8
Reconstructed, 5×5	18	3.3	19	5.6
Reconstructed, 6×6	18	2.7	18	5.0

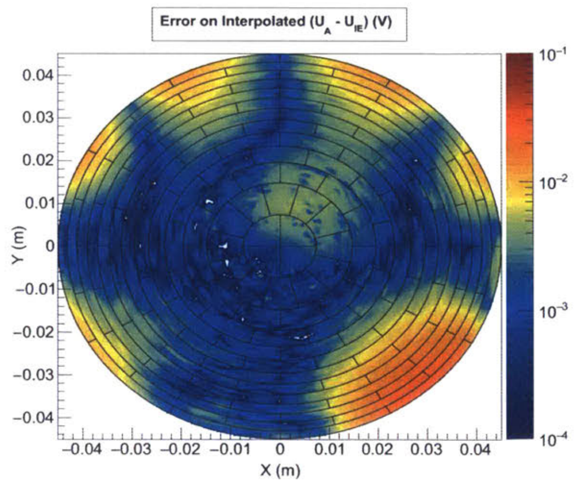
Table 9.6: Table of systematic shifts on the $m^2_{\nu_\beta}$ measurement induced by imperfect reconstruction of the spatial variation of the transmission function for several analyzing potential sampling schemes and magnetic field settings.

setting does not seem to exhibit much improvement with an increased sampling density. This is likely because the analyzing potential is much more homogeneous for the 6 Gauss field setting, and the lowest sampling strategy already manages to resolve the variation over the flux tube. However, what is more puzzling, and somewhat counterintuitive, is that even though the spatial inhomogeneities of the 6 Gauss transmission function are less severe than the 3 Gauss setting, the systematic shift in the extracted neutrino mass is larger. Of course, when deciding between the two magnetic field settings, there are other considerations (such as the background rate) that are equally or more important than the optimization of the transmission function behavior.

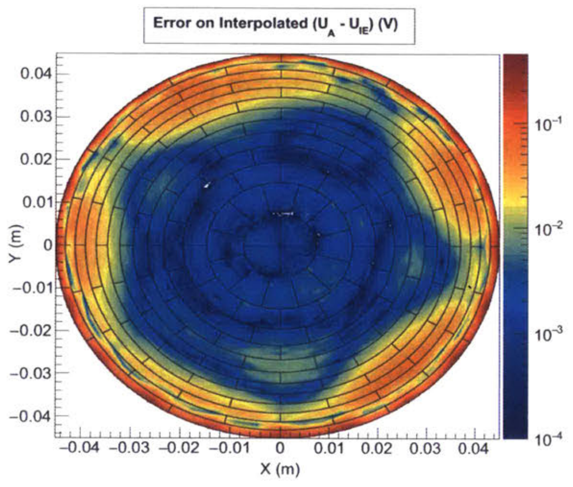
On the basis of this study, we conclude that with a sufficient, but reasonable number of measurement points, the fully, non-axially symmetric behavior of the transmission function throughout the flux tube can be modeled quite accurately using the interpolation method of theorem 9.1 alone. Using the 6×6 sampling scheme to reconstruct the analyzing potential variation can reduce the systematic error percentage caused by our transmission function model by a factor of 5 over that which is induced by using the axially-symmetric, ring-smeared model. In the future, it is expected that combining measurement data with input from the three-dimensional field model when constructing a realistic model of the transmission function may help to reduce the systematic error associated with the field inhomogeneities even further.



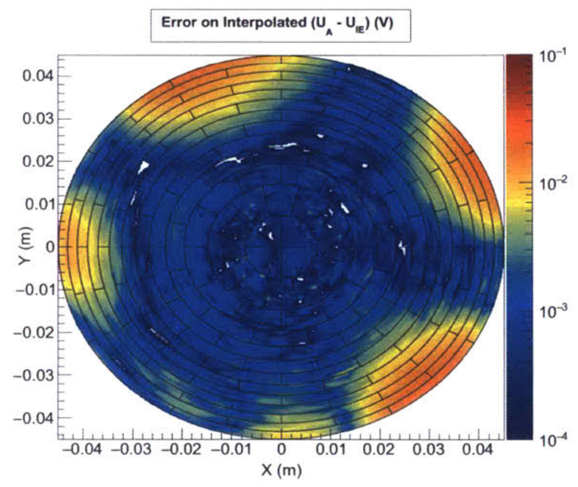
(a) 3 Gauss, 4 $\times$ 4 sampling scheme.



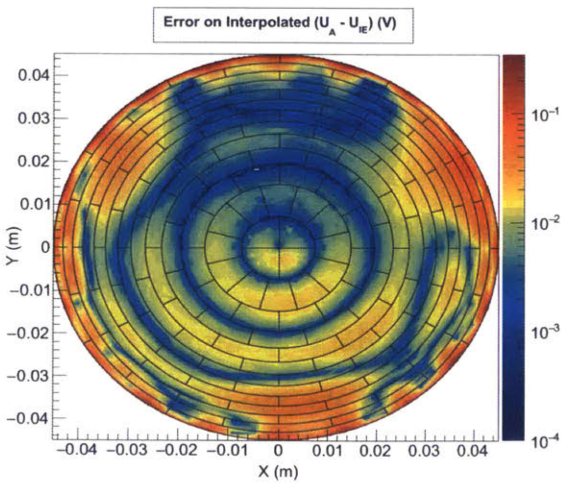
(b) 6 Gauss, 4 $\times$ 4 sampling scheme.



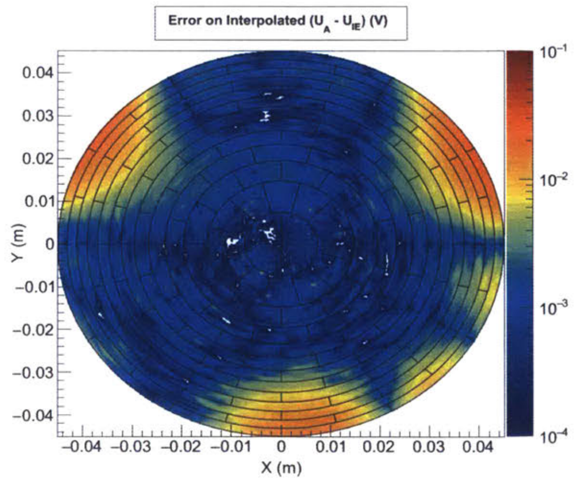
(c) 3 Gauss, 5 $\times$ 5 sampling scheme.



(d) 6 Gauss, 5 $\times$ 5 sampling scheme.



(e) 3 Gauss, 6 $\times$ 6 sampling scheme.



(f) 6 Gauss, 6 $\times$ 6 sampling scheme.

Figure 9-10: The error on the reconstructed analyzing potential for the 3 and 6 Gauss field settings under different sampling schemes. The color axis units are volts.

Chapter 10

Conclusion and Future Outlook

The KATRIN experiment has the ambitious aim of making the most sensitive model independent measurement of the neutrino mass to date. In order to accomplish this goal an enormous number of challenges must be overcome. This thesis has contributed to this effort by developing a novel variant of the fast multipole method in order to rapidly calculate the electrostatic field of the main spectrometer system using the boundary element method. This technique has been implemented as an extension to the open source field solving library `KEMField` and integrated with the particle tracking software `Kassiopeia`. It has also been extended to exploit high performance computing, using both MPI and OpenCL based parallelism, and has used to fully solve the boundary value problem of the three dimensional KATRIN main spectrometer geometry. Furthermore, it has improved the field solving performance by over three orders of magnitude in speed and has, for the first time, enabled high-statistics Monte Carlo simulations of millions of charged particles in a fully three dimensional model of the KATRIN main spectrometer.

In order to validate this fast field calculation model, an electron gun was used to make a detailed survey of the analyzing potential inhomogeneities within the main spectrometer using a transmission function calibrated time-of-flight method. It was found to be accurate to within 0.01-0.03%. The validated model was then used to perform a Monte Carlo simulation of electrons in the main spectrometer using the shorted electrode configuration. The results of this study show that simple axially symmetric models of the transmission function are not sufficient to meet the systematic error budget of the KATRIN experiment. However, further investigation of several sampling techniques, has shown that a reasonable measurement program can resolve the azimuthal variation of the analyzing potential well enough to bring the systematic effect associated with an imperfectly modeled transmission function down to the percent level. This is a promising result and it will help in delivering KATRIN's formidable goal of 200 meV sensitivity to the neutrino mass.

KATRIN's measurement will be the most sensitive model-independent probe of the neutrino mass scale to date, and will help constrain and inform extensions to the standard model and cosmology. With additional developments (such as the MAC-E-TOF method [214]) KATRIN's sensitivity may be able to be pushed even further. However, because of the manner in which the energy resolution of the spectrometer depends on its size, KATRIN will probably be the last terrestrial MAC-E filter to place a new limit on the neutrino mass from tritium β-decay. In the long term, new techniques such as cyclotron radiation emission spectroscopy (CRES) [18] and the use of atomic tritium will likely be necessary in order to go beyond the abilities of KATRIN in its current form. With the rapidly approaching completion of KATRIN's commissioning phase, the prospects for new insight into the absolute neutrino mass scale look very promising, and the next several years will no doubt be very exciting.

Appendix A

The Spherical Multipole Expansion of a Triangle

This appendix has been reproduced from the original publication [28].

A.1 Introduction

The behavior of systems under electrostatic forces is governed by the electric field $\mathbf{E}$, which can be expressed as the gradient of a scalar potential Φ:

$$\mathbf{E} = -\nabla\Phi \tag{A.1}$$

In the absence of free charges, the potential Φ is determined by the Laplace equation,

$$\nabla^2\Phi = 0 \tag{A.2}$$

for all points $\mathbf{x}$ in the simply connected domain Ω. The Laplace equation admits a unique solution for the field $\mathbf{E}$ when the conditions on the boundary of the domain, $\partial\Omega$, are specified. The boundary conditions may be completely specified by associating either a value for the potential Φ (Dirichlet), or the derivative of Φ with respect to the surface normal $\frac{\partial\Phi}{\partial n}$ (Neumann), for every point on $\partial\Omega$.

One technique for numerically solving the Laplace equation is the boundary element method (BEM). Compared to other popular methods designed to accomplish the same goal, such as Finite Element and Finite Difference Methods [190], the BEM method focuses on the boundaries of the system rather than its domain, effectively reducing the dimensionality of the problem. BEM also facilitates the calculation of fields in regions that extend out to infinity (rather than restricting computation to a finite region) [216]. These two features make the BEM faster and more versatile than competing methods when it is applicable.

The basic underlying idea of the BEM involves reformulating the partial differential equation as a Fredholm integral equation of the first or second type, defined respectively as,

$$f(\mathbf{x}) = \int\limits_{\partial\Omega} K(\mathbf{x},\mathbf{y})\Phi(\mathbf{y})d\mathbf{y} \tag{A.3}$$

and

$$\Phi(\mathbf{x}) = f(\mathbf{x}) + \lambda \int\limits_{\partial\Omega} K(\mathbf{x},\mathbf{y})\Phi(\mathbf{y})d\mathbf{y}\ , \tag{A.4}$$

where $K(\mathbf{x},\mathbf{y})$ (known as the Fredholm kernel), and $f(\mathbf{x})$ are known, square-integrable functions, λ is a constant, and $\Phi(\mathbf{x})$ is the function for which a solution is sought. Discretizing the boundary of the domain into N elements and imposing the boundary conditions on this integral equation through either a collocation or Galerkin scheme results in the formation of dense matrices which naively cost $\mathcal{O}(N^2)$ to compute and store and $\mathcal{O}(N^3)$ to solve [160]. This scaling makes solving large problems (much more than $\sim 10^4$ elements) impractical unless some underlying aspect of the equations involved can be exploited. For example, for the Laplace equation there exist iterative methods, such as Robin Hood [152] [84], which take advantage of non-local charge transfer allowed by the elliptic nature of the equation to reduce the needed storage to $\mathcal{O}(N)$ and time of convergence to $\mathcal{O}(N^\alpha)$, with $1 < \alpha < 2$.

Another technique that has been used to accelerate the BEM solution to the Laplace equation, and has also found wide applicability in three dimensional electrostatic, elastostatic, acoustic, and other problems, is the fast multipole method (FMM) [160]. The FMM was originally developed by V. Rohklin and L. Greengard for the two dimensional Laplace boundary value problem [202] and N-body simulation [107]. Fast multipole methods are appropriate when the kernel of the equation is separable or approximately separable so that, to within some acceptable error, it may be expressed as [32],

$$K(\mathbf{x},\mathbf{y}) \approx \sum_{k=0}^{p} \psi_k(\mathbf{x})\xi_k(\mathbf{y}) \ . \tag{A.5}$$

In the case of the Laplace equation, the kernel is often approximated by an expansion in spherical coordinates, with the functions $\psi_k(\mathbf{x})$ and $\xi_k(\mathbf{y})$ taking the form of the regular and irregular solid harmonics [78], [224]. This expansion allows the far-field effects of a source to be represented in a compressed form by a set of coefficients known as the *multipole moments* of the source.

When applying BEM together with FMM in solving the Laplace equation over a complex geometry, it is necessary to determine the multipole moments of various subsets of the surfaces involved. At the smallest spatial scale, this requires a means of computing the individual multipole moments of each of the chosen basis functions (boundary elements). Geometrically, these basis functions usually take the form of planar triangular and rectangular elements, with the charge density on these elements either constant or interpolated between some set of sample points. Since rectangular elements cannot necessarily discretize an arbitrary curved surface without gaps or overlapping elements and can be decomposed into triangles, we consider it sufficient to compute the multipole expansion of basis functions of the triangular type.

In section (A.2) we introduce the integral we wish to compute. In section (A.3) the coordinate system in which the integral is evaluated is described and we demonstrate a recursive evaluation of the multipole moments in the case of constant charge density. The manner by which the multipole moments convert under coordinate transformation is described in section (A.4). We discuss the application of this method to triangular basis functions with non-constant charge density in (A.5), and provide the results of some numerical tests in section (A.6). Unless otherwise specified, throughout this paper we will index all vector and matrix elements starting from zero.

A.2 Mathematical Preliminaries

For an arbitrary collection of charges bounded within a sphere of radius R about the point $\mathbf{x}_0$, there is a remote expansion for the potential $\Phi(\mathbf{x})$ given by [129], [107]:

$$\Phi(\mathbf{x}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \frac{Q_l^m Y_l^m(\theta,\phi)}{r^{l+1}} \,. \tag{A.6}$$

This approximation converges at all points $|\mathbf{x} - \mathbf{x}_0| > R$. The coefficients Q_l^m are known as the multipole moments of the charge distribution. The functions $Y_l^m(\theta,\phi)$ (known as the spherical harmonics), are given by:

$$Y_l^m(\theta,\phi) = N_l^m P_l^{|m|}(\cos\theta) e^{im\phi} \,, \tag{A.7}$$

where the coordinates (r,θ,ϕ) are measured with respect to the origin $\mathbf{x}_0$, and the function P_l^m is the associated Legendre polynomial of the first kind. Several normalization conventions exist for the spherical harmonics; here we use the Schmidt semi-normalized convention with the normalization coefficients given by:

$$N_l^m = \sqrt{\frac{(l-|m|)!}{(l+|m|)!}} \,. \tag{A.8}$$

When the charge distribution $\sigma(\mathbf{x}')$ is confined to a surface Σ, the moments are given by the following integral:

$$Q_l^m = \int_{\Sigma} \sigma(\mathbf{x}) \overline{Y_l^m}(\theta,\phi) r^l d\Sigma \,. \tag{A.9}$$

Here we have used a bar over the spherical harmonic to denote the complex conjugate. The integral given in equation (A.9) can be addressed in a straightforward manner through two dimensional Gaussian quadrature [155]. It can also be reduced to a one dimensional Gaussian quadrature if one first computes an auxiliary vector field and applies Stokes' theorem, as described by Mousa et al [176]. However, for high-order expansions, accurate evaluation of the numerical integration becomes progressively more expensive. It is therefore desirable to obtain an analytic expression of the multipole moments.

For an arbitrary expansion origin and triangular surface element, Equation (A.9) is very difficult to compute analytically. In order to proceed, we therefore make two simplifying restrictions on the general problem: we assume that the charge density σ_0 is constant over the triangle, and that we can always find a special coordinate system S unique to each triangle in which to perform the integral (A.9).

A.3 Coordinate system for integration

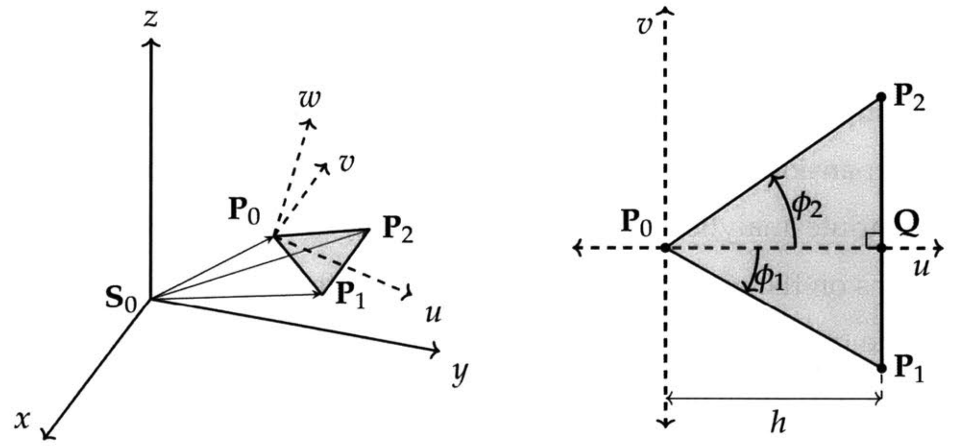
In order to compute the multipole expansion of the triangle Σ defined by points $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$, we first must select the appropriate coordinate system to simplify the integration. Without loss of generality, we choose a system so that the vertex $\mathbf{P}_0$ lies at the origin, and the $\hat{\mathbf{e}}_1$ direction is parallel to the vector $\mathbf{P}_2 - \mathbf{P}_1$. The plane defined by the triangle is then parameterized by the local coordinates (u, v). Formally, this local coordinate system S can be defined with the following origin and basis vectors:

$$S : \begin{cases} \mathcal{O} & = \mathbf{P}_0 \\ \hat{\mathbf{e}}_0 & = \frac{\mathbf{Q}-\mathbf{P}_0}{|\mathbf{Q}-\mathbf{P}_0|} \\ \hat{\mathbf{e}}_1 & = \frac{\mathbf{P}_2-\mathbf{P}_1}{|\mathbf{P}_2-\mathbf{P}_1|} \\ \hat{\mathbf{e}}_2 & = \hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}_1 \end{cases}, \tag{A.10}$$

where $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$ are the points defining the triangle Σ in the original coordinate system. The point $\mathbf{Q}$ lies at $(h, 0)$ in the (u, v)-plane and is the closest point to $\mathbf{P}_0$, which lies on the line joining the points $\mathbf{P}_1$ and $\mathbf{P}_2$. It is given by:

$$\mathbf{Q} = \mathbf{P}_1 + \left(\frac{(\mathbf{P}_0 - \mathbf{P}_1) \cdot (\mathbf{P}_2 - \mathbf{P}_1)}{|\mathbf{P}_2 - \mathbf{P}_1|^2} \right) (\mathbf{P}_2 - \mathbf{P}_1) \ . \tag{A.11}$$

Figure (A-1) shows the arrangement of this coordinate system.



(a) Triangle Σ in global coordinate system.

(b) Triangle Σ in local coordinate system S.

Figure A-1: In (A-1a) the boundary element Σ (shaded region) is shown with arbitrary position and orientation in the global coordinate system. A detailed view of the local coordinate system, S, in which the integration is performed, is shown in (A-1b), where the w axis points out of the page.

Within S the integration takes place entirely in the (u, v)-plane, therefore the integration over the θ coordinate can be trivially evaluated at $\theta = \pi/2$, and the integral reduces to:

$$Q_l^m = N_l^m P_l^m(0) \int_{\phi_1}^{\phi_2} \int_0^{r(\phi)} \sigma_0 e^{-im\phi} r^{l+1} dr d\phi \ . \tag{A.12}$$

As can be seen in figure (A-1) the upper limit on the r integration is given by:

$$r(\phi) = \frac{h}{\cos\phi} \ . \tag{A.13}$$

Performing the integration over the r coordinate leaves us with:

$$Q_l^m = \underbrace{\left(\frac{N_l^m h^{l+2} P_l^m(0)}{l+2}\right)}_{\mathcal{K}_{l,m}} \underbrace{\int_{\phi_1}^{\phi_2} \frac{e^{-im\phi}}{(\cos\phi)^{l+2}} d\phi}_{\mathcal{I}_{l,m}} \ . \tag{A.14}$$

The prefactors $\mathcal{K}_{l,m}$ are easy to compute. To address the integral $\mathcal{I}_{l,m}$, we split our integrand into imaginary and real components, $\mathcal{I}_{l,m} = \mathcal{A}_{l,m} - i\mathcal{B}_{l,m}$, where:

$$\mathcal{A}_{l,m} = \int_{\phi_1}^{\phi_2} \frac{\cos(m\phi)}{(\cos\phi)^{l+2}} d\phi \tag{A.15}$$

and

$$\mathcal{B}_{l,m} = \int_{\phi_1}^{\phi_2} \frac{\sin(m\phi)}{(\cos\phi)^{l+2}} d\phi \ . \tag{A.16}$$

Before evaluating these integrals, we must first introduce the Chebyshev polynomials [4], [169]. The Chebyshev polynomials of the first kind $T_n(x)$ are defined recursively for $n \geq 0$ as:

$$T_0(x) = 1 \tag{A.17}$$

$$T_1(x) = x \tag{A.18}$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \ . \tag{A.19}$$

Similarly, the Chebyshev polynomials of the second kind, $U_n(x)$, are defined as:

$$U_0(x) = 1 \tag{A.20}$$

$$U_1(x) = 2x \tag{A.21}$$

$$U_{n+1}(x) = 2xU_n(x) - U_{n-1}(x) \ . \tag{A.22}$$

These polynomials are noteworthy for our purposes because of the two following useful properties:

$$T_n(\cos\phi) = \cos(n\phi) \tag{A.23}$$

$$U_n(\cos\phi) = \frac{\sin((n+1)\phi)}{\sin\phi} \,. \tag{A.24}$$

We can exploit these in order to evaluate $\mathcal{A}_{l,m}$ and $\mathcal{B}_{l,m}$ recursively. We first address $\mathcal{A}_{l,m}$. Using (A.23) we may rewrite (A.15) as:

$$\mathcal{A}_{l,m} = \int\limits_{\phi_1}^{\phi_2} \frac{T_m(\cos\phi)}{(\cos\phi)^{l+2}} d\phi \,, \tag{A.25}$$

which can be expanded in terms of $T_{m-1}(x)$ and $T_{m-2}(x)$ using (A.19) to give

$$\mathcal{A}_{l,m} = 2\int\limits_{\phi_1}^{\phi_2} \frac{T_{m-1}(\cos\phi)}{(\cos\phi)^{l+1}} d\phi - \int\limits_{\phi_1}^{\phi_2} \frac{T_{m-2}(\cos\phi)}{(\cos\phi)^{l+2}} d\phi \,. \tag{A.26}$$

This yields the recursion relationship for the $\mathcal{A}_{l,m}$:

$$\mathcal{A}_{l,m} = 2\mathcal{A}_{l-1,m-1} - \mathcal{A}_{l,m-2} \,. \tag{A.27}$$

Similarly we may use the Chebyshev polynomial of the second kind (A.24) to rewrite (A.16) as

$$\mathcal{B}_{l,m} = \int\limits_{\phi_1}^{\phi_2} \frac{U_{m-1}(\cos\phi)}{(\cos\phi)^{l+2}} \sin\phi d\phi \tag{A.28}$$

and derive the recursion relationship for the $\mathcal{B}_{l,m}$, which unsurprisingly has the same form:

$$\mathcal{B}_{l,m} = 2\mathcal{B}_{l-1,m-1} - \mathcal{B}_{l,m-2} \,. \tag{A.29}$$

Given these recursion relationships, we can reduce the integrals $\mathcal{A}_{l,m}$ and $\mathcal{B}_{l,m}$ of any degree $0 \leq l$ and order $0 \leq m \leq l$ into a series of terms, for which only the base cases must be evaluated explicitly. Figure (A-2) shows a representation of the recursion relationship.

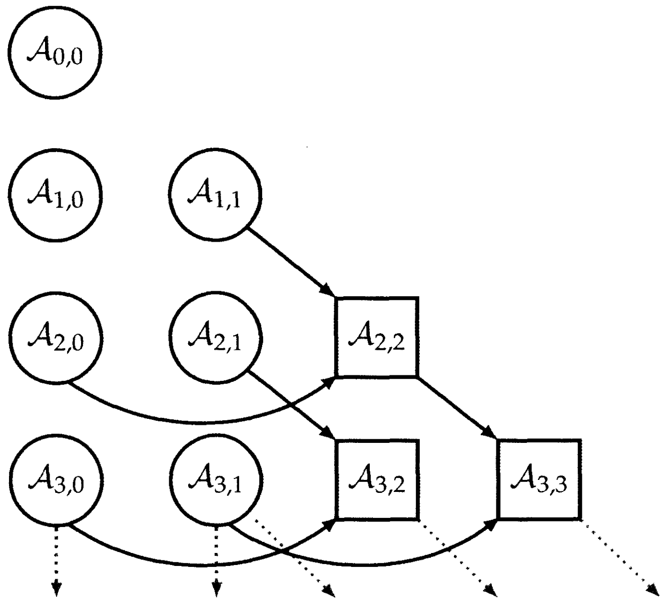


Figure A-2: Graphical representation of recursion given in equation A.27 up to $l = 3$. Circles denote terms which must be computed as a base case. Squares denote terms which may be computed by recurrence. The arrows indicate dependence. Higher order terms extend downwards and to the right, as denoted by the dotted lines and arrows.

Specifically, the integrals that are not further reducible by recursion are the following; $\mathcal{A}_{l,0}$, $\mathcal{A}_{l,1}$, $\mathcal{B}_{l,0}$, and $\mathcal{B}_{l,1}$. Fortunately, all of these base cases have relatively simple solutions that either have a closed form, or a terminating reduction relation. Integrals of the form $\mathcal{B}_{l,0}$ are zero for all $l \geq 0$, while the $\mathcal{B}_{l,1}$ are given as follows:

$$\mathcal{B}_{l,1} = I^1_{l+2} = \int_{\phi_1}^{\phi_2} \frac{\sin\phi}{(\cos\phi)^{l+2}} d\phi \tag{A.30}$$

whereas both $\mathcal{A}_{l,0}$ and $\mathcal{A}_{l,1}$ are integrals of a power of secant I^0_p;

$$I^0_p = \int_{\phi_1}^{\phi_2} (\sec\phi)^p d\phi \tag{A.31}$$

with $\mathcal{A}_{l,0} = I^0_{l+2}$ and $\mathcal{A}_{l,1} = I^0_{l+1}$. The notation I^q_p and these integrals are addressed in A.8.

It should be noted that during the process of computing the value of the Q_l^m moment through recursion, the real and imaginary parts of all moments with degree $\leq l$ and order $\leq m$ will be computed. These values can be stored so that there is no need to repeat the recursion for each individual moment needed. This is useful when determining the multipole expansion of a boundary element since all moments up to certain maximal degree can be computed in one pass through the recurrence.

A.4 Multipole moments under coordinate transformation

We can make use of the results of the preceding section to compute the multipole expansion coefficients of the boundary element Σ with respect to an arbitrary origin and set of coordinate axes. Typically, we are most interested in being able to construct the multipole moments M_j^k of Σ in the coordinate system that has the canonical Cartesian coordinate axes, with an origin at an arbitrary point $\mathbf{S}_0$. We denote this system as S'':

$$S'' : \begin{cases} \mathcal{O} & = \mathbf{S}_0 \\ \hat{e}_0'' & = (1,0,0) \\ \hat{e}_1'' & = (0,1,0) \\ \hat{e}_2'' & = (0,0,1) \end{cases} . \tag{A.32}$$

Therefore, we must first construct the coordinate transformation $A : S \rightarrow S''$, and then determine how this coordinate transform operates on the coefficients Q_l^m of the multipole expansion given in S. The rigid motion $A : S \rightarrow S''$ can be specified by a rotation $U : S \rightarrow S'$ followed by a translation $T : S' \rightarrow S''$. We can describe the translation by the displacement $\mathbf{\Delta} = \mathbf{S}_0 - \mathbf{P}_0$, and the rotation U by the Euler angles (α, β, γ) following the $Z - Y' - Z''$ axis convention of [189] and [92]. The Euler angles allow us to write the rotation U as the composition of three successive

Table A.1: Euler angles in terms of the elements of the matrix U

Angle	$U_{22} \neq \pm 1$	$U_{22} = 1$	$U_{22} = -1$
α	$\mathtt{atan2}\left(\frac{-U_{21}}{\sin\beta}, \frac{-U_{20}}{\sin\beta}\right)$	0	π
β	$\mathtt{acos}(U_{22})$	$\mathtt{atan2}(U_{10}, U_{00})$	$\mathtt{atan2}(U_{01}, U_{11})$
γ	$\mathtt{atan2}\left(\frac{-U_{12}}{\sin\beta}, \frac{-U_{02}}{\sin\beta}\right)$	0	0

rotations $U = U_{Z''}(\gamma)U_{Y'}(\beta)U_Z(\alpha)$. Explicitly, U is given by

$$U = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.33}$$

and can be related to the basis vectors of the coordinate system S by:

$$U = \begin{bmatrix} U_{00} & U_{01} & U_{02} \\ U_{10} & U_{11} & U_{12} \\ U_{20} & U_{21} & U_{22} \end{bmatrix} = \begin{bmatrix} \hat{e}_0 \\ \hat{e}_1 \\ \hat{e}_2 \end{bmatrix}^T . \tag{A.34}$$

It is well known that the Euler angles (α, β, γ) do not uniquely describe an arbitrary rotation matrix U, however, a unique description is not necessary for our purposes. A convenient set of choices is given in table (A.1). With the transformation $A : S \to S''$ specified by the Euler angles (α, β, γ) and the displacement $\mathbf{\Delta}$, we can determine the multipole moments of Σ in S'' through the application of theorems (A.1) and (A.2).

Theorem (A.1), from Wigner [231], originates in quantum mechanics [73]. It appears when needing to express the result of the action of the rotation operator $\mathcal{D}^l(\alpha, \beta, \gamma)$ upon a particular eigenstate $|l, m\rangle$ of total angular momentum l, which is associated with the spherical harmonic $Y_l^m(\theta, \phi)$, in terms of the eigenstates of the rotated frame $|l', m'\rangle$. Note that since total angular momentum is conserved, this rotation operator does not mix states with a distinct value of l (thus $l = l'$). Specifically, Wigner's theorem tells us the matrix elements of the rotation operator

$\mathcal{D}^l(\alpha,\beta,\gamma)$, which is a member of the $(2l+1)\times(2l+1)$ matrix representation of $SO(3)$. A more succinct version of this theorem is given in [92], and is restated here in slightly a modified form.

Theorem A.1 *Assume there are two coordinate systems which share the same origin $S:(O,\hat{e}_0,\hat{e}_1,\hat{e}_2)$ and $S':(O,\hat{e}'_0,\hat{e}'_1,\hat{e}'_2)$, that are related by the rotation $U\in SO(3)$ specified by the Euler angles $\{\alpha,\beta,\gamma\}$ such that $\hat{e}'_i=U\hat{e}_i$, for $i=0,\ 1,\ 2$. Furthermore assume that there is a function $F(\theta,\phi)$ that can be expanded in terms of the spherical harmonics $Y_l^m(\theta,\phi)$ such that:*

$$F(\theta,\phi)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l}Q_l^mY_l^m(\theta,\phi) \tag{A.35}$$

then there exists a function $f(\theta',\phi')$ such that

$$f(\theta',\phi')=F(\theta(\theta',\phi'),\phi(\theta',\phi'))=\sum_{l=0}^{\infty}\sum_{m'=-l}^{l}q_l^{m'}Y_l^{m'}(\theta',\phi') \tag{A.36}$$

where the coefficients $q_l^{m'}$ are given by:

$$q_l^{m'}=\sum_{m=-l}^{l}\mathcal{D}^l_{m',m}(\alpha,\beta,\gamma)Q_l^m \tag{A.37}$$

and elements of the Wigner matrix $\mathcal{D}^l_{m',m}(\alpha,\beta,\gamma)$ are given by:

$$\mathcal{D}^l_{m',m}(\alpha,\beta,\gamma)=e^{im\gamma}\ d^l_{m',m}(\beta)\ e^{im\alpha} \tag{A.38}$$

with

$$\begin{aligned}d^l_{m',m}(\beta)=&\left[\frac{(l+m')!(l-m')!}{(l+m)!(l-m)!}\right]^{1/2}\times\\&\sum_{\sigma}\left\{(-1)^{l-m'-\sigma}\binom{l+m}{l-m'-\sigma}\binom{l-m}{\sigma}\times\right.\\&\left.\left(\cos\frac{\beta}{2}\right)^{2\sigma+m'+m}\left(\sin\frac{\beta}{2}\right)^{2l-2\sigma-m'-m}\right\}\end{aligned} \tag{A.39}$$

where the summation over σ is for all values where the entries of binomial coefficients are non-negative.

The direct evaluation of the coefficients $\mathcal{D}^l_{m',m}(\alpha,\beta,\gamma)$ through the use of the expressions given by Wigner is known to be inefficient, as well as numerically unstable for large values of l and certain angles [47]. However, given the wide applicability of spherical harmonics to quantum chemistry, fast multipole methods, and other areas, there has recently been a large effort to develop efficient and stable methods to perform such rotations in both real and complex spherical harmonic bases. The current state of the field of spherical harmonic rotation is well summarized by [154], with the algorithm developed by Pinchon et al. [189] being one of the fastest and most accurate. We will provide a brief description their algorithm here.

To avoid the need of complex matrix-vector multiplication, the method proposed by Pinchon et al. [189] is executed in the basis of real spherical harmonics $S_l^m(\theta,\phi)$. The real spherical harmonics as defined by Pinchon (note the difference in the normalization convention) are related to our definition of the complex spherical harmonics given in equation (A.7) by

$$S_l^0(\theta,\phi) = \left(\sqrt{\frac{2l+1}{4\pi}}\right) Y_l^0(\theta,\phi) \tag{A.40}$$

$$S_l^m(\theta,\phi) = \begin{cases} \frac{1}{\sqrt{2}}\left(\sqrt{\frac{2l+1}{4\pi}}\right)\left[Y_l^m(\theta,\phi) + (-1)^m Y_l^{-m}(\theta,\phi)\right] & : m > 0 \\ \frac{i}{\sqrt{2}}\left(\sqrt{\frac{2l+1}{4\pi}}\right)\left[(-1)^m Y_l^m(\theta,\phi) - Y_l^{-m}(\theta,\phi)\right] & : m < 0 \end{cases} . \tag{A.41}$$

To apply a rotation to the set of multipole moments $\{Q_l^m\}$ with l fixed and m ranging from $-l$ to l we first must calculate the corresponding real basis $\{R_l^m\}$ coefficients, such that

$$\sum_{m=-l}^{l} Q_l^m Y_l^m(\theta,\phi) = \sum_{m=-l}^{l} \frac{4\pi}{2l+1} R_l^m S_l^m(\theta,\phi) , \tag{A.42}$$

using the relations given in (A.40) and (A.41). Then, to prepare this set of moments

$\{R_l^m\}$ for the rotation operator we arrange them to form the column vector $\mathbf{R}_l$:

$$\mathbf{R}_l = \left[R_l^{-1},\, R_l^{-l+1},\, R_l^{-l+2},\, \dots,\, R_l^{l-1},\, R_l^{l}\right]^T . \tag{A.43}$$

The application of the Wigner $\mathcal{D}^l$-matrix to this column vector produces the corresponding vector of rotated moments $\mathbf{r}_l$. For efficiency, the $\mathcal{D}^l$-matrix is itself decomposed into several matrices, each of which may be applied to the vector $\mathbf{R}_l$ in succession:

$$\mathbf{r}_l = \mathcal{D}^l(\alpha,\beta,\gamma)\mathbf{R}_l = \left[X_l(\alpha)J_lX_l(\beta)J_lX_l(\gamma)\right]\mathbf{R}_l \tag{A.44}$$

In this notation, the X_l matrices effect a rotation about the z-axis, while the J_l matrices perform an interchange of the y and z axes. The advantage to this method is that the X_l matrices have a simple sparse form whose action on the vector $\mathbf{R}_l$ can be computed quickly, as they consist only of non-zero diagonal and anti-diagonal terms. For example, for $l = 2$:

$$X_2(\alpha) = \begin{pmatrix} \cos(2\alpha) & 0 & 0 & 0 & \sin(2\alpha) \\ 0 & \cos(\alpha) & 0 & \sin(\alpha) & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -\sin(\alpha) & 0 & \cos(\alpha) & 0 \\ -\sin(2\alpha) & 0 & 0 & 0 & \cos(2\alpha) \end{pmatrix} . \tag{A.45}$$

The interchange matrices J_l, on the other hand, are completely independent of the rotation angles and therefore only need to be computed once. While the computation of J_l is beyond the scope of this paper, there is an elegant recursive scheme to compute them up to any degree l given by Pinchon et al. [189]. After the rotated moments $\mathbf{r}_l$ have been computed in the real basis, we need only convert them back to the complex basis through the inverse of equations (A.40) and (A.41) to obtain the set of moments $\{q_l^{m'}\}$.

Now that we have obtained the multipole moments $\{q_l^{m'}\}$ in the coordinate system S', we need to determine how they are modified by a displacement of the expansion origin. This can be accomplished by the application of theorem (A.2). This theorem, presented by Greengard and Rohklin [202], [107], is a principle part of the fast multipole method, applied during the operation of gathering the multipole expansions of smaller regions into larger collections, and describes how a multipole expansion about one origin can be re-expressed as an expansion about a different origin. Graphically, this is represented in figure A-3.

Theorem A.2 *Consider a multipole expansion with coefficients $\{O_n^m\}$ due to charges located within the sphere D with radius a centered about the point $\mathbf{P}_0$. This expansion converges for points outside of sphere D. Now consider the point $\mathbf{S}_0 \notin D$ such that $\mathbf{\Delta} = \mathbf{S}_0 - \mathbf{P}_0 = (\rho, \alpha, \beta)$. We may form a new multipole expansion about the point $\mathbf{S}_0$ due to the charges within D which converges for points outside of the sphere D' which has its center at $\mathbf{S}_0$ and radius $a' = \rho + a$. The multipole moments of the new expansion $\{M_j^k\}$ are given by:*

$$M_j^k = \sum_{n=0}^{j} \sum_{m=-n}^{m=n} \frac{O_{j-n}^{k-m} i^{|k|-|m|-|k-m|} A_n^m A_{j-n}^{k-m} \rho^n Y_n^{-m}(\alpha, \beta)}{A_j^k} \qquad \text{(A.46)}$$

where

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)!(n+m)!}} \,. \qquad \text{(A.47)}$$

Immediately applying this theorem to the set of moments $\{q_l^{m'}\}$ results in the final objective of obtaining the multipole moments of the boundary element Σ in the coordinate system S''. However, the number of arithmetic operations required by the application of theorem (A.2) scales like $\mathcal{O}(p^4)$. This high cost can be mitigated by the use of a special case of theorem (A.2), by White et al. [230], and is stated in a form similar to that stated by [108], [46] below:

Theorem A.3 *Consider the situation given in theorem (A.2), in the case that the point $\mathbf{S}_0 \notin D$ is such that $\mathbf{\Delta} = \mathbf{S}_0 - \mathbf{P}_0 = (\rho, 0, 0)$. That is, $\mathbf{S}_0$ lies on the z-axis above $\mathbf{P}_0$. We may form a new multipole expansion about the point $\mathbf{S}_0$, with the multipole moments of the*

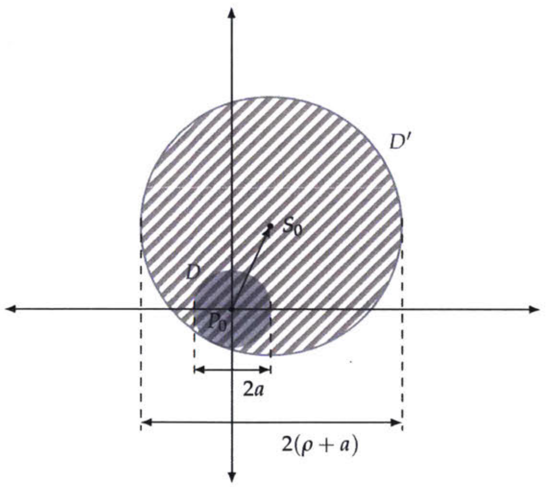


Figure A-3: Multipole to multipole translation. The solid shaded area indicates the region where the original multipole expansion $\{O_n^m\}$ does not converge. The striped area indicates the region where the new multipole expansion $\{M_j^k\}$ does not converge.

new expansion $\{M_j^k\}$ *given by:*

$$M_j^k = \sum_{n=0}^{j} \frac{O_{j-n}^k A_n^0 A_{j-n}^k \rho^n Y_n^0(0,0)}{A_j^k} \, . \tag{A.48}$$

Theorem (A.3) by itself is of course only directly applicable in rare circumstances. However, White et al. [230] noted that it can be applied to perform a multipole-to-multipole translation along any axis needed if a rotation is performed through the use of theorem (A.1) before and after the translation operation. The first rotation applied aligns the z-axis with the vector $\mathbf{S}_0 - \mathbf{P}_0$, while the second rotation is the inverse of the first. The use of the rotation operator together with the axial translation has a cost which scales like $\mathcal{O}(p^3)$, which for high-degree expansions can provide useful acceleration when compared to the implementation of theorem (A.2) alone.

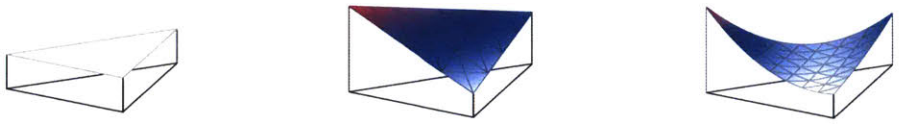

(a) Zero-th order ($N = 0$). (b) First order ($N = 1$). (c) Second order ($N = 2$).

Figure A-4: Planar boundary elements with various orders of charge density interpolation. Height above the element indicates the value of the local charge density.

A.5 Application to higher order basis functions

There are many schemes for function interpolation over triangular domains, such as the natural orthogonal polynomial basis put forth by [198], [67], [182] and [142], and the more commonly used variations on Lagrange and Hermite interpolation [226], [217], [26], [44]. For the sake of simplicity, we avoid these more advanced interpolation schemes in favor of a simpler but less well-conditioned bivariate monomial basis for the charge density. This basis is well suited to the method of integration described in the preceding sections, as it can be represented more naturally in the same coordinate system S. Making a change of basis from some other interpolation method to the bivariate monomials is relatively straightforward; however, we will defer discussion of this change of basis and its application to low-order Lagrange interpolation to A.9.

We make the assumption that the interpolated charge density on the triangle can be expressed in terms of the local orthogonal coordinates (u, v) by:

$$\sigma(u,v) = \begin{cases} \sum_{a=0}^{N} \sum_{b=0}^{N-a} s_{a,b} u^a v^b & : (u,v) \in \Sigma \\ 0 & : (u,v) \notin \Sigma \end{cases}, \tag{A.49}$$

where N is the order of the interpolation, the variables (u, v) are as defined in figure (A-1), and $s_{a,b}$ are the interpolation coefficients. For an example of various orders N of interpolation, see figure (A-4). It is possible to perform a change of basis on the interpolating polynomials [88] to compute the $s_{a,b}$ coefficients in terms of the coefficients of some other polynomial basis; we leave discussion of this change of basis to A.9. As can be seen from figure (A-1) the local coordinates (u, v) in terms of the polar coordinates (r, ϕ) are given by

$$u(r, \phi) = r \cos \phi \tag{A.50}$$

$$v(r, \phi) = r \sin \phi \; . \tag{A.51}$$

Inserting our expression for the charge density (A.49) and local coordinates (A.51) into (A.12) and exchanging the order of integration and summation results in:

$$Q_l^m = \sum_{a=0}^{N} \sum_{b=0}^{N-a} s_{a,b} N_l^m P_l^m(0) \int_{\phi_1}^{\phi_2} \int_0^{r(\phi)} (\cos \phi)^a (\sin \phi)^b e^{-im\phi} r^{a+b+l+1} dr d\phi \; . \tag{A.52}$$

Performing the integration over the r coordinate leaves us with:

$$Q_l^m = \sum_{a=0}^{N} \sum_{b=0}^{N-a} \underbrace{\left(\frac{s_{a,b} h^{a+b+l+2}}{a+b+l+2} \right) N_l^m P_l^m(0)}_{\mathcal{K}_{l,m}^{a,b}} \underbrace{\int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b e^{-im\phi}}{(\cos \phi)^{b+l+2}} d\phi}_{\mathcal{I}_{l,m}^{b}} \; . \tag{A.53}$$

We compute the integrals of the form $\mathcal{I}_{l,m}^b$ by splitting into imaginary and real components, $\mathcal{I}_{l,m}^b = \mathcal{A}_{l,m}^b - i\mathcal{B}_{l,m}^b$:

$$\mathcal{A}_{l,m}^b = \int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b \cos(m\phi)}{(\cos \phi)^{b+l+2}} d\phi \tag{A.54}$$

$$\mathcal{B}_{l,m}^b = \int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b \sin(m\phi)}{(\cos \phi)^{b+l+2}} d\phi \; . \tag{A.55}$$

Using (A.23), we may rewrite (A.54) as

$$\mathcal{A}_{l,m}^b = \int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b T_m(\cos \phi)}{(\cos \phi)^{b+l+2}} d\phi \; . \tag{A.56}$$

Expanding this using (A.19) gives

$$\mathcal{A}_{l,m}^b = 2 \int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b T_{m-1}(\cos \phi)}{(\cos \phi)^{b+l+1}} d\phi - \int_{\phi_1}^{\phi_2} \frac{(\sin \phi)^b T_{m-2}(\cos \phi)}{(\cos \phi)^{b+l+2}} d\phi \; , \tag{A.57}$$

which yields the recursion relationship for the $\mathcal{A}^b_{l,m}$:

$$\mathcal{A}^b_{l,m} = 2\mathcal{A}^b_{l-1,m-1} - \mathcal{A}^b_{l,m-2} \, . \tag{A.58}$$

Similarly for the $\mathcal{B}^b_{l,m}$, we have:

$$\mathcal{B}^b_{l,m} = 2\mathcal{B}^b_{l-1,m-1} - \mathcal{B}^b_{l,m-2} \, . \tag{A.59}$$

Reducing the integrals $\mathcal{A}^b_{l,m}$ and $\mathcal{B}^b_{l,m}$ with this recursion relationship leaves us with the task of evaluating the base case integrals that are not further reducible: $\mathcal{A}^b_{l,0}$, $\mathcal{A}^b_{l,1}$, $\mathcal{B}^b_{l,0}$, and $\mathcal{B}^b_{l,1}$. Integrals of the form $\mathcal{B}^b_{l,0}$ are zero for all $l \geq 0$ and $b \geq 0$, while the remaining base cases can all be expressed as:

$$\mathcal{A}^b_{l,0} = I^b_{b+l+2} \tag{A.60}$$

$$\mathcal{A}^b_{l,1} = I^b_{b+l+1} \tag{A.61}$$

$$\mathcal{B}^b_{l,1} = I^{b+1}_{b+l+2} \, , \tag{A.62}$$

where

$$I^q_p = \int_{\phi_1}^{\phi_2} \frac{(\sin\phi)^q}{(\cos\phi)^p} d\phi \, . \tag{A.63}$$

The solution of I^q_p is addressed in A.8.

The application of the coordinate transform to the moments Q^m_l then follows as detailed in section (A.4). A summary of the full method by which to compute the multipole moments of a triangle is detailed in algorithm (11).

A.6 Numerical Results

In order to gain some understanding of the accuracy and efficiency of the algorithm presented in this work, some numerical tests were performed with regard to the problem of evaluating the electrostatic potential of a uniformly charged triangle. This simple scenario was chosen because there exists an analytic solution to the

Algorithm 11 Computing the multipole moments of a triangular boundary element.

```
Input: Triangle $\Sigma : \{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$ and associated charge density interpolation coefficients $\{s_{ab}\}$.
 1: Compute height $h$ and coordinate system $S$ for triangle $\Sigma$ according to equation (A.10).
 2: for $l = 0$ to $p$ do
 3:     for $m = 0$ to $l$ do
 4:         for all $s_{a,b} \neq 0$ do
 5:             Compute the prefactor $\mathcal{K}^{a,b}_{l,m}$ according to equation (A.53).
 6:             Recursively compute the integral $\mathcal{I}^{b}_{l,m}$ according to equations (A.58) and (A.59).
 7:         end for
 8:         Compute the multipole moment $Q_l^m = \sum_a \sum_b \mathcal{K}^{a,b}_{l,m}\mathcal{I}^{b}_{l,m}$ and $Q_l^{-m} = \overline{Q_l^m}$.
 9:     end for
10: end for
11: Compute the Euler angles $(\alpha, \beta, \gamma)$ of the rotation $U : S \to S'$ according to table (A.1).
12: Using theorem (A.1) compute the effect of the rotation $U$ on the set of moments; $\{Q_l^m\} \to \{q_l^{m'}\}$.
13: Using theorem (A.2) alone, or according to [230] through the use of theorem (A.1) and theorem (A.3) together, compute the effect of the translation $\mathbf{\Delta} : S' \to S''$ on the moments; $\{q_l^{m'}\} \to \{q_l^m\}$.
Output: The multipole moments $\{q_l^m\}$ of the triangle $\Sigma$ in coordinate system $S''$.
```

potential of a uniformly charged triangle [84], which makes absolute accuracy comparisons feasible. All of the following tests were performed in double precision.

Since the integrals required to compute the multipole expansion of boundary elements are typically evaluated using numerical quadrature, a straightforward two dimensional Gauss-Legendre quadrature method was used as a benchmark against which to compare the speed and accuracy of the analytic algorithm. The benchmark numerical integration is performed by first converting the integral over the triangular domain given by the points $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$ to an integral over a rectangular domain through the use of a slightly modified version of the transform described by Duffy [69]. We can then write the surface integral given in equation (A.9) as:

$$Q_l^m = \int_0^{L_1}\int_0^{L_2} \sigma_0 \overline{Y_l^m}(\theta(\mathbf{r}), \phi(\mathbf{r}))|\mathbf{r}|^l \left|\frac{\partial \mathbf{r}}{\partial u} \times \frac{\partial \mathbf{r}}{\partial v}\right| dv du = \int_0^{L_1}\int_0^{L_2} f(u,v) dv du \,, \qquad \text{(A.64)}$$

where $\mathbf{r}(u,v) = \mathbf{t}(u,v) - \mathbf{x}_0$, and $\mathbf{t}(u,v)$ is given by

$$\mathbf{t} = \begin{bmatrix} (\mathbf{P}_0 + u\hat{\mathbf{n}}_1 + v\,(1 - u/L_1)\,\hat{\mathbf{n}}_2) \cdot \hat{\mathbf{x}} \\ (\mathbf{P}_0 + u\hat{\mathbf{n}}_1 + v\,(1 - u/L_1)\,\hat{\mathbf{n}}_2) \cdot \hat{\mathbf{y}} \\ (\mathbf{P}_0 + u\hat{\mathbf{n}}_1 + v\,(1 - u/L_1)\,\hat{\mathbf{n}}_2) \cdot \hat{\mathbf{z}} \end{bmatrix} , \qquad \text{(A.65)}$$

with $L_i = |\mathbf{P}_i - \mathbf{P}_0|$ and $\hat{\mathbf{n}}_i = (\mathbf{P}_i - \mathbf{P}_0)/L_i$, where $i = 1, 2$, and the point $\mathbf{x}_0$ is the origin of the expansion. The two dimensional integral over the (u, v)-plane is then performed using m-th order two dimensional Gauss-Legendre quadrature [4], given by:

$$Q_l^m = \frac{L_1 L_2}{4} \sum_{i=1}^{m} \sum_{j=1}^{m} w_i w_j f(u_i, v_j) \qquad \text{(A.66)}$$

where

$$\begin{aligned} u_i &= \tfrac{L_1}{2}(x_i + 1) \\ v_j &= \tfrac{L_2}{2}(x_j + 1) \end{aligned} \qquad \text{(A.67)}$$

while w_i and x_i are the one-dimensional Gauss-Legendre weights and abscissa, respectively. These can be calculated as described by Golub et al. [103]. The orders of the quadrature rules that were considered were $m = \{4,\ 6,\ 8,\ 10\}$. It should be noted that this numerical integration routine is not the most efficient possible, and is only meant to provide a point of reference to a more commonly used means of computing the multipole coefficients. There are several techniques to accelerate the numerical integration over the benchmark we provide, such as adaptive quadrature [35] or quadrature rules specifically formulated for triangular domains. For example, Cowper [56] gives a 6-point rule with a degree of precision of 4, and a 12-point rule with a degree of precision of 6, which require roughly 3 times fewer function evaluations than the corresponding 4×4 and 6×6 two-dimensional rules used in this study. However, the computation of the weights and abscissa for an arbitrary order quadrature rule on a triangular domain is more complicated than the simple two-dimensional scheme, which are trivially generated from the one dimensional Gauss-Legendre weights and abscissa. For the sake of simplicity, these adaptive techniques and triangle-specific quadrature rules were not considered for this study.

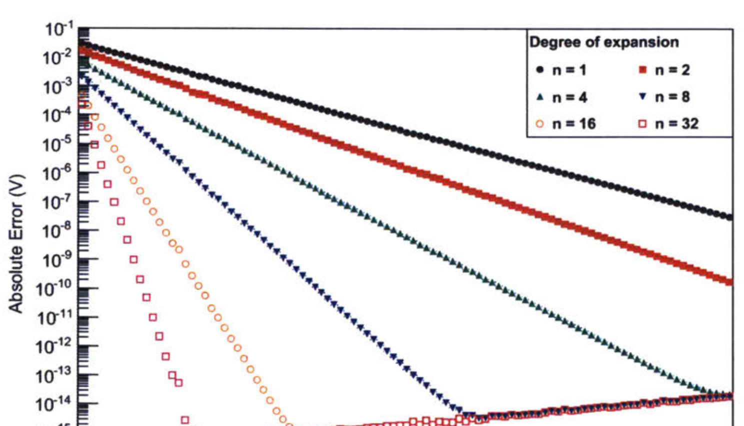


Figure A-5: Comparison of the accuracy of the multipole expansion against the direct method of evaluating the potential. Coefficients of the multipole expansion are calculated using the analytic method described in this paper. Absolute error is shown as a function of the ratio $|\mathbf{x} - \mathbf{x}_0| / R$, where $|\mathbf{x} - \mathbf{x}_0|$ is the distance of the evaluation point from the expansion origin, and R is the radius of the smallest sphere enclosing the charge distribution.

The first study consisted of 100 randomly generated acute triangles, whose vertices were chosen to lie on the unit sphere. For each triangle the charge density was selected such that, at a single collocation point (the centroid), the potential as evaluated by the direct method was unity. This was done in order to normalize the scale of the absolute error such that the smallest measurable error is on the order of 10^{-15}. Then, the multipole expansion of each triangle (up to degree $n = 32$) about the origin $\mathbf{x}_0 = (0,0,0)$ was formed and 10^4 points $\mathbf{x}$ were selected within the volume $1 < |\mathbf{x} - \mathbf{x}_0| < 100$. These points had angular coordinates that followed a uniform distribution over the unit sphere, but the radial coordinate was chosen with a higher weight towards smaller radii in order to provide sufficient statistics in this region. At each point the absolute error between the direct potential and the potential given by the multipole expansion was computed for expansions of degree $n = \{1,\ 2,\ 4\ ,8,\ 16,\ 32\}$. Using the algorithm described in this work to compute the multipole coefficients, the absolute error on the potential is plotted as a function distance from the expansion origin divided by the radius of the region enclosing the charge in figure (A-5). For comparison, the accuracy of the multipole expansion when using two-dimensional Gauss-Legendre quadrature to compute the multipole coefficients is shown in figure (A-7).

It should be noted that the minimum possible error obtainable by the multipole expansion is a slightly increasing function of distance. This trend is observed regardless of the technique used to compute the multipole moments of the source so it is likely attributable to round off error in either the direct potential calculation or in the calculation of the potential from the multipole expansion. However for our purposes this is not an important feature, since we are interested in demonstrating how quickly the expansion converges to this limiting error. As a general rule, as the degree of the expansion is increased the multipole approximation will converge to the minimum possible error at a smaller distance from the source. However, this is only true so long as the method used to compute the multipole moments of the expansion respects the oscillatory behavior of the spherical harmonics. For low degree expansions numerical quadrature rules with a small number of function evaluations can compute the the multipole moments exactly to within machine precision. However, as the degree of the expansion is increased the higher order spherical harmonics oscillate more rapidly and progressively more expensive quadrature rules are needed to evaluate the coefficients to equivalent accuracy. This effect can be seen in figure (A-7). For example, up to an expansion degree of $n = 8$, the 4×4 Gauss-Legendre quadrature rule is sufficient to compute the multipole coefficients to the same accuracy as our algorithm. However continuing to use the 4×4 Gauss-Legendre quadrature rule while increasing the degree of the expansion up to $n = 32$ does not result in a more accurate evaluation of the potential. To obtain the full benefit of a high degree expansion one must correspondingly increase the number of function evaluations used by numerical integration.

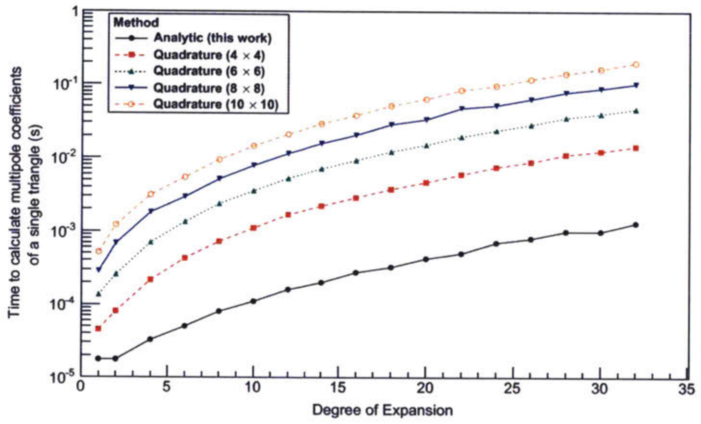
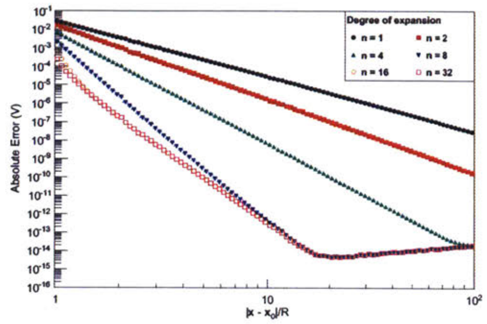
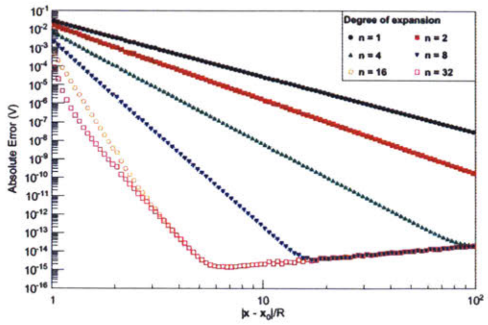


Figure A-6: Time required to evaluate all of the multipole coefficients of a single triangle for the method detailed in algorithm (11) and various $m \times m$ point Gauss-Legendre quadrature.

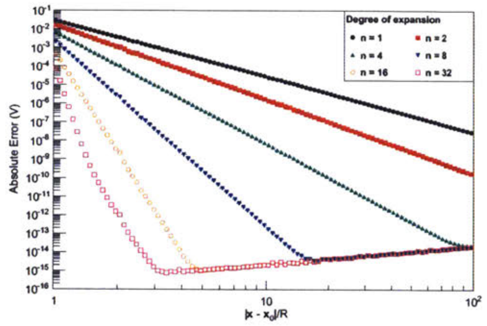
The second study demonstrates the efficiency of this algorithm as an alternative to simple two dimensional numerical integration. To do this, a comparison was made between the time needed to compute all of the multipole expansion coefficients of a single triangle (up to a certain degree) using the analytic algorithm and the time needed when using numerical integration. This test was carried out on a computer with an Intel i7 processor running at 1.9GHz, results are shown in figure (A-6). For all but the lowest degree $p \leq 4$ expansions, the performance of the algorithm presented in this work is approximately an order of magnitude faster than the lowest accuracy Gauss-Legendre quadrature rule considered, while for high order expansions $p \geq 16$, it is nearly two orders of magnitude faster than the quadrature rule which obtains equivalent accuracy.



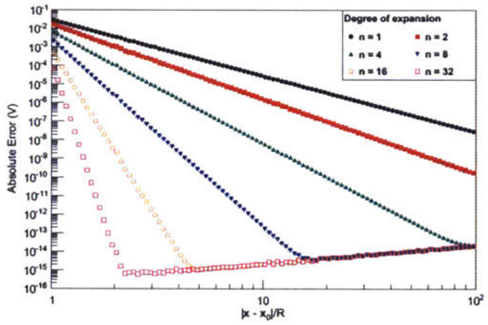
(a) 4×4 quadrature rule



(b) 6×6 quadrature rule



(c) 8×8 quadrature rule



(d) 10×10 quadrature rule

Figure A-7: Comparison of the accuracy of the multipole expansion against the direct method of evaluating the potential. Coefficients of the multipole expansion are computed using two-dimensional Gauss-Legendre quadrature rules of varying precision.

A.7 Conclusion

We have presented a novel technique to evaluate the multipole expansion coefficients of a triangle. This method evaluates the necessary integrals through recursion within the context of a coordinate system with special orientation and placement. The results of the integration can then be generalized to the case of an arbitrary system through the well known transformation properties of the spherical harmonics under rotation and translation. Furthermore we have demonstrated that the application of this method to the multipole expansion of triangles with uniformly constant charge density compares favorably in terms of accuracy and speed to numerical integration. This method can also be extended to the case of non-uniform charge density, provided the interpolant can be represented as a sum over the bivariate monomials. We expect this method may find use in solving the three dimensional Laplace equation with the fast multipole boundary element method (FMBEM). We speculate that other boundary integral equation (BIE) problems, such as the Helmholtz equation in the low frequency limit $k \to 0$, could benefit from this approach if the integrand in the multipole coefficient integrals can be expanded in terms of the solid harmonics. Such will be the study of a following paper.

A.8 Integrals

The solutions to the integrals found in equations (A.30), (A.31), and (A.63) can be found in any standard table of integrals [122], [185]; however, for the sake of completeness we include the solutions and reduction formula here. Starting with equation (A.30) we have an integral of the form

$$I_p^1 = \int_{\phi_1}^{\phi_2} \frac{\sin\phi}{(\cos\phi)^p} d\phi \,, \tag{A.68}$$

which may be solved by simple u-substitution, with $u = \cos(\phi)$, which yields,

$$I_p^1 = -\int_{\cos\phi_1}^{\cos\phi_2} \frac{du}{u^p} = \left.\frac{u^{1-p}}{p-1}\right|_{\cos\phi_1}^{\cos\phi_2} . \tag{A.69}$$

The integral in equation (A.31) is of the type:

$$I_p^0 = \int_{\phi_1}^{\phi_2} (\sec\phi)^p d\phi \,. \tag{A.70}$$

This can be addressed with integration by parts, which yields the reduction relation,

$$I_p^0 = \frac{\sin\phi(\sec\phi)^{p-1}}{(p-1)}\Bigg|_{\phi_1}^{\phi_2} + \left(\frac{p-2}{p-1}\right) I_{p-2}^0 \tag{A.71}$$

with the non-trivial base case:

$$I_1^0 = \int\limits_{\phi_1}^{\phi_2} \sec\phi d\phi = \ln\left|\tan\left(\frac{\phi}{2}+\frac{\pi}{4}\right)\right|\Bigg|_{\phi_1}^{\phi_2} . \tag{A.72}$$

Both of the above integrals turn out to be special cases of equation (A.63), which has the form,

$$I_p^q = \int_{\phi_1}^{\phi_2} \frac{(\sin\phi)^q}{(\cos\phi)^p} d\phi \tag{A.73}$$

where p and q are positive integers. When $p \neq q$, this integral can be simplified by the reduction relation:

$$I_p^q = \frac{-(\sin\phi)^{q-1}}{(q-p)(\cos\phi)^{p-1}}\Bigg|_{\phi_1}^{\phi_2} + \left(\frac{q-1}{q-p}\right) I_p^{q-2} \tag{A.74}$$

until the base cases I_p^0 and I_p^1 are reached. If $p = q$, we simply have an integral of a power of tangent,

$$I_p^p = \int_{\phi_1}^{\phi_2} (\tan\phi)^p d\phi \tag{A.75}$$

which in turn can be reduced with

$$I_p^p = \frac{(\tan\phi)^{p-1}}{p-1} - I_{p-2}^{p-2} \tag{A.76}$$

until reaching the non-trivial base case,

$$I_1^1 = -\ln|\cos\phi|\big|_{\phi_1}^{\phi_2} . \tag{A.77}$$

Although most of these integrals do not have a simple closed form, the implementation of the base cases and reduction formula in computer code is a fairly simple task.

A.9 Change of interpolating basis

Since the calculation of section (A.5) proceeds assuming that the interpolant on the boundary element can be expressed in the basis of the bivariate monomials, in order to make these results relevant to the various interpolation methods often used (see for example, [226], [217], [26], [44]) we need to be able to change the basis of the interpolant. Explicitly, we would like to express the interpolant as a sum over the bivariate monomials. To do this, we must determine the coefficients of the bivariate monomials in terms of the original interpolation parameters. To motivate this section, we will consider the example task of changing from the bivariate Lagrange to bivariate monomial basis. The objective we seek is to replace the tedious symbolic manipulation often encountered when performing a polynomial change of basis with a well defined numerical procedure. We expect that the results may apply to a wider class of interpolants other than Lagrange, though this extension is beyond the scope of this paper. To start, we will first introduce some basic definitions, with the assumption that the reader is familiar with the concept of a group and ring such as presented by [183] or [30].

Let $R[u,v]$ be the polynomial ring over the real numbers in the variables u and v. Then for all $F(u,v) \in R[u,v]$, we may write $F(u,v)$ as the series,

$$F(u,v) = \sum_{a=0}^{n_f} \sum_{b=0}^{m_f} f_{a,b} u^a v^b \tag{A.78}$$

where the coefficients $f_{a,b} \in \mathbb{R}$, and $n_f,\ m_f \in \mathbb{N}_0$. The sum and product operations on this ring are defined in the usual sense as follows; for $F(u,v),\ G(u,v) \in R[u,v]$, the sum is given by:

$$F(u,v) + G(u,v) = H(u,v) = \sum_{a=0}^{n_h} \sum_{b=0}^{m_h} h_{a,b} u^a v^b \quad \in R[u,v] \tag{A.79}$$

where $h_{a,b} = f_{a,b} + g_{a,b}$, and $n_h = \max(n_f, n_g)$ with m_h defined similarly. The product is given by:

$$F(u,v) \cdot G(u,v) = K(u,v) = \sum_{a=0}^{n_k} \sum_{b=0}^{m_k} k_{a,b} u^a v^b \quad \in R[u,v] \tag{A.80}$$

where

$$k_{a,b} = \sum_{i=0}^{a} \sum_{j=0}^{b} f_{i,j} \cdot g_{a-i,b-j} \tag{A.81}$$

and $n_k = n_f + n_h$ with m_k similarly.

For a given polynomial $F(u,v)$, the greatest integer $a+b$ for which the coefficient $f_{a,b}$ is nonzero is called the maximal combined order of $F(u,v)$. We will denote the set of all bivariate polynomials $F(u,v) \in R[u,v]$ whose maximal combined order is N as P_N. In general we may write any polynomial $S^{(N)}(u,v) \in P_N$ as follows

$$S^{(N)}(u,v) = \sum_{a=0}^{N} \sum_{b=0}^{N-a} s_{a,b} u^a v^b \ . \tag{A.82}$$

Consider for example the first order bivariate polynomial,

$$s^{(1)}(u,v) = s_{0,0} + s_{0,1} u + s_{10} v \ . \tag{A.83}$$

This function can be also represented as the matrix vector product:

$$s^{(1)}(u,v) = (1,u) \underbrace{\begin{bmatrix} s_{0,0} & s_{0,1} \\ s_{1,0} & 0 \end{bmatrix}}_{R^{(1)}} \begin{pmatrix} 1 \\ v \end{pmatrix} \ . \tag{A.84}$$

The ability to write the above example in this manner motivates us to find a map between P_N and the set of $(N+1) \times (N+1)$ upper left triangular matrices, T_N. In general, we expect that the bivariate polynomial $S^{(N)}(u,v) \in P_N$, may be written in terms of a matrix vector product involving an upper left triangular matrix $R^{(N)} \in T_N$ whose entries correspond to the coefficients $s_{a,b}$ as follows:

$$s^{(N)}(u,v) = (1, u, \ldots, u^N) \underbrace{\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & \cdots & s_{0,N} \\ s_{1,0} & s_{1,1} & \cdots & s_{1,(N-1)} & 0 \\ s_{2,0} & \cdots & s_{2,(N-2)} & 0 & \vdots \\ \vdots & \iddots & 0 & \iddots & 0 \\ s_{N,0} & 0 & \cdots & 0 & 0 \end{bmatrix}}_{R^{(N)}} \begin{pmatrix} 1 \\ v \\ \vdots \\ v^N \end{pmatrix}. \tag{A.85}$$

Clearly, the set T_N forms a group under matrix addition, and this corresponds to the fact that P_N is also closed under addition. Unfortunately, P_N is not closed under the operation of polynomial multiplication $(\cdot)$, because repeated multiplication can produce a polynomial of arbitrarily large order. In order to construct a proper ring from the set P_N we must restore the property of closure by replacing the traditional product operator $(\cdot)$, with a new operator $(\odot)$ which we will define as multiplication combined with the truncation of terms with combined order larger than N. Formally, for any two polynomials $F(u,v),\ G(u,v) \in P_N$, this operator is given by:

$$F(u,v) \odot G(u,v) = H(u,v) = \sum_{a=0}^{N} \sum_{b=0}^{N-a} h_{a,b} u^a v^b \quad \in P_N \tag{A.86}$$

where,

$$h_{a,b} = \sum_{i=0}^{a} \sum_{j=0}^{b} f_{i,j} \cdot g_{a-i,b-j} \,. \tag{A.87}$$

We note the the $(\odot)$ product defined in equation (A.86) only differs from the definition of normal polynomial multiplication in equation (A.80) by the limits on the summation. This definition leads us to the following lemma.

Lemma A.1 *The set P_N together with the binary operations $+$ and $\odot$ forms a ring.*

In light of lemma (A.1) we would also like to find a binary operator on two matrices $A,\ B \in T_N$ which mirrors the action of multiplication on the set P_N of bivariate polynomials. It is clear from inspection of equations (A.80) and (A.81) that multiplication $(\cdot)$ over the polynomials in $R[u,v]$ corresponds with the two dimensional convolution ($*$) of the two matrices formed from the monomial coefficients. However, the set T_N is also not closed under the convolution operator ($*$). To restore this closure we will instead consider a different operator $\circledast$, specified in definition (A.1).

Definition A.1 *Let the two matrices A and B be elements of T_N, then the action of the binary operator $\circledast$ on A and B produces another matrix $C \in T_N$, whose elements are given by:*

$$C_{a,b} = \begin{cases} \sum\limits_{i=0}^{a} \sum\limits_{j=0}^{b} A_{i,j} B_{a-i,b-j} & a+b \leq N \\ 0 & a+b > N \end{cases} \tag{A.88}$$

Choosing the $\circledast$ operator to be defined as the product operation over T_N produces the following lemma.

Lemma A.2 *The set T_N together with the binary operations of matrix addition $+$ and the operator $\circledast$ forms a ring.*

To make use of the two rings $(P_N, +, \odot)$ and $(T_N, +, \circledast)$ in the problem of determining the monomial coefficients of an interpolant, we now need a bijective map between the two which preserves the structure of the operations on each ring. Specifically, we need an isomorphism, $\Lambda : (P_N, +, \odot) \to T_N(P_N, +, \circledast)$. Equation (A.85) has already demonstrated the nature of $\Lambda^{-1} : (T_N, +, \circledast) \to (P_N, +, \odot)$ as a matrix vector product, and leads us to definitions (A.2) and (A.3), and theorem (A.4).

Definition A.2 *Since we may write all $F(u,v) \in P_N$ according to equation (A.82), we define the map $\Lambda : P_N \to T_N$ as $\Lambda(F(u,v)) = R$, where the entries of the matrix $R \in T_N$ are given in terms of the monomial coefficients of $F(u,v)$ by $R_{i,j} = f_{i,j}$ and are zero when $N < i + j$.*

Definition A.3 *For all $R \in T_N$, we define the map $\Lambda^{-1} : T_N \to P_N$ as follows,*

$$\Lambda^{-1}(R) = F(u,v) \tag{A.89}$$

where the bivariate polynomial $F(u,v) \in P_N$ is given by the following matrix vector product,

$$F(u,v) = \mathbf{u}^{\mathbf{T}} R \mathbf{v} \tag{A.90}$$

where the column vectors $\mathbf{u}$ *and* $\mathbf{v}$ *of length $N+1$, have their i-th entry given (as powers of the variables u and v) by u^i and v^i respectively.*

Theorem A.4 *The inverse of the map $\Lambda : P_N \to T_N$, is given by $\Lambda^{-1} : T_N \to P_N$, moreover the map Λ is a isomorphism from the ring $(P_N, +, \odot)$ to the ring $(T_N, +, \circledast)$.*

Now that we are in a position to make use of the isomorphism Λ, we will also make some assumptions on the class interpolants upon which we wish to make the change of basis. The first assumption is that interpolant $\Pi_N(u,v)$ of maximal combined order N may be written in terms of a finite set of basis polynomials $\Phi_N \subset P_N$ as,

$$\Pi_N(u,v) = \sum_j U_j p_j^{(N)}(u,v) \tag{A.91}$$

where $p_j^{(N)}(u,v) \in \Phi_N$ and the U_j are know as the interpolation coefficients. The second assumption is that any higher order basis function of the interpolant can be expressed as linear combination of products of the first order basis functions. We will term such a class of interpolants as *simple* according to definition (A.4).

Definition A.4 *Assume that a given class of two dimensional interpolating polynomials has the set of first order basis functions given by*

$$\Phi_1 = \{p_0^{(1)}, p_1^{(1)}, \ldots, p_m^{(1)}\} \subset P_1 \, . \tag{A.92}$$

Now consider all multi-sets C_i of size $1 \leq k \leq N$, formed by making all possible combinations (with repetition allowed) from elements of Φ_1. The number of multi-sets C_i is given by:

$$M = \sum_{k=1}^{N} \binom{m+k}{k} \tag{A.93}$$

If the class of interpolants is such that any N-th order basis polynomial $p_j^{(N)}$ can be written *as,*

$$p_j^{(N)} = \sum_{i=0}^{M-1} \gamma_{i,j} \prod_{x \in C_i} x \tag{A.94}$$

where $\gamma_{i,j} \in \mathbb{R}$ and C_i is the i-th multi-set of size $k \leq N$, and which for all $x \in C_i$, we have $x \in \Phi_1$, then we will call such a class simple. We will call the set of coefficients $\gamma_{i,j}$ together with the corresponding set of multi-sets C_i, the rule of this simple class.

With this definition in mind, we can now approach the problem of converting from a bivariate Lagrange basis to a bivariate monomial basis. Specifically, we wish to find the bivariate monomial coefficients of the polynomial N-th order Lagrange interpolant $\Pi_N(u,v)$. Computationally, this amounts to finding the entries of the matrix $\Lambda(\Pi_N(u,v)) = R^{(N)}$ given the set of interpolation coefficients $\{U_j\}$.

We will follow the notation of [226] and [217], who define the first order Lagrange interpolant for a triangle composed of vertices $\mathbf{P}_j = (u_j, v_j)$ as:

$$\Pi_1(u,v) = \sum_{j=0}^{2} U_j p_j^{(1)}(u,v) \tag{A.95}$$

where,

$$p_j^{(1)}(u,v) = \frac{1}{2A}(\tau_{kl} + \eta_{kl} u - \xi_{kl} v) \tag{A.96}$$

and

$$\tau_{kl} = u_k v_l - v_k u_l \tag{A.97}$$

$$\xi_{kl} = u_k - u_l \tag{A.98}$$

$$\eta_{kl} = v_k - v_l \tag{A.99}$$

while (j,k,l) is any cyclic permutation of $(0,1,2)$. The area of the triangle is denoted by A. Within the context of the coordinate system S, we have $\mathbf{P}_0 = (0,0)$, and

$u_1 = u_2 = h$, so we may directly write down the basis functions $p_j^{(1)}$ as:

$$p_0^{(1)}(u,v) = \frac{1}{2A}\left[(v_1 - v_2)(u-h)\right] \tag{A.100}$$

$$p_1^{(1)}(u,v) = \frac{1}{2A}\left[v_2 u - hv\right] \tag{A.101}$$

$$p_2^{(1)}(u,v) = \frac{1}{2A}\left[-v_1 u + hv\right] \tag{A.102}$$

which have the corresponding coefficient matrices of:

$$R_0^{(1)} = \frac{1}{2A}\begin{bmatrix} h(v_2 - v_1) & (v_1 - v_2) \\ 0 & 0 \end{bmatrix} \tag{A.103}$$

$$R_1^{(1)} = \frac{1}{2A}\begin{bmatrix} 0 & v_2 \\ -h & 0 \end{bmatrix} \tag{A.104}$$

$$R_2^{(1)} = \frac{1}{2A}\begin{bmatrix} 0 & -v_1 \\ h & 0 \end{bmatrix} \tag{A.105}$$

To obtain the bivariate monomial coefficients $\pi_{a,b}$ of the polynomial $\Pi_1(x,y)$ it is then only a simple matter of summing each matrix weighted with the appropriate Lagrange interpolation coefficient.

$$\pi_{a,b} = \left[\sum_{j=0}^{2} U_j R_j^{(1)}\right]_{a,b} \tag{A.106}$$

In order to extend this to N-th interpolation we could again compute the coefficients $\pi_{a,b}$ explicitly through direct inspection of the N-th order basis polynomials. However, for higher orders this quickly becomes tedious even with the use of a computer algebra system. Alternatively we can make use of the isomorphism Λ between the rings $(P_N, +, \odot)$ and $(T_N, +, \circledast)$. We note that since the bivariate Lagrange basis is a *simple* class of interpolating polynomials, we can express any

N-th order basis functions according to equation (A.94) as:

$$\Pi_N(u,v) = \sum_{j=0}^{(N+1)(N+2)/2-1} U_j p_j^{(N)}(u,v) \,. \tag{A.107}$$

Furthermore, under the isomorphism Λ the *rule* of the N-th order Lagrange basis can be re-expressed in the space of T_N by:

$$R_j^{(N)} = \sum_{i=0}^{M-1} \gamma_{ij} \prod_{x\in C_i} \circledast \Lambda(x) \tag{A.108}$$

where we use $\prod \circledast$ to denote a repeated product of the $\circledast$ operator over the matrices given by $\Lambda(x)$. This allows us to compute coefficient matrices $R_j^{(N)}$ directly from from the first order coefficient matrices $R_j^{(1)}$ solely through matrix summation and the use of the $\circledast$ operator. Then, to compute the bivariate monomial coefficients $\pi_{a,b}$ we only need to perform the sum:

$$\pi_{a,b} = \left[\sum_{j=0}^{(N+1)(N+2)/2-1} U_j R_j^{(N)} \right]_{a,b} \,. \tag{A.109}$$

As an example, consider the second order Lagrange interpolant, given by,

$$\Pi_2(u,v) = \sum_{j=0}^{5} U_j p_j^{(2)}(u,v) \tag{A.110}$$

with the *rule* of the second order basis functions defined by:

$$p_j^{(2)}(u,v) = p_j^{(1)}\left(2p_j^{(1)} - 1\right) = 2\left(p_j^{(1)}\right)^2 - p_j^{(1)} \quad : \quad 0 \le j < 3 \tag{A.111}$$

$$p_j^{(2)}(u,v) = 4p_\epsilon^{(1)} p_\delta^{(1)} \quad : \quad 3 \le j < 6 \tag{A.112}$$

where $\epsilon = j \bmod 3$, and $\delta = (j+1) \bmod 3$. Using equation (A.108) to re-express equations (A.111) and (A.112) in terms of coefficient matrices, $R_j^{(2)}$, yields:

$$R_j^{(2)} = 2\left(R_j^{(1)} \circledast R_j^{(1)}\right) - R_j^{(1)} \ : \ 0 \leq j < 3 \tag{A.113}$$

$$R_j^{(2)} = 4R_\epsilon^{(1)} \circledast R_\delta^{(1)} \ : \ 3 \leq j < 6\,. \tag{A.114}$$

Thus the bivariate monomial coefficients of the polynomial $\Pi_2(u,v)$ can be computed in terms of the interpolation coefficients U_j and coefficient matrices $R_j^{(2)}$ of the second order basis functions by:

$$\pi_{a,b} = \left[\sum_{j=0}^{5} U_j R_j^{(2)}\right]_{a,b} \quad . \tag{A.115}$$

In a similar fashion, this method can be applied to any class of *simple* interpolants, summarized in algorithm (12).

```
Algorithm 12 Compute bivariate monomial coefficients of a simple interpolant.
Input: Triangle $\Sigma : \{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$ and set of coefficients $\{U_j\}$ of the $N$-th order simple
    interpolant $S^{(N)}(u,v)$ with rule $(\{\gamma_{i,j}\}, \{C_i\})$.
 1: Compute coordinate system $S$ for triangle $\Sigma$ according to equation (A.10).
 2: Compute $(u,v)$ coordinates of $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2\}$ in $S$.
 3: Form the matrices $R_j^{(1)}$ of the coefficients of the 1st order polynomials in the
    bivariate monomial basis according to equations (A.104) to (A.105).
 4: Compute the coefficient matrices $R_j^{(N)}$ of the $N$-th order basis polynomials
    according to equation (A.108) and the rule $(\{\gamma_{i,j}\}, \{C_i\})$.
 5: Sum the coefficient matrices $R_j^{(1)}$ weighted by their interpolation coefficient $U_j$
    according to equation (A.109) to obtain the matrix $M$.
 6: Map each element of $M$ to the bivariate monomials coefficient $s_{a,b}$ of $S^{(N)}(u,v)$
    according to the isomorphism $\Lambda^{-1} : T_N \to P_N$.
Output: The set of bivariate monomials coefficients $\{s_{a,b}\}$ of $S^{(N)}(u,v)$.
```

Appendix B

The Spherical Multipole Expansion of Rectangle and Wire Elements

B.1 Rectangular Elements

Rectangular mesh elements can be viewed as simply the union of two triangular mesh elements. The moments of which can be computed according to algorithm 11 and then summed. However, in order to cut the number of multipole-to-multipole translation transformations that are necessary in half it is more expedient to divide the rectangle into four triangles as in figure B-1. The moments of triangles A and A' are related by a single plane rotation of π. Therefore, the moments of triangle A' about the centroid $\mathbf{c}$ can be expressed in terms of the moments of triangle A by the simple relation:

$$M_l^m(A') = (-1)^m M_l^m(A) \,. \tag{B.1}$$

Triangles B and B' can be treated similarly. To obtain the moments M_l^m of the rectangle about its centroid $\mathbf{c}$, we first compute the moments of triangles A and B using algorithm 11, and then sum them with moments of their rotated counterparts A' and B'. The resulting rectangle moments can then be transformed through theorems A.2 and A.1 to form an expansion about any other origin.

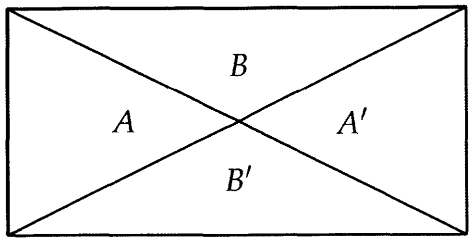


Figure B-1: Division of a rectangle into four triangles to simplify the calculation of its multipole moments.

B.2 Wire Elements

Of the three mesh element types used, the wire elements are the simplest, the parameters describing them consist only of the two vertices ($\mathbf{p}_0$,$\mathbf{p}_1$) spaced apart by a distance, s, and a diameter d. Since the multipole approximation is only useful in the far-field limit we will assume that a wire may be treated as a linear charge density, λ, as if it had no thickness, $d = 0$. Therefore the calculation of its multipole moments reduces to a one dimensional integral. To simplify the integration further we will choose the coordinate system S such that $\mathbf{p}_0$ is at the origin and $\mathbf{p}_1 = (r \sin\theta \cos\phi, r \sin\theta \sin\phi, r \cos\theta)$. In this coordinate system, the expression for the moments is given by:

$$M_l^m = \left(\frac{\lambda s^{l+1}}{l+1}\right) \overline{Y_l^m}(\theta,\phi) \; . \tag{B.2}$$

It is then a simple matter to use theorem A.2 to translate the moments to form an expansion about any other chosen origin.

Appendix C

Embedded Runge-Kutta Integrators

This appendix has been adapted from the original documentation [27].

C.1 Introduction

Efficiently integrating an ordinary differential equation numerically requires the ability to estimate the error on the approximate solution. This is necessary so that we may choose an appropriate time step which is small enough to keep the numerical error under control, but large enough so that we do not waste time computing the solution to unneeded precision. There are several techniques to obtain an estimate of the numerical error accumulated during a single step of a Runge-Kutta integrator. Typically, when the system under consideration has a conserved quantity such as energy or momentum we can use the error in this quantity as a proxy for the error on the position. However, it is also helpful to be able to estimate the error in the particles position directly. This can be accomplished through the use of embedded Runge-Kutta methods. To motivate the use of these more efficient methods we first will review two somewhat more primitive direct error estimation methods.

C.2 Step Doubling

The simplest and most straightforward method of numerical error estimation is called 'step-doubling'. In this method, we only need the use of a single Runge-Kutta routine. We would like to estimate the error on the solution y, as we propagate it from the n-th step to the $(n+1)$-th step over a step of size h_n, (i.e. given $y(x)$, integrate to find $y(x+h_n)$). So in order to estimate the numerical error we first take one step of size $h_1 = h_n$, which gives us some estimate of the exact solution $y(x+h_n)$. We will refer to this estimate as y_1. Then we repeat the process, however, rather than taking just one step of size h_n, we take two steps of size $h_2 = h_n/2$ This gives us another estimate of the exact solution $y(x+h_n)$. We will refer to this estimate as y_2.

Now, for the sake of argument lets assume our Runge-Kutta method is accurate to the 4th power of the step size. That is to say, that if we take a step of size h_n, then difference between the exact solution and the estimate, scales like h_n^5. So in the case of a 4th order Runge-Kutta method, we may relate the estimate of the solution y_1 after taking a step of $h_1 = h_n$ to the exact solution $y(x+h_n)$ by:

$$y(x+h_n) = \underbrace{y(x) + \Delta_{h_1}}_{y_1} + \phi(h_n)^5 + \mathcal{O}h_n^6 = y_1 + \alpha_1 \ . \tag{C.1}$$

Where α_1 is the numerical error on the estimate y_1. And similarly we may relate the estimate of the solution y_2 after taking a two steps of size $h_2 = h/2$, to the exact solution $y(x+h)$ by:

$$y\left(x+\frac{h_n}{2}\right) = y(x) + \Delta_{h_2} + \phi\left(\frac{h_n}{2}\right)^5 + \mathcal{O}h_n^6 \tag{C.2}$$

$$y(x+h_n) = y\left(x+\frac{h_n}{2}\right) + \Delta'_{h_2} + \phi'\left(\frac{h_n}{2}\right)^5 + \mathcal{O}h_n^6 \tag{C.3}$$

$$y(x+h_n) = \underbrace{y(x) + \Delta_{h_2} + \Delta'_{h_2}}_{y_2} + (\phi+\phi')\left(\frac{h_n}{2}\right)^5 + \mathcal{O}h_n^6 \tag{C.4}$$

$$y(x+h_n) = y_2 + (\phi+\phi')\left(\frac{h_n}{2}\right)^5 + \mathcal{O}h_n^6\,. \tag{C.5}$$

Now we note that we may obtain ϕ by equating orders of h_n^5 in a Taylor expansion of the true solution:

$$y(x+h_n) = y(x) + \sum_{m=1}^{\infty} \frac{h_n^m}{m!}\left(\frac{d^m y}{dx^m}\right)\bigg|_x = y_1 + \Delta_{h_1} + \phi(h_n)^5 + \mathcal{O}h_n^6\,. \tag{C.6}$$

Which tells us that that:

$$\phi = \frac{1}{5!}\left(\frac{d^5 y}{dx^5}\right)\bigg|_x\,, \tag{C.7}$$

and similarly for ϕ':

$$\phi' = \frac{1}{5!}\left(\frac{d^5 y}{dx^5}\right)\bigg|_{x+\frac{h_n}{2}}\,. \tag{C.8}$$

Now we may relate ϕ' to ϕ by Taylor expanding $\left(\frac{d^5 y}{dx^5}\right)$ about x, which gives:

$$\phi' = \frac{1}{5!}\left(\frac{d^5 y}{dx^5}\right)\bigg|_{x+\frac{h_n}{2}} = \frac{1}{5!}\left(\frac{d^5 y}{dx^5}\right)\bigg|_x + \sum_{m=1}^{\infty} \frac{(\frac{h_n}{2})^m}{m!}\left(\frac{d^m y}{dx^m}\left(\frac{d^5 y}{dx^5}\right)\right)\bigg|_x\,. \tag{C.9}$$

Now putting this expression for ϕ' back into equation (C.5) and only keeping terms up to 5th order in h_n, we now have:

$$y(x+h_n) = y_2 + 2\phi\left(\frac{h_n}{2}\right)^5 + \mathcal{O}h_n^6 = y_2 + \alpha_2\,, \tag{C.10}$$

where α_2 is the numerical error on the estimate y_2. So we see (ignoring terms of $\mathcal{O}h_n^6$ or higher), that the error on the y_1 estimate is $\alpha_1 \sim \phi h_n^5$, and the error on the y_2

estimate is $\alpha_2 \sim (\phi h_n^5)/16$. So, while we cannot compute the exact solution $y(x+h_n)$ to find the true error, we can obtain a good approximation of the numerical error α_1, by taking the difference of equations (C.1) and (C.10), to find:

$$|y_1 - y_2| = |\alpha_1 - \alpha_2| = \left|\phi h_n^5 - \frac{\phi h_n^5}{16}\right| = \left|\frac{15\phi h_n^5}{16}\right| \approx \left|\phi h_n^5\right| = |\alpha_1| \,. \tag{C.11}$$

We note that the estimate y_2 is more accurate than then estimate y_1, so if we were going to continue integrating the ordinary differential equation from the $(n+1)$-th step on to $(n+2)$-th step, it would seem prudent to take the estimate y_2 as y_{n+1}, rather than the estimate y_1. However, we have only calculated the error α_1 on y_1. Shouldn't we find α_2? In this particular case (4^{th} Runge-Kutta), we could find α_2 by simply dividing our estimate of α_1 by 16. However, in practice, this is not generally necessary, since α_1 is always larger than α_2, and for our purposes it is safe to over estimate the error on our solution.

So now suppose that we wish to integrate an equation with a 4^{th} order Runge-Kutta method, with the constraint that the absolute value of the estimated numerical error of each step should be no larger than a certain size $\beta > 0$. Given this constraint how should we the optimize our process of integration, so that we may take the largest step size possible without the numerical error α on the solution of each step exceeding β?

In order to do this we take note of the fact that our estimate of the numerical error depends on the step size h as:

$$|\alpha| \propto h^5 \,. \tag{C.12}$$

So, if we take a step of size h_a and obtain an error estimate of α, then the step size h_b that would have produced an error of size β is related by:

$$\left|\frac{h_b}{h_a}\right|^5 = \left|\frac{\beta}{\alpha}\right| \qquad \Rightarrow \qquad h_b = h_a \left|\frac{\beta}{\alpha}\right|^{1/5} \,. \tag{C.13}$$

Given this information, we can develop a basic process in which we should approach the integration of the initial value problem:

$$y(x_0) = y_0 \,, \qquad y'(x) = f(x, y(x)) \quad ,$$

while satisfying our error constraint. The process is:

1. Starting at some particular (y_n, x_n), compute an estimate of y_{n+1} for a given step size h_n. Call this estimate y_1.

2. Starting from the same (y_n, x_n), compute another estimate of y_{n+1} by taking two steps of size $h_n/2$. Call this estimate y_2.

3. Compute an estimate of the numerical error $\alpha = |y_1 - y_2|$.

4. Compute the step size h' that would have produced an error of size β, using $h' = \left(h_n \left|\frac{\beta}{\alpha}\right|^{(1/5)}\right)$.

5. If the error $\alpha < \beta$ then the step of size h was good. Then we can set $y_{n+1} = y_2$ and $x_{n+1} = x_n + h$ and repeat the process starting from (y_{n+1}, x_{n+1}) using h' as the new step size. However if $\alpha > \beta$ then the step of size h_n was unsatisfactory, and we must repeat the process starting from (y_n, x_n) with the new step size h'.

This basic process usually works well enough for some purposes. However, it by no means is the most efficient approach. First of all, using h' as the new step size for the next step is not particularly well suited. If we use h' then the error on the next step will be very close to β, and could possibly exceed it, which would result in an unnecessary recalculation of the next step. It is better to replace h' with Sh', where S is some 'safety-factor' which prevents h' from quickly becoming too large or too small. In practice, the authors of [194] have found a 'safety-factor' of $S \sim 0.9$ to be satisfactory for most purposes. This a simple enough fix, however there are other complications. For example, say the user, rather than desiring a fixed limit on the numerical error per step of β, would prefer that the *percent* error on each step not exceed some limit ϵ. If this is the case, then a naive solution would be to use the same process outlined previously, but replace β with some other error limit β_2 which varies with each step as $\beta_2 = \epsilon|y_n|$. However, this is deficient for the following reason. Consider what would happen if for some step $y_n = 0$. If this were to happen then the constraint on the numerical error of this step could never be satisfied without reducing the step size to zero, and our integrator could not proceed. One means of avoiding this problem would be to replace β_2 with some other $\beta_3 = \epsilon h_n|y'_n|$, or $\beta_4 = \epsilon(|y_n| + h_n|y'_n|)$. This way, even if there is a zero crossing in y, our error constraint will not be reduced to zero. However, this introduces a new problem. If we choose this means of calculating our error constraint, with β_i scaling with h_n, then the exponent $\frac{1}{5}$ in equation (C.13) is no longer correct. Instead we should use $\frac{1}{4}$. However if our routine has no way of know which of these methods are being used, a safe means of accounting for both possibilities [194] is to take:

$$h' = Sh_n\left|\frac{\beta_i}{\alpha}\right|^{1/4} \qquad \beta \geqslant \alpha\,, \tag{C.14}$$

$$h' = Sh_n\left|\frac{\beta_i}{\alpha}\right|^{1/5} \qquad \beta < \alpha\,. \tag{C.15}$$

That way whenever we decrease the step size we use the larger exponent, and whenever we increase the step size we use smaller exponent. This ensures we do not underestimate the factor by which the step size is decreased (if it needs to be decreased for the next attempt) and we do not overestimate the factor by which the step size is increased (if it needs to be increased for the next step). With these modifications to the process outlined previously, the 'step-doubling' method can be an effective means of adaptively changing the step size as one proceeds with integration of the initial value problem.

The largest drawback to method of 'step-doubling' is that it is not particularly efficient. The maximal number of evaluations of the derivative function for a 4th order Runge-Kutta method is four per step. However, the step doubling approach uses 12 evaluations for every step. For derivative functions that are computed cheaply this is not such a draw back. However if the derivative is very computationally expensive to calculate (i.e if it involves the evaluation of electric or magnetic fields) then our 'step-doubling' routine will be rather slow in comparison to a method which uses a constant step size, or a method which adapts its step size based on the amount of error of a more cheaply computed conserved quantity, such as energy.

C.3 Two Runge-Kutta Routines of Differing Order

An alternative means of adaptive step size control is to use two separate Runge-Kutta integrators of differing order to create an estimate of the numerical error. This is often more efficient than the 'step-doubling' approach because it generally requires fewer evaluations of the derivative per step.

An example of this would be the use of one 4^{th} and one 5^{th} order Runge-Kutta integrator. So to estimate the error on the solution y, as we propagate it from the n-th step to the $(n+1)$-th step over a step of size h_n we would simultaneously use both integrators to produce estimates y_4 and y_5 of the solution. These estimates are related to the exact solution $y(x+h_n)$ by:

$$y(x+h_n) = y_4 + \phi h_n^5 + \mathcal{O}h_n^6 \,, \tag{C.16}$$

$$y(x+h_n) = y_5 + \mathcal{O}h_n^6 \,. \tag{C.17}$$

Using these expressions we can immediately find an estimate of the numerical error α on our solution (ignoring terms of $\mathcal{O}h_n^6$) by taking:

$$|y_5 - y_4| = \phi h_n^5 = \alpha \tag{C.18}$$

Now in this particular case $\alpha \propto h_n^5$. So essentially the same approach we described when using the 'step-doubling' method to increment the solution, also applies here. The only difference being that in the previous case we took one step of size h_n to produce the first estimate of the solution, and then took two half steps of size $h_n/2$, to produce the second estimate. Where in this case, we instead take one step of size h_n, with the 4^{th} order integrator to produce the first estimate, and then repeat with a step of the same size with the 5^{th} order integrator to produce the second estimate. This method has all of the advantages of 'step-doubling' method, with the benefit of using less function evaluations per step. A optimal 4^{th} order Runge-Kutta method needs four function evaluations, while an optimal 5^{th} order method needs six. So this method is more efficient than the 'step-doubling' method, requiring only ten evaluations of the derivative per step, rather than twelve, to achieve comparable accuracy.

C.4 Embedded Runge-Kutta Routines

While the use of two separate Runge-Kutta routines with differing orders was seen to be more efficient than step-doubling. There is yet another even more efficient means of estimating the numerical error on each step. The downside of using two separate Runge-Kutta routines to produce an error estimate is that many of the derivative evaluations made over the course of taking a $(p-1)^{th}$ order step and a separate p^{th} order step are either repeated unnecessarily or lie very close to one another. While at first glance this feature appears to be unfortunate, if we are careful about choosing the coefficients of our p^{th} and $(p-1)^{th}$ Runge-Kutta methods we can reuse most of the function evaluations needed for the p^{th} order routine in the $(p-1)^{th}$ routine with perhaps one or two extra evaluations needed.

This technique, whereby a lower order Runge-Kutta integrator routine is 'embedded' in a higher order integrator, was first pioneered by E. Fehlberg [79]. The details of choosing the appropriate coefficients which minimize the number of derivative evaluations needed, and the numerical error on the estimate of the solution are far beyond the scope of this summary. However, we will give an example of the method. Those interested in further details are referred to [225], [65], and [81].

One of the simplest examples of the so called embedded Runge-Kutta-Fehlberg method is that which combines a 5^{th} and 4^{th} order Runge-Kutta routine. The

C_i	A_{ij}						
0							
1/4	1/4						
3/8	3/32	9/32					
12/13	1932/2197	-7200/2197	7296/2197				
1	439/216	-8	3680/513	-845/4104			
1/2	-8/27	2	-3544/2565	1859/4104	-11/40		
	25/216	0	1408/2565	2197/4104	-1/5	0	$\}\ B_j$
	16/135	0	6656/12825	28561/56430	-9/50	2/55	$\}\ \hat{B}_j$

Table C.1: Butcher tableau of coefficients for Fehlberg's dual 4^{th} and 5^{th} order embedded Runge-Kutta method.

coefficients of this method are given in the Butcher tableau of table C.1 [79]: Given these coefficients, the initial value problem:

$$y(x_0) = y_0 \,, \qquad y'(x) = f(x, y(x)) \quad ,$$

is then solved by repeatedly progressing from the n-th estimate (denoted y_n) of the exact solution $y(x_n)$ to the $(n+1)$-th estimate (denoted y_{n+1}) of the exact solution $y(x_{n+1}) = y(x_n + h_n)$. So, to find the 5^{th} order $(n+1)$-th estimate $\hat{y}_{n+1}$, given $\hat{y}_n$ we use [195]:

$$\hat{y}_{n+1} = \hat{y}_n + \sum_{i=1}^{6} \hat{B}_i k_i \,, \tag{C.19}$$

where the k_i, are given by:

$$k_1 = h_n f\left(x_n \,,\ \hat{y}\right) \,, \tag{C.20}$$

$$k_i = h_n f\left(x_n + C_i h_n \,,\ \hat{y} + \sum_{j=1}^{i-1} A_{ij} k_j\right) \,. \tag{C.21}$$

Using the same k_i we can also then obtain the 4^{th} order $(n+1)$-th estimate by:

$$y_{n+1} = \hat{y}_n + \sum_{i=1}^{6} B_i k_i \,, \tag{C.22}$$

So now we have two estimates of the exact solution $y(x_{n+1})$, one accurate to 4^{th} order, and the other accurate to 5^{th} order. So we can again obtain an estimate of the numerical error α on the exact solution $y(x_{n+1})$ by once again taking:

$$\alpha = |\hat{y}_{n+1} - y_{n+1}| \,. \tag{C.23}$$

Therefore, given some constraint on the numerical error β, we can also determine if the step was satisfactory and what the appropriate step size h' for the next attempt ought to be, through the following:

$$h' = Sh_n \left|\frac{\beta}{\alpha}\right|^{1/4} \qquad \beta \geqslant \alpha \,, \tag{C.24}$$

$$h' = Sh_n \left|\frac{\beta}{\alpha}\right|^{1/5} \qquad \beta < \alpha \,. \tag{C.25}$$

So we see that by using an 'embedded' Runge-Kutta routine we have solved the same problem using only six derivative evaluations per step, a great improvement over the ten or twelve evaluations per step the previous two methods needed.

C.5 Extensions to Embedded Runge Kutta Routines

So far in this summary we have only considered examples involving 4^{th} and 5^{th} order Runge-Kutta routines. However these methods can be generalized any Runge-Kutta pair of orders p and q, where $q \leqslant p-1$. Using pairs of order (p,q) we now may obtain an estimate on the numeric error α of the n-th step, by taking [219]:

$$\alpha = (h_n)^{p-q-1} \left|y_p - y_q\right| \,. \tag{C.26}$$

In calculating the next appropriate step size we now use [219]:

$$h' = Sh_n \left|\frac{\beta}{\alpha}\right|^{1/(p-1)} \qquad \beta \geqslant \alpha\,,$$
$$h' = Sh_n \left|\frac{\beta}{\alpha}\right|^{1/p} \qquad \beta < \alpha\,.$$

The generalization of this step size control method to the integration of an ordinary differential equation with k components is also relatively straightforward. In this case the user must define a constraint on the error β^i corresponding to i-th component of the solution of the ODE for each of the k components. The numeric error α^i on the i-th component y^i of the estimate of the solution $\vec{y}_n$, is simply:

$$\alpha^i = (h_n)^{p-q-1} \left|y^i_p - y^i_q\right| \,. \tag{C.27}$$

Now in determining the next appropriate step size, we need to ensure that we only increase the step size when the numeric error α^i is less than that of the user defined error constraint β^i for every one of the k components. If there is any component where α^i is larger than the user defined error constraint β^i then we must shrink the step size. So the rules for calculating the next appropriate step size h' become:

$$h' = Sh_n \left(\max_{i=1}^{k}\left|\frac{\beta^i}{\alpha^i}\right|\right)^{1/(p-1)} \qquad \text{if } \forall\ i \in \{1, \cdots k\} \text{ we have } \quad \beta^i \geqslant \alpha^i\,,$$
$$h' = Sh_n \left(\min_{i=1}^{k}\left|\frac{\beta^i}{\alpha^i}\right|\right)^{1/p} \qquad \text{if } \exists \text{ some } i \text{ where } \quad \beta^i < \alpha^i\,,$$

and we can efficiently propagate the solution while limiting the error on all componets.

Appendix D

ToF Trend Fits

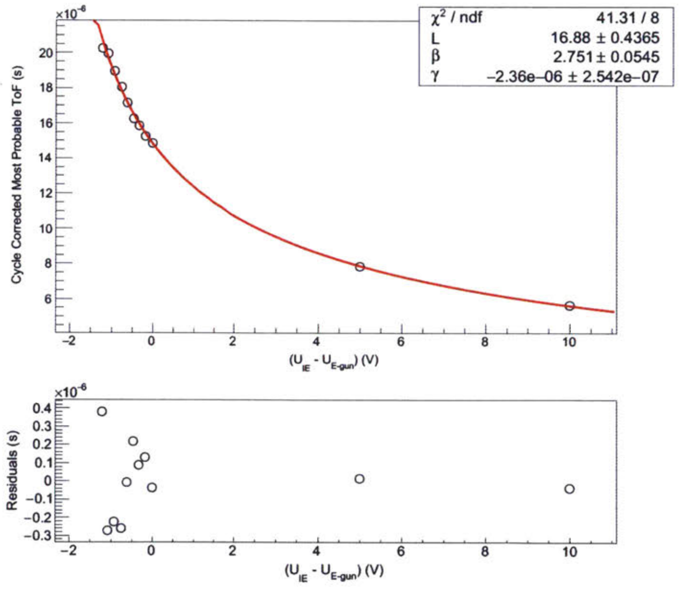


Figure D-1: ToF as a function of surplus energy, run #23839.

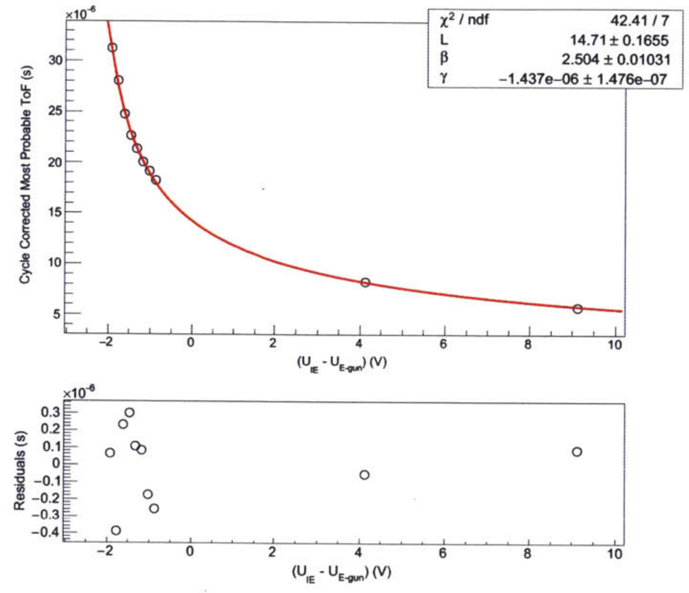


Figure D-2: ToF as a function of surplus energy, run #23844.

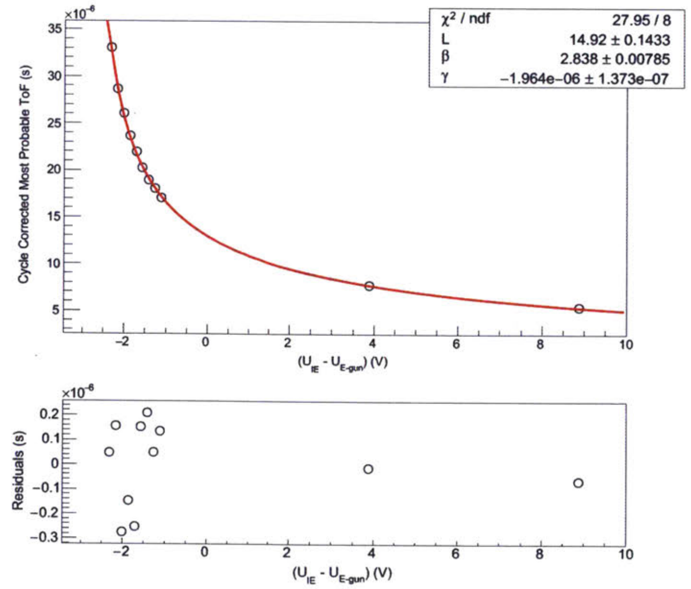


Figure D-3: ToF as a function of surplus energy, run #23847.

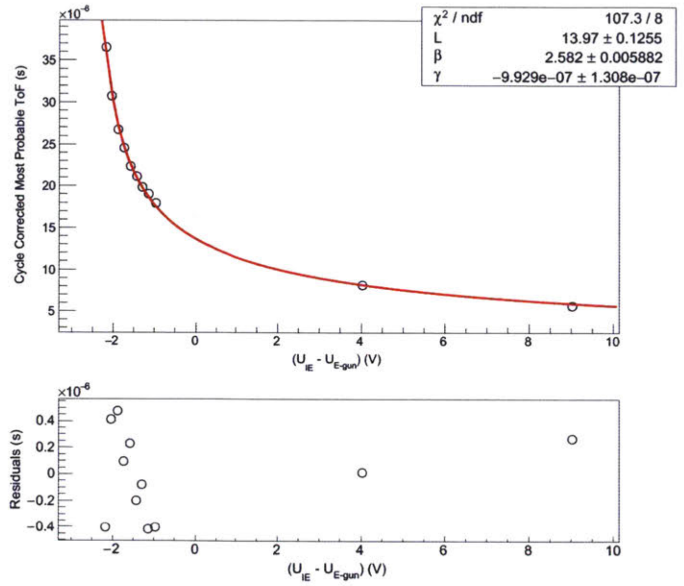


Figure D-4: ToF as a function of surplus energy, run #23850.

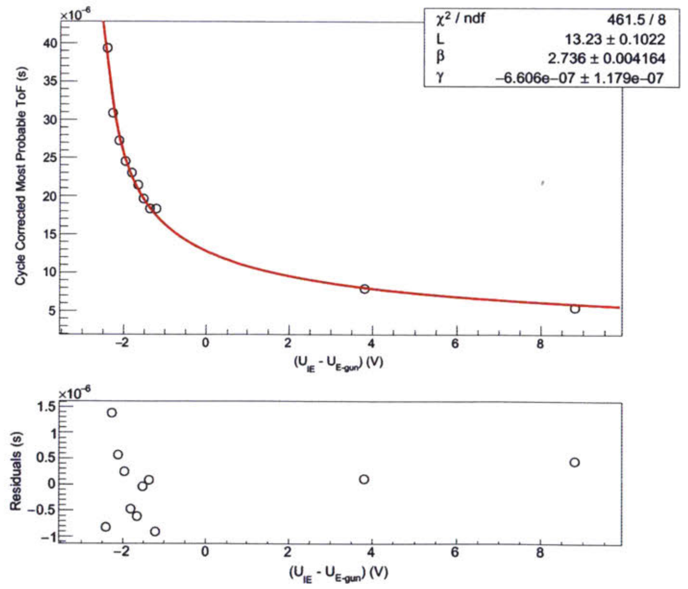


Figure D-5: ToF as a function of surplus energy, run #23863.

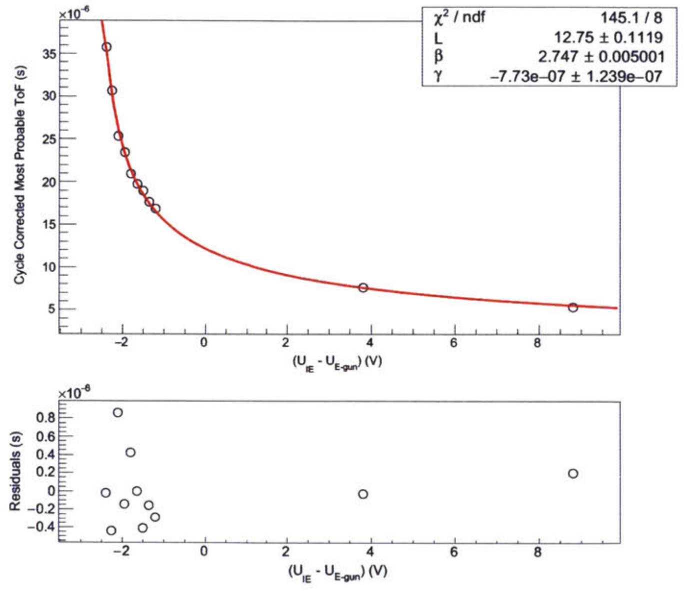


Figure D-6: ToF as a function of surplus energy, run #23866.

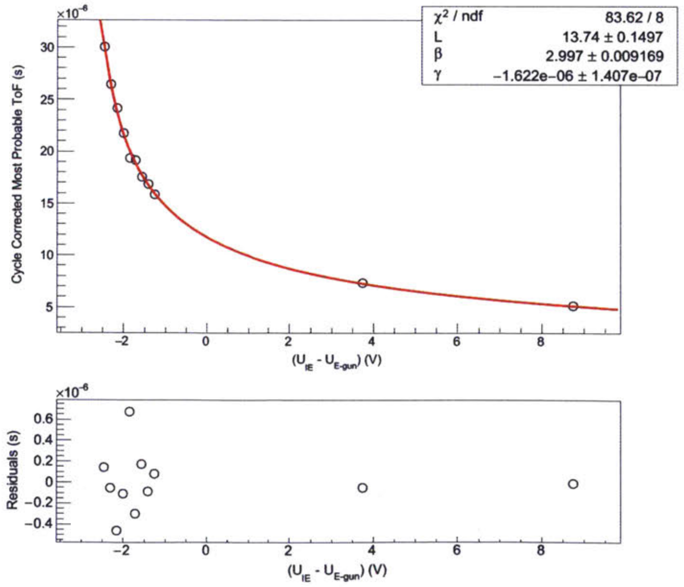


Figure D-7: ToF as a function of surplus energy, run #23869.

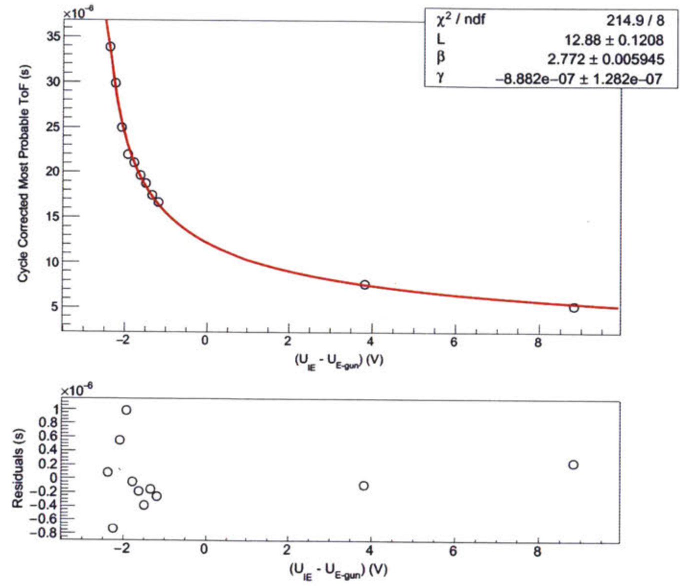


Figure D-8: ToF as a function of surplus energy, run #23872.

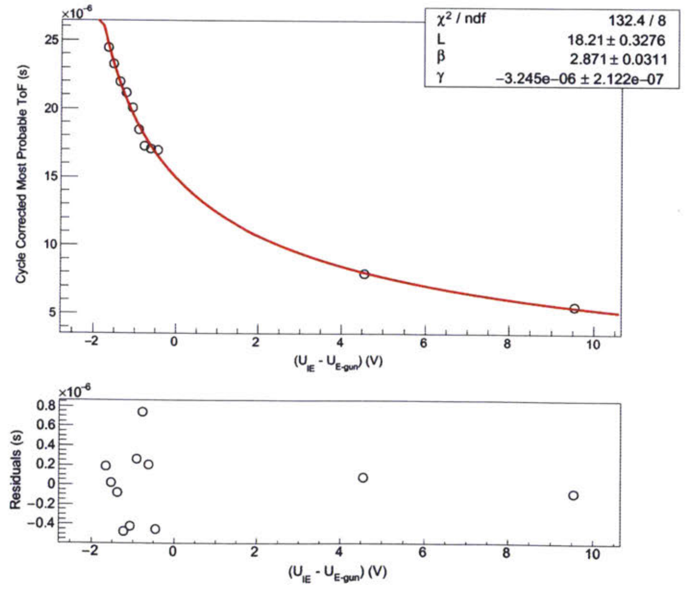


Figure D-9: ToF as a function of surplus energy, run #23933.

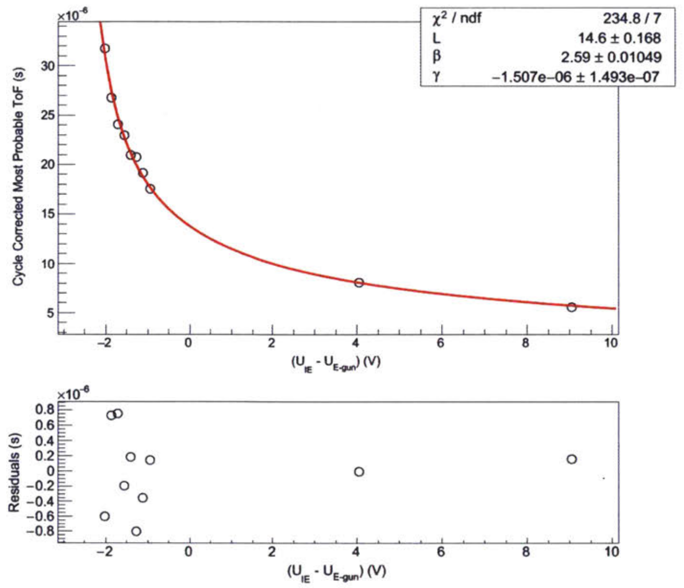


Figure D-10: ToF as a function of surplus energy, run #23938.

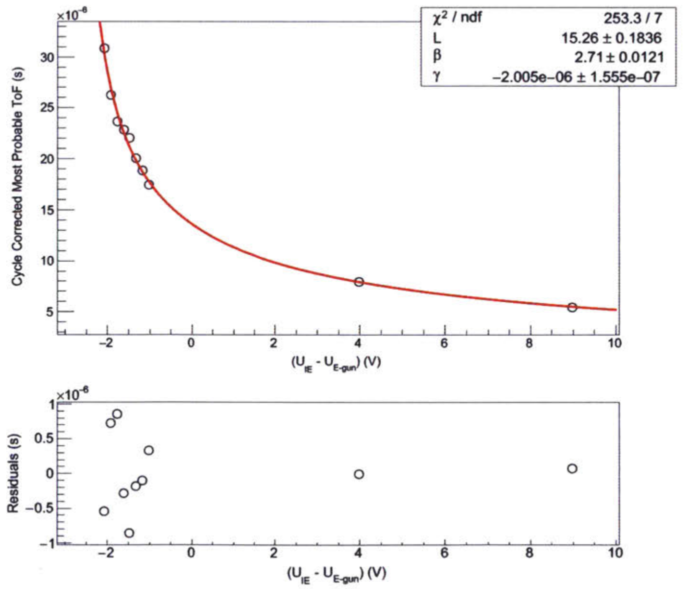


Figure D-11: ToF as a function of surplus energy, run #23941.

Bibliography

[1] Craig E Aalseth, FT Avignone III, A Barabash, F Boehm, Ronald L Brodzinski, JI Collar, PJ Doe, H Ejiri, SR Elliott, E Fiorini, et al. Comment on" Evidence for Neutrinoless Double Beta Decay". *Modern Physics Letters A*, 17(22):1475–1478, 2002.

[2] Kevork N Abazajian, MA Acero, SK Agarwalla, AA Aguilar-Arevalo, CH Albright, S Antusch, CA Arguelles, AB Balantekin, G Barenboim, V Barger, et al. Light sterile neutrinos: a white paper. *arXiv preprint arXiv:1204.5379*, 2012.

[3] Y Abe, Christoph Aberle, JC Dos Anjos, JC Barriere, M Bergevin, A Bernstein, TJC Bezerra, L Bezrukhov, E Blucher, NS Bowden, et al. Reactor ν^{-} e disappearance in the Double Chooz experiment. *Physical Review D*, 86(5):052008, 2012.

[4] Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Courier Dover Publications, 1966.

[5] PAR Ade, N Aghanim, C Armitage-Caplan, M Arnaud, M Ashdown, F Atrio-Barandela, J Aumont, C Baccigalupi, Anthony J Banday, RB Barreiro, et al. Planck 2013 results. XVI. Cosmological parameters. *Astronomy & Astrophysics*, 571:A16, 2014.

[6] QR Ahmad, RC Allen, TC Andersen, JD Anglin, JC Barton, EW Beier, M Bercovitch, J Bigu, SD Biller, RA Black, et al. Direct evidence for neutrino flavor transformation from neutral-current interactions in the Sudbury Neutrino Observatory. *Physical Review Letters*, 89(1):011301, 2002.

[7] Andrei Alexandrescu. *Modern C++ design: generic programming and design patterns applied*. Addison-Wesley, 2001.

[8] JF Amsbaugh, J Barrett, A Beglarian, T Bergmann, H Bichsel, LI Bodine, J Bonn, NM Boyd, TH Burritt, Z Chaoui, et al. Focal-plane detector system for the KATRIN experiment. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 778:40–60, 2015.

[9] FP An, JZ Bai, AB Balantekin, HR Band, D Beavis, W Beriguete, M Bishai, S Blyth, K Boddy, RL Brown, et al. Observation of electron-antineutrino disappearance at Daya Bay. *Physical Review Letters*, 108(17):171803, 2012.

[10] Rene Andrae, Tim Schulze-Hartung, and Peter Melchior. Dos and don'ts of reduced chi-squared. *arXiv preprint arXiv:1012.3754*, 2010.

[11] Andrew W Appel. An efficient program for many-body simulation. *SIAM Journal on Scientific and Statistical Computing*, 6(1):85–103, 1985.

[12] M Arenz, M Babutzka, M Bahr, JP Barrett, S Bauer, M Beck, A Beglarian, J Behrens, T Bergmann, U Besserer, et al. Commissioning of the vacuum system of the KATRIN Main Spectrometer. *arXiv preprint arXiv:1603.01014*, 2016.

[13] Douglas N Arnold and Wolfgang L Wendland. On the asymptotic convergence of collocation methods. *Mathematics of Computation*, 41(164):349–381, 1983.

[14] Walter Edwin Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9(1):17–29, 1951.

[15] VN Aseev, AI Belesev, AI Berlev, EV Geraskin, AA Golubev, NA Likhovid, VM Lobashev, AA Nozik, VS Pantuev, VI Parfenov, et al. Upper limit on the electron antineutrino mass from the Troitsk experiment. *Physical Review D*, 84(11):112003, 2011.

[16] VN Aseev, AI Belesev, AI Berlev, EV Geraskin, OV Kazachenko, Yu E Kuznetsov, VM Lobashev, RP Ostroumov, NA Titov, SV Zadorozhny, et al. Energy loss of 18 keV electrons in gaseous T and quench condensed D films. *The European Physical Journal D-Atomic, Molecular, Optical and Plasma Physics*, 10(1):39–52, 2000.

[17] VN Aseev, AI Belesev, AI Berlev, EV Geraskin, OV Kazachenko, Yu E Kuznetsov, VM Lobashev, RP Ostroumov, NA Titov, SV Zadorozhny, et al. Energy loss of 18 keV electrons in gaseous T and quench condensed D films. *The European Physical Journal D-Atomic, Molecular, Optical and Plasma Physics*, 10(1):39–52, 2000.

[18] David M Asner, RF Bradley, L de Viveiros, Peter J Doe, Justin L Fernandes, M Fertl, Erin C Finn, JA Formaggio, D Furse, Anthony M Jones, et al. Single-electron detection and spectroscopy via relativistic cyclotron radiation. *Physical review letters*, 114(16):162501, 2015.

[19] D Ayres, NOvA Collaboration, et al. NOvA proposal to build a 30 kiloton off-axis detector to study neutrino oscillations in the Fermilab NuMI beamline. *arXiv preprint hep-ex/0503053*, 2005.

[20] John N Bahcall. Solar models: An historical overview. *International Journal of Modern Physics A*, 18(22):3761–3776, 2003.

[21] John N. Bahcall, Hitoshi Murayama, and Carlos Pena-Garay. What can we learn from neutrinoless double beta decay experiments? *Phys. Rev.*, D70:033012, 2004.

[22] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*, volume 11. Siam, 2000.

[23] Allison H Baker, Elizabeth R Jessup, and Thomas Manteuffel. A technique for accelerating the convergence of restarted GMRES. *SIAM Journal on Matrix Analysis and Applications*, 26(4):962–984, 2005.

[24] Abhijit Bandyopadhyay, Sandhya Choubey, Srubabati Goswami, and Kamales Kar. Impact of the first SNO results on neutrino mass and mixing. *Physics Letters B*, 519(1):83–92, 2001.

[25] Josh Barnes and Piet Hut. A hierarchical O (N log N) force-calculation algorithm. *Nature*, 324(6096):446–449, 1986.

[26] R. E. Barnhill and J. A. Gregory. Polynomial Interpolation to Boundary Data on Triangles. *Mathematics of Computation*, 29(131):pp. 726–735, 1975.

[27] J Barrett. Embedded Runge-Kutta Steppers for KMath, 2010. Available from: `https://fuzzy.fzk.de/bscw/bscw.cgi/d717168/EmbeddedRungeKutta-JBarrett.pdf`.

[28] John Barrett, Joseph A Formaggio, and Thomas Joseph Corona. A Method to Calculate the Spherical Multipole Expansion of the Electrostatic Charge Distribution on a Triangular Boundary Element. *Progress In Electromagnetics Research B*, 63:123–143, 2015.

[29] H Barth, Lutz Bornschein, Beate Degen, L Fleischmann, Michael Przyrembel, Hartmut Backe, Alexander Bleile, Jochen Bonn, Daphne Goldmann, Michael Gundlach, et al. Status and perspectives of the Mainz neutrino mass experiment. *Progress in Particle and Nuclear Physics*, 40:353–376, 1998.

[30] John A Beachy and William D Blair. *Abstract algebra*. Waveland Press, 2006.

[31] G Beamson, HQ Porter, and DW Turner. The collimating and magnifying properties of a superconducting field photoelectron spectrometer. *Journal of Physics E: Scientific Instruments*, 13(1):64, 1980.

[32] Rick Beatson and Leslie Greengard. A short course on fast multipole methods. *Wavelets, multilevel methods and elliptic PDEs*, 1:1–37, 1997.

[33] CL Bennett, AJ Banday, KM Gorski, G Hinshaw, P Jackson, P Keegstra, A Kogut, George F Smoot, DT Wilkinson, and EL Wright. Four-year COBE DMR cosmic microwave background observations: maps and basic results. *Astrophysical Journal*, 464:L1–L4, 1996.

[34] Karl-Erik Bergkvist. A high-luminosity, high-resolution study of the end-point behaviour of the tritium β-spectrum (I). basic experimental procedure and analysis with regard to neutrino mass and neutrino degeneracy. *Nuclear Physics B*, 39:317–370, 1972.

[35] Jarle Berntsen, Terje O. Espelid, and Alan Genz. An Adaptive Algorithm for the Approximate Calculation of Multiple Integrals. *ACM Transactions on Mathematical Software*, 17(4):437–451, Dec 1991.

[36] Samoil M Bilenky. Bruno Pontecorvo and Neutrino Oscillations. *Advances in High Energy Physics*, 2013, 2013.

[37] Samoil M Bilenky and B Pontecorvo. Lepton mixing and neutrino oscillations. *Physics Reports*, 41(4):225–261, 1978.

[38] John Markus Blatt and Victor Frederick Weisskopf. *Theoretical nuclear physics*. Springer Science & Business Media, 2012.

[39] LI Bodine, DS Parno, and RGH Robertson. Assessment of molecular effects on neutrino mass measurements from tritium β decay. *Physical Review C*, 91(3):035505, 2015.

[40] Marc Bonnet, Giulio Maier, and Castrenze Polizzotto. Symmetric Galerkin boundary element methods. *Applied Mechanics Reviews*, 51(11):669–704, 1998.

[41] F Capozzi, GL Fogli, E Lisi, A Marrone, D Montanino, and A Palazzo. Status of three-neutrino oscillation parameters, circa 2013. *Physical Review D*, 89(9):093018, 2014.

[42] David J Cartwright. *Underlying principles of the boundary element method*. WIT, 2001.

[43] James Chadwick. Possible existence of a neutron. *Nature*, 129(3252):312, 1932.

[44] G. Chen and J. Zhou. *Boundary element methods*. Computational mathematics and applications. Academic Press, 1992.

[45] Jie Chen, Lois Curfman McInnes, and Hong Zhang. Analysis and practical use of flexible BICGSTAB. *Preprint ANL/MCS-P3039-0912, Argonne National Laboratory*, 2012.

[46] Hongwei Cheng, Leslie Greengard, and Vladimir Rokhlin. A fast adaptive multipole algorithm in three dimensions. *Journal of computational physics*, 155(2):468–498, 1999.

[47] Cheol Ho Choi, Joseph Ivanic, Mark S Gordon, and Klaus Ruedenberg. Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion. *The Journal of Chemical Physics*, 111(19):8825–8831, 1999.

[48] Eleanor Chu and Alan George. *Inside the FFT Black Box: Serial and Parallel Fast Fourier Transform Algorithms*. CRC Press, 1999.

[49] Bruce T Cleveland, Timothy Daily, Raymond Davis Jr, James R Distel, Kenneth Lande, CK Lee, Paul S Wildenhain, and Jack Ullman. Measurement of the solar electron neutrino flux with the Homestake chlorine detector. *The Astrophysical Journal*, 496(1):505, 1998.

[50] KATRIN Collaboration. KATRIN Beamline Image. Available from: `https://fuzzy.fzk.de/bscw/bscw.cgi/d826402/Beamline%20kpl%20-%20schraeg.jpg`.

[51] KATRIN Collaboration. KATRIN design report 2004. *FZKA report*, 7090, 2004.

[52] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[53] Thomas Corona. *Methodology and application of high performance electrostatic field simulation in the KATRIN experiment*. PhD thesis, University of North Carolina, Chapel Hill, 2014.

[54] Thomas Joseph Corona. Tools for electromagnetic field simulation in the KATRIN experiment. Master's thesis, Massachusetts Institute of Technology, 2009.

[55] Clyde L Cowan, Frederick Reines, FB Harrison, HW Kruse, and AD McGuire. Detection of the free neutrino: A Confirmation. *Science*, 124(3212):103–104, 1956.

[56] GR Cowper. Gaussian quadrature formulas for triangles. *International Journal for Numerical Methods in Engineering*, 7(3):405–408, 1973.

[57] Gaillard Danby, JM Gaillard, Konstantin Goulianos, LM Lederman, N Mistry, M Schwartz, and J Steinberger. Observation of high-energy neutrino reactions and the existence of two kinds of neutrinos. *Physical Review Letters*, 9(1):36, 1962.

[58] M Danos and LC Maximon. Multipole matrix elements of the translation operator. *Journal of Mathematical Physics*, 6(5):766–778, 1965.

[59] Raymond Davis Jr, Don S Harmer, and Kenneth C Hoffman. Search for neutrinos from the sun. *Physical Review Letters*, 20(21):1205, 1968.

[60] André de Gouvêa, James Jenkins, and Boris Kayser. Neutrino mass hierarchy, vacuum oscillations, and vanishing $|U_{e3}|$. *Physical Review D*, 71(11):113009, 2005.

[61] Eric De Sturler. Truncation strategies for optimal Krylov subspace methods. *SIAM Journal on Numerical Analysis*, 36(3):864–889, 1999.

[62] D Decamp, B Deschizeaux, J-P Lees, M-N Minard, JM Crespo, M Delfino, E Fernandez, M Martinez, R Miquel, Ll M Mir, et al. A precise determination of the number of families with light neutrinos and of the Z boson partial widths. *Physics Letters B*, 235(3):399–411, 1990.

[63] Henry P Decell, Jr. An application of the Cayley-Hamilton theorem to generalized matrix inversion. *SIAM review*, 7(4):526–528, 1965.

[64] James W Demmel. *Applied numerical linear algebra*. SIAM, 1997.

[65] J. R. Dormand and P. J. Prince. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19 – 26, 1980. Available from: `http://www.sciencedirect.com/science/article/B6TYH-4JS88G2-3/2/502fdefd423026c582b43f0e5e7b3963`.

[66] G Drexlin, V Hannen, S Mertens, and C Weinheimer. Current direct neutrino mass experiments. *Advances in High Energy Physics*, 2013, 2013.

[67] Moshe Dubiner. Spectral methods on triangles and other domains. *Journal of Scientific Computing*, 6(4):345–390, 1991.

[68] Dan E Dudgeon and Russell M Mersereau. *Multidimensional Digital signal Processing*. Prentice-Hall, 1984, 1984.

[69] Michael G Duffy. Quadrature over a pyramid or cube of integrands with a singularity at a vertex. *SIAM Journal on Numerical Analysis*, 19(6):1260–1262, 1982.

[70] Jessica A. Dunmore. Pixelation-Dependence of Transmission Function. Technical report, 2007. Available from: `https://fuzzy.fzk.de/bscw/bscw.cgi/d419247/Pixelation%20dependence%20of%20the%20transmission%20function.pdf`.

[71] Lawrence N Dworsky. *Introduction to Numerical Electrostatics Using MATLAB*. John Wiley & Sons, 2014.

[72] G. Wanner E. Hairer, S. P. Nørsett. *Solving ordinary differential equations I*. Springer-Verlag, Berlin, 2008.

[73] A. R. Edmonds. *Angular Momentum in Quantum Mechanics*. Princeton University Press, 1958.

[74] SR Elliott, AA Hahn, and MK Moe. Direct evidence for two-neutrino double-beta decay in Se 82. *Physical Review Letters*, 59(18):2020, 1987.

[75] Steven R Elliott and Jonathan Engel. Double-beta decay. *Journal of Physics G: Nuclear and Particle Physics*, 30(9):R183, 2004. Available from: `http://stacks.iop.org/0954-3899/30/i=9/a=R01`.

[76] Steven R. Elliott and Petr Vogel. Double beta decay. *Ann. Rev. Nucl. Part. Sci.*, 52:115–151, 2002.

[77] William D Elliott and John A Board, Jr. Fast Fourier transform accelerated fast multipole algorithm. *SIAM Journal on Scientific Computing*, 17(2):398–415, 1996.

[78] Michael A Epton and Benjamin Dembart. Multipole translation theory for the three-dimensional Laplace and Helmholtz equations. *SIAM Journal on Scientific Computing*, 16(4):865–897, 1995.

[79] E. Fehlberg. Klassische Runge-Kutta-Formeln vierter und niedrigerer Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme. *Computing*, 6:61 – 71, 1970. Available from: `http://www.springerlink.com/content/f69256402h71w8m8`.

[80] Richard P Feynman and Murray Gell-Mann. Theory of the Fermi interaction. *Physical Review*, 109(1):193, 1958.

[81] S. Filippi and J. Gräf. New Runge-Kutta-Nystrom formula-pairs of order 8(7), 9(8), 10(9) and 11(10) for differential equations of the form y'' = f(x, y). *Journal of Computational and Applied Mathematics*, 14(3):361 – 370, 1986. Available from: `http://www.sciencedirect.com/science/article/B6TYH-45DHJ4W-2J/2/0a1f1ea94da1da4fdf65c4c280bd8530`.

[82] Sebastian Fischer et al. Monitoring of tritium purity during long-term circulation in the KATRIN test experiment LOOPINO using laser Raman spectroscopy. *Fusion Science and Technology*, 60(3):925–930, 2011.

[83] JA Formaggio and J Barrett. Resolving the reactor neutrino anomaly with the KATRIN neutrino experiment. *Physics Letters B*, 706(1):68–71, 2011.

[84] Joseph A Formaggio, Predrag Lazić, TJ Corona, H Štefančic, Hrvoje Abraham, and F Glück. Solving for Micro-and Macro-Scale Electrostatic Configurations Using the Robin Hood Algorithm. *Progress in Electromagnetics Research B*, 39, 2012.

[85] Daniel Lawrence Furse. *Techniques for direct neutrino mass measurement utilizing tritium [beta]-decay*. PhD thesis, Massachusetts Institute of Technology, 2015.

[86] Edgar Gabriel, Graham E Fagg, George Bosilca, Thara Angskun, Jack J Dongarra, Jeffrey M Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, et al. Open MPI: Goals, concept, and design of a next generation MPI implementation. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 97–104. Springer, 2004.

[87] George Gamow and Edward Teller. Selection Rules for the β-Disintegration. *Physical Review*, 49(12):895, 1936.

[88] W. Gander. Change of basis in polynomial interpolation. *Numerical Linear Algebra with Applications*, 12(8):769–778, 2005.

[89] Bernd Gärtner. Fast and robust smallest enclosing balls. In *Algorithms-ESA'99*, pages 325–338. Springer, 1999.

[90] Anushree Ghosh, Tarak Thakore, and Sandhya Choubey. Determining the neutrino mass hierarchy with INO, T2K, NOvA and reactor experiments. *Journal of High Energy Physics*, 2013(4):1–33, 2013.

[91] Walton C Gibson. *The method of moments in electromagnetics*. CRC press, 2 edition, 2014.

[92] Z Gimbutas and L Greengard. A fast and stable method for rotating spherical harmonic expansions. *Journal of Computational Physics*, 228(16):5621–5627, 2009.

[93] Andrea Giuliani and Alfredo Poves. Neutrinoless double-beta decay. *Advances in High Energy Physics*, 2012, 2012.

[94] Carlo Giunti and Chung W Kim. *Fundamentals of neutrino physics and astrophysics*. Oxford university press, 2007.

[95] Sheldon L Glashow and Murray Gell-Mann. Gauge theories of vector particles. *Annals of Physics*, 15(3):437–460, 1961.

[96] Andrew S Glassner. Space subdivision for fast ray tracing. *IEEE Computer Graphics and applications*, 4(10):15–24, 1984.

[97] Ferenc Gluck. Axisymmetric magnetic field calculation with zonal harmonic expansion. *Progress In Electromagnetics Research B*, 32:351–388, 2011.

[98] Nabil Gmati and Bernard Philippe. Comments on the GMRES convergence for preconditioned systems. In *Large-scale scientific computing*, pages 40–51. Springer, 2008.

[99] Maria Goeppert-Mayer. Double beta-disintegration. *Physical Review*, 48(6):512, 1935.

[100] I Gohberg and V Olshevsky. Complexity of multiplication with vectors for structured matrices. *Linear Algebra and Its Applications*, 202:163–192, 1994.

[101] Maurice Goldhaber, L Grodzins, and AW Sunyar. Helicity of neutrinos. *Physical Review*, 109(3):1015, 1958.

[102] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[103] Gene H Golub and John H Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation*, 23(106):221–230, 1969.

[104] M. C. Gonzalez-Garcia, Michele Maltoni, Jordi Salvado, and Thomas Schwetz. Global fit to three neutrino mixing: critical look at present precision. *Journal of High Energy Physics*, 2012(12):1–24, 2012. Available from: `http://www.nu-fit.org/?q=node/115`.

[105] M Concepción González-Garciá and Yosef Nir. Developments in neutrino physics. *Rev. Mod. Phys.*, 75(hep-ph/0202058):345–402, 2002.

[106] Leslie Greengard and Vladimir Rokhlin. A fast algorithm for particle simulations. *Journal of computational physics*, 73(2):325–348, 1987.

[107] Leslie Greengard and Vladimir Rokhlin. The rapid evaluation of potential fields in three dimensions. *Vortex Methods*, pages 121–141, 1988.

[108] Leslie Greengard and Vladimir Rokhlin. A new version of the fast multipole method for the Laplace equation in three dimensions. *Acta numerica*, 6:229–269, 1997.

[109] David Jeffrey Griffiths and Reed College. *Introduction to electrodynamics*, volume 3. prentice Hall Upper Saddle River, NJ, 1999.

[110] Stefan Groh. *Modeling of the response function and measurement of transmission properties of the KATRIN experiment*. PhD thesis, KIT, February 2015.

[111] S Grohmann, T Bode, H Schön, and M Süsser. Precise temperature measurement at 30K in the KATRIN source cryostat. *Cryogenics*, 51(8):438–445, 2011.

[112] Khronos OpenCL Working Group et al. The OpenCL Specification, version 1.0. 29, 2008. *URL: http://khronos. org/registry/cl/specs/opencl-1.0*, 29.

[113] Pawel Guzowski. A combined limit for neutrinoless double-beta decay. *arXiv preprint arXiv:1504.08285*, 2015.

[114] Fabian Harms. *Characterization and Minimization of Background Processes in the KATRIN Main Spectrometer*. PhD thesis, KIT, October 2015.

[115] Johan Helsing and Karl-Mikael Perfekt. On the polarizability and capacitance of the cube. *Applied and Computational Harmonic Analysis*, 34(3):445–468, 2013.

[116] Magnus Rudolph Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. 1952.

[117] Jason E Hicken and David W Zingg. A simplified and flexible variant of GCROT for solving nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 32(3):1672–1694, 2010.

[118] G Hinshaw, D Larson, E Komatsu, DN Spergel, CL Bennett, J Dunkley, MR Nolta, M Halpern, RS Hill, N Odegard, et al. Nine-year Wilkinson Microwave Anisotropy Probe (WMAP) observations: cosmological parameter results. *The Astrophysical Journal Supplement Series*, 208(2):19, 2013.

[119] Markus Hötzel. *Simulation and analysis of source-related effects for KATRIN*. PhD thesis, KIT-Bibliothek, 2012.

[120] Markus Hötzel. *Simulation and analysis of source-related effects for KATRIN*. PhD thesis, KIT, November 2012.

[121] George C Hsiao. Boundary element methods, an Overview. *Applied numerical mathematics*, 56(10):1356–1369, 2006.

[122] Ralph Gorton Hudson and Joseph Lipka. *A table of integrals*. John Wiley & Sons, 1917.

[123] Masaki Ishitsuka, Takaaki Kajita, Hisakazu Minakata, and Hiroshi Nunokawa. Resolving the neutrino mass hierarchy and CP degeneracy by two identical detectors with different baselines. *Phys. Rev. D*, 72:033003, Aug 2005. Available from: `http://link.aps.org/doi/10.1103/PhysRevD.72.033003`.

[124] J. Behrens T.J. Corona G. Drexlin S. Enomoto M. Erhard F. Fränkle F. Glück S. Görhardt S. Groh M. Haag V. Hannen F. Harms D. Hilk A. Kopmann M. Kraus M. Krause B. Leiber A. Müller O. Rest P. Rovedo J. Schwarz N. Stallkamp N. Steinbrink T. Thümmler N. Trost N. Wandkowsky C. Weinheimer K. Wierman J. Wolf M. Zache J. Barrett, A. Beglarian. Results of the first KATRIN SDS measurement phase. Technical report, February 2014. Available from: `https://fuzzy.fzk.de/bscw/bscw.cgi/d875500/SDSPhase1Report.pdf`.

[125] N. Steinbrink J. Barrett, S. Groh. SDS-1 E-log entry #259. Technical report, September 2013. Available from: `https://neutrino.ikp.kit.edu:8080/SDS-Measurements/259`.

[126] K. Wierman J. Barrett O. Rest D. Hilk J. Behrens, M. Erhard. SDS-2 E-log entry #124. Technical report, March 2015. Available from: `https://neutrino.ikp.kit.edu:8080/SDS-Measurements+Phase+2/124`.

[127] M. Kraus P. Ranitzsch D. Winzen J. Behrens, M. Erhard. SDS-2 E-log entry #88. Technical report, February 2015. Available from: `https://neutrino.ikp.kit.edu:8080/SDS-Measurements+Phase+2/88`.

[128] S. Enomoto J. Behrens. SDS-2 E-log entry #26. Technical report, November 2014. Available from: `https://neutrino.ikp.kit.edu:8080/SDS-Measurements+Phase+2/26`.

[129] John David Jackson. *Classical Electrodynamics*. John Wiley and Sons Inc., 3 edition, 1999.

[130] T.J. Corona F. Glück S. Groh N. Wandkowsky Ch. Weinheimer J.D. Behrens, N. Steinbrink. EMD consequences of electrical shorts at the inner electrode system. Technical report, March 2013. Available from: `https://fuzzy.fzk.de/bscw/bscw.cgi/775422`.

[131] Carsten Jensen and Finn Aaserud. *Controversy and consensus: nuclear beta decay 1911-1934*, volume 24. Springer Science & Business Media, 2000.

[132] Jian-Ming Jin. *Theory and computation of electromagnetic fields*. John Wiley & Sons, 2011.

[133] Wayne Joubert. On the convergence behavior of the restarted GMRES algorithm for solving nonsymmetric linear systems. *Numerical linear algebra with applications*, 1(5):427–447, 1994.

[134] Wolfgang Käfer. *Sensitivity studies for the KATRIN experiment*. PhD thesis, KIT, January 2012.

[135] Wolfgang Käfer. *Sensitivity studies of the KATRIN experiment*. PhD thesis, Ph. D. thesis, Karlsruhe Institute of Technology, 2012.

[136] Takaaki Kajita and Yoji Totsuka. Observation of atmospheric neutrinos. *Rev. Mod. Phys.*, 73:85–118, Jan 2001. Available from: `http://link.aps.org/doi/10.1103/RevModPhys.73.85`.

[137] Tsuyoshi Kato, Shinichiro Omachi, and Hirotomo Aso. Asymmetric gaussian and its application to pattern recognition. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 405–413. Springer, 2002.

[138] HV Klapdor-Kleingrothaus, A Dietz, HL Harney, and IV Krivosheina. Evidence for neutrinoless double beta decay. *Modern Physics Letters A*, 16(37):2409–2420, 2001.

[139] Marco Kleesiek. *A Data-Analysis and Sensitivity-Optimization Framework for the KATRIN Experiment*. PhD thesis, Karlsruhe, Karlsruher Institut für Technologie (KIT), Diss., 2014, 2014.

[140] Marco Kleesiek. *A Data-Analysis and Sensitivity-Optimization Framework for the KATRIN Experiment*. PhD thesis, KIT, September 2014.

[141] K Kodama, N Ushida, C Andreopoulos, N Saoulidou, G Tzanakos, P Yager, B Baller, D Boehnlein, W Freeman, B Lundberg, et al. Observation of tau neutrino interactions. *Physics Letters B*, 504(3):218–224, 2001.

[142] Tom Koornwinder. Two-variable analogues of the classical orthogonal polynomials. In *Theory and application of special functions (Proc. Advanced Sem., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1975)*, pages 435–495. Academic Press New York, 1975.

[143] Joachim Kopp, Pedro AN Machado, Michele Maltoni, and Thomas Schwetz. Sterile neutrino oscillations: the global picture. *Journal of High Energy Physics*, 2013(5):1–52, 2013.

[144] Richard E Korf. Multi-Way Number Partitioning. In *IJCAI*, pages 538–543. Citeseer, 2009.

[145] Kenneth S Krane and David Halliday. *Introductory nuclear physics*, volume 465. Wiley New York, 1988.

[146] Ch Kraus, B Bornschein, L Bornschein, J Bonn, B Flatt, A Kovalik, B Ostrick, EW Otten, JP Schall, Th Thümmler, et al. Final results from phase II of the Mainz neutrino mass search in tritium β decay. *The European Physical Journal C-Particles and Fields*, 40(4):447–468, 2005.

[147] Prem Kythe. *Fundamental solutions for differential operators and applications*. Springer Science & Business Media, 2012.

[148] H. Rechenberg L. M. Brown. *The Origin of the Concept of Nuclear Forces*. CRC Press, 1968.

[149] LM Langer and RJD Moffat. The beta-spectrum of tritium and the mass of the neutrino. *Physical Review*, 88(4):689, 1952.

[150] P Lazić, D Dujmić, Joseph A Formaggio, H Abraham, and H Štefancić. New approach to 3D electrostatic calculations for micro-pattern detectors. *Journal of Instrumentation*, 6(12):P12003, 2011.

[151] Predrag Lazić, Hrvoje Štefančić, and Hrvoje Abraham. The Robin Hood method–A novel numerical method for electrostatic problems based on a non-local charge transfer. *Journal of Computational Physics*, 213(1):117–140, 2006.

[152] Predrag Lazić, Hrvoje Štefančić, and Hrvoje Abraham. The Robin Hood method–A new view on differential equations. *Engineering analysis with boundary elements*, 32(1):76–89, 2008.

[153] Tsung-Dao Lee and Chen-Ning Yang. Question of parity conservation in weak interactions. *Physical Review*, 104(1):254, 1956.

[154] C Lessig, T De Witt, and E Fiume. Efficient and accurate rotation of finite spherical harmonics expansions. *Journal of Computational Physics*, 231(2):243–250, 2012.

[155] Frank G Lether. Computation of double integrals over a triangle. *Journal of Computational and Applied Mathematics*, 2(3):219–224, 1976.

[156] Yu-Feng Li, Jun Cao, Yifang Wang, and Liang Zhan. Unambiguous determination of the neutrino mass hierarchy using reactor neutrinos. *Phys. Rev. D*, 88:013008, Jul 2013. Available from: `http://link.aps.org/doi/10.1103/PhysRevD.88.013008`.

[157] Jörg Liesen and Zdenek Strakos. *Krylov subspace methods: principles and analysis*. Oxford University Press, 2012.

[158] Jörg Liesen and Petr Tichý. Convergence analysis of Krylov subspace methods. *GAMM-Mitteilungen*, 27(2):153–173, 2004.

[159] Kian Meng Lim, Xuefei He, and Siak Piang Lim. Fast Fourier transform on multipoles (FFTM) algorithm for Laplace equation with direct and indirect boundary element method. *Computational Mechanics*, 41(2):313–323, 2008.

[160] Yijun Liu. *Fast multipole boundary element method: theory and applications in engineering*. Cambridge university press, 2009.

[161] VM Lobashev, VN Aseev, AI Belesev, AI Berlev, EV Geraskin, AA Golubev, OV Kazachenko, Yu E Kuznetsov, RP Ostroumov, LA Rivkis, et al. Direct search for mass of neutrino and anomaly in the tritium beta-spectrum. *Physics Letters B*, 460(1):227–235, 1999.

[162] VM Lobashev and PE Spivak. A method for measuring the electron antineutrino rest mass. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 240(2):305–310, 1985.

[163] Larry L Lucas and Michael P Unterweger. Comprehensive review and critical evaluation of the half-life of Tritium. *Journal of research-National institute of standards and technology*, 105(4):541–550, 2000.

[164] Cecilia Lunardini and Alexei Yu Smirnov. Probing the neutrino mass hierarchy and the 13-mixing with supernovae. *Journal of Cosmology and Astroparticle Physics*, 2003(06):009, 2003. Available from: `http://stacks.iop.org/1475-7516/2003/i=06/a=009`.

[165] V Lyubimov, E Novikov, V Nozik, E Tret'yakov, V Kozik, and N Myasoedov. Estimation of the neutrino rest mass from measurements of the tritium beta spectrum. *Sov. Phys.-JETP (Engl. Transl.);(United States)*, 54(4), 1981.

[166] Ziro Maki, Masami Nakagawa, and Shoichi Sakata. Remarks on the unified model of elementary particles. *Progress of Theoretical Physics*, 28(5):870–880, 1962.

[167] Evgeny Margolis and Yonina C Eldar. Nonuniform sampling in polar coordinates with applications to computerized tomography. In *Electrical and Electronics Engineers in Israel, 2004. Proceedings. 2004 23rd IEEE Convention of*, pages 372–375. IEEE, 2004.

[168] Farokh Marvasti. *Nonuniform sampling: theory and practice*. Springer Science & Business Media, 2012.

[169] John C Mason and David C Handscomb. *Chebyshev polynomials*. Chapman & Hall/CRC, 2002.

[170] Samina S Masood, Salah Nasri, Joseph Schechter, María Amparo Tórtola, José WF Valle, and Christian Weinheimer. Exact relativistic β decay endpoint spectrum. *Physical Review C*, 76(4):045501, 2007.

[171] Olga Mena and Stephen Parke. Unified graphical summary of neutrino mixing parameters. *Physical Review D*, 69(11):117301, 2004.

[172] G Mention, M Fechner, Th Lasserre, Th A Mueller, D Lhuillier, M Cribier, and A Letourneau. Reactor antineutrino anomaly. *Physical Review D*, 83(7):073006, 2011.

[173] S Mertens, T Lasserre, S Groh, G Drexlin, F Glueck, A Huber, AWP Poon, M Steidl, N Steinbrink, and C Weinheimer. Sensitivity of next-generation tritium beta-decay experiments for keV-scale sterile neutrinos. *Journal of Cosmology and Astroparticle Physics*, 2015(02):020, 2015.

[174] MD Messier. First neutrino oscillation measurements in NOvA. *Nuclear Physics B*, 908:151–160, 2016.

[175] Rabindra N Mohapatra, Stefan Antusch, KS Babu, Gabriela Barenboim, Mu-Chun Chen, A De Gouvêa, P De Holanda, B Dutta, Y Grossman, A Joshipura, et al. Theory of neutrinos: a white paper. *Reports on Progress in Physics*, 70(11):1757, 2007.

[176] M-H Mousa, Raphaëlle Chaine, Samir Akkouche, and Eric Galin. Toward an efficient triangle-based spherical harmonics representation of 3D objects. *Computer Aided Geometric Design*, 25(8):561–575, 2008.

[177] M Nakagawa. Birth of neutrino oscillation. *Arxiv preprint hep-ph/9811358*, 1998.

[178] Masayuki Nakahata. Kamiokande and Super-Kamiokande. *AAPPS Bulletin*, 13(4):7–12, 2003.

[179] Wandkowsky Nancy. *Study of background and transmission properties of the KATRIN spectrometers*. PhD thesis, KIT, July 2013.

[180] ET Ong, KM Lim, KH Lee, and HP Lee. A fast algorithm for three-dimensional potential fields calculation: fast Fourier transform on multipoles. *Journal of Computational Physics*, 192(1):244–261, 2003.

[181] EW Otten and Ch Weinheimer. Neutrino mass limit from tritium β decay. *Reports on Progress in Physics*, 71(8):086201, 2008.

[182] RG Owens. Spectral approximations on the triangle. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1971):857–872, 1998.

[183] Aigli Papantonopoulou. *Algebra: Pure & Applied.* Prentice Hall, 2002.

[184] Wolfgang Pauli. On the earlier and more recent history of the neutrino. In *Writings on Physics and Philosophy*, pages 193–218. Springer, 1994.

[185] Benjamin Osgood Peirce. *A short table of integrals.* Ginn & company, 1910.

[186] Arno A Penzias and Robert Woodrow Wilson. A Measurement of Excess Antenna Temperature at 4080 Mc/s. *The Astrophysical Journal*, 142:419–421, 1965.

[187] Martin L Perl. The discovery of the tau lepton. In *History of Original Ideas and Basic Discoveries in Particle Physics*, pages 277–302. Springer, 1996.

[188] David G Phillips, Till Bergmann, Thomas J Corona, Florian Fränkle, Mark A Howe, Matthias Kleifges, Andreas Kopmann, Michelle Leber, Alexander Menshikov, Denis Tcherniakhovski, et al. Characterization of an FPGA-Based DAQ System in the KATRIN Experiment. In *IEEE Nuclear Science Symposuim & Medical Imaging Conference*, pages 1399–1403. IEEE, 2010.

[189] Didier Pinchon and Philip E Hoggan. Rotation matrices for real spherical harmonics: general rotations of atomic orbitals in space-fixed axes. *Journal of Physics A: Mathematical and Theoretical*, 40(7):1597, 2007.

[190] Dragan Poljak and Carlos A Brebbia. *Boundary element methods for electrical engineers*, volume 4. WIT Press, 2005.

[191] BM Pontecorvo. Inverse β-processes and non-conservation of lepton charge. Technical report, Joint Inst. for Nuclear Research, Moscow (USSR). Lab. of Nuclear Problems, 1957.

[192] Frank C Porter. Testing consistency of two histograms. *arXiv preprint arXiv:0804.0380*, 2008.

[193] Povh, Rith, Scholz, and Zetsche. *Particles and Nuclei, An Introduction to the Physical Concepts.* Springer, 2010.

[194] William Press, Saul Teukolsky, William Vetterling, and Brian Flannery. *Numerical Recipes in C.* Cambridge University Press, Cambridge, UK, 2nd edition, 1992.

[195] P. J. Prince and J. R. Dormand. High order embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 7(1):67 – 75, 1981. Available from: `http://www.sciencedirect.com/science/article/B6TYH-4JS88FS-B/2/e4dabebc068a171dfac49f5db4394a3d`.

[196] G Prior, SNO Collaboration, et al. Results from the Sudbury Neutrino Observatory Phase III. *Lawrence Berkeley National Laboratory*, 2009.

[197] B. Pritychenko. On Double-Beta Decay Half-Life Time Systematics. 2010.

[198] Joseph Proriol. Sur une famille de polynomes á deux variables orthogonaux dans un triangle. *CR Acad. Sci. Paris*, 245:2459–2461, 1957.

[199] Pascal Renschler. *KESS-A new Monte Carlo simulation code for low-energy electron interactions in silicon detectors.* PhD thesis, Karlsruhe, Karlsruher Institut für Technologie (KIT), Diss., 2011, 2011.

[200] Jorge Revelles, Carlos Urena, and Miguel Lastra. An efficient parametric algorithm for octree traversal. 2000.

[201] RGH Robertson, TJ Bowles, GJ Stephenson Jr, DL Wark, John Franklin Wilkerson, and DA Knapp. Limit on ν_e mass from observation of the β decay of molecular tritium. *Physical review letters*, 67(8):957, 1991.

[202] Vladimir Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of Computational Physics*, 60(2):187–207, 1985.

[203] Youcef Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[204] Youcef Saad and Martin H Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869, 1986.

[205] Yousef Saad. *Iterative methods for sparse linear systems.* SIAM, 2003.

[206] Alejandro Saenz, Svante Jonsell, and Piotr Froelich. Improved molecular final-state distribution of HeT+ for the β-decay process of T_2. *Physical review letters*, 84(2):242, 2000.

[207] Abdus Salam and N Svartholm. Elementary particle theory. *Almquist and Wiksell, Stockholm*, page 367, 1968.

[208] RC Salgo and HH Staub. Re-determination of the β-energy of tritium and its relation to the neutrino rest mass and the Gamow-Teller matrix element. *Nuclear Physics A*, 138(2):417–428, 1969.

[209] Dorothea FE Samtleben. KM3NeT/ORCA status and plans. In *EPJ Web of Conferences*, volume 116, page 11008. EDP Sciences, 2016.

[210] Johannes Schwarz. *The Detector System of the KATRIN Experiment – Implementation and First Measurements with the Spectrometer.* PhD thesis, KIT, July 2014.

[211] Henry Stark. Sampling theorems in polar coordinates. *JOSA*, 69(11):1519–1525, 1979.

[212] Martin Babutzka John Barrett Jan Behrens Thomas Corona Sanshiro Enomoto Moritz Erhard Joseph Formaggio Ferenc Glück Marco Kleesiek Fabian Harms Daniel Hilk Susanne Mertens Noah S. Oblath Pascal Renschler Christian Pommranz Johannes Schwarz Nikolaus Trost Nancy Wandkowsky Stefan Groh, Daniel Furse and Kevin Wierman. Kassiopeia: A Modern, Extensible C++ Particle Tracking Package. To be published, 2016.

[213] EO Steinborn and K Ruedenberg. Rotation and translation of regular and irregular solid spherical harmonics. *Adv. Quantum Chem*, 7:1–81, 1973.

[214] Nicholas Steinbrink, Volker Hannen, Eric L Martin, RG Hamish Robertson, Michael Zacher, and Christian Weinheimer. Neutrino mass sensitivity by MAC-E-Filter based time-of-flight spectroscopy with the example of KATRIN. *New Journal of Physics*, 15(11):113020, 2013.

[215] Roger H Stuewer. The Seventh Solvay Conference: Nuclear Physics at the Crossroads. In *No Truth Except in the Details*, pages 333–362. Springer, 1995.

[216] Miklos Szilagyi. *Electron and ion optics*. Springer, 1988.

[217] Robert L Taylor. On completeness of shape functions for finite element analysis. *International Journal for Numerical Methods in Engineering*, 4(1):17–22, 1972.

[218] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. SIAM, 1997.

[219] CH. Tsitouras and S. N. Papakostas. Cheap Error Estimation for Runge–Kutta Methods. *SIAM Journal on Scientific Computing*, 20(6):2067–2088, 1999. Available from: `http://link.aip.org/link/?SCE/20/2067/1`.

[220] L. Josten H.W. Ortjohann C. Weinheimer D. Winzen M. Zacher M. Zboril V. Hannen, S. Bauer. Main Spectrometer Electron-Gun Specification. Technical report.

[221] M. Zacher V. Hannen, H.W. Ortjohann and C. Weinheimer. Electrical short circuits in the main spectrometer wire electrode. Technical report.

[222] K Valerius. The wire electrode system for the KATRIN main spectrometer. *Progress in Particle and Nuclear Physics*, 64(2):291–293, 2010.

[223] Henk A Van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing*, 13(2):631–644, 1992.

[224] Martin Van Gelderen. The shift operators and translations of spherical harmonics. *DEOS Progress Letter*, 1:57–67, 1998.

[225] J. H. Verner. A Contrast of Some Runge–Kutta Formula Pairs. *SIAM Journal on Numerical Analysis*, 27(5):1332–1344, 1990. Available from: `http://link.aip.org/link/?SNA/27/1332/1`.

[226] R. Wait and A.R. Mitchell. *Finite Element Analysis and Applications*. Books on Demand, 1985.

[227] Steven Weinberg. A model of leptons. *Physical review letters*, 19(21):1264, 1967.

[228] Christian Weinheimer and Kai Zuber. Neutrino masses. *Annalen der Physik*, 525(8-9):565–575, 2013.

[229] Wolfgang L Wendland. *Boundary element methods and their asymptotic convergence*. Springer, 1983.

[230] Christopher A White and Martin Head-Gordon. Rotating around the quartic angular momentum barrier in fast multipole method calculations. *The Journal of Chemical Physics*, 105:5061, 1996.

[231] Eugene Wigner and Griffin J. J. *Group theory and its application to the quantum mechanics of atomic spectra*. Academic Press, New York, 1959.

[232] John Franklin Wilkerson, TJ Bowles, JC Browne, MP Maley, RGH Robertson, JS Cohen, RL Martin, DA Knapp, and JA Helffrich. Limit on $\bar{\nu}_e$ Mass from Free-Molecular-Tritium Beta Decay. *Physical review letters*, 58(20):2023, 1987.

[233] Fred L Wilson. Fermi's theory of beta decay. *American Journal of Physics*, 36(12):1150–1160, 1968.

[234] Lincoln Wolfenstein. CP properties of Majorana neutrinos and double beta decay. *Physics Letters B*, 107(1-2):77–79, 1981.

[235] Chien-Shiung Wu, Ernest Ambler, RW Hayward, DD Hoppes, and R Pl Hudson. Experimental test of parity conservation in beta decay. *Physical review*, 105(4):1413, 1957.

[236] Michael Zacher. *Electromagnetic design and field emission studies for the inner electrode system of the KATRIN main spectrometer*. PhD thesis, Diploma thesis, University of Münster, 2009.

[237] Yu G Zdesenko, Fedor A Danevich, and VI Tretyak. Has neutrinoless double β decay of 76 Ge been really observed? *Physics Letters B*, 546(3):206–215, 2002.

[238] A Zee. A theory of lepton number violation and neutrino Majorana masses. *Physics Letters B*, 93(4):389–393, 1980.