Advances in Fully-Kinetic PIC Simulations of a Near-Vacuum Hall Thruster and Other Plasma Systems

by

Justin M. Fox
B.S., California Institute of Technology (2003)
S.M., Massachusetts Institute of Technology (2005)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

[September 2007]
June 2007

Author . . . . [illegible] . . . .
Department of Aeronautics and Astronautics
June 11, 2007

Certified by . . . . [illegible] . . . .
Manuel Martinez-Sanchez
Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by . . . .
Oleg Batishchev
Principal Research Scientist
Thesis Supervisor

Accepted by . . . . [illegible] . . . .
Jaime Peraire
Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

Advances in Fully-Kinetic PIC Simulations of a Near-Vacuum Hall Thruster and Other Plasma Systems

by

Justin M. Fox

Submitted to the Department of Aeronautics and Astronautics
on June 11, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Aeronautics and Astronautics

Abstract

In recent years, many groups have numerically modeled the near-anode region of a Hall thruster in attempts to better understand the associated physics of thruster operation. Originally, simulations assumed a continuum approximation for electrons and used magnetohydrodynamic fluid equations to model the significant processes. While these codes were computationally efficient, their applicability to non-equilibrated regions of the thruster, such as wall sheaths, was limited, and their accuracy was predicated upon the notion that the energy distributions of the various species remained Maxwellian at all times. The next generation of simulations used the fully-kinetic particle-in-cell (PIC) model. Although much more computationally expensive than the fluid codes, the full-PIC codes allowed for non-equilibrated thruster regions and did not rely on Maxwellian distributions. However, these simulations suffered for two main reasons. First, due to the high computational cost, fine meshing near boundaries which would have been required to properly resolve wall sheaths was often not attempted. Second, PIC is inherently a statistically noisy method and often the extreme tails of energy distributions would not be adequately sampled due to high energy particle dissipation.

The current work initiates a third generation of Hall thruster simulation. A PIC-Vlasov hybrid model was implemented utilizing adaptive meshing techniques to enable automatically scalable resolution of fine structures during the simulation. The code retained the accuracy and versatility of a PIC simulation while intermittently recalculating and smoothing particle distribution functions within individual cells to ensure full velocity space coverage. In addition, this simulation extended the state of the art in Hall thruster anomalous diffusion modeling by adopting a "quench rule" which is able to predict the spatial and temporal structure of the cross-field transport without the aid of prior empirical data. Truly predictive computations are thus enabled. After being thoroughly tested and benchmarked, the simulation was then applied to the near vacuum Hall thruster recently constructed at MIT. Recommendations to improve that thruster's performance were made based on the simulation's results, and those optimizations are being experimentally implemented by other researchers.

This work was conducted with the aid of Delta Search Labs' supercomputing facility and technical expertise. The simulation was fully-parallelized using MPI and tested on a 128 processor SGI Origin machine. We gratefully acknowledge that funding for portions of this work has been provided by the United States Air Force and the National Science Foundation.

Thesis Supervisor: Manuel Martinez-Sanchez
Title: Professor of Aeronautics and Astronautics

Thesis Supervisor: Oleg Batishchev
Title: Principal Research Scientist

Acknowledgments

What a long road it has been…one that I could never have walked alone. I want to first thank my advisor Professor Martinez-Sanchez for his invaluable guidance and for the opportunity to conduct this research. For Dr. Oleg Batishchev, my co-advisor, and his wife Dr. Alla Batishcheva, my "boss," words cannot even express the depth of my gratitude. Two more hard-working, generous, and kind-hearted individuals I could never hope to meet again, and I can't thank them enough for everything they have done for me and my wife. I hope this thesis brings them some small amount of happiness and pride, for without their steadfast support, none of this would have been possible!

I must further thank my colleagues in the Space Propulsion Lab for all of their help and friendship along the way. I wish them all luck wherever their next adventures may lead. A very special thanks also to Kamal Jaffrey and my other friends at Delta Search Labs for their generosity, laughter, and faithful support throughout my research here at MIT. I also must acknowledge the National Science Foundation for funding the majority of this research.

Of course, last but certainly not least, I'd like to thank my family. I am so fortunate to have those around me who have been willing to set aside their own happiness and own agendas to give me the chance to make this long journey. My mother and father have sacrificed so much for their two sons over the years, and I am filled with nothing but the profoundest pride, respect, and love for them. No child could have been more loved. To my newly-wedded wife who lovingly followed me to this strange northern land and braved its frigid wonders with an open and adoring heart, thank you! I can't wait to begin our next adventure together, hand in hand, forever and always.

And if there's anyone I haven't mentioned, it's only because there's not enough paper in the world to thank everyone who has helped me along the way! Thank you all so very much!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is a Plasma?

This thesis details the development of a newly-created accurate, scalable, and general simulation for the study of plasma systems and their associated physical phenomena. Since the conventional researcher of Aeronautics and Astronautics may not, as I did not, have extensive exposure to that mysterious fourth state of matter, our first step is to lay a very basic, common foundation in plasma physics before discussing the detailed operation of our model.

A plasma is simply a gas in which, usually under the influence of extremely high temperatures, a sizable portion of the atoms exist in an ionized or dissociated state. This implies that one or more electrons in the atom has been torn from its usual one-to-one relationship with a nucleus and is instead drifting free. This occurs because at high temperatures, the process of recombination, where an electron becomes reattached to an ion, is dominated by the process of ionization, in which a collision with a free electron gives an attached electron enough energy to break the ion's hold on it.

Of course the above definition is not exact, and indeed, the definition of a plasma is not without ambiguity. What specifically do we mean when we say "a sizable portion" of the atoms are ionized? To understand the answer to this question, we must first understand the concept of a Debye length. This quantity can in principle be understood as a scale length over which a perturbation to the electric potential in a plasma is reduced due to the plasma's tendency to rearrange its charges to counteract that disturbance. Take for example a situation in which an imaginary metal grid set to a certain specific electric potential, ϕ_g, is placed within an otherwise neutral, quiescent, and infinite in extent plasma. Since the electrons are initially in thermal equilibrium at a temperature of say T_e, given in electronvolts, they will exist in a Boltzmann distribution [42]:

$$n_e = n_\infty \exp\left[\frac{e\phi}{T_e}\right] \tag{1.1}$$

This is simply a consequence of statistics and is discussed more in the relevant references. The ion density, on the other hand, can be assumed to be initially unperturbed by the grid's potential since these particles are many times heavier and slow to respond. Thus, the ion density is uniform, $n_i = n_\infty$. Next, we use Poisson's Equation (Equation *1.6* below) to discern the shape of the potential given the distribution of the electric charges:

$$\frac{d^2\phi}{dx^2} = \frac{en_\infty}{\varepsilon_0}\left(\exp\left[\frac{e\phi}{T_e}\right] - 1\right) \tag{1.2}$$

Making the approximation that far from the disturbance the quantity $e\phi/T_e << 1$, this equation can then be solved to find that the potential varies as:

$$\phi = \phi_g \exp\left[\frac{-|x|}{\lambda_D}\right] \tag{1.3}$$

where in these units, the Debye length:

$$\lambda_D \equiv \sqrt{\frac{\varepsilon_0 T_e}{e^2 n_\infty}} \tag{1.4}$$

It is the scale length over which the potential is reduced due to the shielding effect of the plasma charges. This derivation of the Debye length and others can also be found in much greater detail in innumerable plasma physics references [42][43][75].

This then gives us an approximate metric by which we can better define a plasma. A plasma usually must have a physical size that is much greater than its Debye length. In addition, the number of particles in a sphere with radius on the order of the Debye length must be much larger than 1. This latter quantity is known as the plasma parameter and the condition can be written as:

$$n\frac{4}{3}\pi\lambda_D^3 >> 1 \tag{1.5}$$

1.2 General Equations of Plasmas

Now that we have a slightly better idea of the animal that our simulation wishes to study, let us now briefly discuss its habits and the stimuli that cause it to react. Because of the charged nature of the plasma, it is susceptible to the full-range of electromagnetic forces. Thus, in the general case, the full set of Maxwell's equations must be used to calculate the motion of the plasma particles under the influence of external and self-created electric and magnetic fields. A separate branch of the present simulation was developed by Batishcheva and deals exclusively with these full electromagnetic cases. The portion of the simulation which will be discussed in this thesis, however, assumes that the magnetic fields applied on the plasma are static in time. In addition, those magnetic fields which originate through the flow of plasma currents are assumed to be negligible. Under this electrostatic assumption then, Maxwell's equations can be reduced to the single Poisson's Equation relating the charge distribution in the plasma to the potential:

$$\frac{d^2\phi}{dx^2} = \frac{\rho(x)}{\varepsilon_0} \tag{1.6}$$

where ρ refers to the density of charges and is a function of the spatial location. The permittivity of free space, ε_0, is a physical constant related to the speed of light in a vacuum. This allows us to solve just one equation per time step in order to update the electric field while keeping the magnetic field unchanged.

Under the influence of such electric and magnetic fields, charged particles experience what is known as the Lorentz force which gives us an equation of motion:

$$m\frac{dv}{dt} = q(E + v \times B) \tag{1.7}$$

This equation, under various conditions of electric and magnetic fields, is responsible for the numerous peculiarities of motion associated with plasmas, such as the ExB drift and the Hall currents from which the thruster's under study in thesis draw their name. Full analyses of these motions can be found in many plasma texts [42][43][75]. For the present we merely note that our simulation attempts to solve through physical approximation the full set of non-linear kinetic equations which govern plasma motion:

$$\begin{cases} \frac{\partial f_e}{\partial t} + \vec{\mathrm{v}}\frac{\partial f_e}{\partial \vec{\mathrm{r}}} + q_e(\vec{\mathrm{E}} + \frac{\vec{\mathrm{v}} \times \vec{\mathrm{B}}}{\mathrm{c}})\frac{\partial f_e}{\partial \vec{\mathrm{p}}} = C_e \\ \frac{\partial f_i}{\partial t} + \vec{\mathrm{v}}\frac{\partial f_i}{\partial \vec{\mathrm{r}}} + q_i(\vec{\mathrm{E}} + \frac{\vec{\mathrm{v}} \times \vec{\mathrm{B}}}{\mathrm{c}})\frac{\partial f_i}{\partial \vec{\mathrm{p}}} = C_i \\ \frac{\partial f_N}{\partial t} + \vec{\mathrm{v}}\frac{\partial f_N}{\partial \vec{\mathrm{r}}} = C_N \end{cases} \tag{1.8}$$

The collisional terms here, namely C_e, C_i, and C_N, account for various inelastic and elastic types of collisions such as charge exchange, excitation, and ionization. For example, the ionization collisional term might take the form:

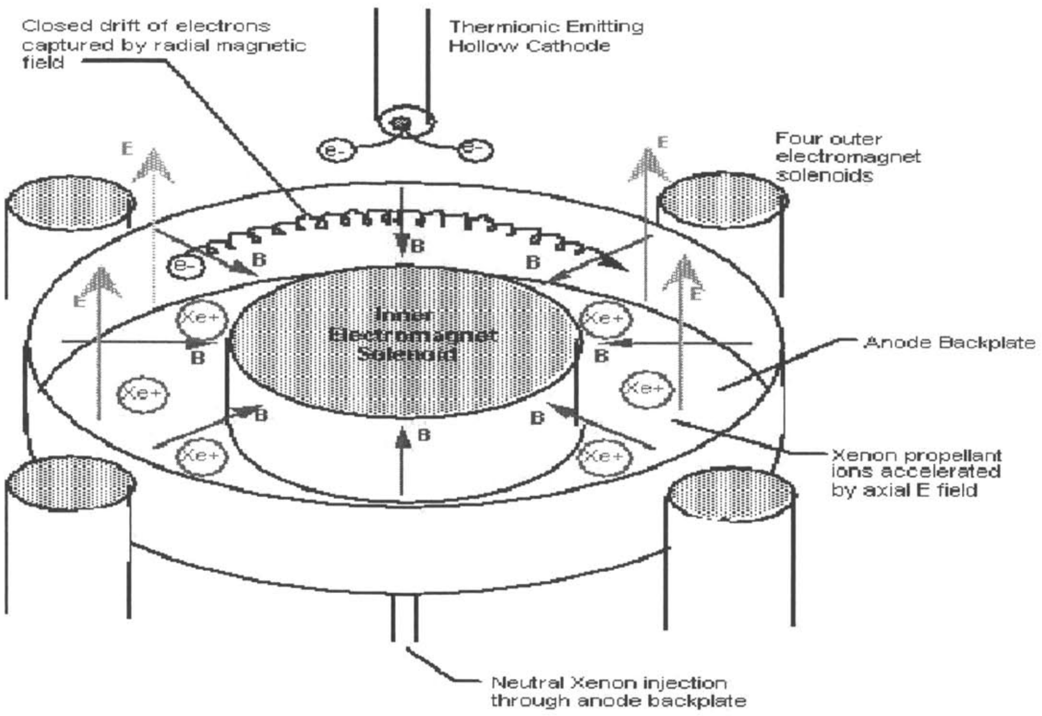
$$\begin{gathered} C_e^I = N\,\mathrm{v}\,\{-\sigma_I(\mathrm{v})\,f_e(\mathrm{v})\theta(\mathrm{v} - V_I) + \\ \sigma_I(a_I\mathrm{v})a_I^3\,f_e(a_I\mathrm{v})\} + G_2(\mathrm{v})\int_{V_I}^{\infty} N\,\mathrm{v}\,\sigma_I(\mathrm{v})\,f_e d\vec{\mathrm{p}} \end{gathered} \tag{1.9}$$

By approximating the solutions to these equations self-consistently, we can in turn calculate a distribution of charge which allows us to calculate the fields and forces on the particles, closing the loop.

1.3 Hall-Effect Thrusters

Experimentation with Hall-effect thrusters began independently in both the United States and the former Soviet Union during the 1960's. While American attentions quickly reverted to seemingly more efficient ion engine designs, Russian scientists continued to struggle with Hall thruster technology, advancing it to flight-ready status by 1971 when the first known Hall thruster space test was conducted. With the relatively recent increase in communication between the Russian and American scientific communities, American interest has once more been piqued by the concept of a non-gridded ion-accelerating thruster touting specific impulses in the range of efficient operation for typical commercial North-South-station-keeping (NSSK) missions.

A schematic diagram of a Hall thruster is shown in Figure 1-1. The basic configuration involves an axially symmetric hollow channel centered on a typically iron electromagnetic core. This channel may be relatively long and lined with ceramic in the case of an SPT (Stationary Plasma Thruster) type engine or metallic and much shorter in the case of a TAL (Thruster with an Anode Layer). Surrounding the channel are more iron electromagnets configured in such a way as to produce a more or less radial magnetic field. A hollow cathode is generally attached outside the thruster producing electrons through thermionic emission into a small fraction of the propellant gas. A portion of the electrons created in this way accelerate along the electric potential toward the thruster channel where they collide with and ionize the neutral gas atoms being emitted near the anode plate. Trapped electrons then feel an $E \times B$ force due to the applied axial electric field and the radial magnetic field, causing them to drift azimuthally and increasing their residence time. The ions, on the other hand, have a much greater mass and therefore much larger gyration radius, and so are not strongly affected by the magnetic field. The major portion of these ions is then accelerated out of the channel at high velocities producing thrust. Farther downstream, a fraction of the remaining cathode electrons are used to neutralize the positive ion beam which emerges from the thruster.



Closed drift of electrons captured by radial magnetic field
Thermionic Emitting Hollow Cathode
Four outer electromagnet solenoids
Inner Electromagnet Solenoid
Anode Backplate
Xenon propellant ions accelerated by axial E field
Neutral Xenon injection through anode backplate
E
B
e-
Xe+

Figure 1-1: A schematic representation of a Hall thruster.

The Xenon ion exit velocity of a Hall thruster can easily exceed 20,000 m/s, translating into an Isp of over 2000s. This high impulse to mass ratio is characteristic of electric propulsion devices and can be interpreted as a significant savings in propellant mass necessary for certain missions. In addition to this beneficial trait, Hall thrusters can be relatively simply and reliably engineered. Unlike ion engines, there is no accelerating grid to be eroded by ion sputtering which could enable fully mature Hall thrusters to outlast their ion engine brethren. Finally, Hall-effect thrusters are capable of delivering an order of magnitude higher thrust density than ion engines and can therefore be made much smaller for comparable missions.

1.4 The Near Vacuum Hall Thruster

As discussed above, Hall thrusters are valued for station-keeping and drag-compensation missions where their efficient use of propellant enables them to extend the lifetime of missions substantially over the more common chemical propellant systems. However, even the typical Hall thruster mission is limited by the necessity of carrying heavy and exhaustible onboard propellant along with the associated complicated tanks and pumping systems. What if we could design an engine that could "live off the land," so to speak, and would use the ambient particles available in low earth orbit as its propellant? The thruster could act as an ionospheric ramjet, scooping up electrons and atoms from the atmosphere and then using the ions to produce enough thrust to maneuver or simply maintain its orbit velocity. Such a thruster would then only be limited by the lifetime of its materials, extending present satellite lifetimes by leaps and bounds. In addition, no onboard propellant would mean reduced payload masses and launch costs, further slashing the price to orbit for innumerable commercial and other applications.

This exciting concept has recently been investigated by MIT in the form of a near-vacuum Hall thruster. Pigeon and Whitaker [62] first constructed a prototype of the NVHT and tested their engine in our vacuum chamber. In addition they, with the help of Martinez-Sanchez, developed a theoretical model to analyze the basic properties and performance of the thruster. The Pigeon and Whitaker experiment was meant to be a first-cut proof of concept study and measured only the total current produced by the thruster. Their experiments did indicate that the concept of an NVHT was a feasible one, with their thruster producing a significant plasma jet and output current.

Bashir and Sullivan [7] attempted to investigate the concept still further. They redesigned the initial prototype to include tunable electromagnets rather than the discrete-setting permanent magnets employed by the first authors. Furthermore, they obtained a retarding potential analyzer (RPA) with which they examined not only the total current produced by the thruster, but also the energy of the ions being produced. The results of their experiments seemed to indicate that even though the anode potential in the thruster was being set to 3-5 kV, only about 600-800 V of that energy was finding its way into the produced ions. Given that particular design of the thruster, the experimenters were forced to conclude that it would not produce enough thrust to compensate for the drag of an LEO satellite. However, that was only one particular design implementation, and one which experiments indicated was highly inefficient. The concept of the NVHT was proven to be feasible and functional. It seemed very probable that with some diligent engineering and experimentation, future researchers should be able to more closely optimize the thruster's performance and remove the impediments to efficient operation.

Hopefully, the numerical simulation results presented in this thesis will play a role in that effort.

1.5 Previous Hall Thruster Modeling

Due to both their recent increase in popularity and also a lack of complete theoretical understanding, there has been a significant amount of simulation work directed toward the modeling of Hall thrusters. Lentz created a one-dimensional numerical model which was able to fairly accurately predict the operating characteristics and plasma parameters for the acceleration channel of a particular Japanese thruster [50]. He began the trend of assuming a Maxwellian distribution of electrons and modeling the various species with a fluidic approximation. Additional one-dimensional analytic work was performed by Noguchi, Martinez-Sanchez, and Ahedo. Their construction is a useful first-stab approximation helpful for analyzing low frequency axial instabilities in SPT-type thrusters [59].

In the two-dimensional realm, Hirakawa was the first to model the Rθ-plane of the thruster channel [39]. Her work has recently become even more relevant upon our understanding that azimuthal variations in plasma density may play a significant role in anomalous transport. Fife's contribution was the creation of an axisymmetric RZplane "hybrid PIC" computational model of an SPT thruster acceleration channel [28][29][30][61]. This simulation also assumed a Maxwellian distribution of fluidic electrons while explicitly modeling ions and neutrals as particles. The results of this study were encouraging and successfully predicted basic SPT performance parameters. However, they were unable to accurately predict certain details of thruster operation due to their overly strict Maxwellian electron assumption. Roy and Pandey took a finite element approach to the modeling of thruster channel dynamics while attempting to simulate the sputtering of thruster walls [65]. A number of studies have also numerically examined the plasma plume ejected by Hall thrusters and ion engines. At MIT, for example, Celik, Santi, and Cheng evolved a three-dimensional Hall thruster plume simulation [22]. In addition, Cheng has adapted Fife's well-known hybrid-fluid simulation to incorporate the effects of sputtering and attempted to conduct lifetime studies of some typical Hall thrusters [23].

A number of important strides in the realm of anomalous diffusion as it pertains to Hall thrusters have also recently been made by the group of Mark Capelli at Stanford [2][21][36]. Their calculations and simulations give strong evidence for the relationship between the ExB shear and the anomalous transport in the thruster channel.

Finally, the most relevant work to the current project was Szabo's development of a two-dimensionl, RZ-plane, fully-kinetic PIC Hall thruster model [73][74]. This simulation treated all species, electrons, ions, and neutrals, as individual particles and did not rely on the fluidic assumptions of the past. Blateau and Sullivan's later additions to that code extended it to deal with SPT type thrusters [18] and the incorporation of sputtering models [72]. I, myself, added parallelization to the implementation and began my attempts to improve upon the constant anomalous diffusion assumption previously employed. This code was able to identify problems with and help redesign the magnetic field of the mini-TAL thruster built by Khayms [47]. In addition, the code was able to predict thrust and specific impulse values for some experimental thrusters to within 30%. It did have several potential drawbacks, however, and the current work was in part an attempt to address those issues.

1.6 Goals of the Present Thesis

The main focus of this thesis was the design, development, and implementation of a novel fully-kinetic particle-in-cell simulation for use in modeling Hall-effect thrusters. In creating this new model, we hoped to address some of the limitations and weaknesses of the former simulations. The advancements over the former simulation include:

- Our simulation was designed to operate on unstructured meshes which could easily be created and modified. This improved upon the former generation's structured meshes and the somewhat enigmatic method of mesh generation.

- We utilized an adaptive recursive refinement and coarsening technique which would enable us to efficiently resolve structures in the plasma which before had been left unresolved or had required exorbitant computational effort to handle.
- A third advancement made by this simulation was the use of a more accurate particle motion calculation scheme which improved upon the former model's linearized approximations.
- The varied statistical weighting of particles within the simulation was also undertaken. This allowed us to more completely cover the entirety of velocity space while reducing the noise often associated with Monte Carlo simulations. The particle re-weighting scheme helped to ensure that high velocity, low probability, and physically important particles were not excluded from the simulation.
- A concerted effort was made to understand and calculate a priori the spatial and temporal profile of the anomalous diffusion within the Hall thruster. Former models had relied on a constant assumption or at best empirical evidence from pre-existing experiments. The addition of a "quench model" for anomalous diffusion calculation improves the accuracy of our simulations while also enabling the true predictive capability of the model. The model does not require previous experimental measurements of a thruster's anomalous diffusion in order to yield accurate results.

On top of creating this model and advancing the general state of the art in Hall thruster simulations, our secondary objective was to prove our model's usefulness by aiding the re-design efforts of the near-vacuum Hall thruster being studied at MIT. Our simulation results have identified several possible sources of inefficiency in this prototype thruster, including the structure of the magnetic field as well as the geometry and material properties of the acceleration channel. At the time of this writing, efforts are underway to experimentally improve that thruster's performance based on the recommendations made by our novel numerical model.

It is hoped that our labor has made a positive contribution to the field of Hall thruster computations and improved our modeling capabilities to aid in the design and predictive simulation of future plasma engines.

1.7 Summary of Methodology

The basic overarching method exploited by this work is the particle-in-cell simulation technique. This involves the simulation of discrete particles, in our fully-kinetic case this includes electrons, ions, and neutrals. These particles are allowed to move freely across a computational mesh or grid. After each movement step, the particles are weighted to, in our case, the centers of grid elements according to some pre-defined procedure. The scheme we used for this weighting is described in detail in Section 2.6. The individual properties of the particles, including their charge and density, are spread out over several neighboring cells and associated to the mesh vertices. This allows for the matrix computation of quantities such as the electric potential and electric field. Once these fields are calculated at the nodes, the forces are then re-interpolated back onto each particle. Thus, the electric field in each of the surrounding cells can affect the particle in a given cell. With this re-interpolation of the fields, the trajectories of the particles are then calculated and their positions updated.

Such is the basic PIC method. The following sections are meant to be a brief introduction to the various novel aspects and adaptations to that method which have been incorporated into this simulation. More detail on each of the topics is provided in the body of the thesis and, in many cases, in the associated references.

1.7.1 Basic Simulation Flowchart

With a project this large in scope, I find it is important to quickly get a broad landscape view of the individual modules or algorithmic pieces before we become inexorably entwined in the thorny details. The flowchart shown in Figure 1-2 is designed to give just such an overview and provide the reader a basis on which to build his mental model of our simulation.

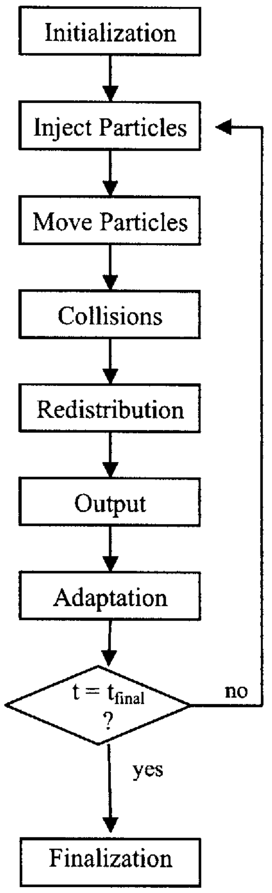


Figure 1-2: A high-level flowchart of the simulation

The first step of the process involves creation of the relevant data structures, allocation of the required memory, and initialization of all variables. In addition, during the initialization phase, the base mesh geometry is loaded and pre-adapted if necessary, the magnetic field is loaded and interpolated, and the appropriate parameters for the particles under consideration are prepared, including particle mass and initial temperature and density distributions. The file *input.txt* controls many of the important parameters for a simulation run and is parsed during this step. Sections of the thesis involving the initialization phase include Sections 2.1 and 2.3.

After the initialization step, the main loop of the program begins. The first step in that loop is to inject new particles that might be entering the simulation region from the boundaries. This includes neutral gas flows, cathode injections, or any other injected particle for the given application. Neutral gas flow injection is covered in Section 2.10.2

while cathode boundary conditions are found in Section 2.10.3, Section 4.1.3, and Section 5.2.5 for general applications, the P5 thruster, and the NVHT thruster, respectively.

Next, at each iteration the particles are moved according to, for the electrostatic branch of the code covered in this thesis, a static magnetic field and a time-varying, self-consistently calculated electric field. The necessary calculations for particle motion are covered in Section 2.5 and numerous benchmarking tests of this module also appear in Section 3.1.

Collision events are then resolved. Due to the non-uniform statistical weighting of particles in our simulation, a new method of collisions was developed. This process is outlined in Section 2.8 and some test results for it appear in Section 3.4.

The next step of the simulation, Vlasov-like particle redistribution, is one of the pieces that makes our simulation a step-up in complexity and generality. Through this step, we are able to place a bound on the number of simulated particles while still managing to adequately cover the entire velocity space. More details about this process are given in Section 2.9.

Every so often during the main loop of the program, output files will be generated. These files contain information about the temperatures, densities, and drift velocities of different species, as well as plots of the relevant fields and collision locations. Section 2.12 outlines the different types of data files that are output and the information to be found in each.

Also performed at intervals during the main loop is the recursive refinement and coarsening grid adaptation step. This powerful technique allows the simulation length scales to vary by several orders of magnitude while still maintaining an efficient number of mesh elements. The adaptation step is discussed in Section 2.2.

Finally, once the specified simulation time has elapsed, the main loop terminates and the Finalization portion of the code is executed. This step deallocates memory, cleans up any temporary files or information that may have been stored, and outputs the final results of the simulation.

1.7.2 New Features of the Grid

One of the most complication-inducing advancements made by this work is the introduction of much more general and powerful meshing structures which enable improved resolution and domain-shaping capabilities. Fortunately, the complications were all made to the implementation of the code, and the actual creation and manipulation of these innovative meshes for the general user was made even simpler than the outdated and quite brittle mesh generation tool used in the past.

The first improvement involved the incorporation of unstructured meshing techniques to replace the formerly structured-only meshes. A structured mesh is one which can be derived from a regular, initially square or rectangular shape and must often be awkwardly stretched to fit non-trivial boundaries. Unstructured meshes can retain high element quality while still very accurately following even the most difficult and frustrating of boundaries, including multiple void regions and highly non-convex shapes. The difference stems from the representation of the mesh in the simulation. A structured mesh relies on the fact that it can enumerate the elements in an x and y numerical order. Thus, element 0 is, say, the lower-left corner and then element 1 must be located directly above it. Element 2 must be located just above that all the way up to the top of the grid. Then the numbering begins again, for example, with element 11 being element 1's right neighbor and element 12 being element 2's right neighbor. This set-up makes the resolution of matrix equations much simpler to conceive, understand, and implement. However, in an unstructured context, the element numbering is arbitrary and each interior node may have as many or as few neighbors as is necessary for the application. The nodes need not be transformable to a regular, square mesh. Instead, the unstructured mesh carries with it a connectivity array. This array describes, for a given element, who its neighbors are and is used to navigate between elements on the grid. Adding this element of abstraction and generality enables the unstructured meshes to be considerably more flexbile while sacrificing the ease of implementation which is gained from a regular, structured mesh. More on the unstructured meshes can be found in Section 2.1.

The second improvement in our meshing techniques comes in the form of an automatic adaptive meshing procedure known as recursive-refinement and coarsening. This procedure allows the code to efficiently locate regions of high gradients in the solution which require further resolution to accurately treat. The mesh then refines itself automatically in those regions temporarily. If conditions change and the region no longer warrants such high resolution, the mesh can then efficiently coarsen itself back up to the level of the original base mesh. In this way, structural singularities such as shocks and sheaths can be resolved without the waste of computational resources in less interesting regions of the simulation. Of course, a price must be paid for this impressive capability and that price is mostly paid in terms of the complexity of the implementation of robust solution techniques for the unorthodox meshes produced. The process of developing these ground-breaking solvers is described in Section 2.7 while the RRC procedure is further discussed in Section 2.2.

1.7.3 Particle Weighting and Redistribution

One of the major drawbacks of typical PIC-DSMC Hall thruster simulations is the depletion of the statistically small but very significant high energy tail of the electron distribution. In order to eliminate this problem, our method institutes periodic redistributions in velocity space and also supports a novel collisional methodology which takes our weighted particles into account and maintains the velocity space coverage as much as possible.

The computational particles in our simulation are each given a different statistical weight. For instance, if an initial two-dimensional Maxwellian distribution is created, a grid in velocity space is first imagined spanning from, for instance, $-3v_M$ to $+3v_M$ in both the x and the y directions where v_M is the particle's thermal velocity discussed below in Section 2.9. Then, an equal number of particles are assigned to each cell in the velocity grid. However, the particles are assigned statistical weights equal to the value of the Maxwellian distribution at that energy, and so particles in the center of the velocity grid are considered statistically "bigger" or "heavier" in terms of the remaining simulation. Particles on the tails of the distribution have a relatively small statistical weight. At the same time, the number of such particles is still large and statistically significant, effectively reducing the noise often associated with full-PIC models which, given their uniform particle weighting, might not assign any computational particles to these low-weight tails.

However, as ionization events occur and more and more cold secondary electrons are created, it becomes necessary to "sweep up" these redundant particles. Otherwise the simulation would quickly become bogged down computationally by a large number of very similar in location, temperature, and energy cold particles that can, without great loss of accuracy or precision, be treated as much larger units. The reconstitution process is accomplished by first imagining all particles within a given computational cell to be placed on a regular velocity grid. Then, all of the particles in a given velocity grid cell are combined into two new particles, the minimal number required to enforce conservation laws, whose statistical weight and velocities are chosen so as to preserve mass, momentum, and energy. In this way, we efficiently retain our coverage of velocity space while ensuring that the number of particles in the simulation remains bounded and computational effort is not squandered on a multitude of cold, redundant particles. This idea is discussed in more detail in Section 2.9.

1.7.4 New Collisional Model

As discussed above and below in Section 2.8, our method of giving particles varying statistical weights compelled us to create new models for collisions other than simple direct Monte Carlo. Our method eliminates Monte Carlo's reliance on random numbers and thus also the associated statistical noise that arises for instance when there is a 1/100 chance of a collision occurring, but only 90 particles present. Instead of choosing a fraction of the particles to collide in their entirety, our method opts to collide a fraction of every particle all of the time. This concept is demonstrated graphically in Figure 1-3.

The method begins by calculating the densities of all the species in each computational cell. Then, examining just a single cell, the relevant collision frequencies are calculated using the formulas discussed in more detail in Section 2.8. Each particle is then split and divided into the proper-sized pieces, each with its own properties of weight and velocity. This process occurs for every cell in the simulation separately. At the end

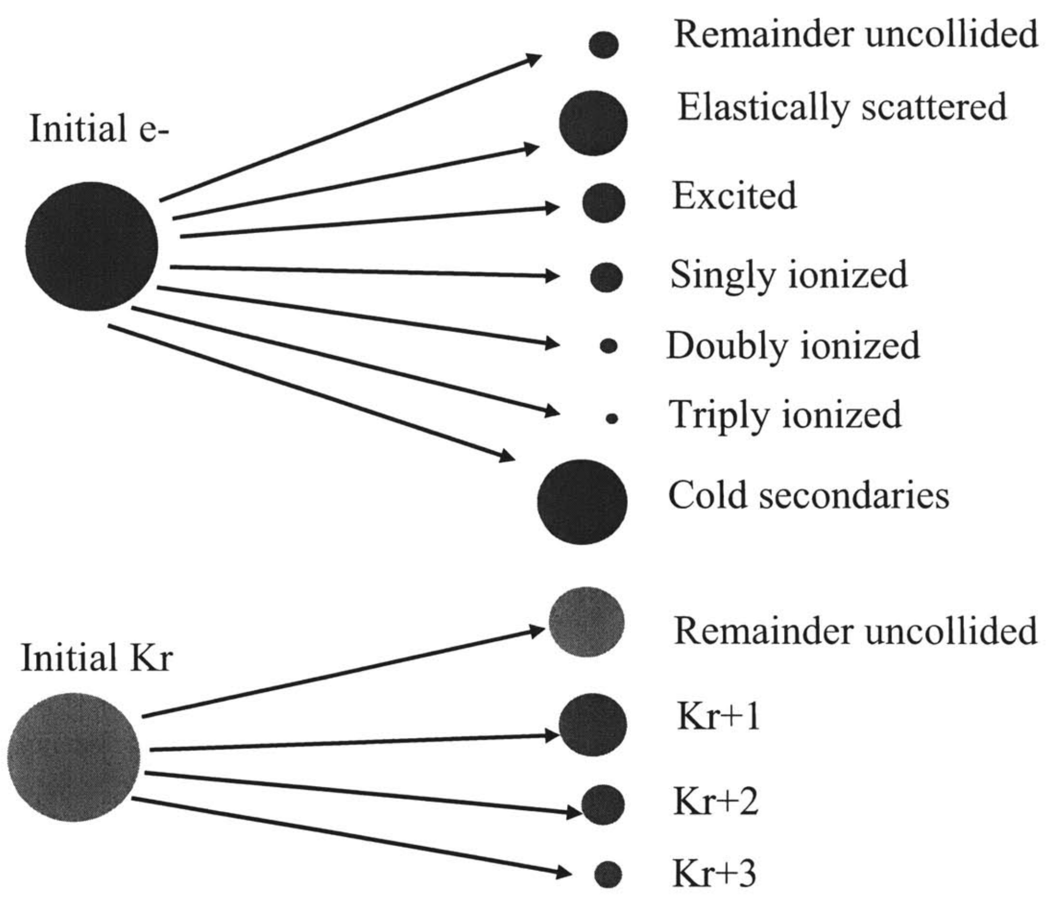


Figure 1-3: A graphical representation of the splitting procedure that occurs during neutral-electron inelastic collisions. This differs from the typical Monte Carlo model in that instead of all of a fraction of the particles colliding, our model collides a fraction of all of the particles, thus eliminating one source of statistical noise.

of a collision step, the number of particles may be as much as ten times what it was before the collisions occurred, but the total density of those particles will remain the same. The newly-formed particles are quickly velocity-redistributed in order to then reduce their number and combine particles with similar energy properties.

This method requires no randomization. The collisions are entirely deterministic, very unlike the previous Monte Carlo methods. Tests of the collision module are demonstrated in Section 3.4.

1.7.5 Anomalous Diffusion Modeling

One of the most important advancements discussed in this thesis was our investigation of the anomalous transport in a Hall thruster's channel and the implementation of an entirely analytical method for approximating this diffusion in our spatially two-dimensional simulations. The name "anomalous" transport is actually a misnomer. There is nothing anomalous about the transport really; it has been known to exist for decades now. The term refers to the extra cross-field mobility experimentally observed which had not been predicted by classical collisional transport theory. Instead, the anomaous transport is driven by turbulent and non-linear effects which have been difficult to accurately grasp or theoretically define. Chapter 4 discusses our own contribution to these efforts, with merely the bare essentials of that model being outlined here.

Many studies have recently indicated the existence of a drastic reduction or barrier to the anomalous diffusion located near the exit plane of the thruster's channel [33][2][21]. The level of transport is clearly not constant in time or in space as previous simulations had usually assumed. In fact, we show that the transport profile has an enormously significant impact on the properties of the discharge, including temperatures and electric potential, which was being overlooked in simulations prior to this time.

The identification of this transport barrier and its correlation with the high ExB shear rate near the thruster's exit plane helped us make the analogy between our Hall thruster plasma system and the plasmas handled by the controlled fusion community. The extensive literature of this field exhibits an astounding abundance of work regarding the mitigation of anomalous diffusion and the purposeful creation of diffusion barriers which have enabled recent fusion researchers to reach unprecendented levels of confinement [79][20][27]. Accordingly, intense effort has been expended investigating the properties of these barriers and attempting to simulate their creation effectively. Waltz and his team, in particular, have developed what they call the "quench rule" which adjusts the level of anomalous transport locally based on the rate of ExB shear and an estimate of the growth rate of the turbulence deemed most responsible for the higher-than-classical transport. It was this concept that the present thesis adapted for use in simulating our Hall thrusters. We determined the turbulent mode, the transit-time mode, which appears to be most strongly linked to the anomalous transport. We then adjusted the quench rule for our geometries and implemented it in the former PIC-MCC simulation developed by Szabo. In this way, we could test the effects of the quench model in a well-established, pre-existing simulation which has previously published results with which we could compare.

The benefits of this model are quite momentous and the results of our tests extremely promising. The quench model was shown to, without any empirical prompting, predict the exit plane barrier in the anomalous transport as was hoped. In addition, the properties of the discharge were significantly improved in comparison with the constant transport case, and the frequency and amplitude of the calculated discharge oscillations much more closely match those of experiment. The incorporation of this analytical calculation of the anomalous transport removes our simulation's dependence on previous experimental data for each specific thruster which would otherwise be our best means of approximating the transport. Instead, it allows the codes to truly become predictive tools which can give a good general overview of a thruster's operation before that particular engine is constructed.

1.7.6 Parallelization

The algorithms employed in this thesis were all developed with their impact on parallelization efficiency at the forefront of our concerns. Since the code was developed for multiple processors upfront, it could be much more efficient than the older PIC-MCC simulation which had been developed in serial first and later twisted and squeezed to fit into a parallel framework.

Details of our efforts to maintain this parallel efficiency are further discussed in Section 2.13. The Message Passing Interface (MPI) standard library was chosen over shared-memory algorithms such as OpenMP. This should allow the code to be portable and provide a relatively simple transfer from the supercomputers it is now operated on to newer models or even our own lab's PC network. Each processor was obliged to maintain a copy of the entire simulation mesh, although at any one time, it was only assigned the particles which existed in its own unique section of the grid. If particles crossed from one processor's assigned region to another, they would be physically sent to the new processor. If the number of particles on each processor began to change, the mesh would be re-divided and sections re-assigned so that an equal number of particles would generally exist on each processor. This allowed the very efficient computation of the time-consuming collision steps and the velocity redistribution module since these processes depend only on the particles within a given cell and can therefore be performed entirely in parallel. The parallelization of the electric potential was accomplished through the use of a pre-designed direct solution algorithm known as MUMPS. The parallelization of the majority of the modules was seen to be highly efficient for the large-scale problems on which we focused as Section 3.7 demonstrates.

1.8 Summary of Tools

The generosity of Delta Search Labs, Inc. enabled this simulation to be developed on a number of different platforms and in multiple programming languages. Thus, the present work is hoped to be more generally applicable and more easily adaptable than previous simulations. The great number of tools used to accomplish this thesis are outlined below.

1.8.1 Hardware

Initially, the code was developed in a Windows environment using a fairly high-end PC with 2 GB available RAM and a dual-core processor system with 3.0 GHz clock speed. Many of the benchmarks provided in Chapter 3 of the thesis were completed using this machine. As the simulation developed and became more complex, it was adapted for use on parallel architectures. Further code development and simple testing was performed on an SGI Octane two processor machine. Full-scale simulations and computationally intensive tests were then accomplished on Delta Search Labs' 128 processor SGI Origin machine with 64 GB total available shareable memory. The individual processors of this machine were singularly much less powerful than the PC with individual clock speeds less than 1 GHz. Taken as a whole, however, and paired with efficient parallelization techniques this machine could achieve a vast reduction in computational time. Also, without its exceptional amount of memory, many of the full-scale simulation trials would have been unfeasible to conduct. Again, I cannot stress how grateful I am for and how invaluable the assistance of Delta Search Labs was in completing this project.

1.8.2 Software

As mentioned, initial code development was completed in an MS-Windows environment. Compaq Visual Fortran90 was employed for code management, and the majority of the code was written in the Fortran programming language. A separate effort was begun to convert key sections of the simulation to the Java language, including the grid-parsing algorithms and the particle trajectory calculators. This work was also done under Windows, but using simple text editors. The SGI machines both ran under the Tru-Unix64 environment. Code was developed again using simple text editors such as nedit.

A number of third-party software packages and programs were also employed throughout the research process. Probably the most important of these was the Tecplot data visualization tool [90]. All of our input geometries and output results were written to be compatible with this program, and the vast majority of the plots found in this thesis were created with this program. A small amount of work was also conducted in Matlab, which program is also responsible for a few figures in the thesis. The MPI standard was chosen for all of our parallelization needs and MPICH was chosen as our Windows-based implementation of that standard. [91] The magnetic field for the Near Vacuum Hall Thruster was modeled using the student version of Maxwell 2D. Finally, the parallel direct solution solver MUMPS [86] was at times used to solve Poisson's equation and compared to a homemade iterative implementation of SOR.

1.9 Summary of Results

It was the goal of this thesis to advance the state of the art in Hall thrusters and their simulations, however modestly. Three primary thrusts comprised the main body of our attack on this problem. First, a novel simulation algorithm was developed, implemented, and benchmarked. Second, an analytic model for the calculation of anomalous transport was adapted for use in simulating our thrusters. Finally, a cutting-edge thruster was modeled with our newly-designed algorithm, and suggestions were made regarding the improvement of that engine's design. The outcome of these three endeavors is discussed briefly below.

1.9.1 Code Validation

Chapter 3 focuses on a subset of the innumerable tests which were conducted in order to demonstrate the validity of our theoretical model and its physical implementation. These tests range from the very simple, including single particle motion under various electromagnetic field conditions, to suitably complex, such as the calculation of the plasma period of Langmuir cold plasma oscillations. The benchmarking demonstrates that particle motion can be computed accurately on unstructured and refined meshes. Further, we show the accuracy of our freshly devised potential solver and electric field interpolation algorithms. The Maxwellianizing intra-species elastic collision model is qualitatively and quantitatively evaluated as is the accuracy of the velocity redistribution module. Finally, the efficiency of our parallelization efforts is measured and reported.

With the successful completion of these component level tests, we then move in Chapter 5 to a full system level test of the code. The well-known and intensively-studied P5 thruster is simulated using our new algorithm. The results of those simulations are compared to the results from the former MIT PIC-MCC simulation and to experimental data. The analysis of this comparison highlights the areas of agreement and disagreement between our simulation and both the former numerical model and the empirical data. The new algorithm is shown to be effective at predicting many important features of the thruster's operation and performance.

1.9.2 Anomalous Diffusion Model

Chapter 4 details the adaptation of the fusion community's "quench rule" for the a priori estimation of the anomalous transport profile in a two-dimensional simulation. This rule relies on the idea that the shear in the ExB drift velocity is able to stabilize and decorrelate the transit time turbulent mode responsible for cross-field mobility in the thruster. The model was implemented in the former MIT PIC-MCC simulation which has previously been studied in detail. While simulating the P5 thruster, the quench rule is able to predict, without the use of any empirical transport data, the exit plane transport barrier which is expected. The use of the quench model also greatly improves the simulation's agreement with experimental temperature and potential profiles while concurrently adjusting the frequency and amplitude of the main discharge oscillation to levels more closely resembling the physical data.

The successful demonstration of this model frees Hall thruster simulations from the post-experimental realm. Experimental data on the anomalous transport coefficient may no longer be needed to create accurate simulations. Rather, the quench model enables the predictive calculation of thruster performance and features without the guesswork and overfitting formerly associated with the choice of transport coefficient.

1.9.3 Near Vacuum Hall Thruster

While the majority of our effort was focused on developing and implementing the model summarized above, it was important as well that we demonstrate its usefulness and applicability. The near vacuum Hall thruster recently constructed and tested by two teams of MIT students seemed an excellent choice for this task. Very little is known about the thruster's operation ensuring our simulation results would have ample impact.

The results of these NVHT simulations and analysis of them are provided in Chapter 6 of this thesis. We there first present a brief background of the thruster and its general characteristics. Next we simulated the NVHT as it was originally configured by Tim Pigeon and Ryan Whitaker [62]. These results agreed with the RPA analysis experimental results of Bashir and Sullivan [7] and indicated that the energies of the ions produced by the thruster were significantly lower than desired.

The initial simulation results suggested that the potential in the channel fell off very rapidly due to the low potential metallic walls sitting very near the high-potential anode. As such, a re-design effort was undertaken by Lozano and his undergraduate student Aung. It was reasoned that the potential could be increased in the channel if the anode was moved closer to the exit plane. As such the physical NVHT is currently under re-construction with an extended anode configuration. Until those experiments could be conducted, we decided to use our simulation to predict if and by how much the new configuration would alter the collected ion energies.

Unfortunately, the simulations seem to indicate that the reconfiguration will not have the hugely significant impact on the ion energies that had been originally hoped. They do suggest that those energies will be increased slightly, but pushing the anode farther down the channel seemed to push the ionization zone farther down the channel as well, keeping the potential of created ions roughly constant. The results did indicate, however, that the divergence of the thruster's beam would likely be improved by the re-design since the ions are formed in a region of lower radial electric field under this configuration. In addition, it is very likely that the new thruster will be able to ignite a plasma and operate at much lower anode voltages than the older model. This is because the anode potential is now more exposed and is not as significantly depressed by the nearby metallic walls. The thrust and anode current of the newly-designed thruster were predicted to be much higher than the original design at decreased anode voltages.

While the experimental re-configuration that is initially being attempted will not necessarily produce the hoped for improvement in the ion energies produced, it does not invalidate the results of the simulations. The electric potential is clearly not being effectively used in the NVHT, and if the current experiments have not solved that problem, another design should most certainly be investigated. The concept of an NVHT offers too many possible benefits to be ignored, especially when our simulation has already laid bare one of the most likely causes of its deficiency.

Chapter 2

Numerical Method

This chapter presents the most important aspects of the numerical technique leveraged by this simulation along with some details regarding the specific implementations in our code. The simulation as a whole draws heavily on the theory presented in the past PIC model developed by Szabo [73][74], but exhibits a substantial number of upgrades and advancements to that original simulation which extend its range of applicability, improve its efficiency, and increase its accuracy. Those changes begin with our powerful new meshing algorithms.

2.1 General Mesh Structure and Creation

The previous generation of MIT Hall thruster simulations was significantly limited in the geometries that it could efficiently and accurately simulate because it relied heavily on the use of a single structured mesh. While this type of mesh made computations and equation solving extremely simple and uncomplicated, it was not general enough to support the RRC adaptive meshing discussed in Section 2.2 nor could it easily be used to simulate complex geometries such as the NVHT simulation required. In addition, the former MIT simulation used a homemade mesh creator that had no graphical user interface and was, to say the least, sometimes difficult to operate.

Hoping to improve upon this previous work, we chose to implement a simulation capable of handling fully unstructured, quadrilateral meshes. This choice does make the coding of the simulation significantly more complicated and leads to some increased computational costs due to the fact that structured grids provide more information and so are easier to use for calculations such as solving Poisson's equation. However, as the example meshes in Figure 2-1 and in Section 2.2 which follows demonstrate, the added capabilities and increased control over mesh spacing and boundary shape are invaluable. For example, a typical structured mesh for the pseudospark device in Figure 2-1(a) would either need to waste memory and computation power by actually gridding the baffle region shown as a void in the figure or significantly bend and contort its grid to fit through the opening beneath the baffle. Not only would this second technique result in numerous non-orthogonal and non-optimal grid cells, but it would also be nearly impossible to maintain an even mesh-spacing throughout the entire space if that was the goal. Unstructured meshes make this and the other geometries depicted extremely easy to create.

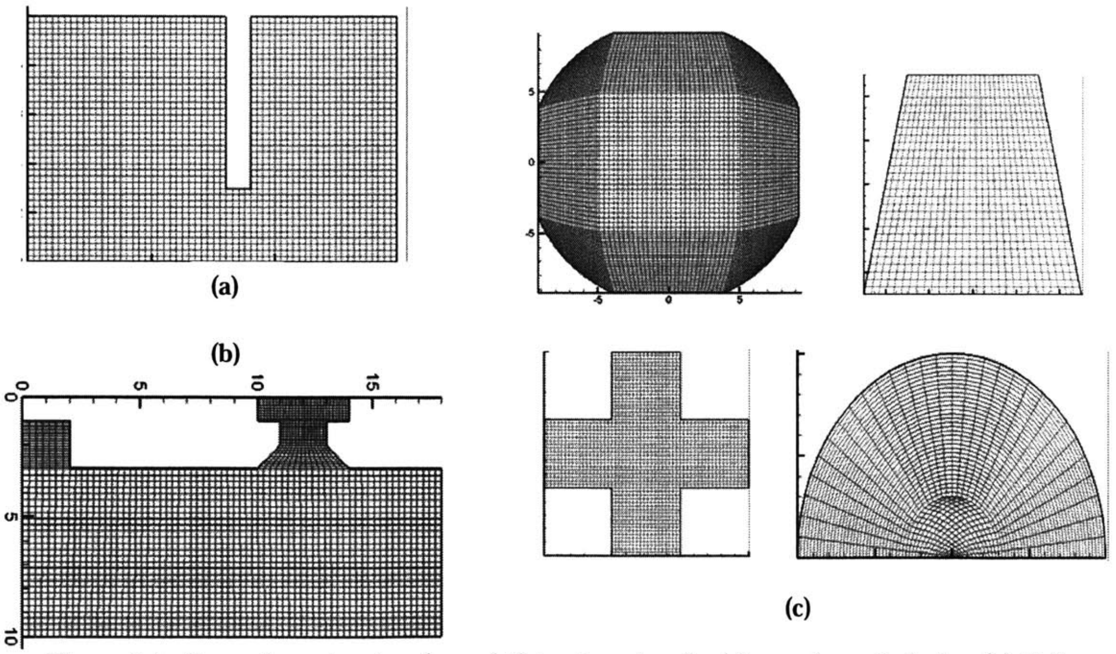


Figure 2-1: Example unstructured, quadrilateral meshes for (a) pseudospark device, (b) Hall thruster with central cathode, and (c) other difficult or awkward geometries.

For simple mesh geometries, we developed a number of our own simple grid-generating functions that could be hand-tuned to produce the desired mesh. However, in order to create the majority of our unstructured meshes efficiently, the proper tool was required. Rather than spending a great deal of time writing code to develop an already mature and available tool, we chose to utilize Tecplot's Mesh Generator Add-on [90] to create the vast majority of our unstructured grids. This powerful package is readily available, stable, reliable, and extremely simple to learn and to use. A more detailed description of this tool and its usage in our simulation is provided in Appendix A, specifically for those interested in using the Mesh Generator Add-on for their own research or for anyone who might be using the present simulation in the future. The easiest way of creating complex, unstructured geometries, in most cases, is to first break down the whole mesh into a number of smaller structured regions with simpler geometries. For each of these regions, regular meshes may be created with good cell qualities. Tecplot then is able to compile these structured regions for the user into one single unstructured mesh suitable for out simulation requirements. This technique was used to create all of the meshes found in Figure 2-1.

2.1.1 Example Meshes and Qualitative Correctness Demonstrations

To demonstrate some of the capabilities of the new Tecplot Mesh Generator meshes, we first created an arbitrary, unusually-shaped grid depicted below in Figure 2-2. The first feature of note on this grid is the large hole in the center of the mesh. This type of geometrical void is not impossible to create using a structured mesh but the required calculations can become extremely complicated especially if there are numerous voids at different points in the mesh. Another feature that this unstructured mesh exhibits is the element qualities around the sharp concave corners and into the small space in the bottom left of the geometry. I, myself, know from experience how difficult it was to model such a region using the previous structured mesh generator as the central cathode thruster's mesh in [33] plainly shows. Finally, the mesh in Figure 2-2 depicts a region of refined meshing, discussed further in Section 2.2, which could only be handled by the use of unstructured meshes .

To test our mesh-reading and mesh-writing functions we then undertook several single particle motion tests on this unusual mesh, qualitatively examining the particle's motion for irregularities. The calculations required for computing particle motion are described in Section 2.5 and further quantitatively benchmarked in Section 3.1. However, the results of these simple qualitative tests are shown in Figures 2-2 through 2-5 and discussed briefly in the figures' captions.

Overall, we can conclude from these tests that the Tecplot Mesh Generator is powerful and capable of creating multi-zone quadrilateral unstructured meshes simply and effectively. In addition, the present simulation is capable of parsing these meshes into our Fortran program, refining them further, and accurately maintaining element connectivity, node location, and neighboring element information.

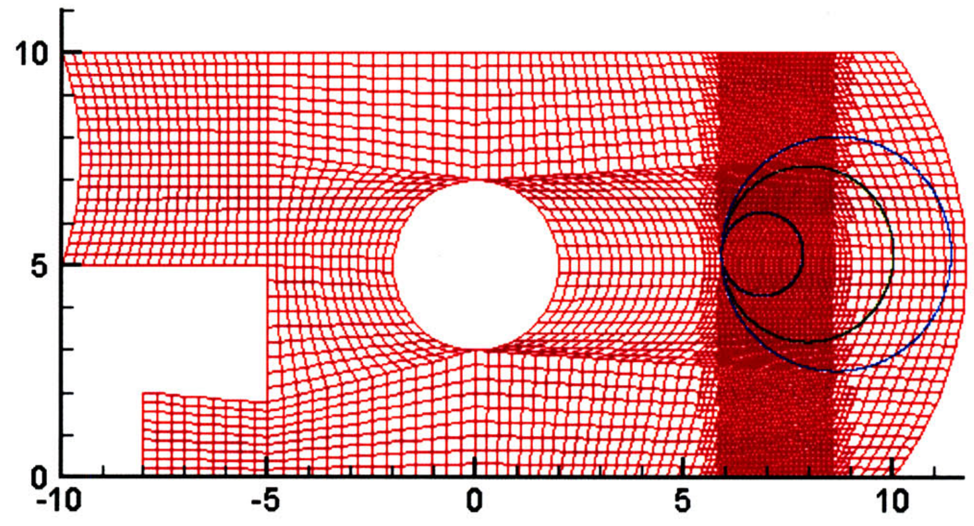


Figure 2-2: Three different particle trajectories are here overlayed on a Tecplot Mesh Generator generated mesh. The adaptation region lies between 5 and 9 x units. There is no electric field present in this trial and the B-field is directed out of the page. The particle trajectories should therefore be perfect circles as they are seen to be. Each of the three particles has a different initial velocity accounting for the varying path radii.

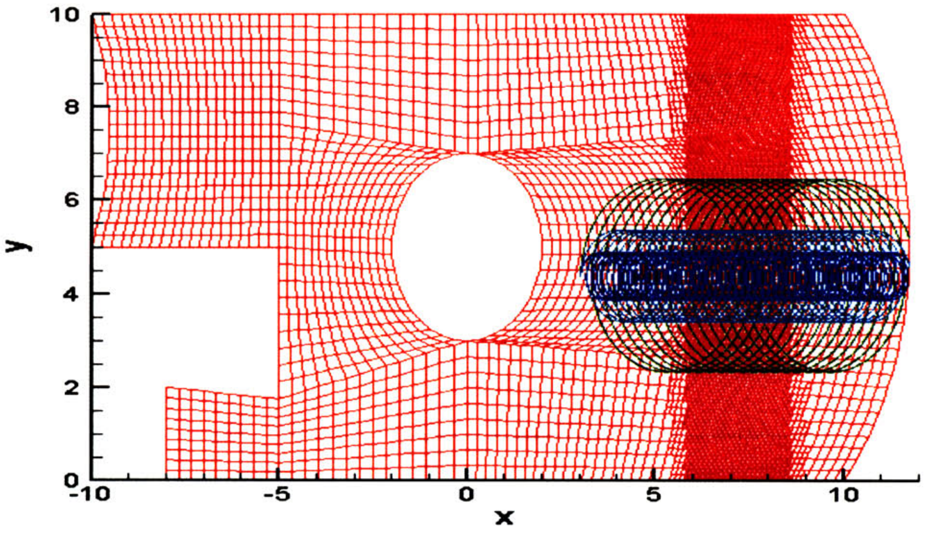


Figure 2-3: This test added a small electric field in the y-direction to the test above. Again the particles have differing initial velocities accounting for the difference in radius. The particles are easily able to move from cell to cell and from zone to zone in this multi-zone quadrilateral mesh, illustrating that our functions are capable of maintaining the mesh structure as well as properly calculating the neighboring element array.

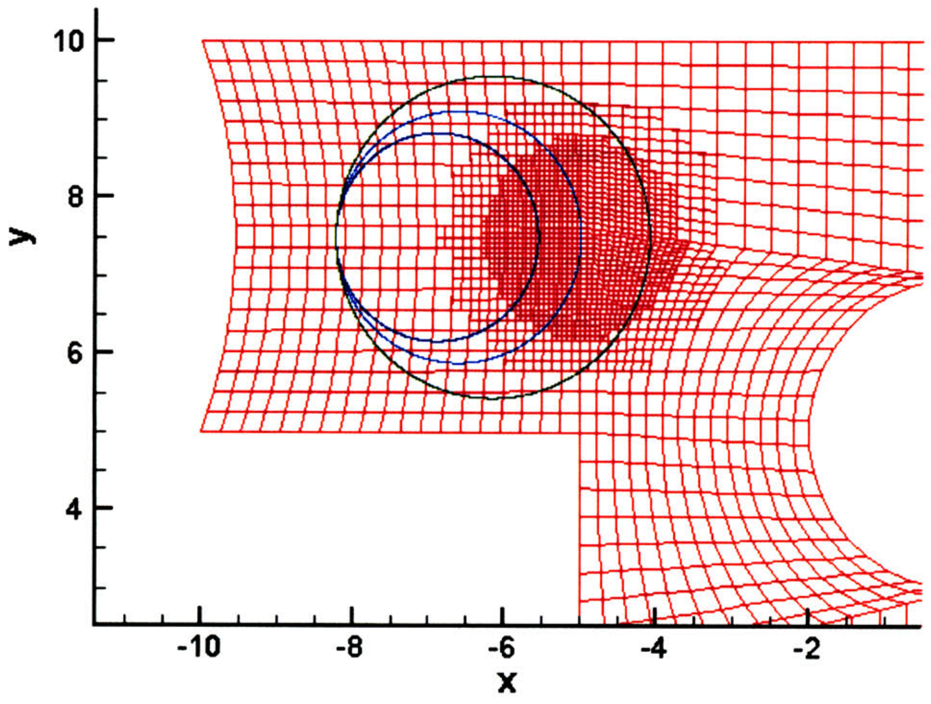


Figure 2-4: Another similar test with zero electric field showing that multi-zone meshes can be integrated with our computational model's cluster adaptation.

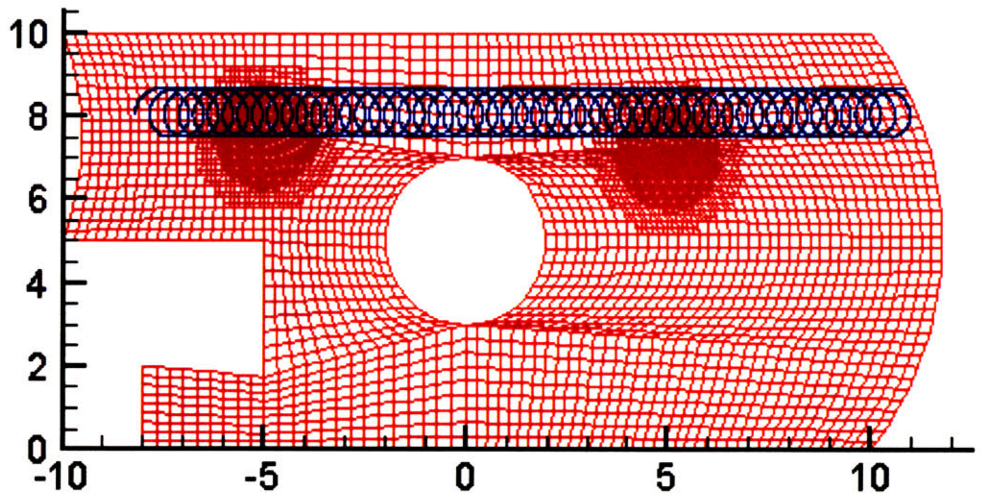


Figure 2-5: One final demonstration. The particle is easily able to traverse the multiple zones of this mesh and pass through the multiple regions of adaptation.

2.2 Recursive Refinement and Coarsening Procedure

One of the most innovative and difficulty-inducing features of our simulation is our usage of the adaptive meshing technique known as RRC, or recursive refinement and coarsening. "Adaptive" meshing refers to the procedure whereby the resolution of the original base mesh may be automatically increased or decreased locally over the course of the simulation. Thus, the number of computational elements may vary from time step to time step as the needs of the simulation dictate. This allows us a great savings in computational effort, especially in multi-scale systems where the plasma may be very dense in one region but tenuous in another. RRC alleviates the need to refine the tenuous region and perform calculations on elements that are not actually needed.

Batishcheva developed the foundations of this method for the present implementation of the simulation. For the results presented in this thesis, it was decided that only trials with static meshing would be illustrated. Thus, the mesh depicted in Figure 2-6 presents a statically-refined Near Vacuum Hall Thruster grid which was used for some initial simulations of that thruster. The hanging nodes and local-refinement of the RRC technique were still used, but we simply chose not to change the adaptation's location during the tests for simplicity, repeatability, and clarity. However, the design of electrostatic solution methods, boundary conditions, particle motion calculators, and innumerable other aspects of the simulation were strongly affected by the requirement that they not interfere with the RRC meshing capabilities. As such, the relevant basics of the method are described in the following sections.

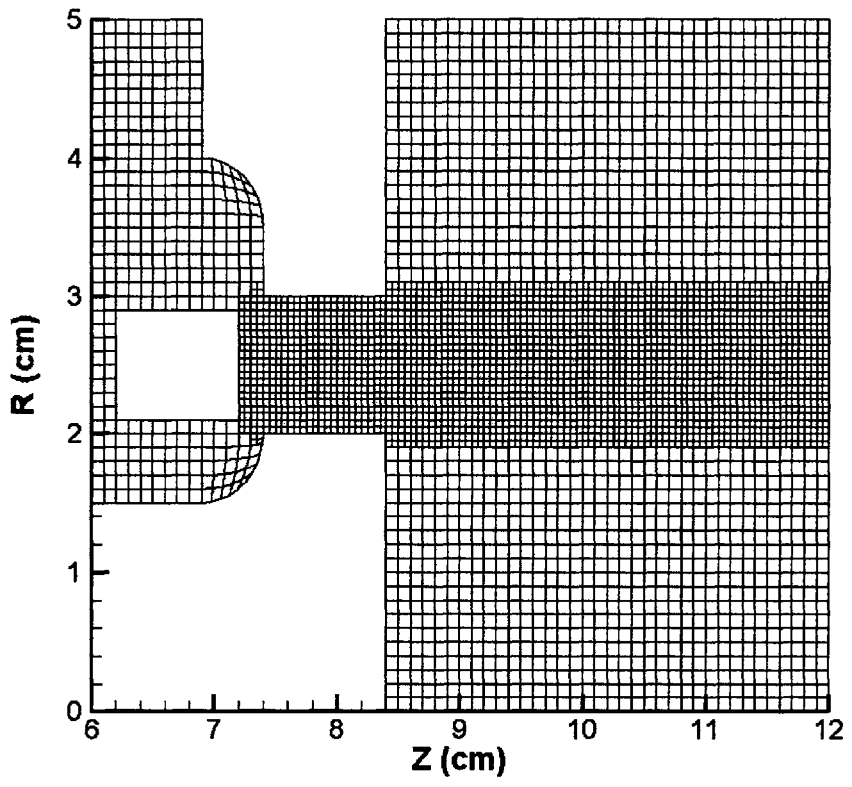


Figure 2-6: An example of the static local-refinement employed in some NVHT tests. For the results presented in this thesis, we did not allow the mesh to adapt automatically as in full RRC, but the simulation was designed to allow such automatic adaptation and all solvers and algorithms were implemented with this technique in mind.

2.2.1 Details of the RRC Technique

At periodic intervals during the simulation, the refinement and coarsening module may be called. This module is keyed to some quantity or quantities of interest, such as the electric field, magnetic field, plasma density, or Debye length of the plasma. If the module finds that in the vicinity of a given element the gradient of that quantity is greater than some $\varepsilon_{\text{refine}}$ chosen by the user, then the cell will be subdivided into four new subcells. The new vertices required for this subdivision are located at the midpoint of the base cell's faces and at either the geometric center or median center of the original cell, depending upon the user's preference. Similarly, if the gradient of the key quantity falls below some pre-defined ε_{coarse} in the four neighboring subcells, the coarsening procedure is enacted. This procedure removes the added vertices and conglomerates the subcells back into the original format. The procedure is entirely reversible. A cell that is refined can later be unrefined as the plasma shifts and high accuracy is no longer needed in the region. Another useful feature of this feature is that if desired, only the base mesh coordinates actually need to be stored in memory. If the history of the subdivision is stored properly, all of the remaining node coordinates can very efficiently be calculated on the fly. This feature could be used to trade computational time for memory under a very space-constrained application.

As the splitting and re-combining of cells occurs, several quantities must be interpolated or extrapolated depending on the situation. When a cell is subdivided, the electric field, as well as all other cell-centered calculated quantities like temperature, magnetic field, and charge distribution, must be re-interpolated to the new subcells. Currently, the implementation of this step is simply to assign the original value of the quantities to each of the four subcells. After one iteration, these values, with the exception of the static magnetic field in the case of Hall thrusters, will be updated anyway and can be calculated much more accurately than they can be interpolated. One could imagine a more complicated weighting scheme being employed during refinement, but our implementation avoids this added complexity. Upon coarsening, these same quantities must also be re-calculated for the new larger cell. We use a simple weighting scheme based on the area of the subcells to perform this step. The computational particles are also of course divided between subcells in a trivial manner, with the particle's location remaining unchanged and simply being assigned to the subcell in which they now reside.

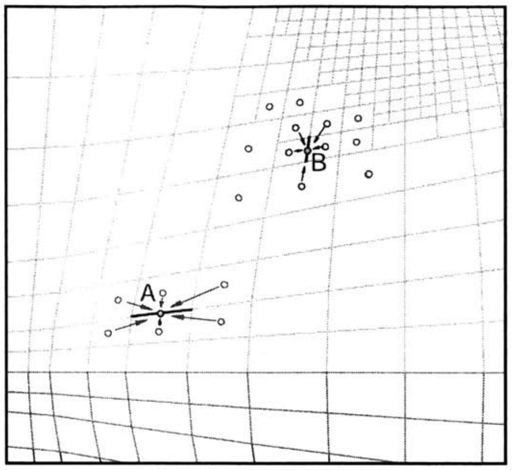


Figure 2-7: An example RRC mesh demonstrating both (a) a regular six-point stencil for a cell face and (b) an irregular five-point stencil for a cell face. Since RRC guarantees that refinement level only changes by +/- 1 from neighbor to neighbor, these two stencil cases are the only two that need to be considered with the exception of boundary elements and faces.

One implicit constraint that must also be guaranteed by the implementation is that the neighboring elements of a cell differ by at most +/-1 level. This one property allows us to ensure that the five-point stencil situation depicted in Figure 2-7 is the most complicated one that can arise in general terms. Knowing this, we can efficiently design solution methods that will function on both regular six-point stencils and on irregular five-point stencils. Both of these types of stencils are illustrated in Figure 2-7.

2.2.2 Conservation of Splined Quantities Across Cell Faces

As Batischev and Batishcheva have previously described [11], the method of regularizing irregular cells has the benefit of conserving fluxes across cell faces when the associated splines are created. This ensures that, for instance, the current out of a cell is equal to the current entering the cell, and allows us to use Stokes Theorem while automatically conserving the necessary quantities. Since the regularization process is deterministic and always results in the same six-point stencil across a given cell face, regardless of the direction being analyzed, the only remaining check we must conduct to ensure conservation is that the splines that are produced are themselves identical, regardless of direction.

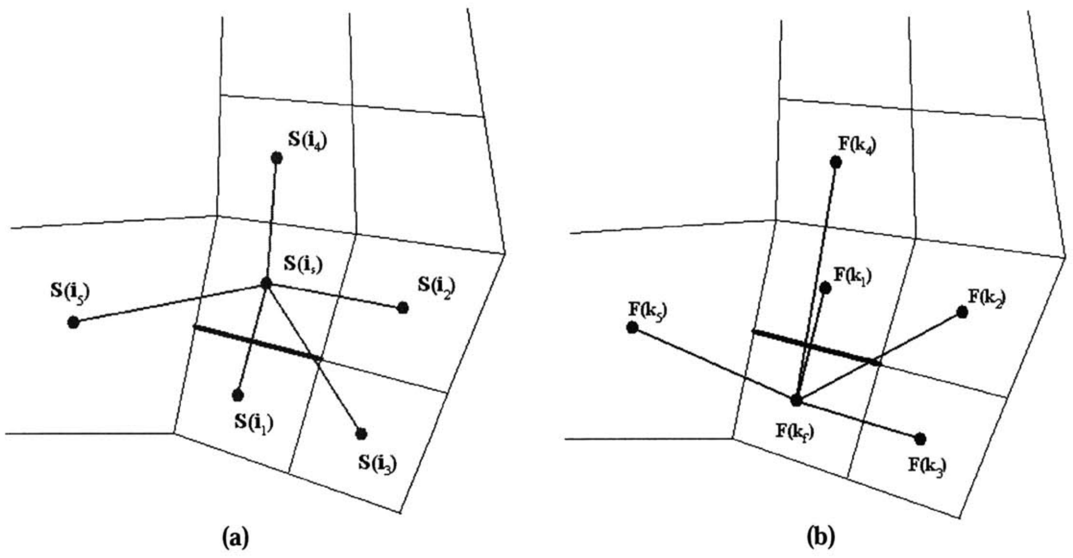


Figure 2-8: The definition of the points in the two spline functions (a) S and (b) F.

To demonstrate that our procedure does produce identical spline functions across a given cell face yielding conservation of the necessary quantities, we consider the two spline functions $S(\vec{r}_i)$ and $F(\vec{r}_k)$, with the *i* and *k* indices representing the neighboring nodes numbered from their respective cell's center and ranging from 1 to 5. The situation is illustrated in Figure 2-8 for clarity. These six point quadratic spline functions can be defined by:

$$\begin{aligned} S &\equiv g_s + a_s\Delta x_s + b_s\Delta y_s + \alpha_s\Delta x_s^2 + \beta_s\Delta y_s^2 + \gamma_s\Delta x_s\Delta y_s \\ F &\equiv g_f + a_f\Delta x_f + b_f\Delta y_f + \alpha_f\Delta x_f^2 + \beta_s\Delta y_f^2 + \gamma_f\Delta x_f\Delta y_f \end{aligned} \qquad (2.1)$$

where g_n represents the value of the function to be interpolated at node n and a_n, b_n, α_n, β_n, and γ_n are the ten unknown spline coefficients which can be solved for as discussed in Section 2.7. The remaining geometrical parameters can be defined simply as:

$$\begin{aligned} \Delta x_n &= x - x_n \\ \Delta y_n &= y - y_n \end{aligned} \qquad (2.2)$$

and are the difference between the component-wise location of the originating node of the spline and the position being interpolated.

Given this arrangement, we can show that the two functions are exactly equivalent if their solutions exist. Assuming that both functions do exist, the two systems of equations:

$$\begin{cases} a_s\Delta x_s^f + b_s\Delta y_s^f + \alpha_s\Delta x_s^{f2} + \beta_s\Delta y_s^{f2} + \gamma_s\Delta x_s^f\Delta y_s^f = \Delta S^f \\ a_s\Delta x_s^1 + b_s\Delta y_s^1 + \alpha_s\Delta x_s^{1^2} + \beta_s\Delta y_s^{1^2} + \gamma_s\Delta x_s^1\Delta y_s^1 = \Delta S^1 \\ a_s\Delta x_s^2 + b_s\Delta y_s^2 + \alpha_s\Delta x_s^{2^2} + \beta_s\Delta y_s^{2^2} + \gamma_s\Delta x_s^2\Delta y_s^2 = \Delta S^2 \\ a_s\Delta x_s^3 + b_s\Delta y_s^3 + \alpha_s\Delta x_s^{3^2} + \beta_s\Delta y_s^{3^2} + \gamma_s\Delta x_s^3\Delta y_s^3 = \Delta S^3 \\ a_s\Delta x_s^4 + b_s\Delta y_s^4 + \alpha_s\Delta x_s^{4^2} + \beta_s\Delta y_s^{4^2} + \gamma_s\Delta x_s^4\Delta y_s^4 = \Delta S^4 \end{cases}$$
$$\begin{cases} a_f\Delta x_f^s + b_f\Delta y_f^s + \alpha_f\Delta x_f^{s^2} + \beta_f\Delta y_f^{s^2} + \gamma_f\Delta x_f^s\Delta y_f^s = \Delta F^s \\ a_f\Delta x_f^1 + b_f\Delta y_f^1 + \alpha_f\Delta x_f^{1^2} + \beta_f\Delta y_f^{1^2} + \gamma_f\Delta x_f^1\Delta y_f^1 = \Delta F^1 \\ a_f\Delta x_f^2 + b_f\Delta y_f^2 + \alpha_f\Delta x_f^{2^2} + \beta_f\Delta y_f^{2^2} + \gamma_f\Delta x_f^2\Delta y_f^2 = \Delta F^2 \\ a_f\Delta x_f^3 + b_f\Delta y_f^3 + \alpha_f\Delta x_f^{3^2} + \beta_f\Delta y_f^{3^2} + \gamma_f\Delta x_f^3\Delta y_f^3 = \Delta F^3 \\ a_f\Delta x_f^4 + b_f\Delta y_f^4 + \alpha_f\Delta x_f^{4^2} + \beta_f\Delta y_f^{4^2} + \gamma_f\Delta x_f^4\Delta y_f^4 = \Delta F^4 \end{cases} \tag{2.3}$$

must have solutions. Notice that here the nodes 1...4 are exactly the same nodes while *s* and *f* are the remaining two nodes and interchange in the equations for the S-spline and the F-spline.

For these two systems to be resolvable, they must have non-zero and equal in absolute value determinants. To make progress on these systems, we must first realize that by definition:

$$\Delta\vec{r}_s = \vec{r}_s - \vec{r}_f + \Delta\vec{r}_f \tag{2.4}$$

This follows from simple vector geometry. Given this formulation, we can then see that, for example:

$$\Delta x_s^i = \Delta x_f^i - \Delta x_f^s \tag{2.5}$$

which in particular, setting $i = f$ shows us that $\Delta x_s^f = -\Delta x_f^s$ which is as we would expect. Now, putting all of this together, we can find that:

$$|S| = \begin{vmatrix} \Delta x_s^f & \Delta y_s^f & \Delta x_s^{f^2} & \Delta y_s^f & \Delta x_s^f \Delta y_s^f \\ \Delta x_s^1 & \Delta y_s^1 & \Delta x_s^{1^2} & \Delta y_s^1 & \Delta x_s^1 \Delta y_s^1 \\ \Delta x_s^2 & \Delta y_s^2 & \Delta x_s^{2^2} & \Delta y_s^2 & \Delta x_s^2 \Delta y_s^2 \\ \Delta x_s^3 & \Delta y_s^3 & \Delta x_s^{3^2} & \Delta y_s^3 & \Delta x_s^3 \Delta y_s^3 \\ \Delta x_s^4 & \Delta y_s^4 & \Delta x_s^{4^2} & \Delta y_s^4 & \Delta x_s^4 \Delta y_s^4 \end{vmatrix} \qquad (2.6)$$

$$= \begin{vmatrix} -\Delta x_f^s & -\Delta y_f^s & \Delta x_f^{s^2} & \Delta y_s^{f^2} & \Delta x_f^s \Delta y_f^s \\ \Delta x_s^1 - \Delta x_f^s & \Delta y_s^1 - \Delta y_f^s & (\Delta x_s^1 - \Delta x_s^f)^2 & (\Delta y_s^1 - \Delta y_s^f)^2 & (\Delta x_s^1 - \Delta x_f^s)(\Delta y_s^1 - \Delta y_f^s) \\ \Delta x_s^2 - \Delta x_f^s & \Delta y_s^2 - \Delta y_f^s & (\Delta x_s^2 - \Delta x_s^f)^2 & (\Delta y_s^2 - \Delta y_s^f)^2 & (\Delta x_s^2 - \Delta x_f^s)(\Delta y_s^2 - \Delta y_f^s) \\ \Delta x_s^3 - \Delta x_f^s & \Delta y_s^3 - \Delta y_f^s & (\Delta x_s^3 - \Delta x_s^f)^2 & (\Delta y_s^3 - \Delta y_s^f)^2 & (\Delta x_s^3 - \Delta x_f^s)(\Delta y_s^3 - \Delta y_f^s) \\ \Delta x_s^4 - \Delta x_f^s & \Delta y_s^4 - \Delta y_f^s & (\Delta x_s^4 - \Delta x_s^f)^2 & (\Delta y_s^4 - \Delta y_s^f)^2 & (\Delta x_s^4 - \Delta x_f^s)(\Delta y_s^4 - \Delta y_f^s) \end{vmatrix}$$

$$= -\begin{vmatrix} \Delta x_f^s & \Delta y_f^s & \Delta x_f^{s^2} & \Delta y_f^s & \Delta x_f^s \Delta y_f^s \\ \Delta x_f^1 & \Delta y_f^1 & \Delta x_f^{1^2} & \Delta y_f^1 & \Delta x_f^1 \Delta y_f^1 \\ \Delta x_f^2 & \Delta y_f^2 & \Delta x_f^{2^2} & \Delta y_f^2 & \Delta x_f^2 \Delta y_f^2 \\ \Delta x_f^3 & \Delta y_f^3 & \Delta x_f^{3^2} & \Delta y_f^3 & \Delta x_f^3 \Delta y_f^3 \\ \Delta x_f^4 & \Delta y_f^4 & \Delta x_f^{4^2} & \Delta y_f^4 & \Delta x_f^4 \Delta y_f^4 \end{vmatrix} = -|F|$$

Thus, we have shown that the two systems are both solvable and do exist. With this result, it is now trivial to prove that the two spline functions are equivalent. To show this, we first examine the difference between the two spline functions:

$$\begin{aligned} S - F &= g_s + a_s \Delta x_s + b_s \Delta y_s + \alpha_s \Delta x_s^2 + \beta_s \Delta y_s^2 + \gamma_s \Delta x_s \Delta y_s \\ &- g_f - a_f \Delta x_f - b_f \Delta y_f - \alpha_f \Delta x_f^2 - \beta_f \Delta y_f^2 - \gamma_f \Delta x_f \Delta y_f = \\ &A \Delta x_s + B \Delta y_s + \mathrm{A} \Delta x_s^2 + \mathrm{B} \Delta y_s^2 + \mathrm{X} \Delta x_s \Delta y_s \end{aligned} \qquad (2.7)$$

To simplify this expression further we apply the condition that both splines must have the same value at every node, in particular the *s* node. This tells us that the constant term, X, must be zero. Therefore, we can solve for the remaining five coefficients by solving the matrix equation:

$$\begin{bmatrix} \Delta x_f^s & \Delta y_f^s & {\Delta x_f^s}^2 & \Delta y_f^s & \Delta x_f^s \Delta y_f^s \\ \Delta x_f^1 & \Delta y_f^1 & {\Delta x_f^1}^2 & \Delta y_f^1 & \Delta x_f^1 \Delta y_f^1 \\ \Delta x_f^2 & \Delta y_f^2 & {\Delta x_f^2}^2 & \Delta y_f^2 & \Delta x_f^2 \Delta y_f^2 \\ \Delta x_f^3 & \Delta y_f^3 & {\Delta x_f^3}^2 & \Delta y_f^3 & \Delta x_f^3 \Delta y_f^3 \\ \Delta x_f^4 & \Delta y_f^4 & {\Delta x_f^4}^2 & \Delta y_f^4 & \Delta x_f^4 \Delta y_f^4 \end{bmatrix} \begin{bmatrix} A \\ B \\ \mathrm{A} \\ \mathrm{B} \\ \mathrm{X} \end{bmatrix} = 0 \qquad (2.8)$$

The 0 vector is a solution of this equation, telling us that indeed S-F = 0. This fact ensures that using these spline functions will conserve quantities across cell faces. In addition, we could save half of our computations in this module at each iteration if we organized the calculations correctly.

Simulation Unit of Length	**[x] = 1e-6 m**
Simulation Unit of Density	[ρ] = 1/[x^3] = **1e18 m^{-3}**
Simulation Unit of Velocity	**[v] = 3e8 m/s**
Simulation Unit of Time	**[t]** = [x]/[v] = **3.33e-15 s**
Simulation Unit of Charge	**[q] = 1.602e-19 C**
Simulation Unit of Mass	**[m] = 9.11e-31 kg**
Simulation Unit of Potential	**[ϕ]** = [m][v^2]/[q] = **5.12e5 V**
Simulation Unit of Energy	**[T]** = [ϕ] = **5.12e5 eV**
Simulation Unit of Electric Field	**[E]** = [ϕ]/[x] = **5.12e11 V/m**
Simulation Unit of Magnetic Field	**[B]** = [m]/([q][t]) = **1.71e3 T**

Table 2.1: Conversions from simulation units to physical (mks) units without yet accounting for acceleration factors.

2.3 Simulation Units

This RRC PIC/Vlasov model was developed with the intention of investigating not only low-density gas discharges such as Hall-effect thrusters but also multi-scale and high-density plasma phenomena such as the interaction of ultrafast laser pulses with matter. Due to the broad spectrum of possible applications, the simulation normalizations were not tailored to a particular regime but were instead keyed to basic physical constants. The units of normalization are presented in summary in Table 2.1.

2.3.1 Length Scale

The typical length scale employed when dealing with Hall thrusters is the Debye length given by:

$$\lambda_D = \sqrt{\frac{\varepsilon_0 T_e}{e n_e}} \tag{2.9}$$

For a typical thruster like the P5, given a nominal electron temperature of 30eV and a nominal plasma density of 1e13 cm^{-3} [73], the debye length is on the order of 10 μm. For the Near-Vacuum Hall thruster under consideration, the anode potential is 10 times that of the P5 thruster, so the electron temperature can be assumed to be on the order of 300eV. The nominal plasma density for the NVHT as discussed in Section 5.2.3 can be approximated to be 1e10 cm^{-3}, yielding a debye length of about 1000 μm.

While normalizing all lengths by the Debye length is appropriate for the study of Hall thrusters, it is not necessarily appropriate for the other applications to which this simulation might be applied. For example, when an ultrafast laser pulse bores into a solid aluminum target the plasma density inside the material can easily exceed 1e22 cm^{-3} while the nearby pre-plasma density may be many orders of magnitude smaller, less than 1e18 cm^{-3} [13]. Temperatures in both plasma regions may be of the same magnitude leading to Debye lengths of widely different values. In this case, where the Debye length is spatially radically variable, or in highly time-dependent cases where the Debye length changes greatly throughout the process, a "nominal density and temperature" often are not available or simply do not apply.

To alleviate this difficulty and simplify the system of units, a generally acceptable 1 μm was chosen as the length scale for simulation normalization. Clearly, this is not a one-size-fits-all choice. Should the need arise to simulate extremely low-density plasma phenomena, such as those often studied in astrophysical situations where length scales grow to the order of meters or kilometers [46], then this unit of normalization can easily be changed within the code. However, for our present purposes, 1 μm represented a "goldilocks" solution, neither too big nor too small for any of the applications at hand.

2.3.2 Time Scale

As with the length scales, the appropriate time scales for the wide-range of plasma applications under consideration are often diverse, variable, or difficult to define. In most cases, the electron plasma frequency would seem to be the relevant scaling factor [73][13]:

$$\omega_{pe} = \sqrt{\frac{ne^2}{m\varepsilon_0}} \tag{2.10}$$

but again deciding on a nominal plasma density proves difficult for many systems. Therefore, it was decided to choose a physical time scale which would be a constant across all systems. Seeing as the speed of light seems to be the universe's choice for the normalization of velocities, it was hoped that our humble simulation might also benefit from such a choice. All velocities in the simulation are normalized by c, the speed of light. Given, therefore, length and velocity scaling factors, the normalization of time was simply calculated as the ratio of the former to the latter.

2.3.3 Other Derived Units

The final two choices required to fully ground our system of simulation units are quantities of charge and mass. As is often done for plasmas, the unit of mass was simply chosen to be m_e, the fundamental mass of an electron, and the unit of charge to be q, the magnitude of the electron's charge. From these values, units for electric field, magnetic field, voltages, and any other necessary parameters were derived in a straightforward manner. For reference, all of the multiplication factors used for conversion from simulation units back into physical units are provided, as mentioned previously, in Table 2.1 above.

2.4 Acceleration Techniques

Full-PIC simulations are notoriously computationally expensive and sluggardly for three primary reasons, each of which have been mediated by previous authors through the use of certain computational techniques [73][75]. The current simulation modified significantly the first of these acceleration techniques, super-particles, but used the remaining two, artificial permittivity and mass ratio, in exactly the same manner as previous authors. For the sake of completeness, brief descriptions of all three techniques are provided below, along with the rationale and effects of each on simulation results, but further treatments can readily be found in the references.

2.4.1 Reducing the Simulated Particle Number – Statistical Weighting

If we imagine simulating each and every particle in the channel of a Hall thruster with a volume on the order of 100 cm^3 and a plasma density on the order of 1e13 cm^{-3}, we would be attempting to track around 1 trillion particles, which would clearly strain even the best computers' memory capabilities. Even assuming the particles could be stored, the amount of time required to compute the trajectories and interactions of this many particles would certainly be unreasonable given any practical current computational device. As such, it is important that the number of particles simulated be reduced.

The previously preferred method for reducing the number of particles in the simulation is known as "super-particles", allowing a single computational particle to represent some number (usually ~10^6) of real, physical particles. This technique is pioneered in numerous references [63][41][17]. Szabo [73] also clearly detailed the derivation of a "nominal super-particle" size, noting also that because the neutral density in a typical Hall thruster is usually much larger than the plasma density, a separate value for nominal plasma super-particle size and nominal neutral super-particle size should be calculated. In his simulations, Szabo also found that the super-particle size could indeed be increased without significant loss of accuracy up to a certain point, beyond which the number of computational particles per grid cell became too small and particle statistics began to grow inaccurate. Thus, a trillion real particles were easily reduced to 1 million computational particles.

In the current simulation, we extend this idea one step further. Instead of calculating a "nominal" super-particle size, we give each individual computational particle its own relative statistical weight, a decimal number unique to that particle. Thus, a single computational particle may represent large numbers of, likely low-energy particles, or merely a few high-energy particles from the extreme tail of the energy distribution function. This seemingly minor change in thinking has deceptively profound benefits for the simulation as a whole.

First and foremost, as hinted to in the previous paragraph, the statistical sampling error inherent in previous simulations with uniform particle weighting which caused the statistically-small but extremely relevant high-energy tails of particle distributions to become undersampled and underrepresented can now be eliminated. If particles are seeded with a uniform distribution of velocities but weighted by their statistical relevance, proper coverage of velocity space can be assured throughout the simulation. Conversely, computational effort is not wasted simulating a large number of very similar particles such as cold, secondary electrons which otherwise might rapidly become the most common, but uninteresting, members of the electron population. These cold particles can, carefully at intervals, be conglomerated together into a single particle with greater weight without significant loss of accuracy.

Further benefits and effects of this adjustment in the statistical weighting of particles are discussed in Sections 2.8 and 2.9. For the present discussion, however, it is clear that the individual weighting of particles will generally have the same effect on the computational expense of simulation that previously was accomplished by super-particles, that is the drastic reduction in the number of particles required.

2.4.2 Artificial Permittivity

The permittivity of free space defines the constant of proportionality between displacement in a vacuum electric field and the field's intensity. Therefore, if this constant is hypothetically increased as in:

$$\varepsilon_0^{sim} = \varepsilon_0 \gamma^2 \qquad (2.11)$$

with $\gamma > 1$, then the displacement of a particle in a field of a given strength will be decreased. Conversely, if we think in terms of Poisson's equation, we see that for a given separation of charge, increasing ε_0 leads to a smaller electric potential and thus a smaller electric field.

These facts help reduce the cost of simulation in two main ways. First, the Debye length is increased by a factor of γ increasing the length scale of many significant plasma features such as plasma-wall sheaths. This allows us to reduce the resolution of our computational mesh by the factor γ in each direction, saving us γ^2 in terms of number of elements. Secondly, the plasma frequency and other oscillations of significance are also reduced, allowing us to take bigger timesteps and still resolve these features.

In our simulation, this artificial permittivity results in a number of changes to the normalization constants previously discussed. Table 2.2 extends previous Table 2.1 and contains the units of normalization now with the artificial permittivity taken into account. The primary changes, as mentioned above, are that length and time scales in the simulation are increased by a factor of γ. The remaining changes follow from these two facts.

It is worth mentioning the calculation of ε_0 in our simulation since we are not working in a unit system which reduces the free space permittivity to 1. In MKS units:

$$\varepsilon_0 = 8.854\text{e-}12\ \mathrm{C^2 s^2 m^{-3} kg^{-1}} \qquad (2.12)$$

We need to apply the inverses of the conversion factors listed in Table 2.2 to move this value from physical units into simulation units. Charge and mass are both unaffected by the artificial permittivity factor, but time and distance are. So, in the end, we see that the cubic distance term is not entirely cancelled by the squared time term, leaving a single γ factor which already therefore incorporates itself into the permittivity. This γ is separate from the γ^2 multiplication factor which we must also then apply, leaving us, in the end with:

$$\varepsilon_0^{sim} = \gamma^2 \varepsilon_0 \frac{[x]^3[m]}{[q]^2[t]^2} = \gamma^3\left(2.829*10^7\right) \qquad (2.13)$$

	Without Artificial ε_0	**With Artificial ε_0**
Sim Unit of Length	$[x] = 1e\text{-}6$ m	$[x] = (1e\text{-}6 \text{ m})\gamma$
SimUnit of Density	$[\rho] = 1/[x^3] = 1e18 \text{ m}^{-3}$	$[\rho] = 1/[x^3] = (1e18 \text{ m}^{-3})/\gamma^3$
Sim Unit of Velocity	$[v] = 3e8$ m/s	$[v] = 3e8$ m/s
Sim Unit of Time	$[t] = [x]/[v] = 3.33e\text{-}15$ s	$[t] = [x]/[v] = (3.33e\text{-}15 \text{ s})\gamma$
Sim Unit of Charge	$[q] = 1.602e\text{-}19$ C	$[q] = 1.602e\text{-}19$ C
Sim Unit of Mass	$[m] = 9.11e\text{-}31$ kg	$[m] = 9.11e\text{-}31$ kg
Sim Unit of Potential	$[\phi] = [m][v^2]/[q] = 5.12e5$ V	$[\phi] = [m][v^2]/[q] = 5.12e5$ V
Sim Unit of Energy	$[T] = [\phi] = 5.12e5$ eV	$[T] = [\phi] = 5.12e5$ eV
Sim Unit of Electric Field	$[E] = [\phi]/[x] = 5.12e11$ V/m	$[E] = [\phi]/[x] = (5.12e11 \text{ V/m})/\gamma$
Sim Unit of Magnetic Field	$[B] = [m]/([q][t]) = 1.71e3$ T	$[B] = [m]/([q][t]) = (1.71e3 \text{ T})/\gamma$

Table 2.2: Conversions from simulation units to physical (mks) units accounting for artificial permittivity.

2.4.3 Artificial Mass Ratio

Electron kinetics occur on a much faster time scale than their heavier counterparts, ions and neutrals. Since full-PIC attempts to resolve this electron motion rather than assume a fluidic approximation, our simulation time step must be short enough to resolve the electron plasma frequency, or about 10^{-11} s^{-1}. The transit time of an ion through the typical P5 Hall thruster simulation domain could conservatively be estimated to be 5 nanoseconds assuming a channel length of 4 cm and a velocity given by a 200 Volt potential drop. Neutrals are even slower since they are not electrostatically accelerated and could take a million or more plasma times to cross the channel.

To speed the rate of convergence of the simulation, we follow the lead of Szabo and others in the plasma community before him [73][75]. The mass of the heavy particles in the simulation, both ions and neutrals, is scaled by an artificial mass ratio factor, $f < 1$, such that:

$$f = \frac{M^{sim}}{M^{real}} \tag{2.14}$$

The main benefit of adding this factor and reducing the effective mass of the heavy particles is that the thermal velocity will then be increased by a factor of the square root of *f*:

$$\frac{v_t^{sim}}{v_t^{real}} \propto \sqrt{\frac{m^{real}}{m^{sim}}} \propto \sqrt{\frac{1}{f}} \tag{2.15}$$

A number of other factors within the code must then be scaled because of this artificial mass ratio. The details of those alterations can be found in Szabo's thesis and have been included in the present simulation. The interested reader can refer to the references for a complete analysis of the effects that an artificial mass ratio may have on the simulation.

2.5 Semi-Analytical Particle Trajectory Integrator

Other than the complicated relativistic calculations required, the particle motion functions are fairly straightforward in implementation. A flowchart of the movement process is shown below in Figure 2-9 for clarity. Basically, the process for each particle is completed separately, a fact which makes parallelization of this step very efficient as discussed later in Section 2.13. For a given particle, the electromagnetic fields acting on its current position are interpolated using a weighted average of the fields in the current cell and in the neighboring cells. This is known as the gathering step. Next, a leapfrog-type algorithm is implemented during which the particle moves first half a time step in the electric field, then experiences a full time step velocity rotation in the magnetic field, and finally accelerates another half time step in the electric field. Boundary conditions are next enacted for particles that have moved outside the simulation domain, and if the particle has not been absorbed, its charge density is added to the overall distribution which will then be used to update the electric field during a later phase of the simulation.

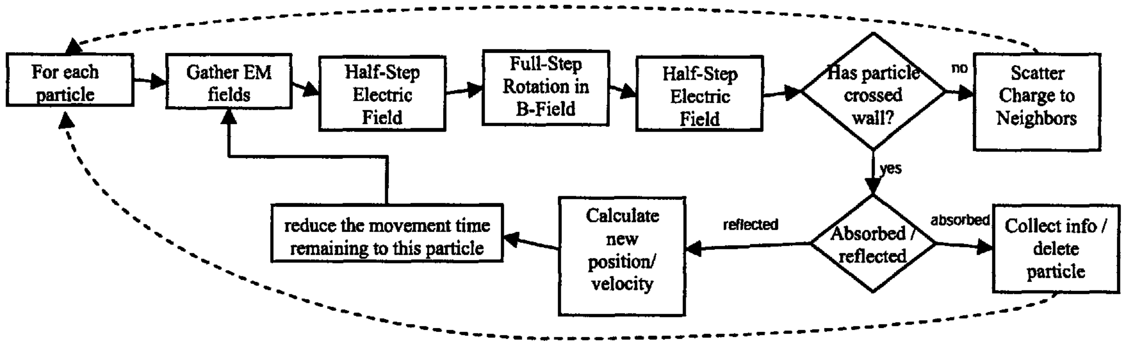

For each particle
Gather EM fields
Half-Step Electric Field
Full-Step Rotation in B-Field
Half-Step Electric Field
Has particle crossed wall?
no
Scatter Charge to Neighbors
yes
Absorbed / reflected
absorbed
Collect info / delete particle
reflected
Calculate new position/ velocity
reduce the movement time remaining to this particle

Figure 2-9: A flowchart depicting the particle movement process in the serial implementation.

The heart of the entire particle movement process is the correct calculation of charged particle motion under the influence of electromagnetic fields, and it is vital to the overall accuracy of the model. The previous MIT PIC simulation [73] opted for a finite-difference approximation to the particle's trajectory which was simpler to calculate and implement. It was decided that the present work would improve upon this calculation and attempt to analytically integrate the equations of motion so as to reduce the error in the particle trajectories and create an unconditionally stable technique. Indeed, the use of these analytical expressions rather than finite-differencing techniques ensures that a particle will remain on the correct helical orbit over an arbitrary number of time steps. The treatment which follows is due to Batishchev and Batishcheva [10]. The present restatement of that work is given for completeness since the motion integrator which was implemented for this simulation incorporates their resulting expressions.

2.5.1 Semi-Analytical Relativistic Orbit Integrator for a 2D2V System

The equations of motion for a general relativistic particle in electromagnetic fields are:

$$\begin{aligned} \frac{d\vec{r}}{dt} &= \vec{v} = \frac{c\vec{p}}{\sqrt{m_0^2 c^2 + p^2}} \\ \frac{d\vec{p}}{dt} &= q\left(\vec{E} + \frac{\vec{v}\times\vec{B}}{c}\right) \end{aligned} \tag{2.16}$$

where m₀ is the particle's rest mass and $\vec{p} = m_0 \gamma \vec{v}$ is its momentum. The quantity γ is of course the relativistic gamma and is defined as:

$$\gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}} = \sqrt{1 + \frac{p^2}{m_0^2 c^2}} \tag{2.17}$$

The motion of the particle over a single time step will be divided into three separate substeps, all with the same frozen EM fields. A half time step acceleration in the electric field, $\vec{E}$, is followed by a full time step pure rotation in the magnetic field, $\vec{B}$, which is in turn followed by a final half time step in the electric field. For the 2D2V case, we can determine an analytic expression for the particle's trajectory during each of the three separate substeps, the first and last, of course, having matching structures. The momentum equations for the two different types of substeps therefore read:

$$\begin{aligned} \frac{d\vec{p}_1}{dt} &= q\vec{E} \\ \frac{d\vec{p}_2}{dt} &= q\frac{\vec{v}_1 \times \vec{B}}{c} \end{aligned} \tag{2.18}$$

The structure of the system we will be solving for is here elaborated. The electric field is entirely inside the plane, $\vec{E} = (E_x, E_y, 0)$ while the magnetic field is only perpendicular to the plane, $\vec{B} = (0,0,B_z)$. We will take the angle that the electric field makes with the x-axis to be θ, such that $E_x = |\vec{E}|\cos\theta$ and $E_y = |\vec{E}|\sin\theta$. In addition, we are given the initial position, (x^0, y^0), of the particle along with the projections of its initial momentum, $p^0 = \sqrt{p_x^{0^2} + p_y^{0^2}}$.

2.5.1.1 Half-Step in Electric Field

Given this basic set-up, Batishchev and Batishcheva [10] solved for the new displacements and momenta of the particles after the initial half-step in the electric field. These final quantities parallel to the electric field are given by:

$$\begin{aligned} p_\parallel^1 &= p_\parallel^0 + qEt \\ \delta_\parallel &= \frac{c}{qE}\left[\sqrt{m_0^2c^2 + p_\perp^{0\,2} + p_\parallel^{1\,2}} - \sqrt{m_0^2c^2 + p_\perp^{0\,2} + p_\parallel^{0\,2}}\right] \end{aligned} \tag{2.19}$$

The latter represents an analytic expression for the particle's displacement over a half time step in a frozen electric field. The one remaining issue to be resolved with the above expression is the singularity present as E→ 0. Common sense tells us, however, that if the field is negligible, the force on the particle will also be very small, and so in practice, we avoid this singularity by checking that the electric field is greater than some ε before the particle is moved. If the field is significant, we update the particle's momentum and position using the expressions above. If the field is small, however, we simply leave the particle's momentum unchanged and update its position in the straightforward way assuming a constant momentum. Another way to understand the handling of this singularity would involve performing a linear or quadratic expansion of the final parallel momentum term. This would be valid for small E, revealing, as before, that the momentum would remain unchanged and the displacement could be calculated assuming a constant momentum.

In addition, the displacement perpendicular to the electric field was calculated. The lack of force in the perpendicular direction and the constant, unchanging perpendicular momentum make this calculation tractable. The exact expression for the perpendicular displacement is given in reference [10] as:

$$\delta_\perp = \frac{cp_\perp^0}{qE}\ln\left[\frac{p_\parallel^1 + \sqrt{p_\parallel^{1\,2} + m_0^2c^2 + p_\perp^{0\,2}}}{p_\parallel^0 + \sqrt{p_\parallel^{0\,2} + m_0^2c^2 + p_\perp^{0\,2}}}\right] \tag{2.20}$$

At this point we have calculated all of the quantities we need to update the particle's state for this substep. However, for implementation, we also need to rotate these final values back into our original x-y coordinate system. The momentum vectors are returned to this system by a simple -θ rotation:

$$\begin{aligned} p_x^1 &= p_\parallel^1 \cos\theta - p_\perp^1 \sin\theta \\ p_y^1 &= p_\parallel^1 \sin\theta + p_\perp^1 \cos\theta \end{aligned} \tag{2.21}$$

The displacement vectors also undergo this rotation, yielding the particle's final position after this substep:

$$\begin{aligned} x^1 &= x^0 + \delta_\parallel^1 \cos\theta - \delta_\perp^1 \sin\theta \\ y^1 &= y^0 + \delta_\parallel^1 \sin\theta + \delta_\perp^1 \cos\theta \end{aligned} \tag{2.22}$$

Of course, these expressions are also applied for the third substep given a new electric field vector and new "initial" momenta.

2.5.1.2 Full Step in Magnetic Field

For the magnetic field substep, it is assumed that the particle's position remains constant while its momentum alone is rotated by the present magnetic field. This makes the calculation of the magnetic field's effect shorter and simpler than that of the electric field. In addition, merely the direction of the momentum is changed while its magnitude remains constant, further simplifying by holding the relativistic factor, γ, constant.

Using the relativistic expression for the gyro-frequency, the particle momentum will rotate through the angle:

$$\beta = \frac{2\pi q B}{m_0 c \gamma_1} = \frac{2\pi q B \tau}{\sqrt{m_0^2 c^2 + p_1^2}} \tag{2.23}$$

This rotation implies that the new components of momentum are just:

$$\begin{aligned} p_x^2 &= p_x^1 \cos\beta - p_y^1 \sin\beta \\ p_y^2 &= p_x^1 \sin\beta + p_y^1 \cos\beta \end{aligned} \tag{2.24}$$

2.5.2 Analytical Orbit Integrator in the Classical 3D3V Case

Before tackling the fully relativistic 3D3V case, let us first examine the classical case in which v << c. This type of integrator was implemented along with the relativistic ones for use with some low energy systems (such as most Hall thrusters) for which the added complexity of relativistic calculations is not completely necessary. In the classical limit, Equation *2.16* describing the particle's motion now simply becomes the Newtonian system:

$$\begin{aligned} \frac{d\vec{r}}{dt} &= \vec{v} = \frac{\vec{p}}{m_0} \\ \frac{d\vec{p}}{dt} &= q\left(\vec{E} + \frac{\vec{v}\times\vec{B}}{c}\right) \end{aligned} \tag{2.25}$$

Fortunately, this system is simple enough, albeit geometrically complicated, that it can be analytically integrated under the assumption that the electromagnetic fields remain frozen throughout an entire time step. Batishchev and Batishcheva have reported [10] that the displacements may be given by:

$$\begin{aligned} \delta_x &= \frac{1}{m_0}\left\{\frac{\beta_x\tau}{\omega^2} - \frac{\beta_x(\sin\omega\tau + \cos\omega\tau - 1)}{\omega^3} + \frac{p_x^0\sin\omega\tau + p_y^0(1-\cos\omega\tau)}{\omega}\right\} \\ \delta_y &= \frac{1}{m_0}\left\{\frac{\beta_y\tau}{\omega^2} - \frac{\beta_y(\sin\omega\tau + \cos\omega\tau - 1)}{\omega^3} + \frac{p_x^0(\cos\omega\tau - 1) + p_y^0\sin\omega\tau}{\omega}\right\} \\ \delta_z &= \frac{1}{m_0}\left[p_z^0\tau + \frac{qE_z\tau^2}{2}\right] \end{aligned} \tag{2.26}$$

assuming the coordinate system has been properly rotated. Again, these expressions are entirely analytic and require no approximation, meaning they are accurate for arbitrary τ, unlike the finite-difference expressions generally used in past Hall thruster simulations. Even effects dependent upon gradients in the fields, such as the ∇B drift, can be modeled by decreasing the time step and using the proper value of the fields given each new position of the particle.

2.5.3 Semi-Analytic Orbit Integrator for the Relativistic 3D3V Case

Sadly, it is not possible to obtain full analytic solutions for the relativistic 3D3V case analogous to those described in the previous classical section. However, we can perform the same sub-stepping procedure that was described for the two-dimensional case in Section 2.5.1, analytically solving for the partial trajectories over each separate sub-step. We once again freeze the electric and magnetic fields, step the particle a half time-step in the electric field, a full time-step in the magnetic field, and then a final half time-step in the electric field. The solution process is outlined in the sections which follow.

2.5.3.1 Half Step Acceleration in Electric Field

Reference [10] shows that this calculation proceeds very similarly to that described in Section 2.5.1.1 above, with the significant difference that the momentum and electric field vectors no longer need lie in the same plane. The expressions for the displacements parallel and perpendicular to the electric field are then given by:

$$\begin{aligned} \delta_{\parallel} &= \frac{c}{qE}\left[\sqrt{m_0^2c^2 + p_\perp^{0^2} + p_\parallel^{1^2}} - \sqrt{m_0^2c^2 + p_\perp^{0^2} + p_\parallel^{0^2}}\right] \\ \delta_{\perp} &= \frac{cp_\perp^0}{qE}\ln\left[\frac{p_\parallel^1 + \sqrt{p_\parallel^{1^2} + m_0^2c^2 + p_\perp^{0^2}}}{p_\parallel^0 + \sqrt{p_\parallel^{0^2} + m_0^2c^2 + p_\perp^{0^2}}}\right] \end{aligned} \tag{2.27}$$

2.5.3.2 Full Step Rotation of Velocity Vector in Magnetic Field

The magnetic field rotation is again very similar to the 2D2V case, except for the arbitrary angle between the magnetic field and the momentum. This is accounted for by first breaking down the momentum into components parallel and perpendicular to the magnetic field, just as was done above for the electric field:

$$\begin{aligned} \vec{p}_{\parallel b} &= \left(\vec{p}\cdot\vec{b}\right)\vec{b} = p_{\parallel b}\vec{b} \\ \vec{p}_{\perp b} &= \vec{p} - \vec{p}_{\parallel b} = p_{\perp b}\vec{n} \end{aligned} \tag{2.28}$$

where in this case, $\vec{b}$ and $\vec{n}$ are unit vectors parallel to the magnetic field and to the component of the momentum which is perpendicular to the magnetic field, respectively.

With this transformation completed, the momentum vector is then seen to rotate only in the plane perpendicular to the magnetic field and by an angle, as in the 2D2V case of:

$$\beta = \frac{2\pi qB\tau}{\sqrt{m_0^2 c^2 + p_1^2}} \qquad (2.29)$$

This implies that the final momentum when reconstituted back into the original coordinate system is given by:

$$\vec{p}_2 = \left(p_{\parallel b}\right)\vec{b} + \left(p_{\perp b}\cos\beta\right)\vec{n} + \left(p_{\perp b}\sin\beta\right)\vec{u} \qquad (2.30)$$

where $\vec{u} = \vec{b} \times \vec{n}$ is the final vector in the associated right-handed coordinate system. It is easy to verify that the magnitude of the momentum is unchanged, and that these expressions represent a pure rotation in the magnetic field.

2.5.4 2D3V Methodology

Hall thrusters, as well as numerous other plasma applications, have often been modeled as axisymmetric systems with varying degrees of accuracy. Neglecting the θ-variations greatly simplifies our models and without this assumption, the computational complexity of accurately simulating a full Hall thruster system would be exorbitant and likely unattainable with the present state of hardware. Unfortunately, this computational necessity has recently been found to hinder our modeling efforts significantly, especially in the regime of three-dimensional turbulent effects which appear to control anomalous transport. As such, it is our recommendation that the move to fully three-dimensional models be made as soon as it is computationally possible.

Until that time, however, the best approximation we can make is to allow all vectors, including fields and momenta, to retain all three dimensions while assuming that spatially, θ-variations are negligible, folding any azimuthal motion back into the RZ-plane. This is what is known as the 2D3V approach.

To implement this concept, a few final adjustments needed to be made to our three-dimensional particle orbit integrators. First of all, they could be simplified in the sense that both the electric and magnetic azimuthal components could be assumed to be zero. For the sake of generality, however, our implementation actually maintained the possibility of azimuthal fields, so this simplification will not be detailed.

The more important step in moving from 3D3V to 2D3V is the resolving of particle motion out of the RZ-plane back into it. This process is illustrated in the simple example shown in Figure 2-10. At the end of a given time step, a particle's position will have both x and y components. The R-axis is assumed to be aligned with the y-axis, and so simple Pythagorean theorem tells us that to eliminate the θ-component of position, the new R-position must become:

$$R = \sqrt{x^2 + y^2} \tag{2.31}$$

This "folding" is shown in the figure, and in effect motion in the θ-direction is directly translated into motion in the positive R-direction. The Z-position remains unaffected.

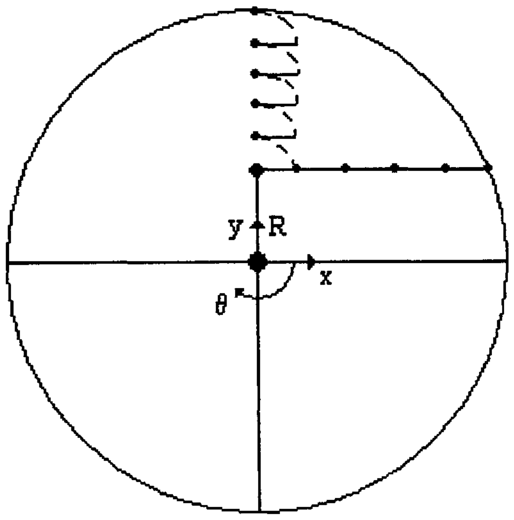


Figure 2-10: Particle motion physically in the θ-direction is folded back into the RZ-plane and becomes motion in the positive R-direction under the 2D3V assumption.

2.6 Particle-in-Cell Weighting Scheme

While the particles are able to move freely about the simulation domain, in order to solve for fields and other quantities, the particle-in-cell methodology requires that the important quantities be interpolated or weighted to a computational mesh. The next sections discuss the implementation of that weighting procedure as it pertains to our code.

2.6.1 Weighting Scheme in 2D

Initially the method chosen to weight the particles to the nodes was to examine, for example, *n* neighbors surrounding the cell currently holding a particle and to use the normalized inverse distances as weights. Thus, if the particle was close to a given cell's center, the distance to the cell's center would be small and the amount of density assigned to that cell would be large. This method had the advantage of being very compatible with our unstructured RRC-adapted meshes. All that was needed to be known was the distance to the *n* closest cell centers, and there were no issues to be addressed concerning the various sizes a cell might have. Unfortunately, the method was not numerically sound. Figure 2-11 attempts to demonstrate this schematically. Here $n=9$ has been chosen. The particle is initally weighted to the closest cell centers, indicated by the filled squares. The intensity of color in the squares is indicative of how much weight from the particle is assigned to that particular cell. After the particle moves a possibly infinitesimal amount and crosses the boundary of the two central cells, the image is drastically changed. The weights in the top three cells would never go to zero in the limit as the particle reached the boundary and crossed it. Similarly, the weights in the bottom three cells jump from zero to some non-zero value non-smoothly as the particle transitions. It was found that this lack of smoothness in the weighting function led to very significant self-forces being calculated on particles as they crossed from one mesh cell to another. As such, the inverse-distance weighting scheme needed to be abandoned and a new, smoothly-changing method developed.

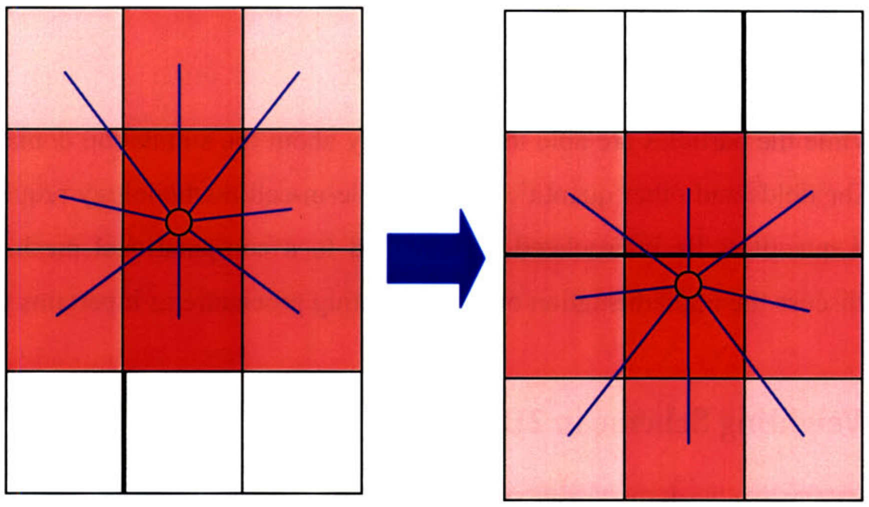
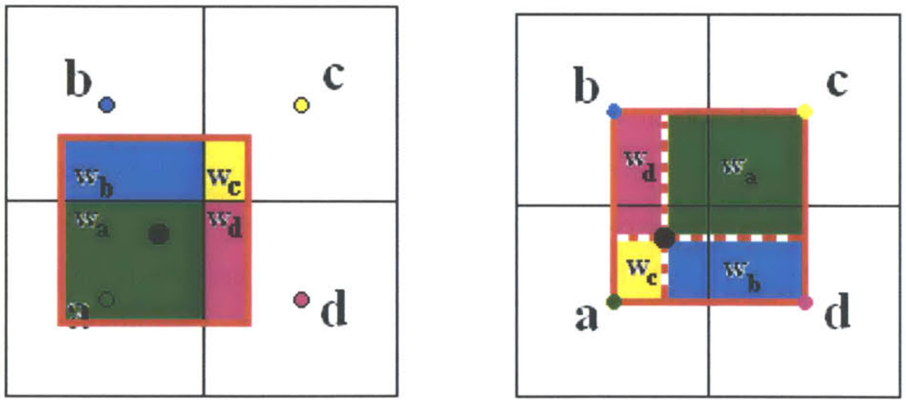


Figure 2-11: Under the inverse-distance weighting scheme, the weighting factors could change non-smoothly even during an infinitesimal particle motion. These discontinuities in weighting sometimes led to a significant self-force being calculated on an individual particle.

The essence of the final adopted PIC weighting scheme which did exhibit the smooth transition between cells that we required is shown graphically in Figure 2-12 and Figure 2-13 below. The method employed, given the simplest case of an orthogonal, unrefined mesh, can be boiled down to what is essentially a calculation of overlapping volumes. Figure 2-12(a) shows how the implementation of this method can be visualized in that simple case. Basically, the particle is assumed to occupy a volume of space equal to the volume of the cell in which it currently resides. This volume is centered on the particle and extends partially into neighboring cells. The fraction of the particle's weight assigned to a given cell is then just the amount of the particle's imagined volume which overlaps with the cell. Thus, only a small amount of the particle's volume overlaps cell *c* and so the portion of the weight it receives is much smaller than that assigned to cell *a* where the particle actually rests.

In reality, the computation is not performed in exactly that way since such a method does not generalize well to RRC-adapted meshes with highly-varying cell volumes. Figure 2-12(b) gives a more accurate picture of the actual method of computation. Basically, the four closest cell centers to the particle are first located. Then two new points are defined, being the intersection points of the opposite edges of the solid red square in the figure. That is, one point would be the intersection of segments *ab* and *cd* and the other point would be the intersection of segments *bc* and *ad*. These points are shown in Figure 2-13 for a general, non-uniform and non-orthogonal mesh. In the degenerate case where the segments are parallel, the points are taken to be directly perpendicular to the particle's location and on the solid red square in the positive x and positive y direction respectively. Using these points, a cross like the dotted red one shown in the figures, is then constructed. Each line defining that cross is created such that it passes through one of the two intersection points and through the particle's position. With that cross as a division boundary, the volumes of the four quadrilaterals thus formed are computed. These volumes, when normalized, are assigned as the weight of the cell directly opposite, as shown in the figure. Thus, the small yellow square formed defines the weight assigned to element *c* while the volume of the largest square formed is used as the weight for element *a*.

Comparing Figure 2-12(a) to Figure 2-12(b) one can see that in the simple unrefined, orthogonal case, the two methods are exactly equivalent. However, the method outlined in Figure 2-12(b) can deal easily with cells of varying sizes. In some cases, the RRC-adaptation may necessitate the usage of a three-point triangular method, as well, which is deftly handled by the latter computation as well.



(a) **(b)**

Figure 2-12: Illustration of PIC-weighting method. (a) Conceptually, we calculate the overlapping volume of a particle with the neighboring cells. (b) For the general refined and non-orthogonal case, a different computational implementation is actually used to compute the weights which does reduce in the simple case to an overlapping volume scheme.

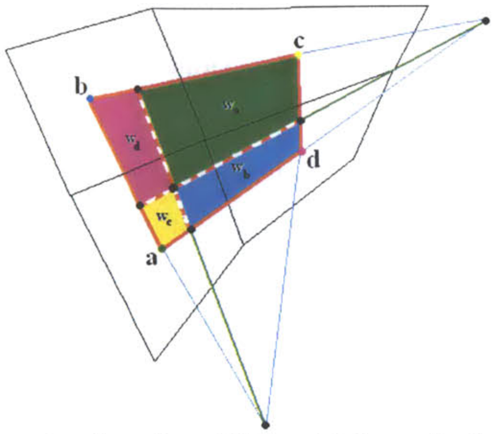


Figure 2-13: The "overlapping volume" PIC-weighting method on a general mesh.

Most importantly, the above method yields smooth transitions in weighting as a particle moves throughout the grid. If the particle were to move south and pass over segment *ad* in Figure 2-12(b), for example, the weight assigned to cells *b* and *c* would reach zero at the point where the particle rested exactly on the segment. In addition, as the particle moved farther south, the weights assigned to the new cells would initially begin smoothly at zero and not start with a discontinuous jump. This method was able to alleviate the self-force on particles as is demonstrated in Section 3.3.1.

2.6.2 Weighting Scheme for RZ Geometries

The PIC weighting functions were designed with two-dimensional grids in mind. As such, they did not automatically take into account certain effects inherent to an axisymmetric simulation. We here address the relatively minor adjustment that needed to be taken into account to correct that problem.

Our desire is that the weighting scheme should approximate the effect of an overlapping volume scheme. The previous section discussed the alternative method of implementation that was used in order to accommodate our RRC meshes and make the calculations more efficient. This method, when naively transferred to an RZ geometry no longer acts like an overlapping volume method and creates erroneous results. For example, imagine we have the same situation that is demonstrated by Figure 2-12 above. The naïve implementation would simply revolve the squares labeled *w* about the main axis of symmetry in the simulation to form ring elements. This axis was taken to be y=0 for all of our simulations, and we can picture it as being parallel to segment *ad* and, without loss of generality, running along the bottom of those cells. Thus, after being revolved, one can easily see that in Figure 2-12(a), the weight of w_b relative to w_a has been increased since the radius of revolution is larger in w_b's case. That is how an overlapping volume (overlapping rings) scheme should function. However, if we examine the volumes in Figure 2-12(b) after revolution, we would find that using this method w_a has now been increased relative to w_b. This is exactly the opposite of what we want. The problem stems from the fact that the computational method used does compute the overlapping volumes, but that their positions are more or less flipped both horizontally and vertically.

Figure 2-14(a) below demonstrates the detrimental effect such a weighting scheme would have on the simulation. In this figure, a spatially uniform distribution of neutral particles has been seeded in an axisymmetric geometry. Despite this uniform distribution of particles, the erroneous method outlined above assigns too large a weight from a particle to the cells closer to the axis and too little to the cells farther away. For a general cell, the effect is small and is more or less balanced since its inner edge receives too little density, but its outer edge receives too much. For the cells on the centerline, the effect is unbalanced. In addition, the differences in the revolved volumes of an inner cell and an outer cell are relatively much larger right near the centerline, making the weighting error very significant.

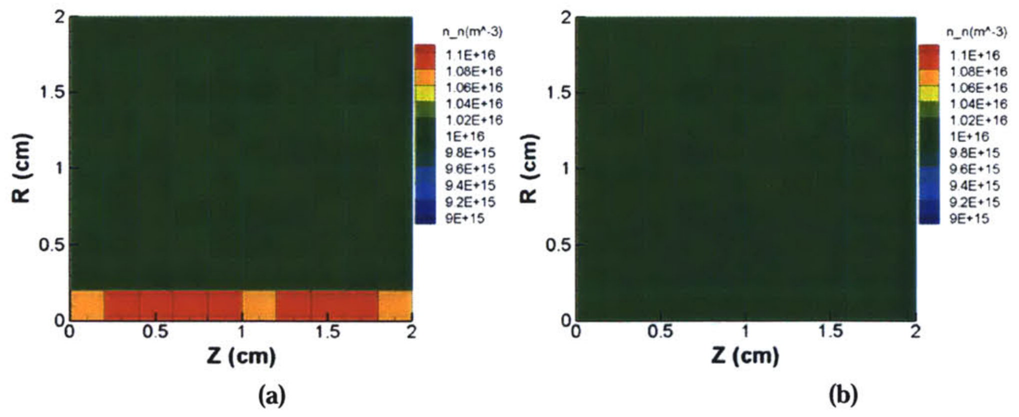


Figure 2-14: The density after scattering for a uniform distribution of neutrals was calculated using (a) the naïve generalization of the two-dimensional scheme and (b) a method which accounts for the RZ geometry.

Fortunately, this problem can be fixed with one fairly inexpensive adjustment. Basically, we cannot simply calculate the axisymmetric volumes of the boxes in Figure 2-12(b) above, because they are out of position and rotated compared to Figure 2-12(a). Instead, in the RZ case, we revolve the boxes calculated in Figure 2-12(b) not about y=0 as we usually would, but around a line above the cells which makes the horizontal dotted red line in Figure 2-12(b) be the same distance that the *ad* boundary is from y=0. This in effect creates the same volumes of revolution that we would have found from revolving Figure 2-12(a) about y=0. The result of this change gives a much more uniform distribution of density at the centerline as shown in Figure 2-14(b). This weighting method may still not be exactly correct in approximating an overlapping volume scheme on non-orthogonal or refined meshes. However, it does have the property of smoothness required to minimize the problem of self-fields on particles, is relatively simple to calculate, and exhibits reasonable trends as the particle moves near to or far from a cell center and near to or far from the axis of revolution.

2.7 Self-Consistent Calculation of Electric Field

The forces in a plasma may often depend not only upon boundary conditions and externally applied fields but also on the configuration of the plasma itself. Maxwell's equations govern these forces, and can be found in any common physics textbook, for example [67]. The present simulation was designed with two major branches, an electromagnetic branch and an electrostatic one. The electromagnetic branch assumes that the currents in the simulated plasma are strong and that the self-induced magnetic fields are not negligible. This branch of the simulation requires a very different approach and more computationally-intensive solution techniques to interpolate the full set of Maxwell's equations at every time step. Work on this branch was conducted by Batishcheva, and her results are reported elsewhere [9].

The present document will instead focus primarily on the electrostatic branch of the code while assuming that the magnetic field remains constant throughout. This enables us to reduce the complete set of Maxwell's equations to the much simpler

Poisson's Equation at each time step, which in its two-dimensional Gauss's Law form, is given by equation *2.32* below. This equation tells us that the flux of the electric field through the boundary of a computational element is equal to the total charge contained inside the area of that element. In three dimensions, this extends to the flux through the surface of an element and the charge inside the element's volume. The following sections will detail a number of different methods implemented for solving this equation at each time step for the new electric field given the current distribution of charge. While this problem can readily be solved for simple mesh geometries, the addition of our RRC adaptive meshing procedure and our insistence on using unstructured quadrilateral meshes required the development of entirely new solution techniques. These techniques all require approximations that range from simple to much more complicated but accurate. These approximations are outlined in Section 2.7.1 while the solution methods used to solve the resulting matrix equations are discussed in Section 2.7.2. Each approximation method can be combined with each of the solution methods, resulting in a large number of different Poisson solver permutations, each with its own strengths and weaknesses.

2.7.1 Approximations to Gauss's Law

Simulations always require approximations. In this case, we wish to discretize the Gauss's Law equation, Equation *2.32*, over a two-dimensional mesh which might also represent an axisymmetric, three-dimensional geometry. This must be accomplished while accounting for the fact that neighboring cells may be of very different sizes and that, due to the RRC meshing, each element may have anywhere between one and eight neighboring elements.

2.7.1.1 Approximating Gauss's Law Using Simple Linear Interpolation

The equation we seek to solve is Gauss's Law:

$$\oint \bar{E} \cdot ds = \iint \frac{q}{\varepsilon_0} dV \qquad (2.32)$$

with q here representing the charge distribution normalized by the dielectric constant, ε_0, and of course:

$$\bar{E} = -\nabla\Phi = -\frac{\partial\Phi}{\partial x} i_x - \frac{\partial\Phi}{\partial y} i_y \tag{2.33}$$

The simplest method of solving this equation such that the least amount of geometrical information is required involves linearly approximating the gradient in the above expression. We will outline that process for a single cell edge while in practice all four edges would need to be treated in a similar fashion to complete the integration.

Given the situation as demonstrated in Figure 2-15, we begin by realizing from trigonometry that an element of the face, *ds*, over which we are integrating the electric flux is given by:

$$\vec{ds} = \left\{ \frac{-(y_2 - y_1)\vec{i}_x + (x_2 - x_1)\vec{i}_y}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \right\} \tag{2.34}$$

A first order approximation in one-dimension to the potential gradient is given by:

$$-\nabla\vec{\Phi} = -\frac{\Phi_n - \Phi_s}{\sqrt{(x_s - x_n)^2 + (y_s - y_n)^2}} \vec{i}_{sn} \tag{2.35}$$

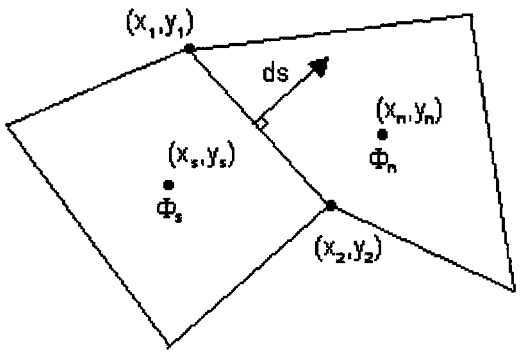


Figure 2-15: The gradient in potential between cell *s* and cell *n* must be integrated over facet 1-2.

where $\vec{i}_{sn}$ is a unit vector which points along the direction from cell center *s* to cell center *n*. Now breaking this expression into its x and y components to find a two-dimensional representation, we find:

$$-\nabla\vec{\Phi} = \left[-\frac{\Phi_n - \Phi_s}{\sqrt{(x_s - x_n)^2 + (y_s - y_n)^2}}\right]\frac{\left[(x_n - x_s)\vec{i}_x + (y_n - y_s)\vec{i}_y\right]}{\sqrt{(x_s - x_n)^2 + (y_s - y_n)^2}} \qquad (2.36)$$

Our next assumption is that this gradient value is constant over the entire face, and so the integration is simply transformed into a multiplication by the face area. Putting everything together, and assuming a constant charge distribution, q_s, throughout the entire cell volume, our final expression, which will be solved either by SOR iteration or by use of a direct matrix solver, is:

$$-\frac{(\Phi_n - \Phi_s)\left[(x_n - x_s)(y_1 - y_2) + (y_n - y_s)(x_2 - x_1)\right]}{\left[(x_s - x_n)^2 + (y_n - y_s)^2\right]}\frac{face\_area}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} = \frac{q_s}{\varepsilon_0}(cell\_vol) \qquad (2.37)$$

In the case where only two-dimensions are assumed in the simulation, one might notice that the expression for *face_area* and the denominator associated with *ds* are in fact the same and cancel. However, in the axisymmetric case, these two terms are actually different, with the *face_area* being equal to:

$$face\_area_{RZ} = \pi(y_1 + y_2)\sqrt{(y_1 - y_2)^2 + (x_1 - x_2)^2} \qquad (2.38)$$

This represents not just the mesh line's length, but its surface area when revolved about the centerline of the simulation, and it is therefore included above.

2.7.1.2 Approximating Gauss's Law Using Splines

Again, the equation we seek to solve is:

$$\oint \vec{E} \cdot ds = \iint \frac{q}{\varepsilon_0} dV \tag{2.39}$$

To accomplish this, we will herein attempt to use the iterative technique of successive over-relaxation. First we remember that:

$$\vec{E} = -\nabla\vec{\Phi} = -\frac{\partial \Phi}{\partial x}\vec{i}_x - \frac{\partial \Phi}{\partial y}\vec{i}_y \tag{2.40}$$

and that an element of the boundary with a direction outwardly normal to the face is just:

$$\vec{ds} = (-dy)\vec{i}_x + (dx)\vec{i}_y \tag{2.41}$$

These two facts allow us to rewrite equation *2.39* as:

$$\sum_{i=0}^{3}\left(\int_{y_i}^{y_{(i+1)\bmod 4}} \frac{\partial \Phi}{\partial x} dy - \int_{x_i}^{x_{(i+1)\bmod 4}} \frac{\partial \Phi}{\partial y} dx \right) = \iint q dV \tag{2.42}$$

We can discretize this equation by assuming a spline approximation to Φ, namely:

$$\Phi = \Phi_s + a_s \Delta x_s + b_s \Delta y_s + \alpha_s \Delta x_s^{\ 2} + \beta_s \Delta y_s^{\ 2} + \gamma_s \Delta x_s \Delta y_s \tag{2.43}$$

where Φ_s is the value of Φ at the center of the current element, and Δx is defined as the difference in x coordinate between the point being interpolated and the center of the current element:

$$\Delta x_s^i = x_i - x_s \tag{2.44}$$

The coefficients a_s, b_s, α_s, β_s, and γ_s are the spline coefficients and can be calculated by solving the equation Ax=b, where:

$$A = \begin{bmatrix} \Delta x_s^f & \Delta y_s^f & \Delta x_s^{f^2} & \Delta y_s^{f^2} & \Delta x_s^f \Delta y_s^f \\ \Delta x_s^1 & \Delta y_s^1 & \Delta x_s^{1^2} & \Delta y_s^{1^2} & \Delta x_s^1 \Delta y_s^1 \\ \Delta x_s^2 & \Delta y_s^2 & \Delta x_s^{2^2} & \Delta y_s^{2^2} & \Delta x_s^2 \Delta y_s^2 \\ \Delta x_s^3 & \Delta y_s^3 & \Delta x_s^{3^2} & \Delta y_s^{3^2} & \Delta x_s^3 \Delta y_s^3 \\ \Delta x_s^4 & \Delta y_s^4 & \Delta x_s^{4^2} & \Delta y_s^{4^2} & \Delta x_s^4 \Delta y_s^4 \end{bmatrix} \qquad (2.45)$$

$$x = \begin{bmatrix} a_s & b_s & \alpha_s & \beta_s & \gamma_s \end{bmatrix}^T$$

$$b = \begin{bmatrix} \Delta\Phi_s^f & \Delta\Phi_s^1 & \Delta\Phi_s^2 & \Delta\Phi_s^3 & \Delta\Phi_s^4 \end{bmatrix}^T$$

Here the superscript *f* refers to the element directly across the facet from element s and superscripts 1-4 indicate the other elements in the spline's stencil, as previously introduced in Section 2.2.2 and Figure 2-8. Now, from equation *2.43*, we first analytically find the components of the gradient of the potential, Φ:

$$\frac{\partial\Phi}{\partial x} = [a_s + 2\alpha_s(x - x_s) + \gamma_s(y - y_s)]$$
$$\frac{\partial\Phi}{\partial y} = [b_s + 2\beta_s(y - y_s) + \gamma_s(x - x_s)] \qquad (2.46)$$

Realizing that (x_s,y_s) is just the center of the current element and is a known constant, we need only find a relation between x and y in order to substitute into equation *2.42* and make the integrals tractable. In this case, with our mesh facets being all straight lines, the relation is just the simple equations of a line:

$$y = cx + d \qquad x = \frac{y - d}{c} \qquad (2.47)$$

where c and d depend on which facet we are examining. For the rest of these calculations we will assume we are dealing with the 1-2 facet (i.e. the facet whose starting point is (x_1,y_1) and whose ending point is (x_2,y_2)). The reader can easily extend the ideas to the other facets by a simple substitution of indices. Thus, for the 1-2 facet, c and d become:

$$c = \frac{y_2 - y_1}{x_2 - x_1} \qquad d = y_1 - cx_1 \qquad (2.48)$$

So substituting these expressions into equations *2.46* and *2.47*, and writing out the complete integrals in equation *2.42*, we see that:

$$\sum_{i=0}^{3}\left(\begin{array}{c}\displaystyle\int_{y_i}^{y_{(i+1)\bmod 4}}\left[a_s+2\alpha_s\left(\frac{y-d}{c}-x_s\right)+\gamma_s\left(y-y_s\right)\right]dy- \\ \displaystyle\int_{x_i}^{x_{(i+1)\bmod 4}}\left[b_s+2\beta_s\left(cx+d-y_s\right)+\gamma_s\left(x-x_s\right)\right]dx\end{array}\right)=\iint q\,dV \qquad (2.49)$$

Next we analytically carry out the integration in the above equation. For face 1-2 for example, the integrals on the left side of the equation above would, after some rearranging of terms, become:

$$\left(\int_{y_1}^{y_2}\frac{\partial\Phi}{\partial x}dy-\int_{x_1}^{x_2}\frac{\partial\Phi}{\partial y}dx\right)=\left(\begin{array}{l}\left(y_2-y_1\right)\left[a_s-2\alpha_s\left(\frac{d}{c}+x_s\right)-\gamma_s y_s\right]+\left(x_2^2-y_1^2\right)\left[\beta_s c+\frac{\gamma_s}{2}\right]+ \\ \left(y_2^2-y_1^2\right)\left[\frac{\alpha_s}{c}+\frac{\gamma_s}{2}\right]-\left(x_2-x_1\right)\left[b_s+2\beta_s\left(d-y_s\right)-\gamma_s x_s\right]\end{array}\right) \qquad (2.50)$$

Now, to carry out the successive over-relaxation, we need to find an expression for each Φ_s in terms of the other Φ values. Believe it or not, such an expression is embedded in equation *2.42*, and using equation *2.50* and its implied sister equations for the other facets, we can extract this relationship. Indeed, the spline coefficients in equation *2.50* depend on the Φ values via the equation $A^{-1}b = x$. Thus:

$$\begin{array}{lll} a_s = A^{-1}_{(1,:)}\cdot b & b_s = A^{-1}_{(2,:)}\cdot b & \\ & & \gamma_s = A^{-1}_{(5,:)}\cdot b \\ \alpha_s = A^{-1}_{(3,:)}\cdot b & \beta_s = A^{-1}_{(4,:)}\cdot b & \end{array} \qquad (2.51)$$

where the syntax $A^{-1}_{(i,:)}$ denotes the i^{th} row of the inverse of A. Since b_i is just Φ_i-Φ_s, we can therefore solve explicitly for Φ_s in terms of geometrical parameters and the other nodal Φ values. This allows us to create a matrix equation, solving for the value of Φ_s in terms of its neighbors.

2.7.1.3 Using Splines for an Axisymmetric Geometry

We also adapted our two-dimensional Cartesian spline approximation method for use with an axisymmetric geometry. Without loss of generality, we assume that the axis of symmetry lies along the horizontal or Z-axis. The R-axis then is vertical and represents the radial position.

Only relatively few changes need to be made to the above approximation scheme in order to model the axisymmetric geometry. The equation we are now attempting to solve becomes:

$$\oint E \cdot d\vec{A} = \oint (-\nabla \Phi) \cdot dA = \iiint q dV \qquad (2.52)$$

In cylindrical polar coordinates, assuming there is no azimuthal variation in potential, the gradient operator is practically the same as its Cartesian counterpart:

$$\nabla \Phi = \frac{\partial \Phi}{\partial z}\hat{z} + \frac{\partial \Phi}{\partial r}\hat{r} \qquad (2.53)$$

The major difference lies in the vector dA which has now become:

$$d\vec{A} = 2\pi r[(-dr)\hat{z} + (dz)\hat{r}] \qquad (2.54)$$

This area vector makes the integrals we need to perform analytically a little more difficult, but otherwise is unproblematic. Using a substitution similar to that used before for x and y, namely:

$$r = cx + d \qquad z = \frac{r-d}{c}$$

$$c = \frac{r_2 - r_1}{z_2 - z_1} \qquad d = r_1 - cz_1 \qquad (2.55)$$

we can analytically compute the surface integral given above. The result, as before, is an equation relating the charge distribution to the potentials through the spline coefficients:

$$2\pi\left\{\begin{array}{l}\left[a_s-2\alpha_s\left(\frac{d}{c}+z_s\right)-\gamma_s r_s\right]\left[\frac{r_2^2-r_1^2}{2}\right]+\left[\frac{2\alpha_s}{c}+\gamma_s\right]\left[\frac{r_2^3-r_1^3}{3}\right]\\ -\left[b_s c+\beta_s(4cd-2cr_s)+\gamma_s(d-cz_s)\right]\left[\frac{z_2^2-z_1^2}{2}\right]-\left[2\beta_s c^2+c\gamma_s\right]\left[\frac{z_2^3-z_1^3}{3}\right]\\ -\left[db_s+2d(d-r_s)\beta_s-dz_s\gamma_s\right]\left[z_2-z_1\right]\\ \end{array}\right\}=\iiint q\,dV \tag{2.56}$$

Using this equation, we can once more employ our matrix solution processes to obtain the final potential vector. During testing it was discovered that the spline coefficient matrix often became singular for certain types of geometries. As such, a patch to this approximation was developed that uses spline approximations where the relevant matrix is non-singular but reverts to a linear interpolation scheme when this matrix is singular. This patch is discussed in more detail in Section 3.2.2.

2.7.2 Solving the Resulting Matrix Equations

The above methods of approximation lead in general to a matrix equation which relates the value of the potential at one point to the value of the potential at neighboring points. Innumerable techniques exist for solving such matrix equations and whole theses could be, and have been, written just discussing and comparing these methods. For our purposes, two such methods were chosen for their convenience, simplicity, and applicability to the present simulation, and these are outlined below.

2.7.2.1 Successive Over-Relaxation Overview

Due to its simplicity, generality, and familiarity to our lab, successive over-relaxation (SOR) was chosen as the primary iterative solution method for the matrix equations resulting from approximating the Gauss's Law equation. By using one of the methods (splines, linear interpolation, etc.) described above, we first derive a matrix equation of the form AΦ=Q, with Q being a vector of integrated charges, Φ being the unknown vector of potentials, and A being a matrix of geometrical properties relating the two. This matrix equation can then be iteratively solved through the application of SOR.

At each iteration of SOR, a new estimate of the unknown Φ^{t+1} is obtained through the extrapolated Gauss-Seidel formula:

$$\Phi^{t+1} = \omega D^{-1}\left(Q - L\Phi^{(t+1)} - R\Phi^{t}\right) + (1-\omega)\Phi^{t} \qquad (2.57)$$

where D, L, and R are respectively the diagonal, left-triangular, and right-triangular portions of A and ω is known as the over-relaxation factor [25].

The parameter ω must be chosen in the interval (0,2) in order for SOR to converge. Choosing ω equal to 1 reduces the SOR iteration to a simple Gauss-Seidel iteration while a selection of ω less than 1 results in the method known as under-relaxation which is often used in multi-grid techniques [53]. Both such methods have generally slower convergence rates than SOR. In practice then, ω for SOR must be chosen in the interval (1,2). Unfortunately, no explicit theory exists as to how ω should be chosen for all general cases such that the convergence rate is a maximum. Worse yet, the benefits in the convergence of SOR are often only significantly observed if ω is chosen in a small interval around the optimum value. However, years of experience and experimentation has led to the common heuristic of choosing $\omega = 2\text{-}O(h)$, where h is the average grid-spacing [25]. From this initial guess for ω, one can then further refine and experiment to determine an approximately optimum value for a particular grid.

2.7.2.2 Direct Solution Technique Using MUMPS

In addition to testing the iterative technique of successive over-relaxation, we decided to also test the solution performance of a common web-available linear algebra solver known as MUMPS (MUltifrontal Massively Parallel sparse direct Solver) [3][4][5]. This direct solver offers the benefits of being easily parallelizable, faster than the iterative technique for the current size of problems on which we are testing, pre-coded, and thoroughly tested and debugged.

MUMPS will solve the matrix equation **KΦ** = **Q** and return to us the solution vector **Φ**. In this case, the vector **Φ** contains the computed cellular potential values, the **Q** vector contains a cell's integrated charge as well as other forcing terms which arise from boundary conditions, and the **K** matrix can be calculated element by element using one of the approximation techniques described above. Indeed, equations *2.50* and *2.56* are still relevant. In this case, however, instead of solving these equations for Φ_s and iteratively approximating this value from its neighbors, we keep the entire system intact and directly solve for the **Φ** vector as a whole.

Each element produces a row in the **K** matrix. The MUMPS interface is such that we only need to create a vector describing the individual elemental contributions to each position in the **K** matrix, and the library will perform the assembly for us. For a typical non-boundary cell, the elemental stencil is found using the techniques described in the previous sections. For a cell containing a boundary, we again assume the potential is known. We analytically integrate this potential over the boundary face and move this term into the right-hand side **Q** vector. Alternatively, if say element i is on the boundary, we can simply set the ith row of the **K** matrix to 1 in the ith column and 0's elsewhere. In this case $\mathbf{Q}_i$ is then just set to the analytic potential at the center of the boundary cell.

2.7.3 Approximating the Electric Field from a Given Electric Potential

The electric potential is of course not very useful for computing forces on charged particles. What we really require is the magnitude and direction of the electric field. Since the electric field is simply the negative of the gradient of the potential, we have the task of computing the x and y derivatives of the electric potential.

Any time derivatives need to be calculated from digital data, it is probable that errors will be incurred, and this case is no exception. Our first attempt at approximating these derivatives was to again use a spline approximation. However, when attempting to calculate the field at the center of a cell we do not always have five readily available neighbors. For a non-boundary non-adapted cell, we generally only have four neighbors. So we used a slightly less accurate spline representation:

$$\Phi = \Phi_s + a_s \Delta x_s + b_s \Delta y_s + \alpha_s \Delta x_s^2 + \beta_s \Delta y_s^2 \qquad (2.58)$$

Once we have calculated an approximate Φ in this manner, we then analytically differentiate it:

$$\frac{\partial \Phi}{\partial x} = \left[a_s + 2\alpha_s(x - x_s)\right]$$
$$\frac{\partial \Phi}{\partial y} = \left[b_s + 2\beta_s(y - y_s)\right] \tag{2.59}$$

Solving a system similar to that represented by equation *2.45*, except that the final row and column will be missing, we can calculate the spline coefficients and therefore the electric field.

As appealing as this five-point stencil may be, initial testing showed that it did not in fact produce accurate results. The reason for this is soon realized upon careful consideration of Figure 2-16 depicting the five points involved in the five-point stencil. One realizes that as the mesh is further and further refined, eventually all cells will numerically begin to look like regular cells. Indeed, in Figure 2-16, the mesh is obviously not uniform in both directions, but if we examined the Δx^2 column of the spline A matrix, we would see that numerically these values would all appear to be the same. Such a situation leads to an ill-conditioned A matrix making it impossible to accurately calculate the spline parameters.

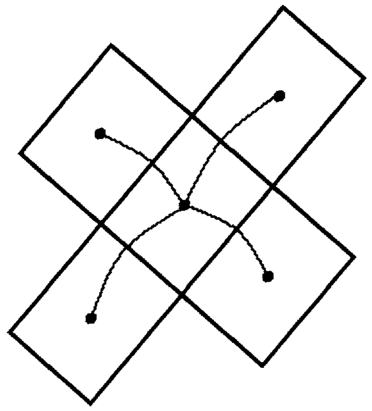


Figure 2-16: A five-point stencil for spline approximation of the electric field.

Due to this difficulty, we could not hope to use the five-point quasi-second-order spline approximation on any mesh, be it uniform or non-uniform, for with enough adaptation, any mesh could exhibit these ill-conditioned spline matrices. Therefore we were forced to reduce the order of the electric field approximations. In place of the five-point stencil, we are able to use only a three-point stencil meaning our approximation is now:

$$\Phi = \Phi_s + a_s \Delta x_s + b_s \Delta y_s \tag{2.60}$$

The derivatives are thus:

$$\begin{aligned} \frac{\partial \Phi}{\partial x} &= [a_s] \\ \frac{\partial \Phi}{\partial y} &= [b_s] \end{aligned} \tag{2.61}$$

With a three-point stencil, there is no longer an inherent symmetry and we must take care in selecting which three points to use in our approximation. Using, for instance, the north and south neighbors of a particular point could easily result in three collinear points and once again an ill-conditioned spline. Thus, we must use the four combinations of points being north and east, north and west, south and east, and south and west, as shown in Figure 2-17.

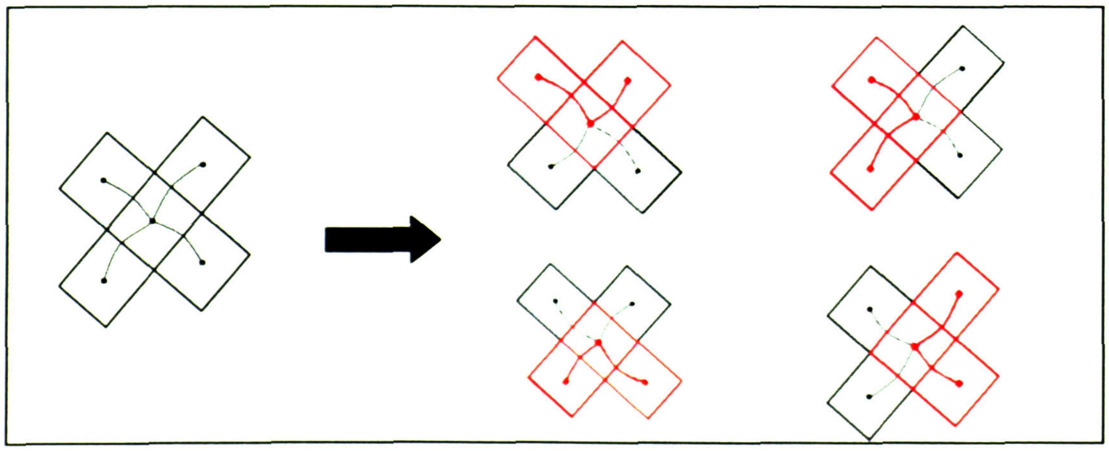


Figure 2-17: The five-point stencil is turned into four three-point stencils in order to avoid ill-conditioned approximation matrices.

These four electric field results are averaged to produce a final result. As Figure 2-17 clearly shows, the same information is being used in the calculation of this three-point approximation as would have been used in the five-point. It is simply being conglomerated in a different way. Unfortunately, this does result in a method which is only first-order accuracy, but this reduced accuracy became necessary in order to deal with the exceptionally general and difficult RRC, unstructured, irregular meshes. For simpler meshes, the higher accuracy spline method could of course be employed.

2.8 Models for Collision Events

Two main factors prompted us to develop new methods of collisions for this simulation rather than rely on the technique previously used in MIT's Hall thruster simulations [73][18], Monte Carlo collisions. First of all, Monte Carlo methods have long been known to introduce sometimes significant numerical noise into a simulation which can usually only be attenuated by a large increase in the number of computed particles and thus a large increase in computational effort. Other inaccuracies and difficulties with Monte Carlo methods have also been reported in past studies [5][6], leading us to propose a hopefully more stable and accurate methodology. The second reason that new collision models were developed was that the typical Monte-Carlo collision model could not be applied in our simulation due to the variable statistical weighting of particles which was used. Imagine for a moment that two particles are located in a grid cell and half of the particles in the cell are supposed to undergo collisions. In the typical MCC model, one of the two particles will probably be chosen, and since the particles all have equal statistical weights, it does not matter which particle. In our model, however, one of the particles could have a hundred times heavier weight than the other. Now if we randomly choose which particle collides, highly variable results could ensue. Given this toy example and the fact that our particles have individual, real-number weights assigned to them, one may easily see that in our simulation's context, colliding entire particles a fraction of the time is no longer the most applicable collision method. Rather, it makes more sense to collide a fraction of each particle the entire time. In the example above then, the logical course of action is to suppose that half the weight of both particles has collided, reduce their weights accordingly, and create new particles of the appropriate weight if necessary. This is the logical basis for the collisional methods developed which will be discussed in the following sections.

2.8.1 Intra-Species Elastic Collisions

We chose to handle the particular case of same-species elastic collisions in a separate and unique manner. This novel collision technique was implemented in our simulation for neutral-neutral elastic collisions, but could, of course, be extended to other types of particles if necessary. The technique is explained in the paragraphs which follow and, for clarity, is outlined in the flowchart shown in Figure 2-18.

The cross-section for neutral-neutral scattering, Q_{nn}, has been shown to be on the order of 10^{-15} cm^2 and generally varies very gradually with neutral temperature. As a first order approximation, we will take the cross-section to be a constant. The frequency of a collision event for an average particle is then just:

$$\nu_{nn} = n_n \langle v_n \rangle Q_{nn} \tag{2.62}$$

where $<v_n>$ is the average velocity in the given cell and n_n is the local neutral density.

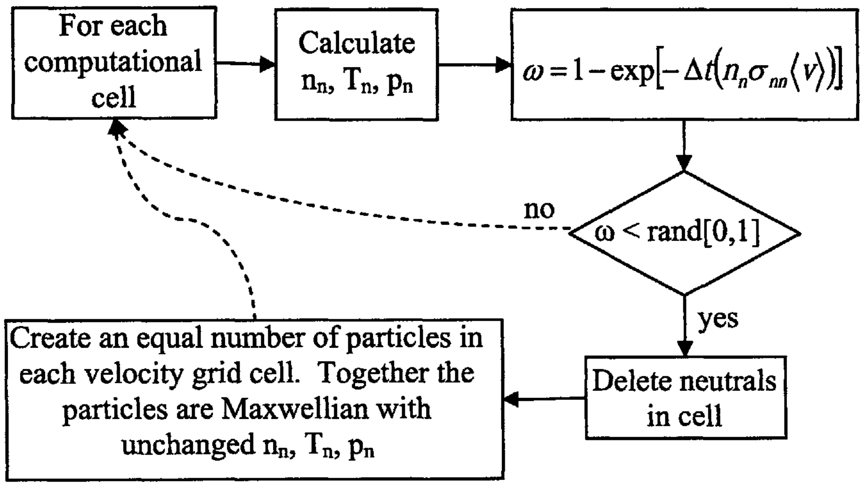

For each computational cell
Calculate n_n, T_n, p_n
$\omega = 1 - \exp[-\Delta t(n_n \sigma_{nn} \langle v \rangle)]$
$\omega <$ rand[0,1]
no
yes
Delete neutrals in cell
Create an equal number of particles in each velocity grid cell. Together the particles are Maxwellian with unchanged n_n, T_n, p_n

Figure 2-18: A flowchart showing the steps in the intra-species elastic collision model.

In the regime in which our simulation time step, τ, is small and the probability of more than one collision event per particle per time step is small, we can calculate the probability that collisions occurs for an average particle during a particular time step via:

$$p = 1 - \exp(-\tau \nu_{nn}) \qquad (2.63)$$

By comparing this p to a random number $r \in [0,1]$, we determine whether or not this particle has undergone collisions during the time step.

If collision events have occurred, a slightly unorthodox method of calculating the effects of those collisions is undertaken. It is assumed that on the small scale of the particular grid cell, intra-species collisions have acted to fully Maxwellianize the temperature distribution of that species. The calculation of individual collision events is not significant on this scale. Instead, a more holistic, global perspective of the effects of all the collisions in a cell has been adopted.

To perform the Maxwellianization, we first remove all of the current particles in the cell. We then create new particles which represent a Maxwellian distribution of velocity, according to the same principles used when initial particle distributions were created. That is, velocity space is more or less uniformly covered, and several particles may have much greater statistical weights than others whose velocities are near the extreme tails of the distribution. The distribution retains the same overall physical density exhibited by the particles in the original cell as well as the same temperature and drift velocity. The actual physical location within the grid cell of each new particle is randomly selected with the assumption that the mesh is refined sufficiently well so that any particular cell can basically be treated as a single point and spatial variations within each cell can be practically overlooked.

This method, it turns out, does not accurately calculate the theoretical rate of a distribution's relaxation to Maxwellian. In the case of Hall thrusters, however, the exact velocity distribution of the neutrals is not absolutely integral to computing accurate results. As long as the plume has roughly the correct spatial distribution so as to not disrupt the physics of the ionization region, we need not be too concerned. As such, the slightly less accurate method described here was used for the Hall thruster applications rather than a more time-consuming and difficult to implement BGK approximation which might have otherwise been selected had the neutral distribution been of paramount importance. In the future work section, Section 7.3, we recommend upgrading this collision methodology to possibly improve our simulation's agreement with theoretical results.

The above method does, however, have the important characteristic of not increasing the number of particles in the simulation on average. The number of particles used to create the reseeded Maxwellian distribution can be chosen by the user's selection of the variable **n_half_v_n** in the file *input.txt.* Since a constant number of particles are used to reseed the Maxwellian distribution within a cell while a constant number is also removed, these two values should eventually reach some balance, and the number of particles will reach an equilibrium. This is demonstrated for a particular case in Figure 2-19 below. This feature of the intraspecies collision model beautifully complements the Vlasov redistribution step, its goals, and its effects.

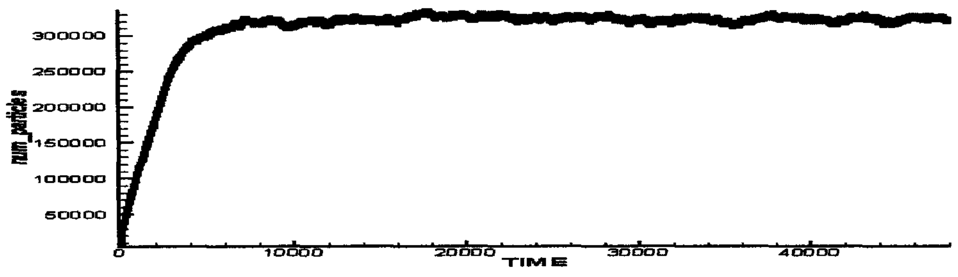
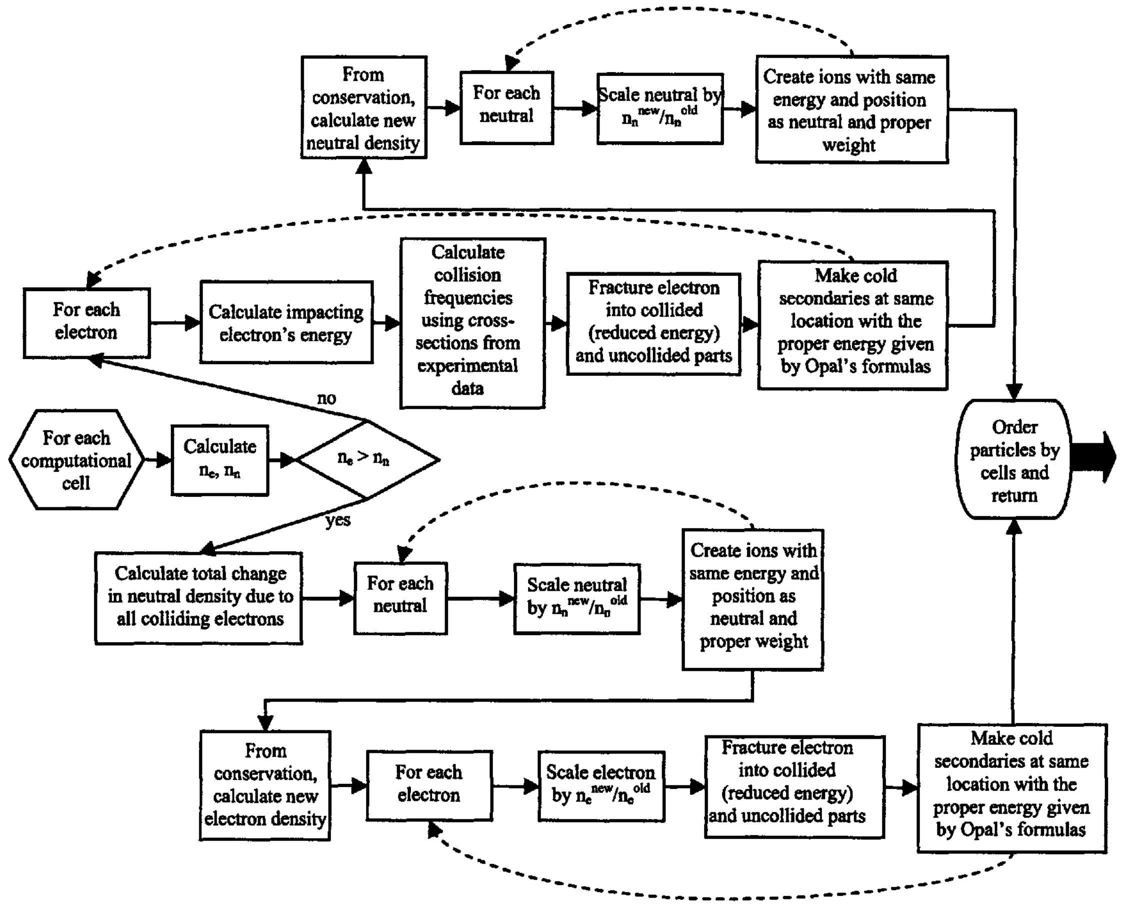


Figure 2-19: The number of computational particles versus time for a simulation with neutral-neutral elastic collisions.

2.8.2 Neutral-Electron Inelastic Collisions

As mentioned above, due to the statistical weighting of particles, our method of performing inelastic collisions, including ionizations and excitations, had to be developed more or less from scratch. The rather complicated flowchart in Figure 2-20 displays the main parts of this algorithm. It is actually not as complicated as it first appears, so we will attempt to step through it here and give a general overview of how the algorithm operates before delving into the specific equations and data necessary for its implementation.



From conservation, calculate new neutral density
For each neutral
Scale neutral by n_n^{new}/n_n^{old}
Create ions with same energy and position as neutral and proper weight
For each electron
Calculate impacting electron's energy
Calculate collision frequencies using cross-sections from experimental data
Fracture electron into collided (reduced energy) and uncollided parts
Make cold secondaries at same location with the proper energy given by Opal's formulas
no
For each computational cell
Calculate n_e, n_n
$n_e > n_n$
Order particles by cells and return
yes
Calculate total change in neutral density due to all colliding electrons
For each neutral
Scale neutral by n_n^{new}/n_n^{old}
Create ions with same energy and position as neutral and proper weight
From conservation, calculate new electron density
For each electron
Scale electron by n_e^{new}/n_e^{old}
Fracture electron into collided (reduced energy) and uncollided parts
Make cold secondaries at same location with the proper energy given by Opal's formulas

Figure 2-20: A flowchart depicting the basic steps of the complicated neutral-electron inelastic collision method.

The neutral-electron collisions are calculated for each cell individually. This allows the method to be easily parallelized as discussed in Section 2.13.4. Depending on the value of the electron density in a given cell relative to the neutral density, one of the two branches of the algorithm is then entered. The reason for these branches stems from the equations outlined below in Section 2.8.2.2.

Assuming first the simpler case that the neutral density is greater than the electron density, the flowchart continues along the upper branch. What happens is that we now loop over the electrons in the given grid cell. For each of those electrons, we calculate its kinetic energy and determine the cross-sections and collision frequencies that would be associated with such a particle colliding with a background density of slow-moving neutral particles. Under this infinite mass ratio assumption, the amount of each type of modeled collision is calculated from the relative collision frequencies as described in Section 2.8.2.3. The single computational electron particle is then fractured into a number of statistically smaller computational electron particles, the sum of whose weights is equal to that of the initial particle. This splitting was illustrated graphically in the previous Figure 1-3 in the introductory section. As the splitting occurs, cold secondary particles are also created with energy stolen from the impacting electron. This is discussed more in Section 2.8.2.1. Once the electrons have collided, we can, through conservation calculate how much of the neutral density must also have collided. This then allows us to scale each individual neutral (or ion if we include electron-ion collisions) to account for the portion of that particle that has not yet collided. The final step is the creation of the newly-formed ions whose weights are evident and are related to the total amount of electrons that underwent a certain type of ionization collision. The ions' initial energies are generally based on the bulk drift velocity of the originating particles. That is, a singly-charged or doubly-charged ion that was created directly from a neutral will begin its life with the neutral bulk velocity while a doubly-charged ion created from a singly-charged ion will initially be given the bulk singly-charged ion velocity. This completes the overview of the upper branch of the flow chart.

The lower branch of the flow chart differs slightly from the upper and is used when the electron density is greater than the neutral density, indicative of a high ionization fraction. In this case, the neutrals must be treated as the limiting species as discussed below in Section 2.8.2.2. Thus, we first calculate the change in each of the neutral's densities given all of the impacting electrons. With this information, we can then obtain, through conservation, the new electron density as well. The rest of the flow chart proceeds basically as before. The neutral's are scaled, ions are created with velocities as described above, and finally electrons are split based on the newly-calculated final electron density and the relative frequency of each collision type given the particular electron's energy.

In both cases, upper and lower branch, the final step of the collision algorithm is one which is common to many other modules in the code, as well. That is to re-order the main particle array such that all of the particles for a given cell appear in a contiguous block of the array. This not only has the benefit of using machine cache most efficiently since nearby particles are very often accessed sequentially, but is necessary for the parallelization algorithm as well.

2.8.2.1 Creation of Cold Secondary Electrons Through Collisions

As the splitting process of electrons occurs, new secondary electrons are also being created. Following the lead of the previous generation of MIT simulation, these particles are given an initial kinetic energy according to Opal's distribution [60], which is given by:

$$P(E_s) \approx \frac{1}{1+\left(\frac{E_s}{E_j}\right)^2} \tag{2.64}$$

where E_S is the energy of the secondary and E_j is an empirical parameter roughly equal to 8.7eV in the case of Xenon. The secondary's energy is subtracted along with the collision energy, be it ionization or excitation energy, from the impacting electron. Thus:

$$E_p^f = E_p^0 - E_s - E_I \tag{2.65}$$

where E_p is the primary's energy, with the f superscript denoting the final value and the 0 superscript denoting the initial. E_I is the ionization energy.

In the code, then, we first choose a random energy for the secondary electron that is between zero and the energy remaining to the primary after subtraction of the ionization energy. This value is subjected to the rejection method, meaning another random value is generated between 0 and 1 and if this value is less than $P(E_S)$ for the chosen E_S, then the energy is accepted. Otherwise, a different secondary energy is again randomly selected and checked.

2.8.2.2 Semi-Analytical Equations For Electron Impact Collisions

Batishchev developed the fundamental basis for our semi-analytical collision model [11]. The analysis is described and expanded here followed by a brief discussion of its possible limitations.

The analysis begins by assuming that there exist K electrons with individual statistical weights defined as $w_k(t),\ k=1...K$. These electrons are undergoing, possibly inelastic, collisions with a background density of neutrals, $N(t)$. The system of non-linear ordinary differential equations which can be used to describe the change in the number densities of these species can be given by:

$$\left\{\begin{array}{l} \dfrac{dw_k}{dt} = -r_k N w_k, \quad k = 1...K \\ \dfrac{dN}{dt} = -N\sum_{k=1}^{K} r_k w_k \end{array}\right\} \qquad (2.66)$$

where $r_k = \sigma(v_k)v_k$ and can be considered constant in time for this analysis. Note that the real difficulty here is that the energy of an individual electron decides how much of a given type of collision it will undergo, and so the electrons cannot be simply conglomerated. Also, both densities are changing at the same time in a fundamentally non-linear fashion making analytical solution of this system improbable.

To make progress on these two equations, we can make two assumptions about the relative densities which will lead to two separate solutions. If we assume that $\sum_{k=1}^{K} w_k$ is large compared to N, then we can assume that the right hand side of the second equation above remains frozen. This allows us to first solve for the neutral density with respect to time and find that:

$$N(t) = N_0 \exp\left[-t\sum_{k=1}^{K} r_k w_k\right] \qquad (2.67)$$

From this formula combined with the obvious formula of conservation:

$$\sum_{k=1}^{K} w_k(\tau) = N(\tau) - N(0) + \sum_{k=1}^{K} w_k(0) \qquad (2.68)$$

we are then able to calculate the final sum of the electron weights. We then assume that the fraction of each electron that collided is in proportion to the collisional cross-sections at its respective energy, enabling us to find the new electron weights.

If we instead assumed that N is large compared to the sum of the electron weights, then we can hold the right hand side of the first equation in the set *2.66* above constant and solve this equation to find:

$$w_k(t) = w_k(0)\exp[-Nr_k t] \tag{2.69}$$

Again, the conservation formula, Equation *2.68* above, enables us to discern the final neutral density, again assuming that each neutral has collided an equal amount relative to its initial weight.

So, we have developed two solutions to the non-linear system *2.66* by assuming that either $n_n >> n_e$ or that $n_e >> n_n$. In effect, we are assuming that the number of the more numerous species does not change appreciably as collisions occur. This is, of course, a good assumption in the given limits. In the particular case of the NVHT, for example, the plasma is so tenuous that we even decide to make the assumption that the background neutral density is constant and does not change throughout the simulation, not just over the course of a collision timestep.

Unfortunately, not all applications fall into one of these two regimes. The P5 thruster, for example, exhibits a nearly equal electron and neutral density in the channel during the peak of its ionization. The assumptions made in the collision model outlined above, then, become untenable. One partial solution to this problem that we implemented for just such a situation was the following. If the densities of the relevant species in a given computational cell were found to be within one order of magnitude of one another, a slightly different alternate collisional method was used. This method used equation *2.95* or, more graphically, the upper branch of the flowchart in Figure 2-20. However, instead of performing the collisions in one fell-swoop, the effective collision time step was reduced by some factor, usually 100. By allowing only a small amount of time to pass, very few collisions occurred during each of these effective time steps. This made more viable the assumption that during each step the neutral density remained constant. Basically, this technique amounted to following a continuous function by moving in the direction of its derivative at discrete points along it. If the step is small enough, the final answer will hopefully remain close to the actual function. Clearly, this is not the most rigorous technique, but it was a useful temporary workaround for the problematic situation described above while simulating the P5 thruster. In the case of the NVHT, fortunately, this time-consuming process was not necessary nor was the error from this very ad hoc method incurred.

Martinez-Sanchez suggested another, more rigorous method to handle this difficulty [56] which involves expanding the exponential decay of the neutrals to second order around the initial density. This method then allows the evolution of the electron densities with respect to the quadratically decaying neutral background. Such a method would likely be more accurate and certainly more computationally efficient than the very ad hoc one described above. Unfortunately, due to time constraints, the technique first described was implemented and used to compute the test results found in Chapters 3, 5, and 6.

2.8.2.3 Calculating the Individual Rates

Another small ripple that was glossed over in the previous section was the matter of calculating the effect of several different categories of collision rates affecting the various species all at the same time. Indeed, the parameter r_k used above, can in truth be broken down into a sum of the rates for single-ionization, double-ionization, excitation, elastic scattering, and even anomalous diffusion. The literature study resulting in cross-sections and relative rates for each of these types of collisions is outlined in Section 2.8.3 below.

To resolve the results of two or more collision processes occurring at the same time we assume, without loss of generality, a simple example in which process A occurs with rate r_A and process B occurs with rate r_B. Under equation *2.69*, these processes will result in a final electron weight of:

$$w(\tau) = w(0)\exp[-(r_A + r_B)N\tau] \qquad (2.70)$$

What we seek to find is the amount of $1\text{-}w(\tau)$ that has undergone process A and the amount that has undergone process B. In the previous PIC/MCC simulation, this calculation was approximated by assuming that the term in the exponent was very small and so the function could be linearized and, for example, the simple fraction:

$$w_A = \frac{r_A}{r_A + r_B} \qquad (2.71)$$

was used to calculate the fraction of electrons that underwent collision-type A [73]. This is not exactly correct, however, and in some cases the term in the exponent may in fact become too large for this approximation to be really viable. Instead, we solve using a transformation to the logarithmic derivative:

$$\frac{d}{dt}[\ln w] = (-r_A - r_B)N \qquad (2.72)$$

From this point on, we assume that *w(0)* = 1, meaning that the value we obtain at the end of the calculation will need to be post-multiplied by *w(0)* to obtain the final result. So, from Equation *2.72*, we can assume an elapsed time of τ and write:

$$\ln w(\tau) - \ln w(0) = (-r_A - r_B)N \qquad (2.73)$$

Using the fact that *w(0)* = 1, we can simplify this to just:

$$\ln w(\tau) = (-r_A - r_B)N \qquad (2.74)$$

Now, by definition, the amount of the change in ln*w(τ)* that is attributable to process A can be extracted as just:

$$\ln w_A(\tau) = -r_A N \qquad (2.75)$$

Then taking the ratio of this partial amount to the whole, we have:

$$\frac{\ln w_A(\tau)}{\ln w(\tau)} = \frac{r_A}{r_A + r_B} \equiv \delta_A \qquad (2.76)$$

Finally, we can calculate the fraction of the weight that underwent process A via:

$$\begin{aligned} \ln w_A(\tau) &= \delta_A \ln w(\tau) \\ w_A(\tau) &= w(\tau)^{\delta_A} \end{aligned} \qquad (2.77)$$

This result will give a proper ratio between collision-types rather than an approximate one as the old PIC/MCC code had done. Figure 2-21 shows the percent error incurred if the simple, linear estimate is used instead of the above calculation as a function of the ratio δ_A. The error becomes more severe as the overall collisionality level increases, such that when one tenth of a particle is supposed to undergo collisions (or 10% of the total number of particles in the previous MCC case), the error from this approximation alone may be as much as 5% at each time step.

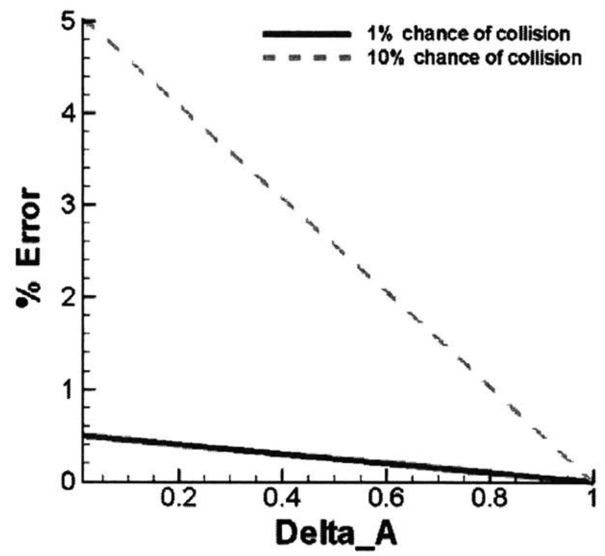


Figure 2-21: The percent error incurred if the linear approximation is used to calculate the amount of each type of collision. In regions of high collisionality, such as the ionization zone, this error can reach very significant levels.

2.8.3 Compiling Cross-Sections

Since the 1930's, there have been a great many studies which have attempted to measure the cross-sections of the noble gas atoms. Unfortunately, these studies have frequently disagreed with one another as to the absolute values of the cross-sections, often severely. As such, the field is still active and better experimental techniques are yet being invented in attempts to once and for all establish the exact values of these experimental cross-sections. The best we can currently do is to examine the available data critically and attempt to discern which sets of data we feel most confident selecting to be the basis for our collisional model.

Generally, the total cross-section for electron-impact events, Q_T is broken down into three parts, being q_s, q_e, and q_i, the elastic scattering cross-section, the total excitation cross-section, and the total ionization cross-section, respectively. Thus:

$$Q_T = q_s + q_e + q_i \tag{2.78}$$

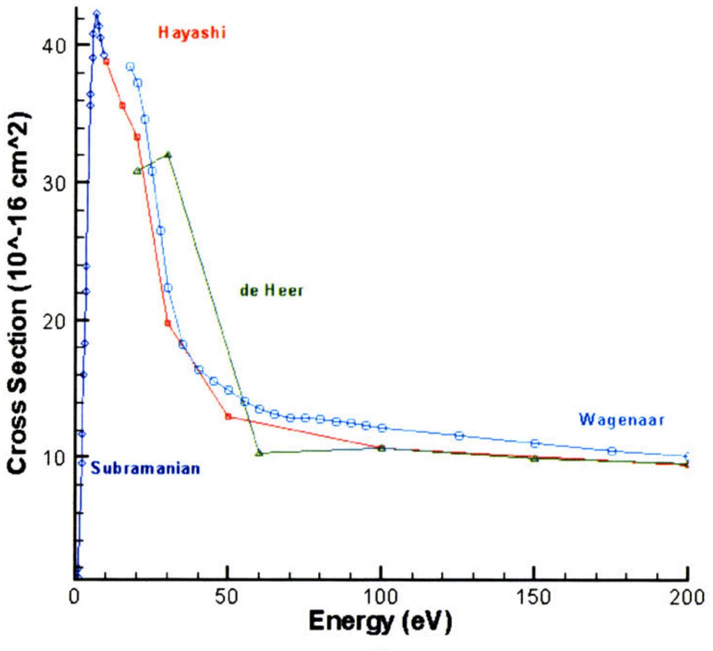
In addition the two latter cross-sections q_e and q_i are sometimes broken down further to enumerate the possible different types of excitation and ionization events. In the sections which follow, the total cross-section along with each of its three pieces is broken down individually and a critique of the data available for Xenon and Krypton is made, rationalizing our eventual selection of cross-section data for our simulation.

2.8.3.1 Total Cross-Section Data

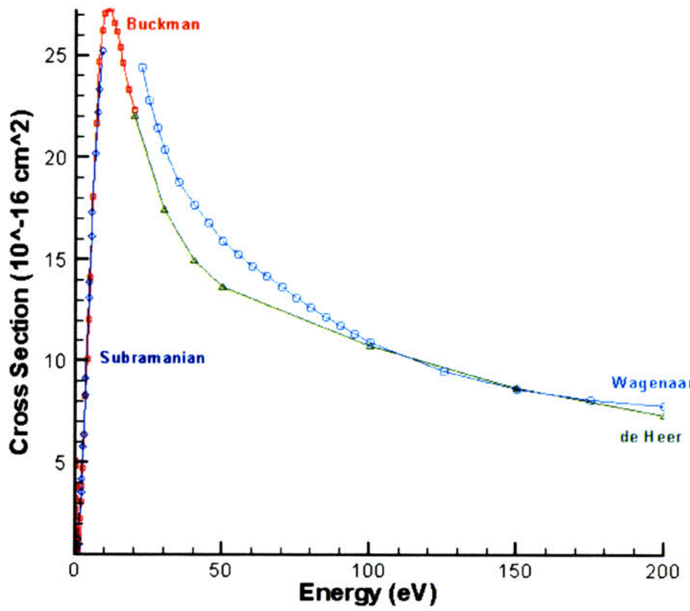
Representing the results of a general literature survey, the plot in Figure 2-22 illustrates a number of the seemingly more reliable data sets for the total event cross-sections for both (a) Xenon and (b) Krypton. Note that the scale of the cross-section is in square Angstroms, or 10^{-16} cm^2. Frequently cross-section data may also be reported in units of a_o^2, the Bohr radius which is equal to $5.29177*10^{-11}$m.

The total cross-section data are perhaps more reliable than, for instance, the various partial cross-section data. The experiments which measure this value are relatively simple to construct and rely on direct empirical observation rather than data transformations or obtuse use of theoretical relationships. In addition, this is the most frequently studied cross-section, and so the general consensus is much more stable than the lesser-studied partial cross-sections.

The recommendation for the usage of the Xenon data is to employ the results of Subramanian's [71] study at low energies and the data of Hayashi [38] at high energies. The accuracy of Wagenaar's study [78] may be slightly suspect since it was performed earlier with a less advanced technique. In addition, de Heer noted the poor quality of his Xenon total cross-section data, stating that "in this [low] energy region in our empirical cross-sections a large contribution comes from σ_{el} based on Williams and Crowe's (1975) data only . . . more experimental data for σ_{el} are needed" [40]. Again when examining the Krypton data, I would hold as suspect the data of Wagenaar [78] which were obtained earlier than that of Subramanian [71] and Buckman [19] and which appeared to be slightly too high for both gases.



(a)



(b)

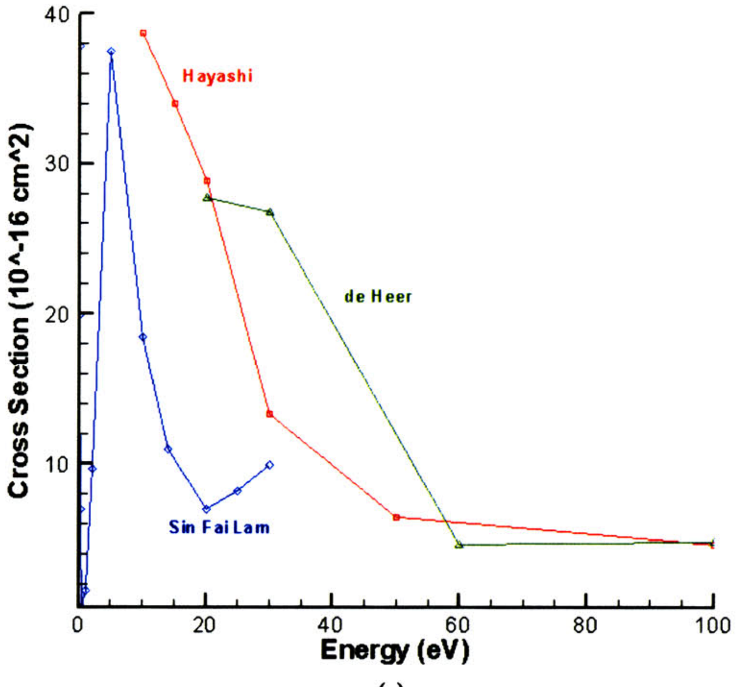
Figure 2-22: Total cross-section data for (a) Xenon and (b) Krypton.

2.8.3.2 Elastic Scattering Cross-Section Data

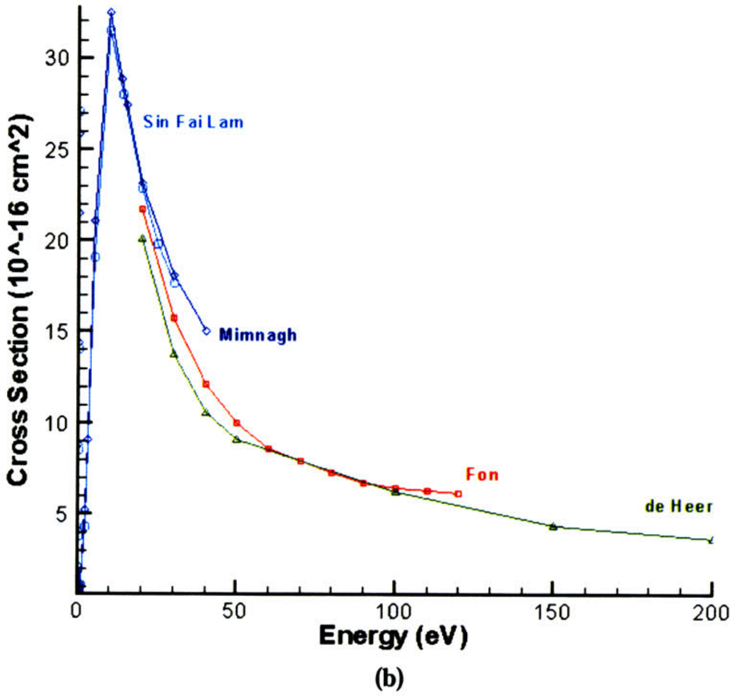
Elastic scattering represents those collisions which do not appreciably dissipate energy but serve only to alter momentum. Figure 2-23 plots the reported elastic cross-section for (a) Xenon and (b) Krypton. The data for Krypton is relatively similar in all cases while the data for Xenon varies widely. It is a general trend in cross-section data that the further down on the periodic table one examines, the less accurate will be the data available for that element. However, after examining the merits of the three included studies, my recommendation is as follows. While Hayashi's study was published last [38], his elastic cross-section data actually originates from studies conducted by de Heer and others in the early 1970's [40]. Sin Fai Lam's low energy study [68], however, used 1981 technology and had as its primary objective the calculation of the elastic cross-section whereas Hayashi merely cited old elastic scattering data in order to check his total cross-section results. Therefore, I would recommend ignoring Hayashi's elastic cross-section data, despite this study's other merits. In de Heer's study, again, the two low energy elastic scattering data points that disagree so strongly with Sin Fai Lam's results were taken from a single study conducted in 1975 by Williams and Crowe [83]. These two data points are therefore suspect and may be ignored. The remaining high energy data, however, represents a semi-empirical averaging of a number of more accurate studies, and may therefore be considered reliable. Thus, we appended the low-energy data of Sin Fai Lam to the high energy data reported by de Heer in order to cover the full range of Xenon elastic scattering. One can clearly see, however, that accurate cross-section measurements, especially in the interesting 10-25 eV range would be a great boon to Hall thruster research in general.

2.8.3.3 Ionization Cross-Section Data

Past experimental results of Hall thrusters have shown that small but relatively significant populations of multiply-charged ions exist in the ion plume. Therefore, it seems important that we obtain ionization cross-section data which have been broken down into the various possible ionization events. Rejoub presents a most comprehensive and modern study which fortunately measured each individual ionization cross-section separately [64]. These data for Xenon (a) and Krypton (b) are plotted in Figure 2-24.

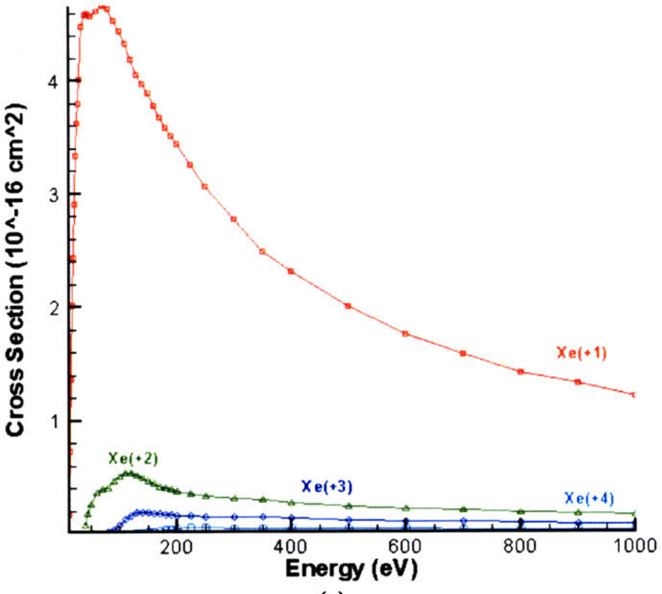


(a)

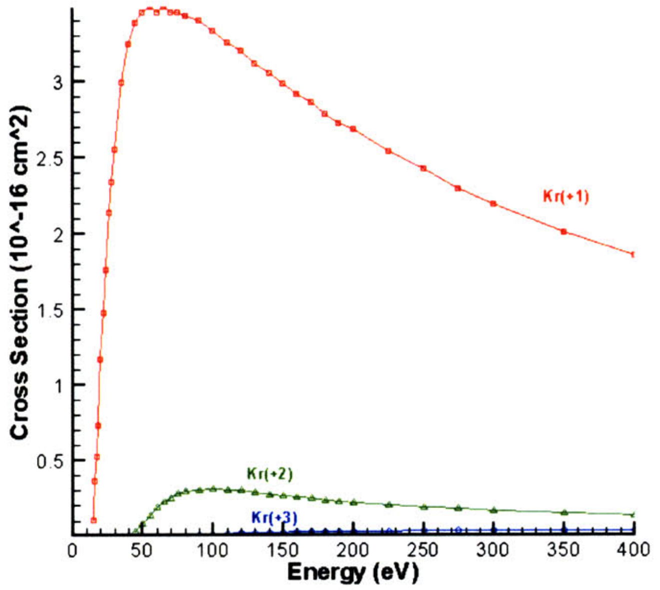


(b)

Figure 2-23: Elastic cross-section data for (a) Xenon and (b) Krypton.



(a)



(b)

Figure 2-24: Ionization cross-section data for (a) Xenon and (b) Krypton broken down by ionization level. All data was taken from Rejoub's 2001 study.

2.8.3.4 Excitation Cross-Section Data

Excitation of electrons within the atomic shell of Xenon and Krypton temporarily removes energy from the Hall thruster system which might otherwise have been used to ionize further atoms and increase the efficiency of the thruster. Therefore, the amount of energy lost to excitation is an important factor in calculating thruster efficiency. Unfortunately, the cross-section data for this event are very difficult to obtain and less comprehensively studied. There are so many different levels of excitation possible, that measuring them all individually has proven time-consuming and difficult. Trajmar presents one attempt at this feat, and his data show the cross-sections for many transitions within Krypton organized by the energy each transition absorbs [76].

Including all of the various paths and states of excitation would severely complicate a simulation such as ours. Therefore, what has been done in past numerical Hall thruster codes was to use a total excitation cross-section, for which data do exist (see for example[38] and [40]), but to assume that all excitation events are the simple 8.32eV, for Xenon, or 9.915eV, for Krypton, first level event. If one examines Trajmar's data, however, it is quickly seen that many other events are just as likely if not more so, including, using Krypton as an example, some events which absorb energies between 10 and 12eV. As such, thruster codes using this assumption may be underestimating the amount of energy being lost to the excitation of ions by factors of up to 50%. Such an inaccuracy is unpleasant but apparently unavoidable without greatly complicating our collisional model. Thus, in the present work, we have assumed that all excitation events remove the first-level energy from the colliding electron as previously described.

2.8.3.5 Inclusion of Cross-Sections Into Simulation

Having thus surveyed the cross-section literature and selected the experimental data to be used, it was next required to write functions which would interpolate these data in an appropriate manner and, given any input electron energy, would return the proper cross-sections. The numerical values of the actual experimental data points chosen have been included in Appendix B, mostly because I sympathized with anyone wading through numerous papers that only report graphs of their results when what is wanted is the actual, numerical data. The interpolation task involved the use of both Matlab and Fortran90, and two different, complete solutions to the problem were invented and implemented.

Solution 1 – Polynomial Fitting and Evaluation

The first method of implementation was chosen for its simplicity, light memory and processor time requirements, and its similarity to the methods used by previous Hall thruster modelers. Once the experimental data were extracted to text files, these files were parsed in Matlab. Matlab *polyfit* functions were then used to fit the data with a fourth order polynomial ratio of the form:

$$f(x) = \frac{p_1 x^4 + p_2 x^3 + p_3 x^2 + p_4 x + p_5}{x^4 + q_1 x^3 + q_2 x^2 + q_3 x + q_4} \tag{2.79}$$

The experimental data out to 200 eV were used to create these curve fits. Once the polynomials had been constructed, the p and q coefficients were hard-coded into functions that, given a particular energy, evaluate the proper polynomial, and return the value $f(x)$, the approximate cross-section. The shape of the rational polynomials is generally in good agreement with the experimental data, but because this interpolation has fewer degrees of freedom than the second method which is described below, it is generally not as accurate. Figure 2-25 depicts the interpolated cross-sections plotted together on a log-log scale.

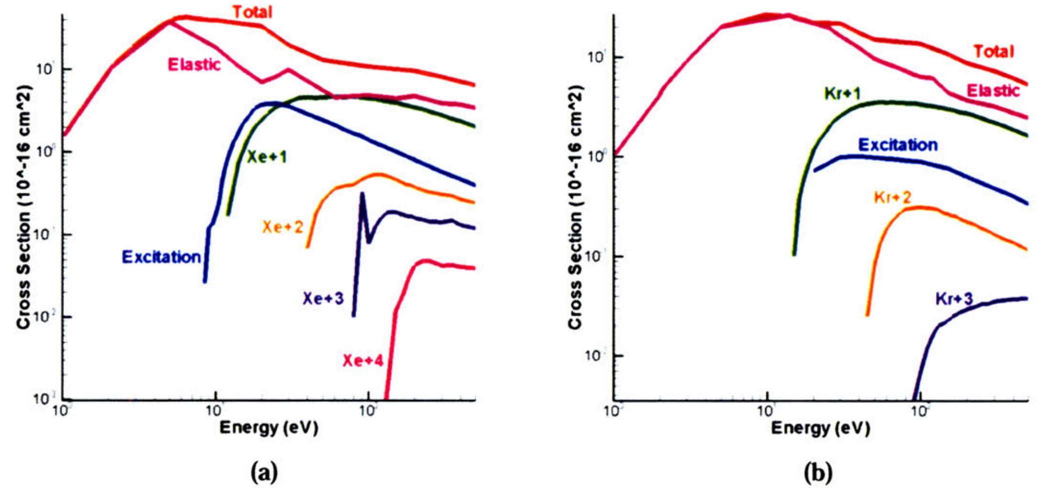


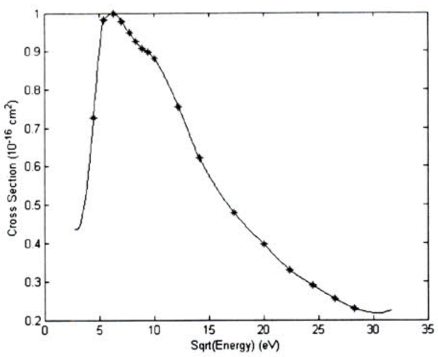
Figure 2-25: The polynomial fit cross-sections for (a) Xenon and (b) Krypton. The jaggedness and possible inaccuracy of these polynomial fits prompted the creation of the second, more intensive interpolation algorithm described below.

Solution 2 – Piecewise Interpolation

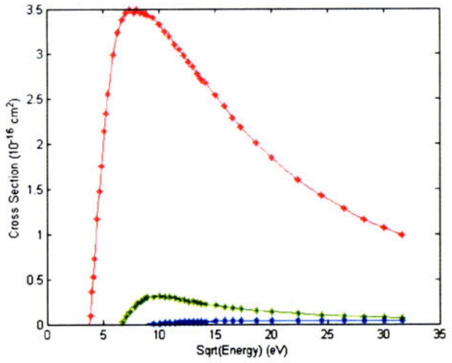
The second implemented method of obtaining cross-section data within the simulation involved more overhead work and more memory storage within the code, but is almost certainly the more accurate of the two methods. The experimental data were first transcribed into separate text files readable by Matlab, with data from different sources being compiled for each of the different types of collision into one coherent unit. These experimental data points are plotted in Figure 2-26 and symbolized by the *'s. Matlab's *interp1* function was then used to produce a piecewise 'cubic' fit to the data. The curve fits are also plotted in Figure 2-26. The values of this interpolation at certain predefined intervals were then recorded in new output files. The reason for this manipulation of the data was twofold. First of all, the experimental data were not evenly spaced. By interpolating them to evenly spaced, known abscissas before runtime, we enabled fast, efficient look up of the proper data during actual execution. Otherwise, the on-the-fly process of laboriously searching for the proper interval of data within which to interpolate would have significantly slowed down the code. The second benefit of pre-interpolation is that Matlab's piecewise cubic interpolation is very powerful and already prewritten for us. This meant that our initial interpolation would be accurately performed with minimal coding on our part.

When the code executes, it loads the proper interpolated output files into particular arrays. When a cross-section is requested for a given energy, the interval within the evenly-spaced arrays is first calculated. In the likely case that the energy does not fall exactly on an interpolated, precalculated point, simple linear interpolation is employed using the values of the arrays to either side of the given energy point.

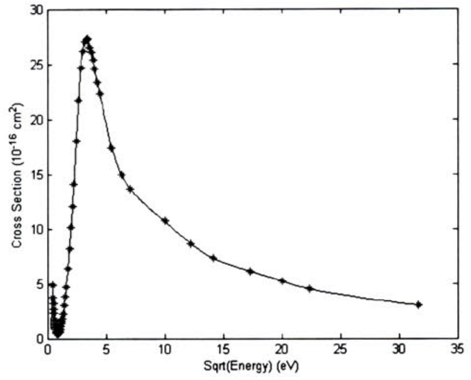
One likely will notice that the plots in Figure 2-26 do not include a plot of the elastic scattering cross-section. It was decided that since our sources significantly disagreed on those data, we would use a slightly roundabout method of calculating it. Since the total cross-section is broken down into three parts and since we know both the total cross-section and its other two parts, we take the remaining part, the elastic cross-section, to be just the difference between the total cross-section and the sum of the excitation and ionization cross-sections. This differencing has been accomplished ahead of time, such that calculation of the elastic cross-section is not handled any differently



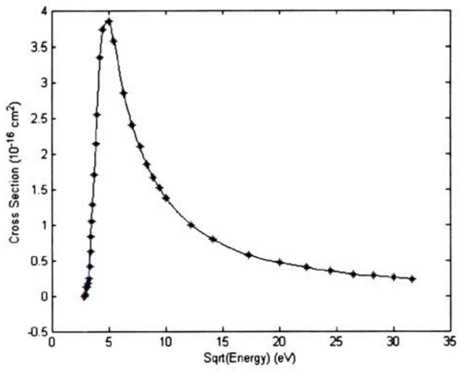
(a)



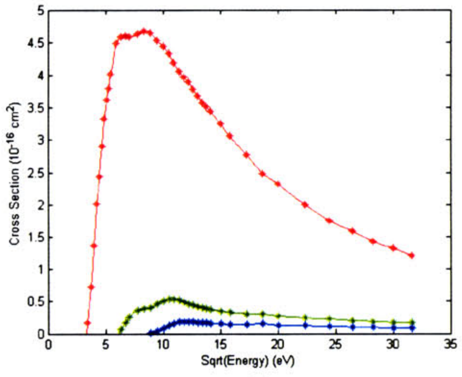
(b)



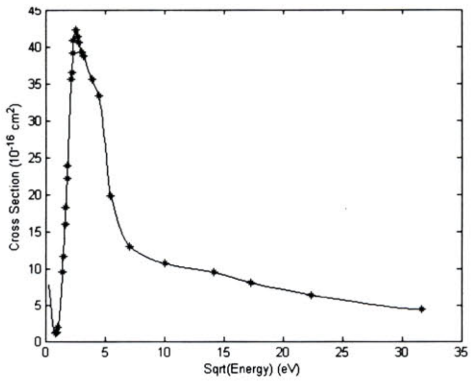
(c)



(d)



(e)



(f)

Figure 2-26: The final compiled cross-section data for (a)-(c) Krypton and (d)-(f) Xenon. The cross-sections for (a)(d) Excitation, (b)(e) Ionization, and (c)(f) the total for classical collisions are shown.

than the other types of collisions during actual execution of the code. Figure 2-27 gives an indication of how this calculated elastic cross-section compares to the experimental data that were compiled. For the most part, the cross-section that was used agrees largely with at least one of the author's data. The obvious exception, however, is that the calculated cross-section is significantly less than the experimental observations for Krypton at high energies. Given the relative disagreement between the experimental data, however, this difference was felt to be acceptable. Again, better refined and higher quality experimental cross-section data in these energy regimes would be highly valued by our community.

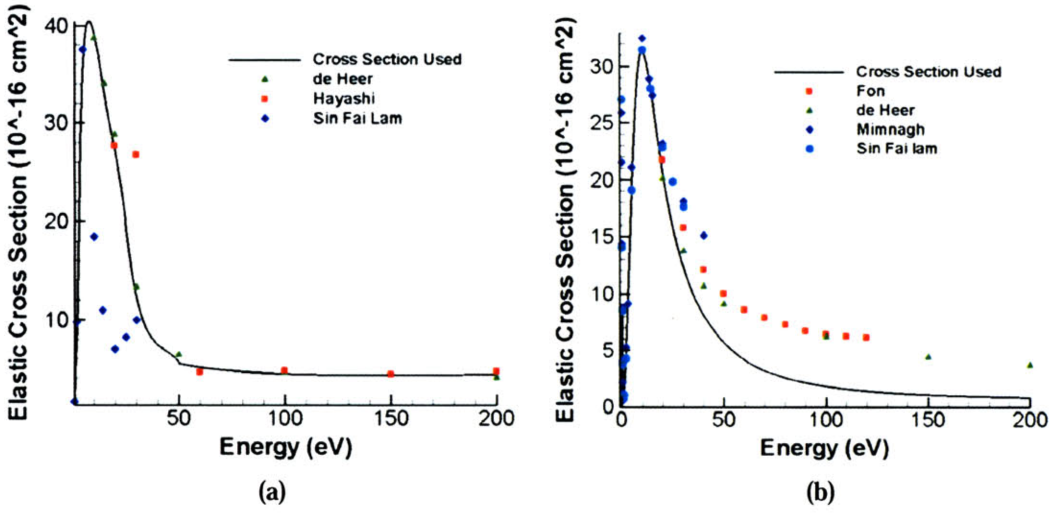


Figure 2-27: The experimental elastic cross-section data compared to the actual cross-section employed by the simulation for (a) Xenon and (b) Krypton.

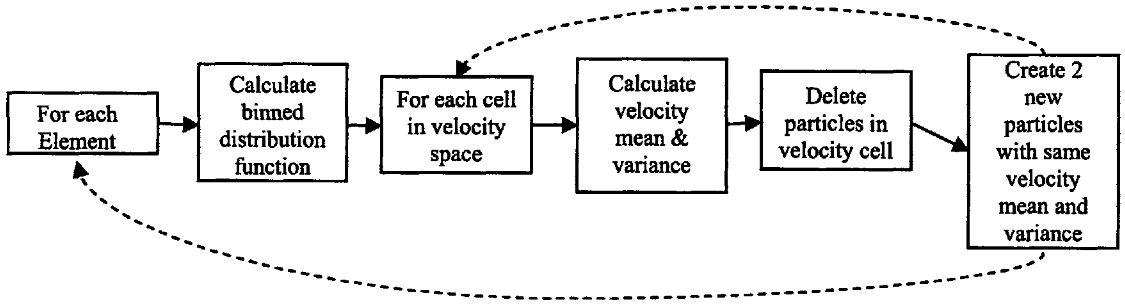
2.9 Velocity Redistribution Step

Particle-in-cell models attempt to simulate every particle of extremely complicated systems. Due to computational constraints of memory and time, actually accomplishing the goal of modeling EVERY particle in a Hall thruster is currently infeasible. Thus, past generations of the MIT Hall thruster simulations have relied upon superparticles, the concept in which 1 computational particle stands for a very large, constant number of real particles. Unfortunately, due in part to the shape of the

Maxwellian energy distribution and also to the fact that collision processes tend to generate large numbers of cold secondary particles, the number of high energy particles important to ionization processes in the thruster is significantly smaller than the number of less important low energy ones. Thus, if the same constant superparticle size is used for all particles, it is difficult to represent the less numerous high energy particles and the very common low energy particles efficiently. It is likely that, due to the random nature of the former models, the high energy particles may be missed all-together.

To alleviate this problem, we chose to extend the concept of superparticles by allowing the number of particles represented by each computational particle to vary. In this way, an equal number of high energy and low energy computational particles can be created which are able to simulate a system with a large number of low energy particles and relatively few high energy ones. However, if we left the system evolve, eventually the number of computational high energy particles would dwindle as before, due to collision processes and other factors. Thus, the velocity redistribution step was implemented as a very important step in the algorithm which allows us to maintain a more or less constant number of particles throughout the entire energy spectrum.

At periodic intervals within the simulation, the process illustrated by the flowchart in Figure 2-28 is performed. First, each element is queried separately. Then the particles of a given type within that spatial element are imagined to be placed within the cells of a regular, uniform grid in velocity space. This concept is graphically represented in Figure 2-29. In general, the number of particles in the various velocity grid cells might be



For each Element
Calculate binned distribution function
For each cell in velocity space
Calculate velocity mean & variance
Delete particles in velocity cell
Create 2 new particles with same velocity mean and variance

Figure 2-28: A flowchart depicting the process for the velocity redistribution module.

relatively similar to the representation in that figure; a large population of cold lower energy particles will be found with a few high energy outliers. Next, each velocity grid cell is examined. The local velocity mean and variance and the total number density associated with the particles within that grid cell is calculated. If there are two or fewer particles in the velocity grid cell, they are left untouched and the next cell is considered. However, if three or more particles exist in the velocity grid cell, they are deleted. Two new particles are then introduced, randomly placed into the spatial grid cell. These particles will average to the same mean velocity, same velocity variance, and same total density that the deleted particles did, only now the extraneous particles will have been removed. To calculate the velocities of these two new particles we examine the following system of equations:

$$\begin{aligned}
n^0 &= \sum_{i=1}^{n} w_i = w_{p1} + w_{p2} \\
p^0_{xyz} &= \frac{\sum_{i=1}^{n} w_i v^i_{xyz}}{n^0} = \left(w_{p1} v^{p1}_{xyz} + w_{p2} v^{p2}_{xyz}\right) / n^0 \\
t^0_{xyz} &= \frac{\sum_{i=1}^{n} w_i {v^i_{xyz}}^2}{n^0} - {p^0_{xyz}}^2 = \left(w_{p1}\left(v^{p1}_{xyz}\right)^2 + w_{p2}\left(v^{p2}_{xyz}\right)^2\right) / n^0 - {p^0_{xyz}}^2
\end{aligned} \tag{2.80}$$

The first equation tells us that the total density within the velocity grid cell, n^0, is equal to the sum of the statistical weights of the n particles initially in that cell. Then, clearly, the sum of the weights of the two new particles must be equal to the sum of the weights of the old particles in order for the total density to remain unchanged. The second and third equations describe similar conservations for the mean and variance of the system's velocity.

Equation *2.80* represents a system of 7 equations with a total of 8 unknowns, being the weight and three components of velocity for each of the two particles. Therefore, we are able to choose one unknown in order to completely constrain the other seven. The choice which seems the most intuitive and which will help maintain the balance of particle number and statistical accuracy of the simulation is to set the weight of the first new particle equal to half of the total weight. This immediately forces the second particle to also have the remaining half of the total weight and makes the solution of system *2.80* above very simplistic:

$$\begin{cases} w^{12} = n_0/2 \\ \bar{v}^{12} = p_0 \pm \sqrt{t_0} \end{cases} \qquad (2.81)$$

where the second equation is meant to represent three individual uncoupled equations, one for each of the three directional components of velocity. The plus sign applies to one of the particles while the minus sign applies to the other. With this selection of new particle weights and velocities, the energy, mean velocity, and density within the velocity grid cell will remain unchanged.

Through this periodic redistribution process, we efficiently retain our coverage of velocity space while ensuring that the number of particles in the simulation remains bounded and computational effort is not squandered on a multitude of cold, redundant particles.

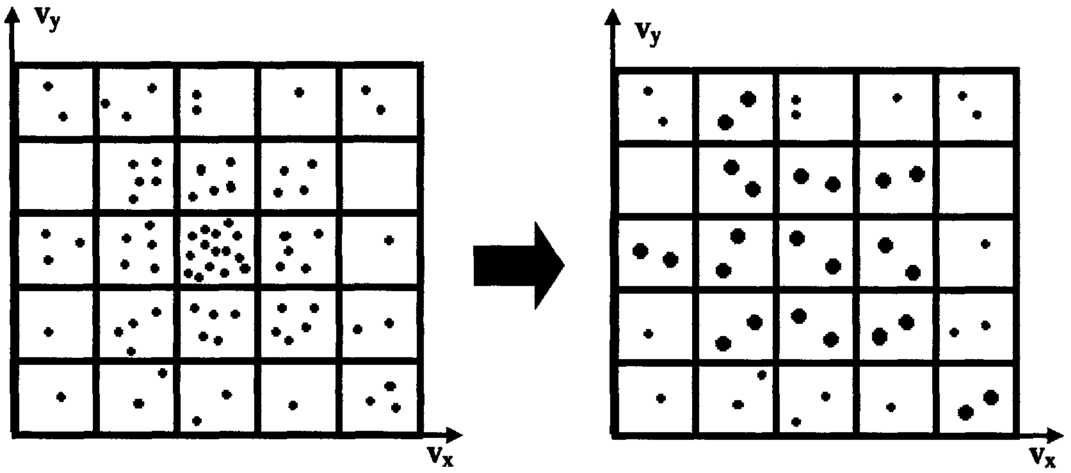


Figure 2-29: Velocity space redistribution is accomplished by particles in a given cell first being placed on a velocity grid. Velocity space cells having more than two particles then have their particles combined while energy, mean velocity, and density are retained.

2.10 Addition of Particles

This code was developed to be a fully-kinetic simulation. This implies that we attempt to physically model all of the various types of particles in a plasma system, neutral gas, electrons, singly-charged ions, douby-charged ions, and even triply-charged ions when appropriate. The methods used to actually introduce these computational particles into the simulation domain are outlined in the sections that follow. These are meant to give a general overview of how the code was designed to perform the particle creation task. For some specific applications, changes to this model needed to be made, and those are described in their own relevant sections, namely Sections 4.1.3 and 5.2.5.

2.10.1 Initial Conditions

In the general case, the initial plasma density and neutral gas density, n_0, are specified by the user in the file *input.txt*. These densities should be entered in units of particles per cubic meter. This file also provides the means to specify various shapes of initial plasma formation, with rectangular blocks and circular regions being the two examples implemented in the code. It would not be hard to extend these basic structures to more complicated initial geometries or unique density gradients, and this is indeed undertaken in later studies of the Hall thrusters which are discussed in later chapters.

Given an initial location for the plasma, the code is in general structured so that the RRC procedure is enacted in the region where the plasma density is non-zero and the refinement will be initially to the maximum level allowed, a parameter also specified in *input.txt*. The neutral density regions are not immediately refined, but adaptations to the code could easily be made if desired. Only singly-charged ions, electrons, and neutrals are at this time able to be initially placed within the code using the input files. If other species or finer control over initial location is desired, these cases must be re-coded via the functions in *init.f90* and *pic_rrc_nort.f90*.

Initial particle placement proceeds in practically the same manner regardless of the particle type and is designed to create an initially Maxwellian distribution in each computational grid cell. First, three user-specified parameters are read in from *input.txt*. The initial temperature of the distribution, specified in the code as *t0_half_v* or simply T_0, should be entered in units of electronvolts. The next parameter, *v_half_v*, defines how many thermal velocities into the tail the most extreme velocity grid cells should be placed. Thus if a typical value of 3d0 (the "d0" suffix specifies to the code that the input number is a double-precision value and is required by Fortran) is used, the particles with the most extreme velocities initially will have velocities of $3v_M$ and $-3v_M$ where:

$$v_M = \sqrt{\frac{T_0}{m_p}} \tag{2.82}$$

is taken to be the thermal velocity of the particle type, *p*. The third and final input parameter, *n_half_v*, defines the number of velocity grid cells to be used to cover one half of the distribution. For example, *n_half_v* = 3, will give a velocity grid that is 6 by 6 by 6 in three-dimensions or just 6 by 6 in two-dimensions. Half of the 6 velocity grid cells in a given dimension will contain particles with velocities greater than zero and the other half will contain their negative velocity counterparts. The image in Figure 2-30 attempts to clarify these three input values graphically.

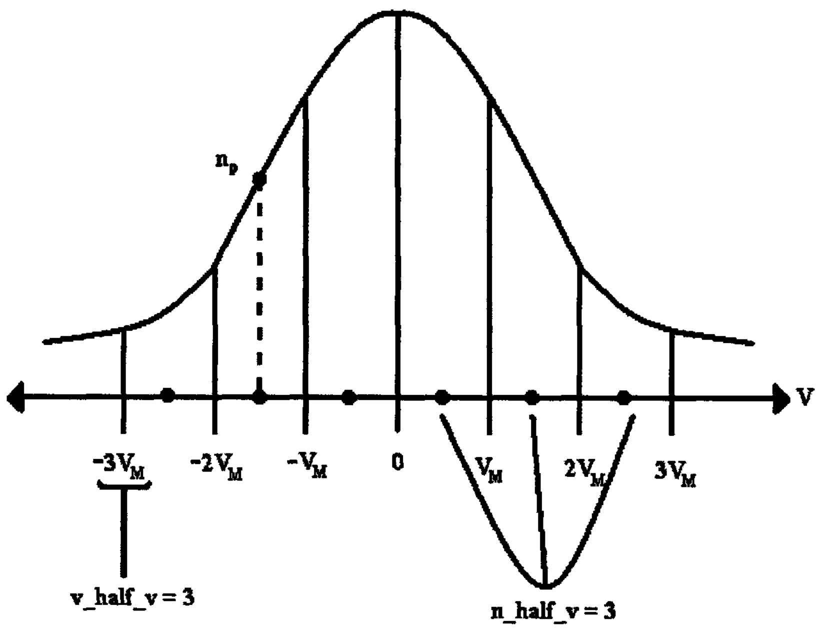


Figure 2-30: The diagram illustrates the discretization of the initial Maxwellian distribution during particle creation. The red dots on the velocity axis indicate the 6 particles that are created. The range of velocities is specified by *v_half_v* and the number of grid cells is controlled by *n_half_v*. The initial statistical weight assigned to a particle, being the value of the Maxwellian at that point, is shown by the dotted line as well.

Given these initial inputs, the proper number and type of particles are then created. They are assigned velocities according to their position on the velocity grid plus a possible user-specified initial drift velocity given in component form as u_x^0, u_y^0, u_z^0. Their statistical weights are then also assigned, being the value of the Maxwellian above their position on the velocity grid. More concretely, the initial density of a given particle with velocity v_x, v_y, and v_z in three-dimensions will be:

$$n_p = \frac{n_0}{\left(2\pi m_p v_M^2\right)^{\frac{3}{2}}} \exp\left[\frac{-\left[\left(v_x - u_x^0\right)^2 + \left(v_y - u_y^0\right)^2 + \left(v_z - u_z^0\right)^2\right]}{2v_M^2}\right] \qquad (2.83)$$

where here m_p is the mass of the given particle type. Note that this particle density has removed all dependence upon mass, and must be thought of as a number density. This density is then multiplied by the volume of the spatial element in which the particle initially rests to give a starting statistical weight:

$$w_p = n_p \cdot cell\_volume \qquad (2.84)$$

The remaining information to specify regarding the initial particle is its exact physical location within the given grid cell. We assume throughout out simulation that the mesh is suitably refined so that a particle's location within a given element is not necessarily significant. Therefore, the initial location for a particle is assigned randomly and more or less uniformly within the desired simulation cell. This process is accomplished by first randomly selecting the first and third corners of the cell or the second and fourth corners. Then a random number between the x-values of the selected corners and a random number between the y-values of the corners is uniformly guessed. That *(x,y)* location is then checked to make sure that it is within the given grid cell. If it is not, then the process is repeated until a location is chosen within the cell. This procedure ensures that the particles will be more or less uniformly placed throughout a given quadrilateral element, even if that element is highly non-orthogonal. Clearly, the method is not perfectly random and will have preference for the central portion of a very long and irregularly shaped cell, but as mentioned before, the particle's exact location in the cell should not be important. Additionally, it is recommended for other numerical reasons that such highly non-orthogonal elements be avoided in practice as often as is feasible and so any non-uniformity should be rare and minor in consequence.

2.10.2 Injection of Neutrals

Neutral injection flow rates are often given in either sccm or mg/s. Our simulation assumes the neutral gas flow is input in mg/s. To convert from sccm to mg/s is fairly simple and common practice. Equation *2.85* below provides a quick reference:

$$\mu[mg/s] = \mu[sccm]\frac{1mL_{STP}}{1cm^3_{STP}}\frac{1L_{STP}}{1000mL_{STP}}\frac{1mol}{22.4L_{STP}}\frac{(M)kg}{1mol}\frac{10^6 mg}{1kg}\frac{1\min}{60s} \qquad (2.85)$$

The simulation handles neutral injection, if necessary for a given simulation, through a single function, *rz_add_neutrals*, which may be commented in or out as the situation demands. Inside this function, the mass flow may be specified as well as the exact elements into which neutrals should be injected. The distribution of these particles is discussed below in Section 2.10.4. The initial injection temperature for the neutrals is by default taken to be the same as *t0_half_v_n* given in the file *input.txt*, but this can easily be adjusted in the code depending on the given application.

2.10.3 Injection of Cathode Electrons

The implicit modeling of the cathode is an exercise which previous Hall thruster numerical simulations have also encountered [73][18], and we chose to more or less follow their lead in implementing boundary conditions to handle this troublesome aspect of the code. For the two Hall thrusters studied in this thesis, the exact method of injection is described below in Sections 4.1.3.4 and 5.2.5.1. However, we include here the theoretical basis for that method in a more general sense.

The basic concept we rely upon is that the cathode serves to neutralize the ion beam downstream of the thruster exit. As such, we define a region in the simulation to be the "cathode plane" and ensure that within this region electrons are injected in order to maintain local neutrality if possible. We generally do not inject electrons if the potential in a given simulation cell is less than the cathode potential, specified by the user. Electrons would not be expected to fall into such a region of low potential and adding further electrons, even if the local charge in that particular cell happens to be positive, could further depress the potential and create an unstable situation.

Thus, the method proceeds by first locating the cells along the boundary specified to be the cathode boundary. If the charge in a given cell is negative or its electric potential is negative, it is excluded from the injection process. Otherwise, the density of electrons required to make the charge within each cell exactly 0 is calculated. The proper density of electrons is then injected into each of the considered cells. The number and distribution of those particles is described below in Section 2.10.4. The temperature of the injected electrons is user specified. In addition, they are given an initial drift velocity comparable to the energy they would have acquired rising from the cathode potential, ϕ_c, to the given local potential, ϕ. In mathematical terms:

$$|v_{drift}| = \sqrt{\frac{2e(\phi - \phi_c)}{m_e}} \qquad (2.86)$$

The direction of this drift can be adjusted, but is generally assumed to be inward and normal to the plane of the boundary which is injecting the electrons.

2.10.4 Injected Particles' Initial Distribution

Two types of initial distributions for injected particles have been implemented and tested. The first is exactly analogous to the method described above in Section 2.10.1 for the creation of initial particle distributions. Basically, the injected particles are placed uniformly on a velocity grid such that a Maxwellian (or half Maxwellian if desired) distribution is created. This, however, involves the injection of a large number of computational particles during each injection iteration. We found that such was not really necessary since the cell would soon be velocity-redistributed anyway and most of these particles would almost immediately be merged with others.

Instead, and in order to limit the increase of particle number in the short term, we also implemented the following injection scheme. Within the injection function, a number of particles to be injected, say *n*, may be specified. Then, the density to be injected into the given grid element is divided evenly among these particles, with each particle being given a density equal to 1/*n* of the total. The Box-Muller method of particle creation is then used, following Szabo and his predecessors [73] to establish the velocity of the injected particles. This method selects the particle velocity via:

$$|v| = v_M \sqrt{-2\ln(rand)} \tag{2.87}$$

where the v_M here is the same as in Equation *2.82* and is based on the specified injection temperature. The variable *rand* is a random number uniformly chosen between 0 and 1. Once the norm of the velocity is calculated, the components of the velocity are then assigned isotropically unless the application demands otherwise. In this way, we can create a smaller number of particles each with equal weight that may not have the best statistical coverage of velocity space, but will save on computational costs.

2.11 Boundary Conditions

The father of computer science, Alan Turing, once wrote "Science is a differential equation. Religion is a boundary condition" [89]. Maybe that is why boundary conditions seem to be the subject of so many doctoral candidates' fervent prayers and, oftentimes, curses.

In this section, a number of the different types of boundary conditions implemented in the simulation will be outlined and explained. Many are application-dependent and very likely would need reimplementation or extension if this model were applied to new problems, but those conditions that are thus far included are sufficient for the tasks at hand. The five-letter heading of each section indicates the string with which an edge created in Tecplot Mesh-Generator should be identified in order for the PIC/Vlasov code to recognize it as a boundary of the proper type. For more information on creating boundaries and meshes, consult Appendix A. Table 2.3 below is intended for quick reference and lists the five-letter identifier strings along with the "common" name of the boundary condition.

SPACE	Free space
METAL	Floating metal wall
DIELE	Insulating ceramic
SAXIS	Axis-of-symmetry
ANODE	Voltage-specified conductor
CATHO	Cathode quasi-neutral boundary
PLATE	Grounded collector plate

Table 2.3: Boundary conditions and their mesh-file identifiers.

2.11.1 SPACE – Free Space Condition

Arguably the simplest of the boundary conditions to implement is, of course, the free space condition. Any particle impacting a free space boundary is immediately deleted from the simulation. In some applications, data about the particle, including its contribution to overall thrust and current, will be accumulated upon impact with a free space boundary. All field solvers implemented treat the free space boundary as a Neumann or natural boundary condition, unless otherwise specified. Thus, the normal derivatives there are zero.

2.11.2 METAL – Floating Metallic conductor

If the designation METAL is given to a number of objects, they are all assumed to be electrically-connected, floating-potential metal conductors in the eyes of the potential solvers. That is, the charge of any particle impacting the surface is absorbed and assumed to be "smeared" out instantaneously across the entire METAL surface. This charge is used along with an input capacitance to determine the potential of the entire surface. If the particle impacting the surface was an ion, it is assumed to have recombined and is diffusely reflected back into the simulation as a neutral with half its original energy. Neutrals impacting this type of boundary are isotropically scattered by default, but versions of the simulation do exist with either specular or inelastic reflection implemented. Electrons are absorbed and destroyed.

2.11.3 DIELE – Dielectric Insulator

In some applications, notably the simulation of SPT-type Hall thrusters, ceramic insulators are a prominent boundary type which must be handled. The modeling of these boundaries follows Blateau [18] and others previous, and it should be noted that its assumptions may not be completely valid outside the realm of Hall thrusters. According to Blateau, the electric field within the dielectric channel-coating of a Hall thruster is negligible in comparison to the electric field inside the plasma itself. Therefore, the interior field is assumed to be zero, and charge that impacts the boundary remains there indefinitely.

For simulation purposes, this translates to the following. First of all, any neutral impacting a dielectric boundary is by default isotropically scattered without loss of energy. Versions of the code do exist in which a level of energy accommodation to the walls can be specified. The charge from any electrons which impact the wall are accumulated at the individual points of impact while the particle itself is deleted. The charge from impacting ions is treated similarly. However, the ions themselves are neutralized and bounced back into the simulation as neutrals. When solving for the potential, the Poisson solvers include the accumulated wall charge into the general charge for that particular grid element. The dielectrics are thus treated as perfect insulators.

2.11.4 CATHO – Implicit Cathode

In most cases, due to computational constraints and, more importantly, symmetry, a thruster's hollow-cathode is not modeled explicitly. Instead various methods have been designed over the history of Hall thruster simulating for dealing with the downstream conditions which an explicit cathode would satisfy. The method implemented herein is one which assumes that the cathode acts to maintain a quasi-neutral plasma in a plane some short distance downstream of the thruster channel.

For the purpose of particle impact, the CATHO boundary acts like a free space boundary. All particles reaching this edge are deleted immediately with the proper statistics first being tallied if necessary.

In terms of the potential solvers, several different conditions were implemented for this type of boundary. All boundaries labeled CATHO can be connected to an input cathode potential. This is in most cases an overly-strict and unnecessary condition, but it does help ensure the potential near the thruster exit remains reasonable. Another possibility is to allow all CATHO boundaries to float as Neumann, free space boundaries with zero normal gradients. The middle-ground between these first two is to allow all but a small subset of CATHO boundaries to float while forcing the rest to a specified cathode potential, usually about -10 Volts. The method desired for the particular trial or application can be easily swapped for in the code.

Regardless of the potential assigned to the cathode boundary cells, they are treated uniquely from other boundaries in order to force quasi-neutrality. At each iteration, after the particles have been moved and a new charge distribution has been calculated, the total charge present in cells bordering a CATHO boundary is computed. If this charge is negative or zero, nothing is done; the cathode cannot correct the non-neutrality. If, however, the charge is positive, then a spatially uniform distribution of electrons is injected into the CATHO cells whose charge will exactly neutralize the cathode boundary. These electrons are generally injected with a Maxwellian temperature distribution centered about a given input temperature, and are created in such a way as to uniformly cover velocity space while randomly placed in space. The current from these electrons is recorded and output to the *perf.dat* output file. One interesting behavior of this method that has been noted in the past [73][18] is the fact that even though there is no solid reason why this cathode injection method should constrain the anode current to be balanced by the cathode current, the two do vary simultaneously. This co-variation is in fact quite strong as evidenced by Figure 2-31 below, a plot of the currents in an SPT Hall thruster created and provided by Szabo using his PIC-MCC simulation.

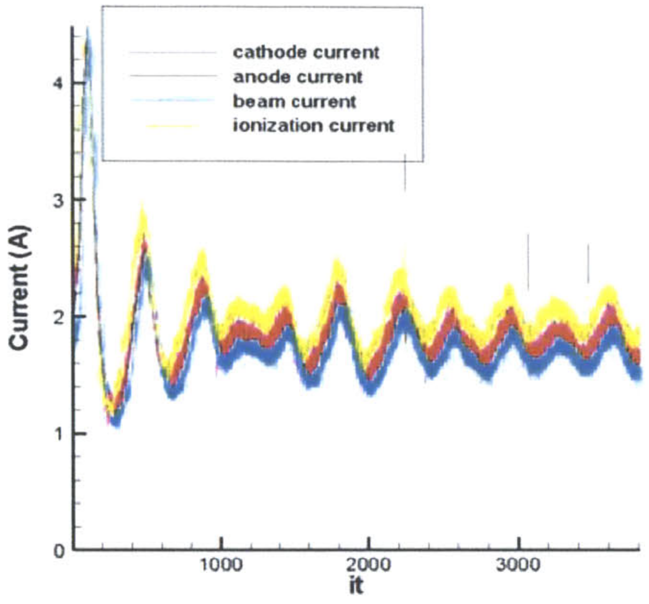


Figure 2-31: A plot depicting the strong co-variance of the anode and cathode currents along with the beam and ionization currents. This plot was generously created and provided by Szabo.

2.11.5 ANODE – Fixed-Potential Metal Anode

As its identifier suggests, an ANODE boundary generally represents the anode of a Hall thruster. The potential of such a boundary is fixed to that value specified by the user. Electrons impacting the anode are absorbed and tallied in the total anode current. Impacting ions also have their charge tallied for the total anode current, but recombine and are reinjected as neutrals with half of their impacting energy and a random velocity direction. Neutrals are accommodated to the specified anode temperature and isotropically bounced back into the simulation region.

2.12 Data Collection

Like a tree falling alone in a forest, a simulation without an observer to give it context and interpret its results might as well not even have existed. Fortunately our simulation provides ample outputs and a multitude of relevant data to keep the interested observer happily interpreting for a long time. The various output files currently supported by the simulation are detailed in the next section. This section is provided in the hopes that it may be a handy reference for another student who follows this work to quickly understand what data the simulation provides and how it is organized. All data files are output in Tecplot-readable formats, unless otherwise specified. When a file name is given as "_xxx," the implication is that these files are output periodically throughout the simulation and are labeled by increasing numbers, i.e. reb_0.plt and reb_100.plt.

2.12.1 Output Files Defined

reb_xxx.plt: Arguably the most important and useful type of output file available, this file provides a general spatial overview of the simulation and its fields. By changing the correct line in *input.txt*, this output file can be created for just a small rectangular region of interest or the entire simulation region if desired. In particular, the output variables include:

r (C): The charge density calculated for each computational element.

n_e, n_i, n_n, n_{i2}, n_{i3} (m^{-3}): The density of the indicated type of particles. Note that n_i takes into account only singly-charged ions while n_{i2} and n_{i3} account for doubly- and triply-charged ions, respectively.

E_r, E_{theta}, E_z (V/m): The three components of electric field at this instant in the simulation for each computational cell.

B_r, B_{theta}, B_z (Tesla): The three components of magnetic field for each cell. The magnetic field is generally assumed constant throughout the electrostatic branch of the code.

Phi (V): The potential at each cell in Volts

nump: The total number of computational particles associated with a given cell, including all simulated species. This can be used to help track the effectiveness of the velocity-redistribution step and to ensure the domain is being fully-covered.

wall_charge (C): The amount of charge that has impacted a dielectric wall associated with a given cell.

nt xxx.plt: This file generally contains the moments calculated for the electrons and the singly-charged ions at each simulation element. This includes temperatures and momenta. The variables are defined as:

n_e, n_i (m^{-3}): The number densities of the electrons and the singly-charged ions, respectively.

v_{te}, v_{re}, v_{ze}: The electron mean velocities in each cell in the θ, R, and Z directions, respectively. These velocities are normalized by the speed of light.

v_{ti}, v_{ri}, v_{zi}: The singly-charged ion mean velocities in each cell in the θ, R, and Z directions, respectively. These velocities are normalized by the speed of light.

t_{te}, t_{re}, t_{ze} (eV): Average one-dimensional temperatures for the electrons in a given cell computed componentwise for the θ, R, and Z directions, respectively.

t_{ti}, t_{ri}, t_{zi} (eV): Average one-dimensional temperatures for the singly-charged ions in a given cell computed componentwise for the θ, R, and Z directions, respectively.

perf.dat: This file summarizes important global simulation information over time, but does not contain any spatial information. There is only one of these files per simulation. The information that it holds includes:

time (s): The time elapsed in the simulation given in actual seconds.

i_{anode} (mA): The current reaching the anode at each given time step.

i_{beam} (mA): The current exiting the simulation domain as part of the particle beam. This may involve currents exiting through the top or right-hand boundary of the simulation in particular.

thrust (mN): The calculated thrust at each instant in the simulation.

coll_xxx.dat: Contained in this file is spatial information about where various collisions are occurring and relatively how much of each type of collision is occurring. Not all collision-types are currently output, but the remaining types could easily be added. The variables that are output include:

> W_{i1}, W_{i2}, W_{i3}, W_{el}, W_{exc}, W_{Bohm} (m^{-3}): These variables represent the number density of single-ionization, double-ionization, triple-ionization, elastic, excitation, and anomalous Bohm collisions which occurred in a given simulation cell during the output time step. Note that if the output time step is not also a collision time step, these values may all return 0.

2.13 Parallelization

Due to the overwhelming computational requirements of simulating millions of particles for hundreds of thousands of time steps, we employed the power of parallel computing to help make our simulation more efficient. For this step, the generosity and patience of the Cambridge, Massachusetts supercomputing services company Delta Search Labs, Inc. was prolific and invaluable. With virtually unfettered access to their high-performance 128 processor SGI supercomputer having 64 GB of memory shared among the processors, the feasibility and scale of this simulation were vastly improved. cannot thank them enough for this opportunity.

The parallelization of the code proceeded alongside code development with both a serial and a parallel version of the code available at most steps of the development process. The MPI standard parallel library was chosen as a communication protocol with the assumption that future users of this simulation might only have access to an Ethernet connected network and not the shared-memory behemoth we were fortunate enough to have at our disposal. The MPI standard is explained in great detail in numerous references [69], and was previously applied to MIT's PIC Hall thruster simulations [33]. A brief synopsis of testing conducted to select the most efficient MPI functions and parallel programming techniques for the problems at hand is given in Section 2.13.1 below. Full parallel performance and ground-truthing results for the current simulation can be found in Section 3.7 below. The remaining parts of this Section, 2.13.2 to 2.13.7, briefly outline the general techniques employed to parallelize each major portion of the simulation. Throughout the discussion, the variables *np* and *myid* will be employed for simplicity. The former represents the number of processors being used in a particular simulation and the latter is a processor's unique ID number ranging from 0 to *np*-1. Clearly, *np* will have the same value on every processor while *myid* will be different and is often used to determine how a processor should interact with the remaining processors. Unless otherwise stated and without loss of generality, it is assumed that processor 0 is treated as the "Master" processor when data compilation or task-management is required.

2.13.1 Comparing MPI Functions Using CVPerf

Many MPI routines have architecture dependent timing characteristics. The structure of our computing environment could therefore have a significant impact on which MPI calls to use in which situations to achieve the maximum parallelization efficiency. To gain a better insight into this trade off we used the Unix *ssrun* and *cvperf* commands. The respective *man* pages for each of these commands may be consulted if usage methods are desired.

Via these commands, we were able to obtain global "butterfly" images of the calling chain of our simulation and the various times spent in each of the subchains. The commands also allowed us to investigate which function calls were requiring the most computational time. For instance, a serial trial that required 764 seconds to complete required only 95 seconds to complete using 8 processors, indicating a nearly linear speed-up slope. More interestingly, in the serial trial, the particle pushing function was called for 668 seconds, or 87% of the total simulation time. In the parallel trial, the particle pushing step was called for only 14.6 seconds out of 95, or 15% of the total simulation time. This tells us that the parallelization we are using for the particle pushing step has been effective and that our parallelization efforts should next focus on a different portion of the simulation, a portion that is requiring more than 15% of the simulation time.

Using these same tools, we can see that of the 95 seconds in the parallel simulation the vast majority, 45 seconds, is spent in the function *MPI_SGI_REQUEST_WAIT*. This function is an internal one which is called by processes like *MPI_Allreduce* and *MPI_Bcast* if processors are not reaching barriers at approximately the same time. With this in mind, we examined a number of different sequences of function calls which accomplish the same task, sharing data among all processors, to see which required the least amount of *WAIT* on our architecture and for our specific simulation. A simple table of example results appears in Table 2.4. What was observed is that the optimal sequence of function calls may be different depending on the number of processors being used, but that in the limit as the number of processors becomes large, using the built-in MPI meta-procedures eventually becomes best, as would be expected. For our initial small simulations with four or eight processors that we will most usually be using for benchmarking results, it is interesting to note that using the standard MPI meta-calls may not be allowing us to achieve the most efficient results and so care must be taken when analyzing parallel performance in the future. It would be expected that only at large numbers of processors would our simulation truly obtain efficiency in parallelization.

Number of Processors	**4**	**8**	**16**	**32**
MPI meta-functions (AllReduce)	179 s	147 s	31 s	19 s
MPI mid-level functions (Reduce)	180 s	149 s	46 s	32 s
MPI low-level functions (Send/Recv/Bcast)	152 s	95 s	55 s	48 s

Table 2.4: The total simulation time required for an example electromagnetic problem using three different levels of MPI functions. For small numbers of processors, the overhead of MPI meta-functions outweighs their benefit, but as the number of processors becomes large, the meta-functions asymptotic complexity properties eventually dominate.

2.13.2 Initialization

The initialization phase of the simulation is performed almost entirely in parallel. All processors first allocate their own arrays and data structures. The *com2d.dat* file defines the size of these arrays on each processor, not for the simulation as a whole. For example, setting *maxpic*, the variable defining the maximum number of particles, to 10^6 in the serial version of the code would allow only 10^6 particles to be created. However, in the parallel version, each processor would be able to hold 10^6 particles enabling a total of 10^6**np* particles to be simulated.

The next step in initialization is the parsing of the Tecplot grid file. Since this step required significant file access, it was deemed more efficient to allow the Master processor alone to read the file, filling the appropriate arrays with vertex locations, cell connectivity information, and boundary conditions. This information is then broadcast to the remaining processes. In this way, all processors have access to the entire grid throughout the whole simulation. This was deemed an appropriate trade-off between memory and processing speed. It will be seen that each processor only truly needs a portion of the grid in memory at each time step. These portions change in order to maintain load-balance as particles move between simulation regions, and it would be possible, if desired, to transfer the actual grid data along with the particles between processors. However, this step would further complicate the implementation and significantly increase processing time as the all-to-all communications required are very expensive. On the other hand, the memory required to maintain the grid on all processors is, for most applications, small in comparison to the memory required to maintain the particles, and is generally not the bottleneck to simulation scale expansion. Thus, it was appropriate to choose storing *np* copies of the grid in favor of the time-consuming process of physically segmenting the grid and transferring portions of it.

With the geometry in place, all processors proceed to load the *input.txt* data file. The information in this file is required for all processors, and the file is small enough that parallel file access is not exorbitantly expensive. This done, each processor then is assigned an initially equal (or as close to equal as divisibility permits) number of grid cells to manage. All processors perform this entire calculation, saving their own assigned cells and all other processor's assigned cells in an array. A processor's cells are not necessarily spatially contiguous as they would be in true domain decomposition, representing a possible inefficiency in later steps. However, due to the inherent numbering of base mesh cells, the cells are in practice usually found close together or at least in large-sized groupings and are not scattered entirely randomly about the simulation domain. Future implementations could improve upon the spatial configuration of the processor's cells, however, if that complexity is required.

The particles are then initialized on each individual processor. The exact method for particle creation is described in more detail in Section 2.10.1. For our present purposes, it is enough to realize that an equal number of particles are created in every grid cell which contains plasma or neutrals, regardless of the density of the plasma. This ensures that the load is initially balanced between processors. The final steps of initialization involve the loading of the magnetic field and the initial adaptation of the mesh if applicable. Both of these steps are completed by all processors.

Thus, by the end of the initialization phase, the only distributed quantity is the particles themselves. All other information, grid structure, magnetic field, input and output parameters, exists in its entirety as multiple copies, one per processor.

2.13.3 Particle Injection

Depending on the application at hand, particles may need to be injected every time step or at other user-controlled intervals during the simulation. In general, the parallelization of such injection is very simply handled. All processors execute the same particle injection function appropriate for the given application, such as *P5_cathode_condition, NVHT_cathode_condition,* or *P5_add_neutrals.* Inside this function, the number, type, and location of particles to be injected is calculated. If the particles are to be created inside cells which are currently controlled by a particular processor, that processor is assigned to create them.

The ramifications of this process are not hard to discern. All of the particles for a given simulation cell will still be located on a single processor, allowing cell-wide statistics and manipulations to be easily undertaken. The disadvantage, however, is that the number of particles per processor is now unbalanced. The processor, for example which controls the cells closest to the anode or neutral injection point in a Hall thruster for example will, at this moment in the simulation at least, now be responsible for all of the newly-injected neutrals by itself. If we did nothing else, this processor would quickly become overburdened, diminishing the parallel efficiency. This problem will, however, be alleviated in Section 2.13.6 below.

2.13.4 Collisions, Vlasov Step, and Scattering

These three steps of the simulation can easily be grouped together in reference to the parallelization method since all three rely on a single principle. As long as all particles in a given simulation cell belong to a single processor, each of these three steps can be completed entirely in parallel. That is, each processor can perform the calculations for the cells currently under its control and needs no extra information from its neighbors.

There are some small complications to be noted, though. Both the collision and Vlasov redistribution steps involve large changes in the numbers of particles assigned to a given cell. At this point in the simulation then, the load will not be entirely balanced, and left unchecked, this problem could ruin our parallelization effort. The load rebalancing step mentioned in Section 2.13.6 overcomes this difficulty, however.

A second detail worth mentioning involves the scattering step. It is conceivable and, in fact, common that the PIC-weighting scheme which assigns a portion of a particle's density to the neighboring cells as well as its current cell, will scatter some density into cells not currently under a processor's control. Since all processors have their own copies of the grid information, the calculations for this have already been enabled. What is fortunate is that the scattering process is entirely associative, up to machine accuracy, and can be completed by each individual processor separately before their results are finally summed. In practical terms, this simply requires a call to *MPI_Allreduce* summing the individual processors' density, charge, and current results and broadcasting a copy of the completed calculation back to each of the individual processors. Thus, at the end of the scattering step, all processors know the charge distribution at each point in the grid, allowing us great freedom in selecting a parallel field solver in the following section.

2.13.5 Field Solution

As discussed in section 2.7.2, the main computation involved in the field solution module simply boils down to the solution of a large matrix equation. Thus, parallelization of this module became a choice of how to best solve a matrix equation in parallel. For my Master's thesis [33], I learned that the parallelization of iterative solvers is usually quite difficult and time-consuming. In search of a simpler method, I was introduced to MUMPS (MUltifrontal Massively Parallel sparse direct Solver). MUMPS was developed by a number of French researchers to be a general use, efficient direct matrix solution package. The Fortran library is available for download from the MUMPS website, along with detailed information about the algorithm and its development history [86][3][4][5]. While MUMPS does require the programmer to have pre-installed copies of certain common linear algebra libraries, such as BLAS and SCALAPACK, the package itself was relatively simple to integrate into our existing simulation. Any questions that arose during installation or use were quickly answered in detail either by consultation with the pre-existing web documentation or by timely and polite correspondence from the package's developers.

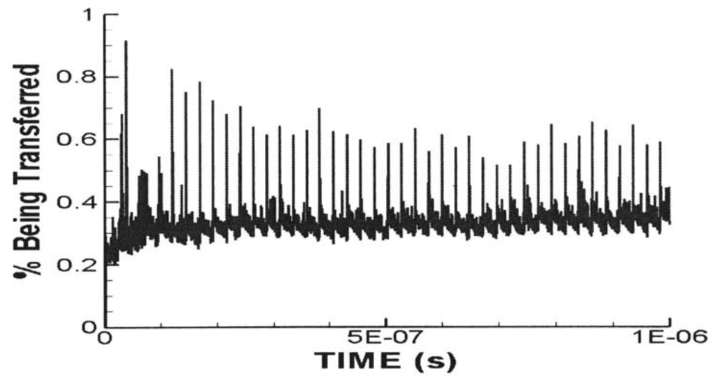
Incorporation of the MUMPS package into our field solution module did incur a small amount of overhead, both in terms of time and memory. Whereas our iterative solvers could solve the matrix implicitly without actually constructing it, MUMPS requires that the matrix be explicitly constructed and passed as a parameter. Fortunately, this memory requirement is not exorbitant due to the sparseness of our matrix and MUMPS' ability to compute with sparse matrix structures. Some time is spent creating this matrix explicitly, however, and this is highlighted in Section 3.7.2. That section also discusses the surprisingly low parallel efficiency of the MUMPS solver.

One other minor problem arose during our use of the MUMPS package. For some reason, either through our incorporation of the package or through its original implementation, our simulation with MUMPS installed exhibited a small, but significant memory leak. Each time the MUMPS solver would be run, the stack would become slightly more and more full. We contacted the developers of MUMPS and notified them of this problem. They responded that they were aware of such memory issues in previous versions of the MUMPS software and that newer versions would hopefully correct this problem [51]. Unfortunately, over several hundred thousand iterations, this small memory leak could easily lead to a stack overflow and the failure of the simulation. As such, for many of the longer simulation runs, we opted to use a serial iterative SOR solver to avoid the memory issue. That solver was obviously slower and less efficient, but had the advantage of being foolproof and wholly stable. Another solution to this problem would be to use MUMPS but run the simulation only for a small number of iterations at a time, save out the data, stop the program and clear it from memory, and then restart the simulation again from the saved point. This method was highly labor-intensive, however, and so in general was not adopted.

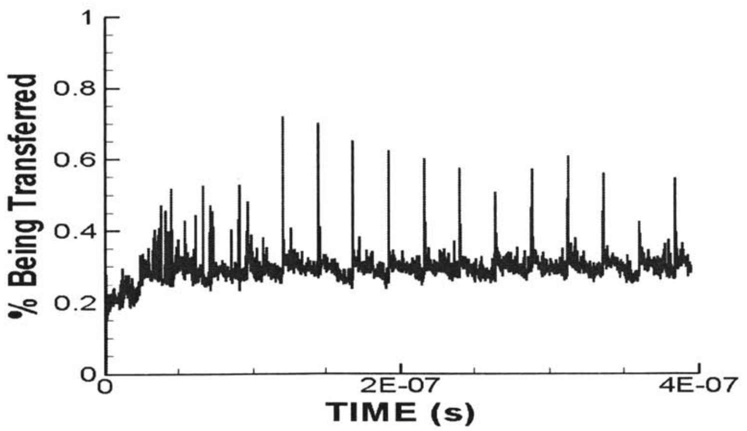
2.13.6 Load Rebalancing

As mentioned in the preceding sections, the number of particles per processor is initially equal, but over the course of an iteration can significantly change due to collisional processes, the Vlasov redistribution step, and application-dependent boundary conditions. Our approach to solving this problem is to reorganize the cells for which a processor is responsible, balancing the number of particles (but not necessarily the number of grid cells) between processors.

The load balancing process occurs after the whole-cell dependent steps outlined above, and before the particle motion step described below. At this point, all processors communicate to the master the number of particles they are maintaining in each of their assigned cells. The master compiles this information and then assigns a chunk of cells to each processor in turn. The number of particles per chunk is kept as equal as possible. In this way, each processor now has a new chunk of cells assigned to it at every time step. If the chunks were simply randomly assigned or assigned in random order, this could in theory mean that a processor would have entirely different particles to tend to at each time step, requiring a great amount of information transfer during the next step of the algorithm, particle movement. However, in practice, the chunks are assigned in a systematic manner which balances the load while attempting to minimize the number of particles that must be communicated between processors. This is accomplished by assigning the chunks in the same numerical order at each iteration, and as Figure 2-32 below indicates, the number of particles changing processors is in most cases only a small percentage of the total number of particles in the simulation. The trial was run with varying numbers of processors to ensure that the method was scalable and particle transfer did not significantly increase with increasing numbers of processors. The results presented here are for 8 processors and 16 processors in Figure 2-32(a) and Figure 2-32(b), respectively, and no significant trend over number of processors was observed. One artifact that might require explanation, though, is the periodic peaking clearly visible in the figures. These peaks coincide with iterations in which the neutrals have been velocity-redistributed. This event is infrequent throughout the simulation since the neutrals tend to move very little. However, when it does occur, the particle number on most every processor is drastically changed instantaneously, unbalancing the load, and requiring a slightly higher amount of particle transfer between processors to complete the rebalancing. This is merely a numerical feature, however, and is a totally nominal behavior.



(a)



(b)

Figure 2-32: The percentage of the total particles being transferred between processors at each time step is displayed for trials with (a) 8 processors and (b) 16 processors. Note that the horizontal axes of these figures are different. The periodic sharp peaks are due to the re-weighting of neutral particles which occurs relatively infrequently, but significantly alters the particle number on each processor requiring greater than usual particle transfer to rebalance the load.

While the communication of particles is an expensive portion of the parallelization algorithm and does somewhat reduce the parallel efficiency, there is unfortunately no better alternative method for parallelization that exhibits so suitable a mix of efficiency, stability, and ease of implementation.

2.13.7 Particle Movement

The final step of the simulation is the actual updating of the individual particles' location given the present electromagnetic fields. The parallelization of this step is more or less straightforward: each processor moves the particles to which it has access. All processors have a full copy of the grid geometry, the fields, and all other relevant quantities making this a possibility. The only complication involves particles that, at the end of the particle motion step, are no longer situated inside the chunk of cells assigned to the processor. These particles originate either by physical movement across chunk boundaries or from the motion of the chunk boundaries themselves as described above in the load rebalancing section. Both types of particles are the same from the code's standpoint and are treated in the same manner.

At the end of the particle's motion, the simulation cell in which it rests is calculated. Assuming this cell is no longer assigned to the current processor, the particle is placed in an array to await communication. After all particles have been moved, the actual transfer of information occurs, a much more efficient method than trying to communicate the particles one at a time. Each processor takes its turn sending particles to each of the other processors. If a processor is not sending on this turn, it waits to receive. This type of all-to-all communication is particularly detrimental to parallel efficiency and should be severely rationed. However, as shown above in Figure 2-32, the number of particles being communicated in this manner is in most cases quite small, and so the communications occur relatively quickly. This algorithm may not scale to extremely large numbers of processors, however, due to the all-to-all communications. Future iterations of this simulation could alleviate this problem by moving to a method of full domain decomposition which would require a processor to communicate with a much smaller number of other processors at each step. Unfortunately, domain decomposition often requires significant a priori knowledge about the application being studied and its spatial tendencies in order to properly balance the load and provide workable parallel efficiency. Our simulation was meant to be generally applicable and such methods appeared to be outside the scope of our objectives. Thus, application-dependent benefits might be gleaned by future alterations to this parallel algorithm, but for our present purposes, it was the optimal choice.

Chapter 3

Code Validations

As the various modules of the simulation were being implemented, we undertook to test them on an individual basis and in combination one with the other. The tests included in this chapter represent only a subset of those actually performed on the code. They range from the very simple single particle trajectory tests to full-scale simulations of Langmuir cold plasma oscillations. By conducting such benchmarking tests ahead of time, we were able to gain significant confidence in the accuracy and applicability of our model before applying it to the Hall thrusters of interest. Debugging a code of this magnitude absorbed at least half of the total development time, and the validation tests discussed below were instrumental in highlighting and correcting the obligatory errors which arose during programming, both minor syntactical and major structural. By the end of the chapter, we hope it will be clear to the reader that the implementation of our simulation is not only theoretically sound but also numerically accurate and robust.

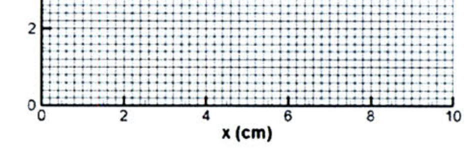
3.1 Constant Field Particle Motion Integrator Tests

In order to ensure that the semi-analytical particle motion trajectory integrator was implemented and operating properly, a number of simple low-level tests were undertaken with the aim of verifying the accuracy of this component. These tests all placed a single computational particle, with either positive, negative, or zero charge and varying computational masses, onto different computational meshes with various constant in time electric and magnetic fields. The results of these tests are demonstrated and described in the next section.

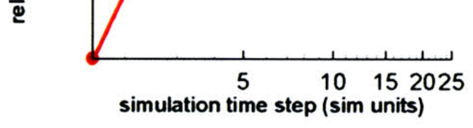
3.1.1 Two-Dimensional Integrator – Zero Fields

The first very simple test placed three different particles, one with zero charge and the other two with the positive and negative fundamental charge, onto a simple square mesh having a grid-spacing of .2 microns. Each particle was assigned the same mass, given the same initial speed of .9c, and placed an equal distance from the edge of the mesh. However, each was sent in a different direction, with the negative particle being given $v_x = -.9c$, the neutral being given $v_y = .9c$, and the positive particle being given $v_x = .9c$.

An image of this test is depicted in Figure 3-1(a). The first important fact to note about this test is that all of the three particles remained on an exactly straight trajectory with no change in position except what was due to their respective velocities. Also important to note, and evident from the even-spacing of the particle positions in Figure 3-1(a), is that all three particles traveled at the same rate. Each reached the edge of the



(a)



(b)

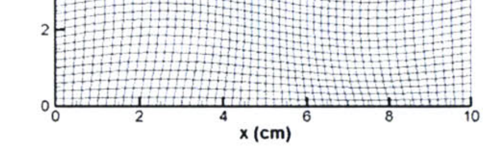
Figure 3-1: (a) Three different charged particles released at the same time in zero electric and magnetic field all travel at the same rate and in straight lines. All three reached the mesh edge at precisely the same moment in the simulation. (b) The relative error in impact time decreases with decreasing time step.

mesh at exactly the same time. So no type of charge is being treated differently than the others in the case of zero fields, despite that a different, pared-down function is actually used to move the neutral particles.

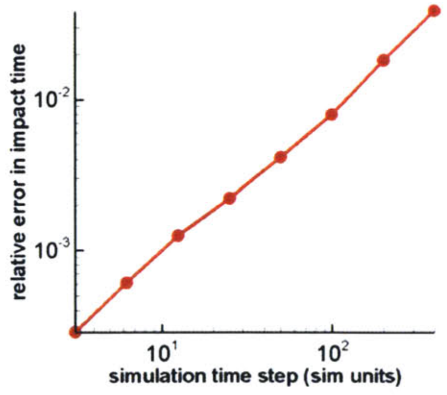
More quantitatively, Figure 3-1(b) depicts the simulation time required for the three particles to reach the mesh edge normalized by the actual time it should have taken. The simulation time step was varied for these trials, and one can clearly see that as the time step, here given in simulation units of μm/c = 1.66e-14s, is decreased, the simulation time required to travel the distance to the mesh edge tends to the theoretical value. The motion trajectory calculator is accurate in the presence of no fields and at sufficiently small time steps

3.1.2 Two-Dimensional Integrator – Zero Fields with Adaptation

The next test which was conducted had a set-up similar to the one above except that in this case, the simple square mesh was skewed to make it non-orthogonal and also adapted in a number of places. The resulting mesh is depicted below in Figure 3-2(a) with the particle trajectories overlaid as before. Again, all three particles were given initial speeds of .9c and sent in different directions.



(a)



(b)

Figure 3-2: (a) Three particles were released in zero field on an adapted mesh. All three reached the mesh boundary at the same time. (b) The relative error in the time elapsed before the particles impacted the boundary decreases with decreasing time step.

All three particles arrived at the mesh boundary at precisely the same instant. The error in that impact time is shown plotted in Figure 3-2(b) for several different time step sizes. Clearly, as the time step is reduced, the error is also reduced and the trajectory becomes more accurate. The adaptation of the mesh and its non-orthogonality did not disturb the convergence of the trajectory calculations.

3.1.3 Two-Dimensional Integrator – Constant B-Field

Having shown the correctness of the particle motion calculator in the absence of fields, it is next imperative that the accuracy also be shown in the presence of electromagnetic field systems. In this case, a magnetic field of strength .2 Tesla is introduced in the positive z-direction. The same three particles are again introduced at the same locations on the unadapted mesh and given .9c initial speeds. In this case, the uncharged particle should be unaffected by the magnetic field, but the two charged particles should execute cyclotron motion with gyration frequency and Larmor radius given by the well-known:

$$\begin{aligned} \omega_c &= \frac{qB}{m} \\ r_L &= \frac{v_\perp}{\omega_c} \end{aligned} \tag{3.1}$$

In this case, the theoretical Larmor radius of each charged particle, having mass of 1 electron mass, can be calculated to be r_L' = 0.76702899cm assuming we are traveling in the reference frame of the particle. However, in the laboratory's rest frame, we must apply the relativistic correction:

$$\begin{aligned} r_L &= r_L'(\gamma) \\ \gamma &= 1/\sqrt{1 - v^2/c^2} \end{aligned} \tag{3.2}$$

where r_L is the radius in the lab frame, r_L' is the radius in the particle's frame, and γ is the relativistic correction factor. With this correction, we find that the Larmor radius in the rest frame should in fact be r_L = 1.759685188cm.

Figure 3-3(a) depicts the particle trajectories for this test overlaid on the mesh. Note that as expected, the negative particle executes right-handed motion while the positive particle executes left-handed motion about the magnetic field. The neutral particle travels in a straight line and exits the simulation. The distance from the particle's position to the theoretical guiding center was calculated at each time step and averaged, giving us a calculated Larmor radius. Figure 3-3(b) shows that as the time step was decreased, the Larmor radius approached the theoretical value.

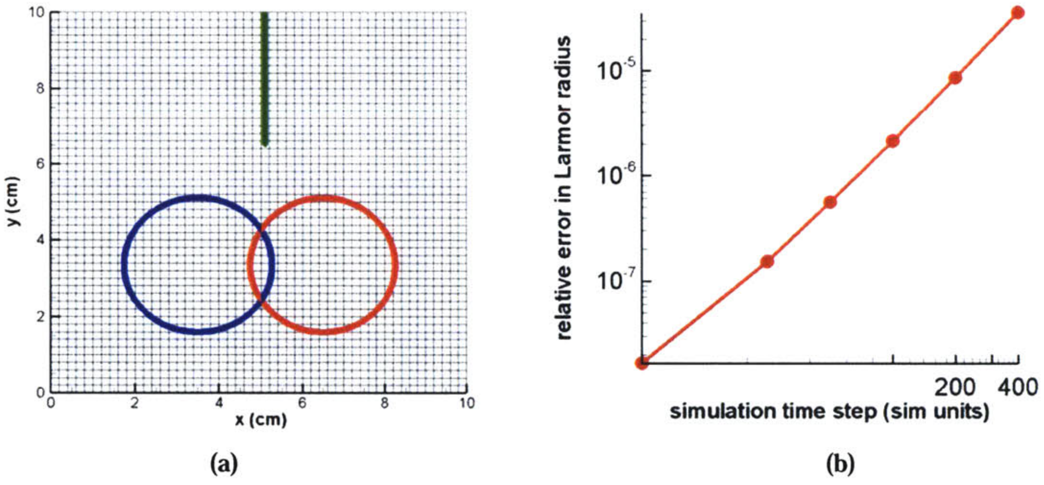


Figure 3-3: (a) The trajectories of the three described particles in a field of B_z = .2 Tesla. (b) The relative error in the calculated Larmor radius is reduced with reducing time steps.

3.1.4 Two-Dimensional Integrator – Constant ExB Field

Given that the magnetic field calculations were accurate, we next added a constant electric field to the tests as well. The set-up for this test began again with the same three particles, all with the mass of an electron and one each of positive e, negative e, and neutral. This time, the negative particle was placed near the bottom of the mesh, the positive nearer the top, and the neutral particle above that. The charged particles were each given the same initial velocity of .9c in the positive x-direction only. The neutral was assigned a positive x-directed velocity of .05c, for reasons that will be clearer in a moment. The magnetic field, still in the positive z-direction, was increased to .4

Tesla to tighten the orbits of the particles. Finally, an electric field, E_y, was added with a strength such that the guiding center drift velocity:

$$v_{E\times B} = \frac{E\times B}{B^2} = .05c \tag{3.3}$$

The trajectories of the three particles in this configuration are shown below for the simple square mesh in Figure 3-4(a). The charged particles very clearly execute the ExB drift expected. At the right side of the mesh where the trajectories end, it can also be seen that the positive particle is revolving in a clockwise direction while the negative particle is revolving in the opposite counter-clockwise direction. This too is as it should be. Finally, one notes that the radii of the gyrations is stable and constant throughout the motion.

Figure 3-4(b) is a plot of the electron's average x-velocity, normalized by the speed of light, over time. Of course, the x-velocity is cyclic and so in the beginning, the running average oscillates strongly. However, over time, it is clear that the average is asymptoting to the expected drift velocity of .05c. The guiding center of the drift motion is seen to move at the same velocity as the neutral particle as also evidenced by the length of the neutral's trajectory trace in the previous Figure 3-4(a).

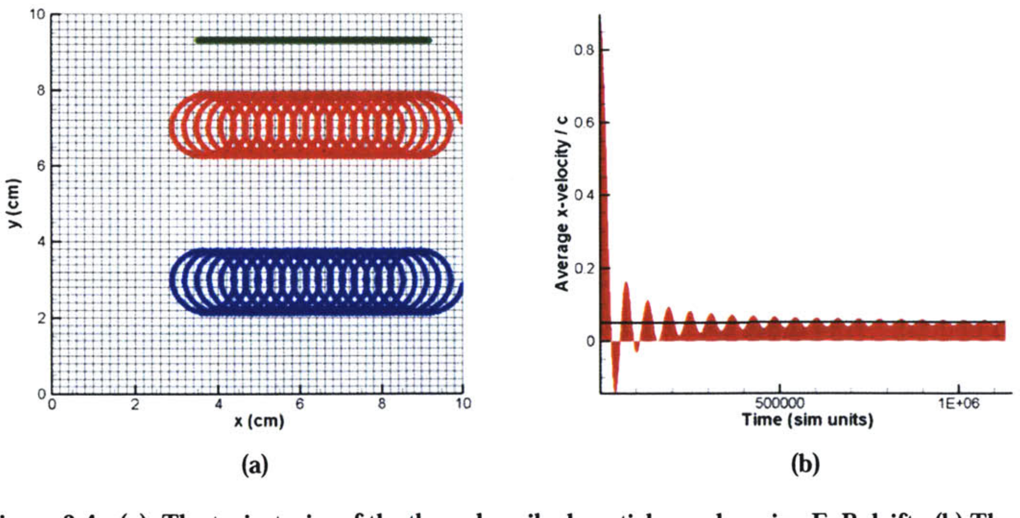
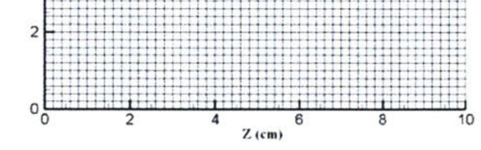


Figure 3-4: (a) The trajectories of the three described particles undergoing ExB drift. (b) The time-averaged x-velocity (shown for the electron) tends to the theoretical drift velocity on average.

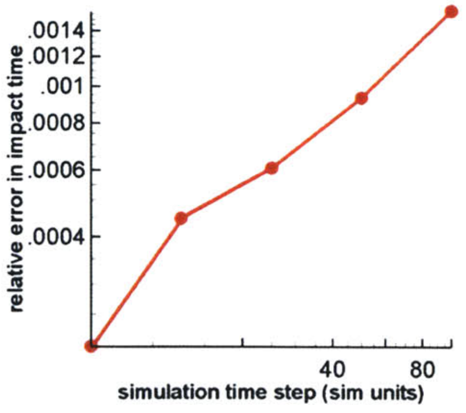
3.1.5 Axisymmetric Integrator – Zero Fields

Having thoroughly tested the two-dimensional particle trajectory calculator, it was next necessary to test the RZ, axisymmetric integrator. The implementations for each case were written separately, and the axisymmetric case includes the more complicated technique of folding the third, azimuthal direction down into the RZ plane.

As before with the two-dimensional integrator, the first test conducted on the axisymmetric integrator was the case of three different particles with negative, positive, and zero charges being placed on a simple, orthogonal square mesh representing a cross-sectional plane which can be imagined to be revolved about the y=0 axis. The particles in this case were again given speeds of .9c and let fly toward the mesh boundary from equal distances. Their trajectories are shown below in Figure 3-5(a). The time elapsed before each particle collided with the boundary was recorded; in every trial the three particles reached the boundary at precisely the same time. Figure 3-5(b) depicts how the accuracy of this impact time was increased as the timestep was decreased. The convergence of the axisymmetric integrator on orthogonal meshes was in this way demonstrated.



(a)



(b)

Figure 3-5: (a) The trajectories of three particles using an axisymmetric solver with no electromagnetic fields. (b) The relative error in the time until the particles impacted the boundary is reduced with reduced time step.

A second related trial was also conducted with zero fields present and the three initial particles. Again, the particles were each given a speed of .9c, but the direction of that velocity was now different. The negatively charged particle was given only velocity in the negative θ-direction while the positive particle and the neutral were sent in the positive θ-direction. In this way we could test the accuracy of the integrator in handling particle motion directed out of the simple RZ-plane. The theoretical motion of the particles is demonstrated below in Figure 3-6. We now must take a perspective looking at the thruster end-on, viewing the Rθ plane. The actual path of the particle is a straight line with the velocity constantly remaining in the x-direction of this image. In the simulation, this motion is folded back into the RZ-plane at every time step, giving the stepped trajectory depicted in the figure.

Figure 3-7(a) shows the trajectory of the three particles as it is viewed in the simulation projected onto the RZ-plane. As an additional test of the axisymmetric integrator's correctness, the mesh used in this trial was the adapted non-orthogonal mesh shown. Despite the out of plane motion and the non-orthogonal mesh, all three particles reached the top boundary of the mesh at the exact same instant in every case. Figure 3-7(b) plots the relative error in this impact time as the simulation time step is made smaller. As in previous cases, the integrator performed admirably and the convergence of the implementation was demonstrated.

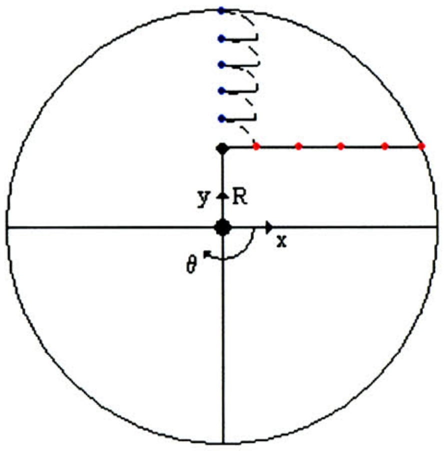
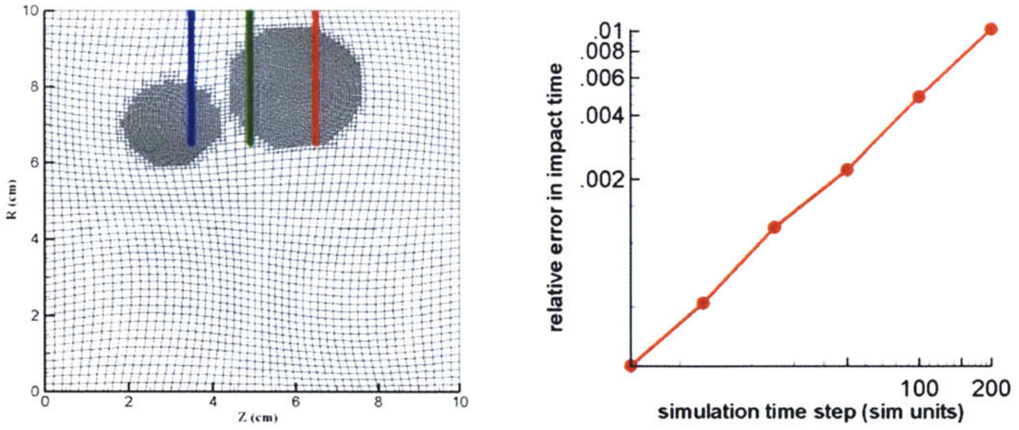


Figure 3-6: The particles were given only x-directed velocity. The actual trajectory then is just in the x-direction to the right toward the simulation boundary. However, in the code, the particle is projected at each timestep back into the RZ plane and so only seems to move in the R-direction.



(a) (b)

Figure 3-7: (a) The trajectories of three particles given only θ-directed velocities appear to be only in the R-direction after they have been projected onto the RZ plane. (b) Even though the particles were not all traveling in the same direction and they traveled over different adaptation boundaries, they still arrived at the mesh boundary at the same moment. The error in this impact time was reduced with decreased time step.

3.1.6 Axisymmetric Integrator – Constant B-Field

Continuing the analogy with the two-dimensional integrator tests, the next step was to test the particle motion implementation with a constant magnetic field. The three particles were placed on an adapted, non-orthogonal mesh, and each was given an initial velocity of .9c. A constant magnetic field of .2 Tesla was associated with the positive theta direction. The negative particle was sent in the negative Z-direction, the positive particle was sent in the positive Z-direction, and the neutral particle was sent in the positive R-direction. Corresponding to the equations presented in Section 2.5, the motion of the charged particles was a circle about a fixed center while the neutral particle was unaffected by the magnetic field and exited through the top boundary of the simulation. Figure 3-8(a) shows the trajectories of the three particles. Notice that in this case, the negative particle moves in a counterclockwise direction while the positive particle moves in a clockwise one. This is because the simulation defines a positive B_θ as being directed into the plane. As such, the negative particle is still executing right-handed motion and the positive is executing left-handed motion as they should. Figure 3-8(b) depicts that the error in the calculated Larmor radius decreased as the simulation time step decreased.

In the axisymmetric case, we must also test that the out of plane dynamics are working properly as well. So a similar test to that described above was also performed. This time, the simulation region around the negative particle was filled with only a constant .2 Tesla B_Z field while the region around the positive particle was assigned a .2 Tesla B_R field. The motion of the particles in each of these cases is, of course, still circular but takes place in the $R\theta$ and θZ planes, respectively. This motion, projected onto the RZ plane, is shown below in Figure 3-9. Again, we must remember, as demonstrated in Figure 3-6, that motion in the θ-direction is turned into motion in the R-direction when the projection onto the RZ plane is made.

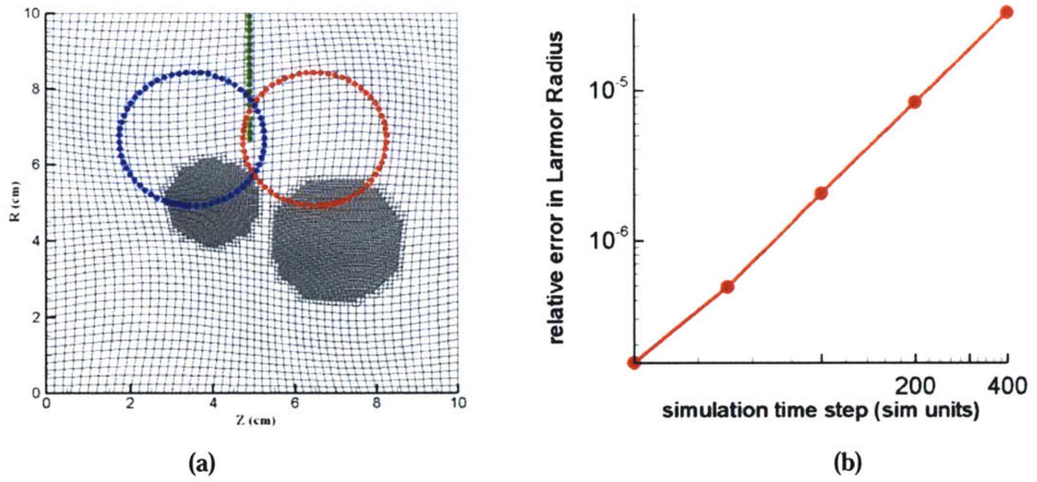


Figure 3-8: (a) The negative particle (left) executes right-handed motion while the positive particle (right) executes left-handed motion about the magnetic field which is directed into the drawing. (b) As the time step is decreased, the error in Larmor radius is decreased.

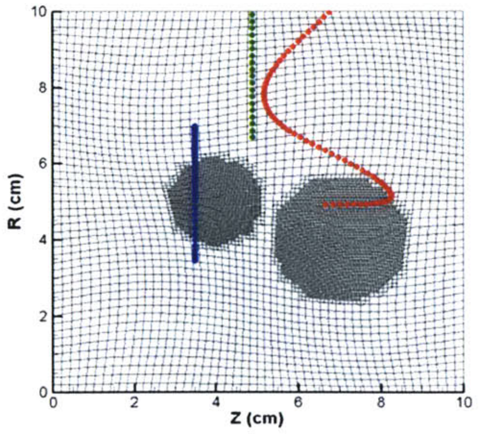


Figure 3-9: The trajectories of a negative particle (left) about a B_z field and a positive particle (right) about a B_R field. The neutral particle is unaffected.

3.1.7 Axisymmetric Integrator – Checking Mass

All of the preceding tests have assumed that the mass of the particles was simply a single electron mass, with a value of 1 in simulation units. It is important to ensure that the motion trajectory calculator does properly scale quantities as the mass is scaled. For this test, two positively charged particles were placed near the leftside of the simple, orthogonal simulation domain shown below in Figure 3-10. The green particle in the image was assigned a mass of 10 electron masses and the blue particle was assigned a mass of 5 electron masses. Both were given initial velocities of .1c directed in the negative-R direction. The magnetic field was set to .1T and directed in the positive-θ direction. Again, the theoretical Larmor radius in each case was calculated, remembering to scale for the different masses and also remembering to incorporate the relativistic correction. For the heavier particle, the theoretical radius should be .017130959m while the lighter particle should have a radius exactly half of that. Comparsion to Figure 3-10(a) indicates qualitatively that the orbit calculator does appear to be properly scaling with mass since the computed radius for the heavy particle appears to be roughly .017m while the lighter particle's radius appears to be just half that. Quantitatively, Figure 3-10(b) shows that the errors between the computed radii and the theoretical are indeed small and that as the time step is reduced, the errors in the computed Larmor radii are also reduced. Note that the abscissa's scale in the figure has been keyed to the Larmor periods of each particle in order to ensure that the number of time steps in a Larmor period was kept constant between trials. This allowed us to more clearly compare the integrator's relative accuracy when computing for particles of different masses. Perhaps unsurprisingly, however, the two computed lines fell exactly on the same place in the graph in Figure 3-10(b), demonstrating that the integrator is computing with the same accuracy for both cases.

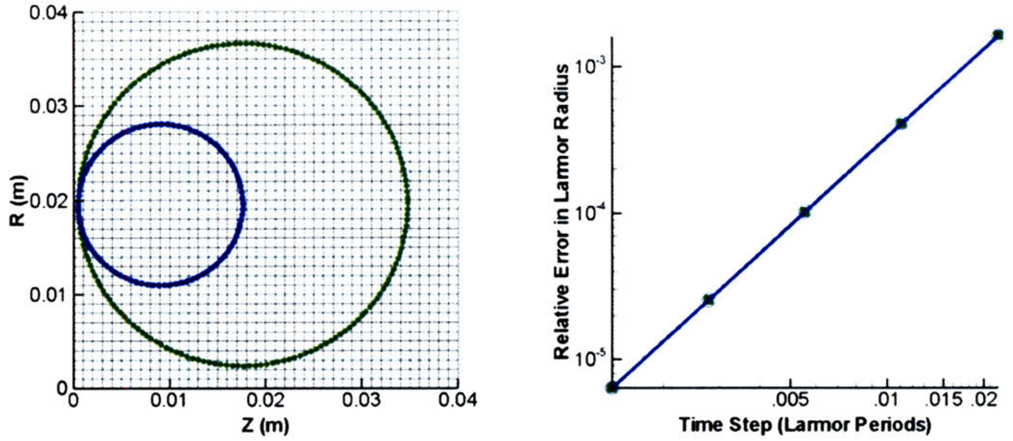


Figure 3-10: (a) The trajectories of two particles in a constant magnetic field. The green particle has twice the mass of the blue particle. (b) The computed Larmor radius of both particles converges to the theoretical value with a small enough time step.

As a demonstration and check of the artificial mass ratio implementation, a further short test was conducted. The artificial mass ratio parameter was set to 1/25, meaning that non-electron particles would now have 1/25 of their physical mass in the simulation. The particle with 10 electron masses was again launched from the left side of the domain with a purely negative velocity. To account for the lighter mass, the particle was given a 5 times greater speed, .5c, as it would have if it had been accelerated by the same force as before. Since the magnetic field seen by heavy particles is reduced by the square root of the mass ratio, the Larmor frequency calculated for the particle was the same as the theoretical frequency. The Larmor radius was also the same as the calculated radius excepting of course that it increased slightly due to relativistic effects. With relativistic effects included, the theoretical radius should have been .019681973m and the average computed radius over one period differed from this value by only .000095m. Figure 3-11 plots this trajectory in blue while the green trajectory is the trace of a particle with no artificial mass ratio. The implementation of the artificial mass ratio in the trajectory integrator is correct.

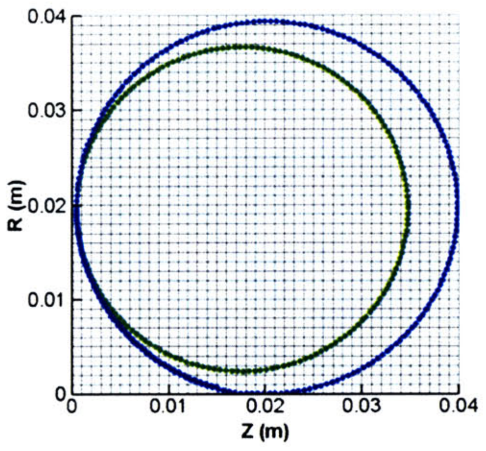


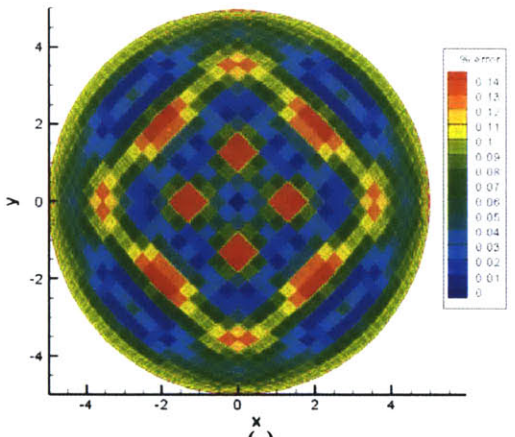
Figure 3-11: The inner green trajectory was calculated with no artificial mass ratio. The outer blue trajectory was calculated with an artificial ratio of 1/25. The only reason for the difference between the two is the relativistic effect of the artificially lighter particle moving faster.

3.2 Electric Potential and Field Solution Tests

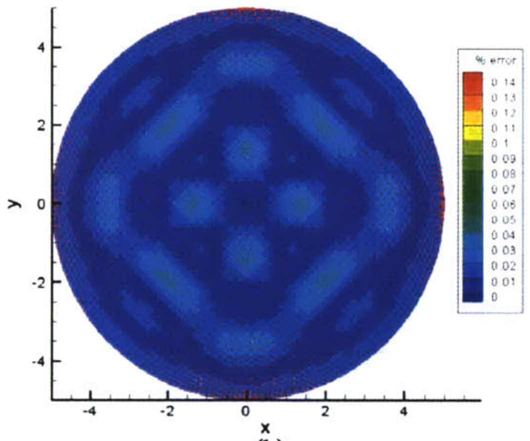
Since we were forced by the complexity of our meshes to develop entirely new algorithms for updating the electric potential and field at each time step, it was important that we determine the correctness and accuracy of these methods. The following sections report on some of the basic tests that were conducted to document the effectiveness of our implemented solution methods.

3.2.1 Examining Solution Convergence on Refined Meshes

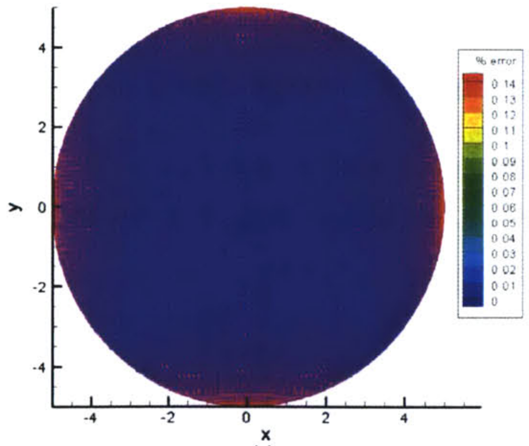
When examining a new solution technique, it is often useful to determine the benefit in accuracy which may be gleaned from further refinement of a mesh. To examine this convergence, we selected three different circular meshes of coarse, medium, and fine quality. The medium mesh is a simple two times refinement of the coarse, and likewise the fine mesh is a naïve two times refinement of the medium mesh. The normalized errors for each of these meshes calculated considering the analytic potential cos(πr) are shown below in Figure 3-12. Figure 3-13 shows a logarithmic plot of the average errors as the mesh is refined. Clearly we can see that refinement does produce the expected reduction in error.



(a)



(b)



(c)

Figure 3-12: The normalized errors for each of the three successively refined meshes. (a) The coarse mesh has half as many elements as (b) the medium mesh and a quarter as many elements as (c) the fine mesh. All three color scales are the same allowing for direct comparison.

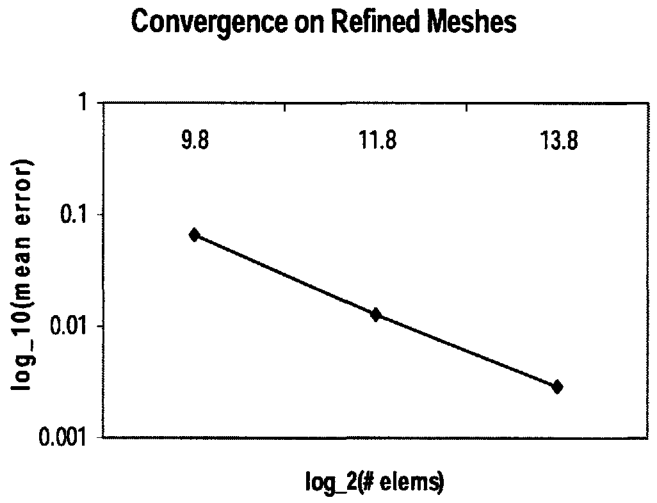


Figure 3-13: Average errors are plotted versus the base 2 log of the number of elements. The error exhibits a roughly linear trend denoting an acceptable convergence rate.

3.2.2 Patching the Spline Method for Adapted Grids

Having successfully created and tested the spline potential solver for non-uniform meshes, our next step was to apply this technique to adapted RRC meshes. Unfortunately, we ran up against a significant difficulty due to one of the drawbacks of the spline-fitting method as described in Section 2.7.1.2. Interpolating a value to a point to approximate the electric field or solving the Gauss's Law equation both assume that the **A** matrix of spline coefficients featured in Equation *2.45* is non-singular. If this matrix becomes theoretically singular or even numerically singular, the spline coefficients cannot accurately be calculated and large errors may arise.

To combat this problem, we were forced to resort to a simpler, less accurate interpolation type for facets whose spline **A** matrix appeared singular. To perform this task, we proceeded by first calculating the spline **A** matrix and attempting to invert it. If the condition number of this matrix was found to be less than some threshold, usually $1*10^{-10}$, a numerically singular matrix was assumed. In this case, a modified linear interpolation was instead used across the facet.

The method used for the linear interpolation is relatively simple but proportionately less accurate when compared to the spline interpolation. Basically, given a facet which forms the boundary between cell i and cell j, we still need to approximate:

$$\oint \vec{E} \cdot ds \tag{3.4}$$

which is just the flux of electric field normal to the facet. So we approximate this electric field by:

$$\vec{E} \cdot ds = (-\nabla\Phi) \cdot ds \approx \frac{\Delta\Phi}{\Delta \underline{x}} = \frac{(\Phi_i - \Phi_j)}{\sqrt{\left[(x_i - x_j)^2 + (y_i - y_j)^2\right]}} \cos\theta \tag{3.5}$$

where the angle θ is defined as the angle between the midpoint of the facet and the point where the segment ($\underline{x}_i$, $\underline{x}_j$) crosses the facet. This angle is depicted below in Figure 3-14.

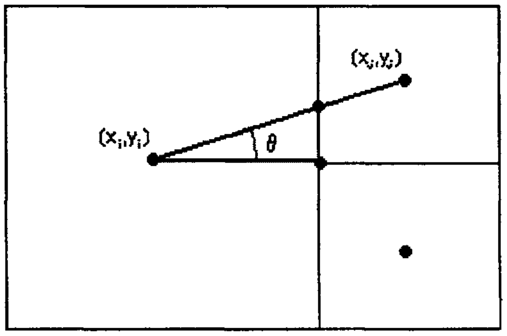


Figure 3-14: The definition of θ accounts for the fact that our linear estimate of the potential gradient is not actually normal to the facet.

3.2.3 Testing the Patched Spline Method on Adapted Grids

In order to test the effectiveness of the above mixed approximation method for adapted grids, we decided to use a common Poisson's Equation Benchmark, the uniformly-charged hollow cylinder. A uniformly-charged, infinite-in-z-direction cylinder with inner radius $R_{\min}$ and outer $R_{\max}$ creates a 3-region field as shown in Figure 3-15. If charge density is ρ_0, and the boundary condition $\Phi_{r\to 0} \to \Phi_0$, then the amplitude of the static electric field is given by the following expressions:

$$E = \begin{cases} 0 & , \ r < R_{\min} \\ E_{\max} \dfrac{r - R_{\min}}{R_{\max} - R_{\min}} & , \ r \in [R_{\min}, R_{\max}] \\ E_{\max} \dfrac{R_{\max}}{r} & , \ r > R_{\max} \end{cases} \tag{3.6}$$

The magnitude of the maximal electric field $E_{\max}$ is obtained by applying Gauss's law to the field divergence equation $\vec{\nabla}\vec{E} = 4\pi\rho$ (in Gaussian, CGS units) to a circular surface coaxial with the cylinder:

$$E_{\max} = \frac{2\pi\rho_0}{R_{\max}}(R_{\max}^2 - R_{\min}^2) \tag{3.7}$$

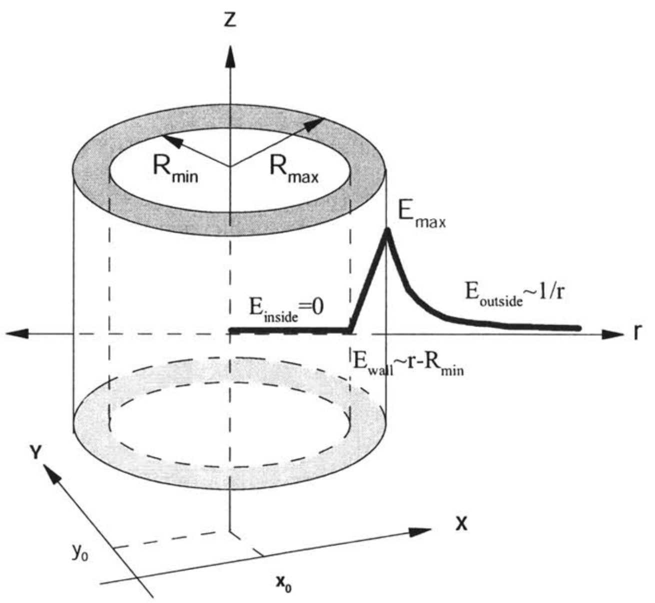


Figure 3-15: Electric field created by uniformly-charged hollow cylinder

The electric potential $\Phi(r)$ is obtained by integrating equation for the radial component of the gradient in cylindrical coordinates:

$$E_r = -(\nabla\Phi)_r = -\frac{\partial\Phi}{\partial r} \tag{3.8}$$

which yields the following explicit expression:

$$\Phi = \Phi_0 - \int_0^r E_r dr \tag{3.9}$$

For the electric field given by equation *3.6* the electric potential is:

$$\Phi = \begin{cases} \Phi_0 & , \; r < R_{\min} \\ \Phi_0 - E_{\max}\dfrac{0.5\,(r - R_{\min})^2}{R_{\max} - R_{\min}} & , \; r \in [R_{\min}, R_{\max}] \\ \Phi_0 - E_{\max}(\,0.5\,(R_{\max} - R_{\min}) + R_{\max}\ln\dfrac{r}{R_{\max}}) & , \; r > R_{\max} \end{cases} \tag{3.10}$$

Given this framework, we examined the convergence rate of our mixed-approximation method on successively refined meshes using the MUMPS direct matrix solver. A mesh of perfectly diamond-shaped elements, it turns out, leads to splines which all have singular matrices. For example, using equation 2.45 of the previous chapter we can calculate the spline matrix, A, for the sample mesh in Figure 3-16. This perfectly regular diamond mesh with distances of 1 unit between adjacent cell centers yields the spline matrix:

$$A_{diamond} = \begin{bmatrix} -1 & 1 & 1 & 1 & -1 \\ -\sqrt{2} & 0 & 2 & 0 & 0 \\ -1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & \sqrt{2} & 0 & 2 & 0 \end{bmatrix} \tag{3.11}$$

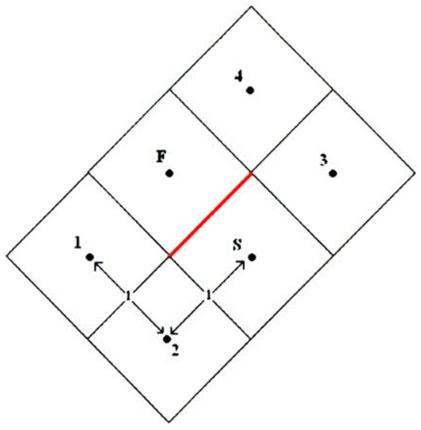


Figure 3-16: A mesh which yields singular spline matrices. The mesh is meant to be completely regular with diamond shaped elements, leading to the spline matrix shown in Equation *3.11*.

The determinant of this matrix can easily be calculated and is exactly zero. Since many of the nodes are collinear, the information contained within them is degenerate and does not allow the full second order spline to be calculated. Knowing this, we chose to really highlight the difficulties associated with the spline approximation method by computing the charged-cylinder benchmark on the mesh shown in Figure 3-17. The elements of this base mesh are not exactly diamonds, but are numerically very close to it. As such, the condition numbers of the spline matrices calculated for this mesh are extremely small, producing significant error in the spline method. In addition, as the mesh is further and further refined, the matrix condition numbers grow progressively smaller as the elements attain even greater numerical resemblance to regular diamonds. Therefore, we expect that as the mesh is refined, many of the elements near the center of the mesh will be forced to switch from the second-order accurate spline method to the simpler, first-order accurate linear interpolation mentioned above. This switch will tend to initially increase the error in the regions that require the less accurate method, but will prevent the poor matrix condition numbers from introducing gibberish and high levels of numerical noise into the solver. Clearly, this patched method is not the most desirable of solutions, but it is a viable one which allowed us to simulate even the most difficult of mesh structures with a suitable degree of accuracy. Of course, it should be noted that in practice, the meshes used for simulations can be constructed with the solver's limitations in mind and for the most part avoid the hassles associated with these highly unusual, irregular meshes.

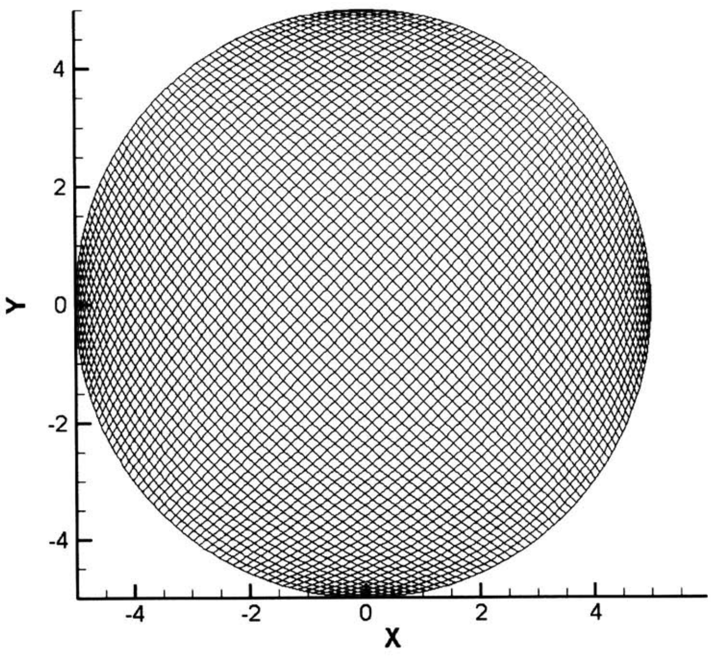
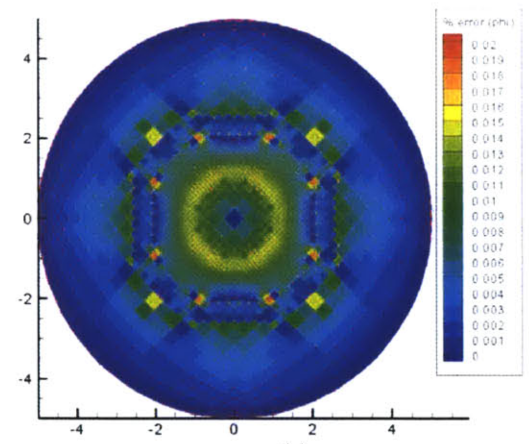


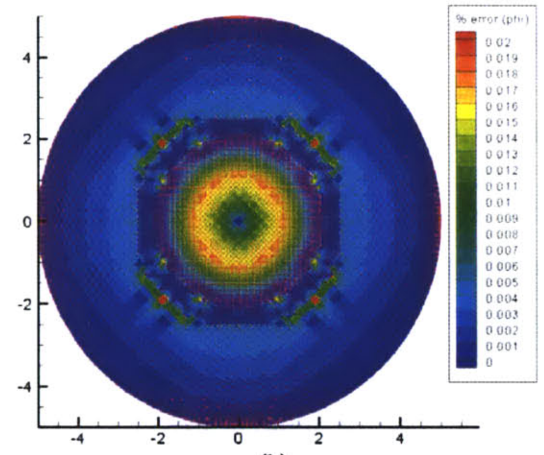
Figure 3-17: The base mesh used for the charged-cylinder, patched spline testing.

The test results of the charged cylinder benchmark now follow. Each mesh featured a two-level cylindrical adaptation which covered the region between Rmin and Rmax. The boundary conditions on the edge of the mesh were Dirichlet, that is the potential at those points was set to the analytical potential at that particular location. The error in the electric potential for these tests is given in Figure 3-18. The regions of highest error generally tend to be those on the boundaries of the adaptation. The error also increases near the center of the highly-refined mesh since, as discussed the elements there are using the less accurate first-order calculation method. The average errors over each entire mesh are plotted in loglog format in Figure 3-19. The trend indicates that even though locally the errors may be increased by switching to the less accurate scheme, the reduction in error granted by refinement in the regions of non-degenerate elements is on average more significant for this particular geometry and the error is convergent.

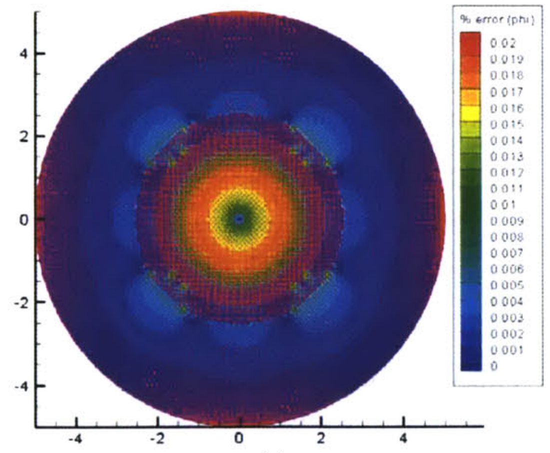
The patched spline scheme implemented in the simulation must be used with some caution. On most meshes, the spline matrices will be well-conditioned and few problems should arise. However, as we have demonstrated, some geometries may require a modified solution scheme with a possible corresponding reduction in accuracy.



(a)



(b)



(c)

Figure 3-18: The normalized error between the analytic potential and calculated potential for three successively refined meshes. (a) The coarse mesh has one half as many elements as (b) the medium mesh and one quarter as many elements as (c) the refined mesh. Note that the color scale is the same on all three meshes to enable direct comparison.

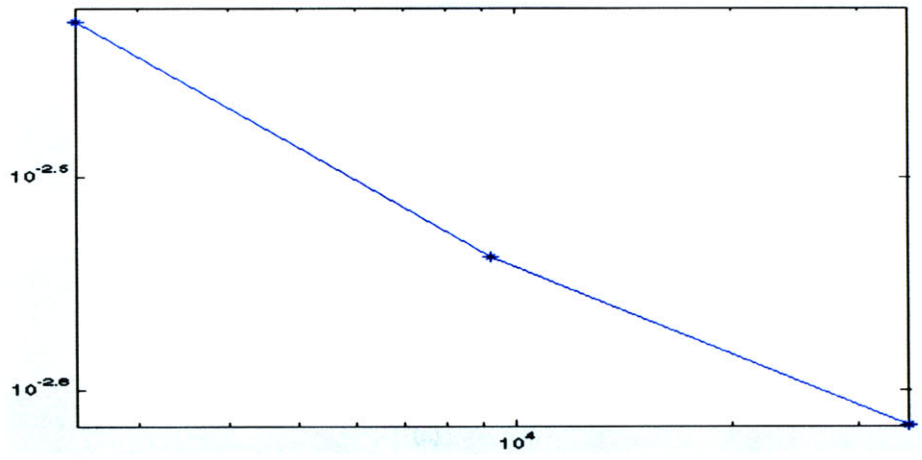
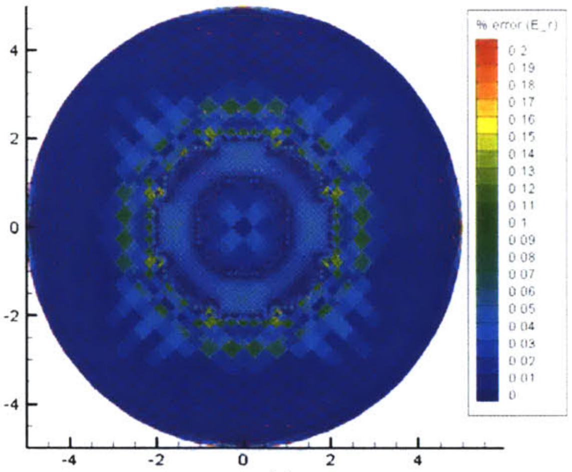


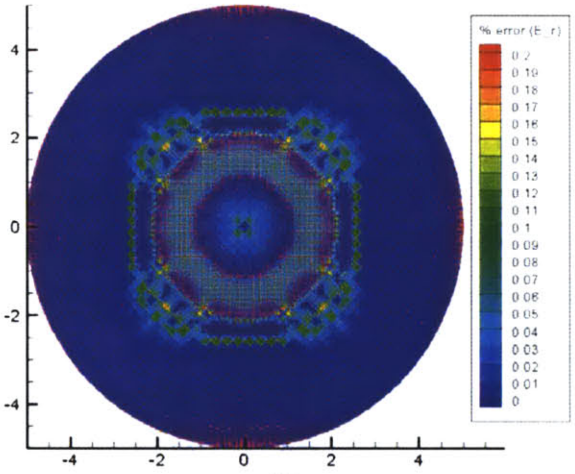
Figure 3-19: The average normalized errors for these three meshes plotted versus the number of elements in the mesh on a log-log scale. The convergence is clear.

3.2.4 Testing the Calculation of Electric Field from the Given Potential

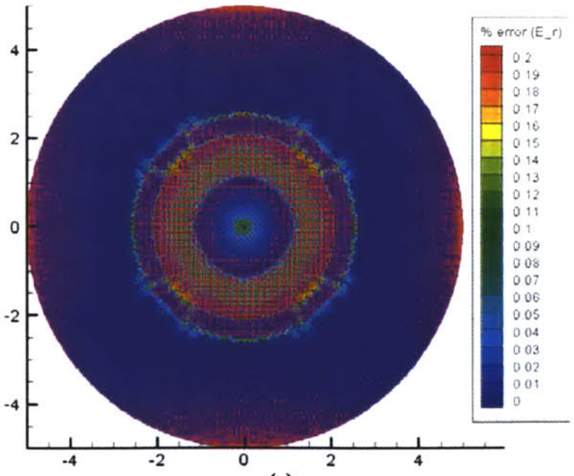
The method described in Section 2.7.3 for approximating the gradient of the potential and thus the electric field was tested on the hollow cylinder benchmark discussed in Section 3.2.3. The error between that analytical solution and the calculated radial electric field is plotted in Figure 3-20 for three successively-refined meshes. In Figure 3-21, the average error for each of these three meshes is plotted versus the number of nodes in the mesh. Due to the necessity of reverting to first order approximations for the electric field, our errors are noticeably larger than those for the electric potential. The maximum errors in the electric field occur generally in the same regions in which the electric potential is inaccurate. Thus, it seems that the method for calculating electric fields is accurate, but only as accurate as the electric potential that is fed to it. It might also be noted that the resolution of the printed images shown here may not be perfect and the reddish cloud which has sometimes been seen to occur in Figure 3-20(c) on paper is actually mostly due to the printing of the mesh which was overlayed with these results to indicate the increasing refinement and the regions of RRC adaptation.



(a)



(b)



(c)

Figure 3-20: The normalized error between the analytic and calculated radial electric field for three successively refined grids. Note that all three plots have the same color scale.

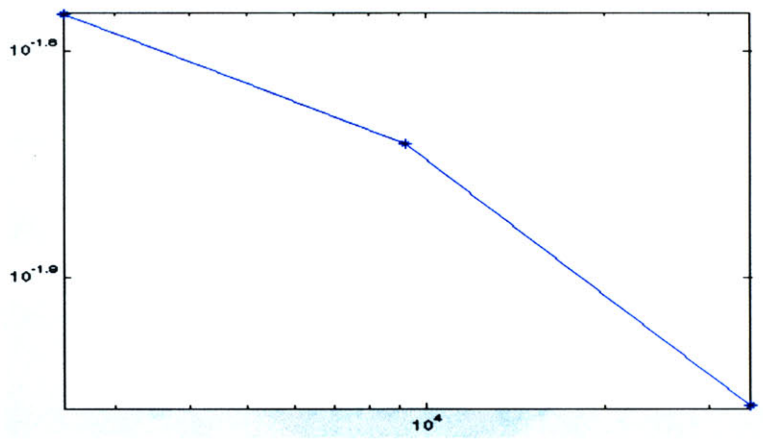


Figure 3-21: A loglog plot of the errors on each of the refined meshes showing a generally covergent trend.

3.2.5 Testing the Axisymmetric Patched Spline Method

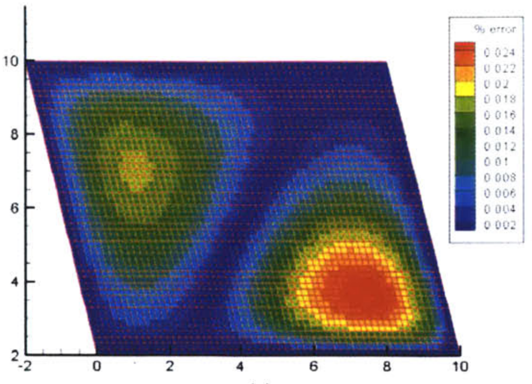
Some further sample simulation results are shown below in Figure 3-22, now for a less problematic mesh. The shape of this mesh was chosen to demonstrate that typical non-orthogonal elements can certainly be handled by the solver with high accuracy. The potential used in these trials was the simple:

$$\Phi = r * z \tag{3.12}$$

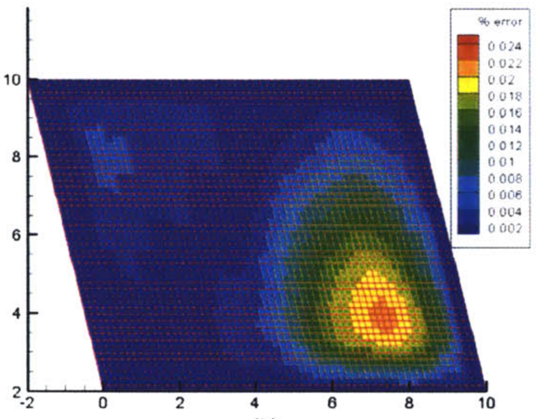
Thus, the analytical charge distribution which was given to the potential solver was:

$$\nabla^2\Phi = q = z / r \tag{3.13}$$

Figure 3-22(a) shows the percent error in the computed potential when only linear interpolation was used for approximation. Figure 3-22(b) shows that this error is dramatically reduced when the combined method of splines and linear interpolation is employed. The color scales in both figures were kept the same, to provide easier direct comparison. In this case, the spline matrices remained well-conditioned allowing the spline method to be used throughout the mesh, greatly increasing the accuracy of the solution over the first-order accurate scheme.



(a)



(b)

Figure 3-22: (a) The normalized error between the analytic and calculated potentials for an axisymmetric trial using only linear interpolation. (b) The error from a trial which employed the spline approximation in elements where the matrix was non-singular and linear interpolation in the remaining elements. The vertical axis is the r-direction, the horizontal axis is the z-direction and the symmetry axis is assumed to be horizontal along r=0.

3.2.6 Comparing the Iterative and Direct Solution Techniques

The method of solving the resulting matrix equation also plays a significant role on the accuracy and the efficiency of the potential calculation. Preliminary testing with the MUMPS direct solution technique shows it to be much faster than the iterative SOR technique for the problem sizes we are currently investigating. Where the iterative mehod may require as much as thirty seconds to converge, the MUMPS solution can be returned in a matter of milliseconds. Even when the SOR scheme is allowed such a long time to converge, the level of accuracy it can attain is much less than the direct solver, as indicated by Figures 3-24 and 3-26. One of the significant drawbacks to the MUMPS solver which forced us to revert to SOR in the final versions of the code was the amount of memory it required. Actually constructing and storing the entire matrix equation and attempting to factor it stretched the memory capacities of our machines with any significantly adapted mesh. Also, it was found that the version of MUMPS we were using, version 4.3.2, exhibited a minor memory leak which, after the course of approximately 10^5 iterations of the Near Vacuum Hall Thruster tests for example, caused the program to overflow.

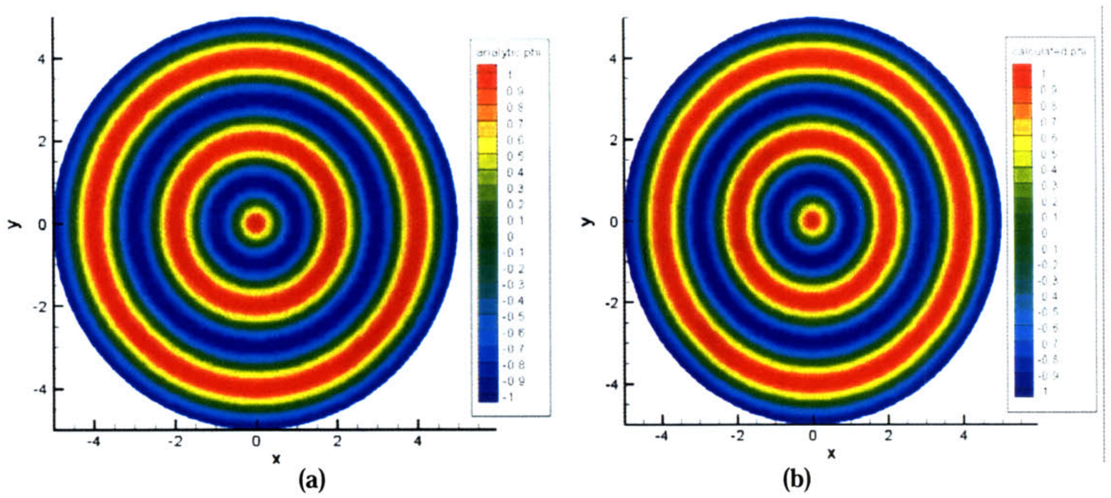


Figure 3-23: (a) The analytic potential cos(πr) compared to (b) the calculated potential with the boundaries set to this analytic potential.

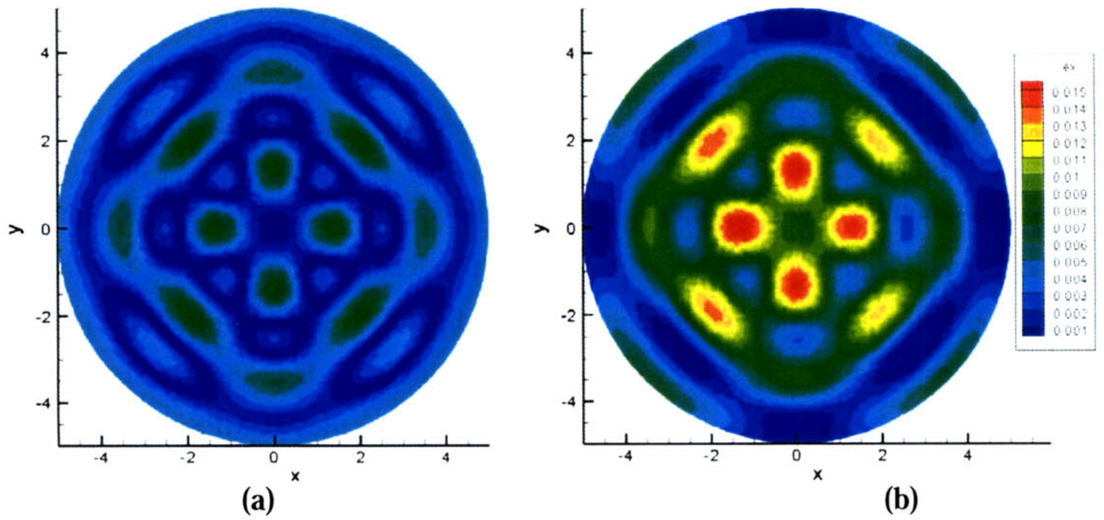


Figure 3-24: (a) The normalized error calculated by comparing the analytical potential cos(πr) to the value calculated by the direct solution technique using MUMPS. (b) The same for the iterative SOR solver. Note that both (a) and (b) have the same color scale.

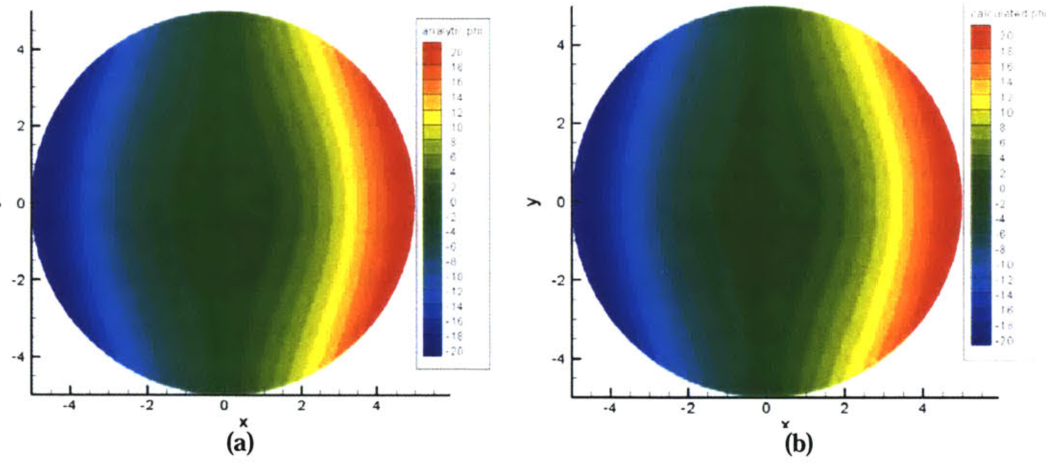


Figure 3-25: (a) The analytic potential $r^2\cos(\theta)$ compared to (b) the calculated potential with this analytic potential on the boundary.

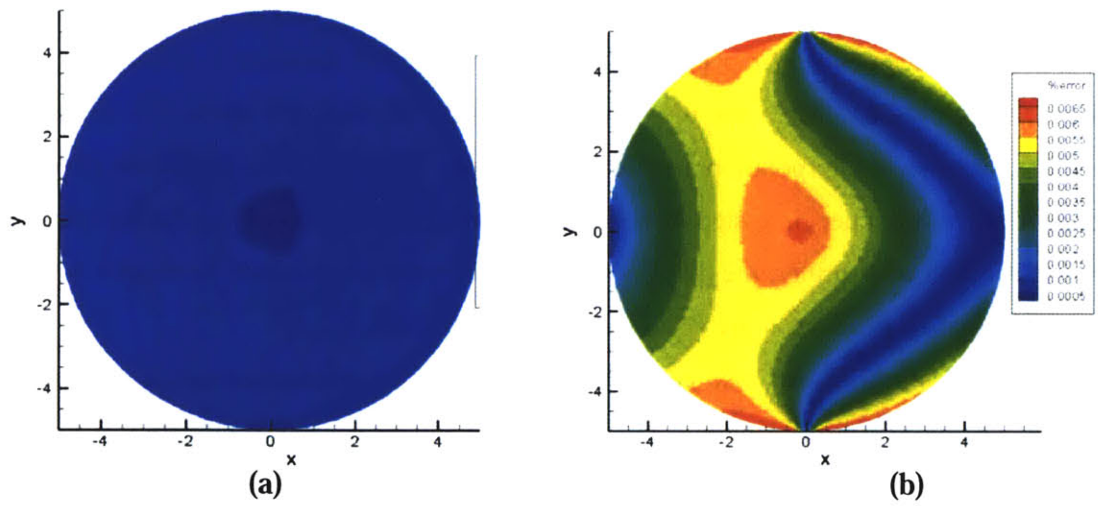


Figure 3-26: (a) The normalized error calculated by comparing the analytical potential $r^2\cos(\theta)$ to the value calculated by the direct solution technique using MUMPS. (b) The same for the iterative SOR solver. Note that both (a) and (b) have the same color scale.

3.3 Self-Consistent Field Calculation Tests

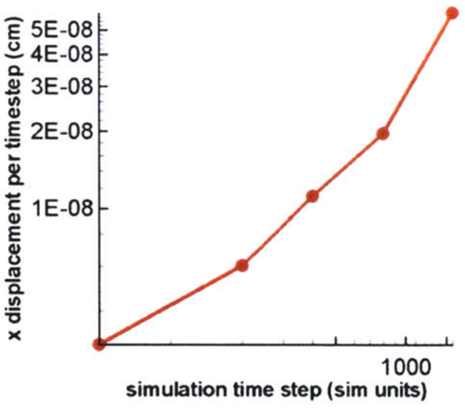
Given that our particle motion calculator is accurately devised and implemented and we have the capability of solving Poisson's equation on our RRC non-orthogonal meshes, the next logical step is to combine these two portions of the code. The next sections discuss several different types of tests which combined these two modules, the real backbones of the simulation.

3.3.1 Self-Consistent Single Particle Motion Test

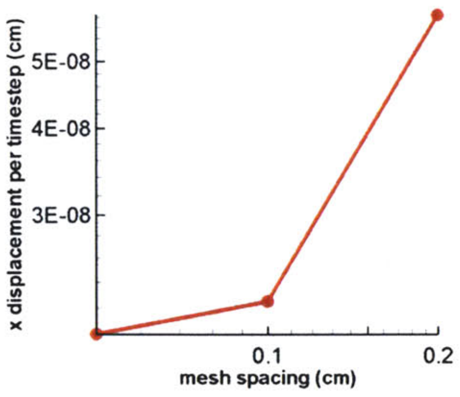
We first conducted a test whereby a single electron was placed near the center of a square, two-dimensional orthogonal mesh. This electron was given a velocity in the y-direction of .9c. The boundary conditions were simply set to 0 Volts on the edges of the simulation under the assumption that the potential of this single charge should be negligible as it neared the boundary. This boundary condition does in itself introduce a small measure of error in the particle's trajectory if the particle does not remain centered in the domain, but the effect should be small and shall be ignored hereafter.

The measurement of interest was the x-displacement of the electron at the moment it reached the simulation boundary. In theory, this displacement should have been 0. However, due to errors in the interpolation of the electric field and in its calculation, some amount of error was introduced. Figure 3-27(a) below quantifies that absolute displacement error per time step as a function of decreasing simulation time step. It is clear that decreasing the time step can help alleviate the errors introduced by the inaccuracies in interpolation and field calculation. Figure 3-27(b) shows the absolute displacement error per time step now as a function of the grid-spacing. As the mesh is refined further and further, the displacement error is reduced as would be expected since the interpolation becomes more accurate at finer mesh sizes.

The same trial was also conducted on a simple square axisymmetric mesh with similar results. Now the single electron can be imagined to be a single ring of charge which is being revolved around the axis of symmetry. Figure 3-28(a)-(d) depicts the calculate potential for this RZ case at different times during the simulation. Figure 3-29(a)-(d) depicts the norm of the electric field at these times as well. Note that where the point charge exists, the potential should of course be infinite. This singularity is smoothed by the interpolation and spreading of the charge between the containing cell and its neighbors. As such, the potential does not exactly match the theoretical $q/(4\pi\varepsilon_0 R)$ point charge potential, but in general our plasmas will be dense enough that this is not a consideration. Figure 3-28 and 3-29 demonstrate by their asymmetry toward the bottom mesh edge that the "point charge" is in this case actually being considered as a ring of charge, so the potential near the centerline would naturally be greater than that outside of the ring.

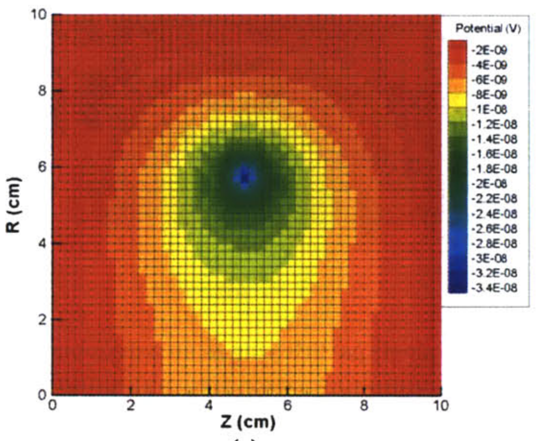


(a)

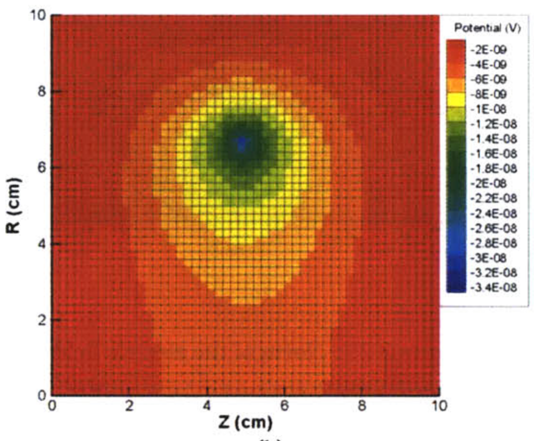


(b)

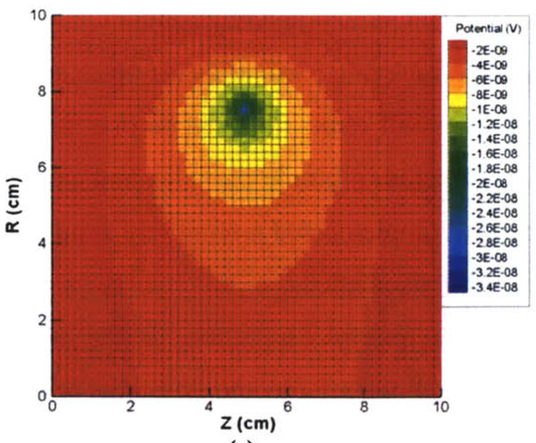
Figure 3-27: (a) The error in x-motion is small and can be arbitrarily reduced by reducing the simulation time step. (b) The error in x-motion can also be arbitrarily reduced by reducing the mesh spacing.



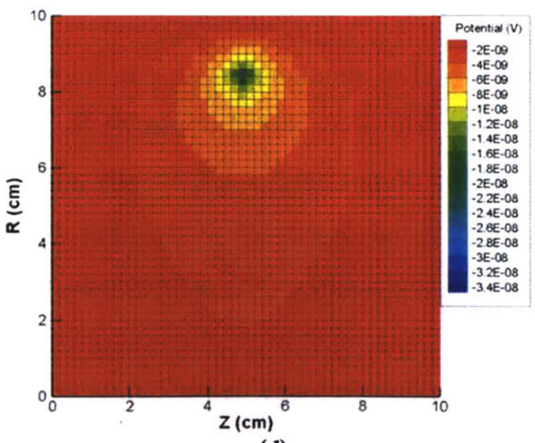
(a)



(b)



(c)



(d)

Figure 3-28: Electric Potential over time for a single particle ring in axisymmetric geometry.

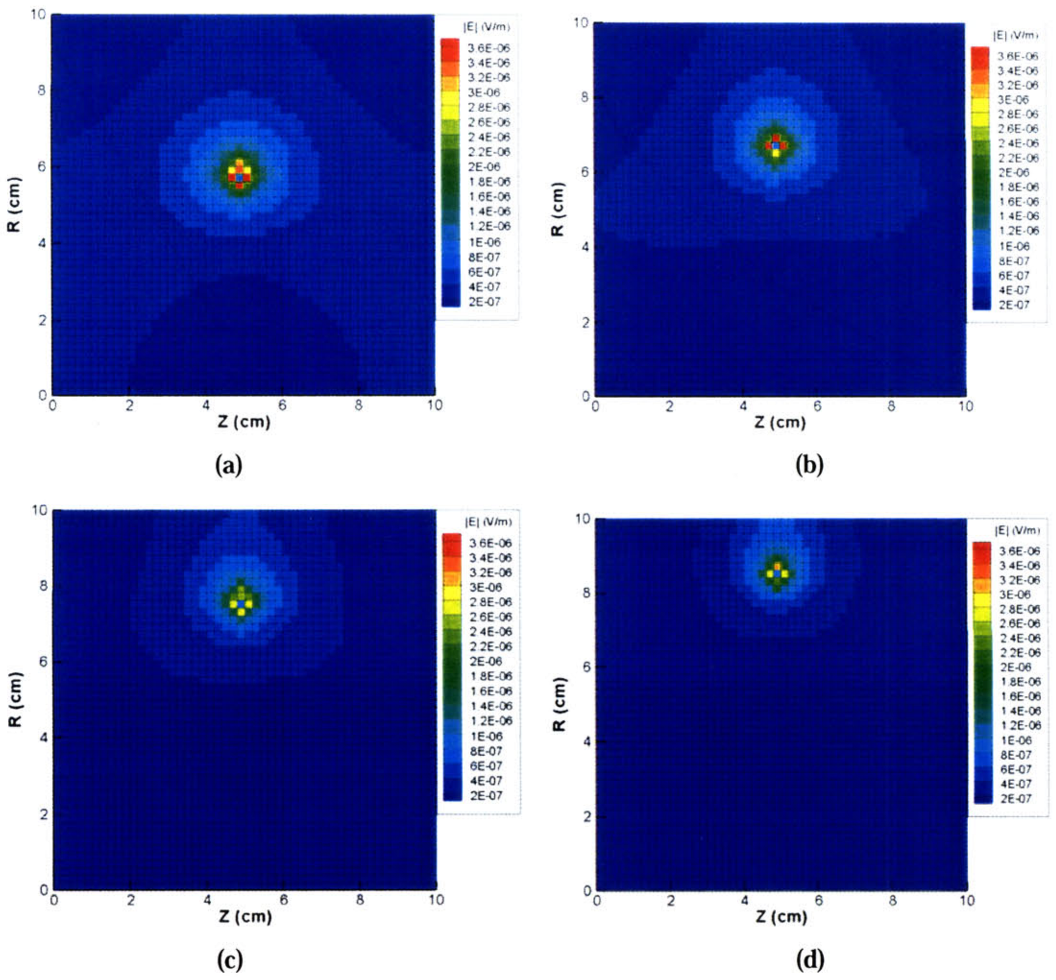


Figure 3-29: Norm of the electric field over time for a single particle ring in an axisymmetric geometry.

3.3.2 2D Langmuir Oscillation Tests

The simulation is able to fairly accurately track one particle as it moves across a mesh. However, we also need to be sure that the small errors incurred on one particle's motion do not have a disastrously additive effect across multiple particles and cause the entire simulation to act wildly. For this reason, we implemented a relatively simple real-world benchmark problem known as cold Langmuir oscillations.

A Langmuir oscillation, also known as a plasma oscillation, is the periodic oscillation of charge density in an otherwise neutral plasma. In our case, we created a plasma in the center of our simplest two-dimensional square, orthogonal mesh. For this trial, the boundary conditions were set to be periodic in both the x and y directions. This plasma contained positively charged particles with a very large individual mass (10^8 electron masses) and negatively charged electrons, each species having density of 1 simulation density unit. The particles were placed roughly in a circle centered at (5,5) on the mesh with a radius of 1 cm. They were initially at rest. However, they were initially placed such that for each positive particle, a negative particle was also placed, only adjusted outward 300 μm radially. This displacement caused a restoring force which initiated a plasma oscillation at the theoretical frequency:

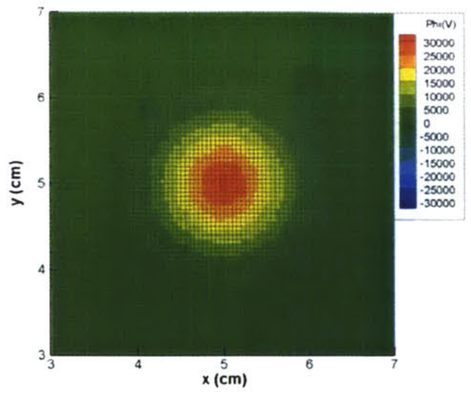
$$\omega_{pe} = \sqrt{\frac{ne^2}{m\varepsilon_0}} \tag{3.14}$$

where n is the electron density, e is the fundamental charge, m is the electron mass, and ε_0 is the permittivity of free space. As described above, all of these quantities were purposely assigned to a value of 1 for our test, except the uncontrollable ε_0. It can be calculated that in terms of the simulation units, $\varepsilon_0 = 2.829*10^7$ yielding us a plasma period of approximately 33420 simulation units.

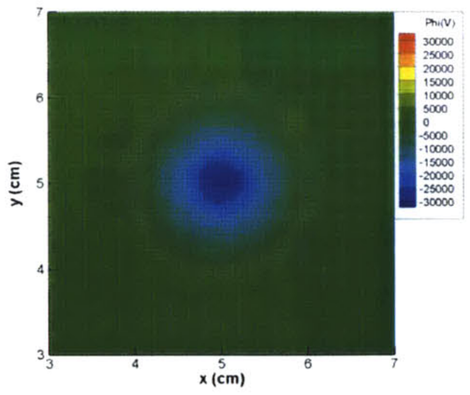
To measure our simulation's adherence to this physical quantity, we chose to measure the potential and kinetic energy. The kinetic energy is just given by ½mv^2 summed over the particles while the potential energy can be described by:

$$\int\left[\frac{1}{2}\varepsilon_0|E|^2\right]dV \tag{3.15}$$

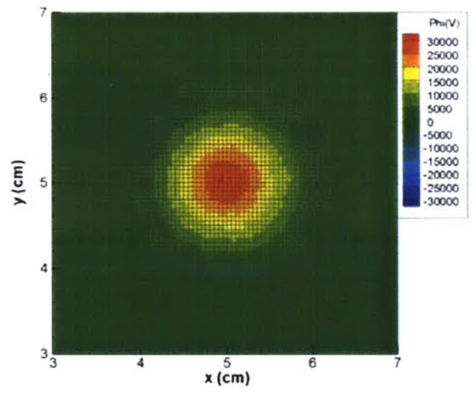
which in our case reduces to a sum of the electric field's squared norm over the elements. Figure 3-30 above shows the electric potential in our test at four times each separated by half the plasma period. This illustrates that the electric potential varies at the same rate as the plasma density, as expected, and moves from maximum to maximum over the course of one plasma period. Figure 3-31 is also a plot of the potential at times separated by one half of the plasma period, only now the mesh refinement has been increased to four levels rather than just two. These images show that the potential becomes much smoother with refinement ensuring more accurate interpolation and field calculations, reducing the error in the overall simulation.



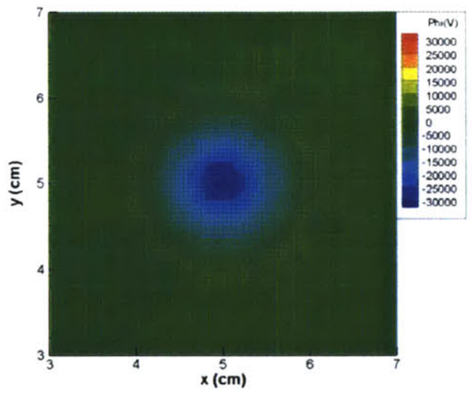
(a)



(b)



(c)



(d)

Figure 3-30: The potential at peak times separated by 16,750 simulation time steps, or about ½ of a plasma period.

The kinetic and potential energies on the other hand, have a period of oscillation equal to ½ of the plasma period, or 16710 simulation time steps. This half-period is apparent when we examine Figure 3-32, a plot of the kinetic, potential, and total energies over time for a simulation having two levels of refinement (as depicted in Figure 3-30). As the plot has been normalized by the theoretical plasma period, we see clearly that the simulation energies oscillate at almost exactly the frequency predicted by theory. Also, we see that the total energy roughly remains constant, for only a slight increase of less than 3% is observed over two full plasma periods. This simulation was run at three successively higher levels of mesh refinement and the relative error in the calculated plasma period versus the minimum grid spacing is plotted below in Figure 3-33. This figure demonstrates that as we increase the mesh refinement, we are able to increase the

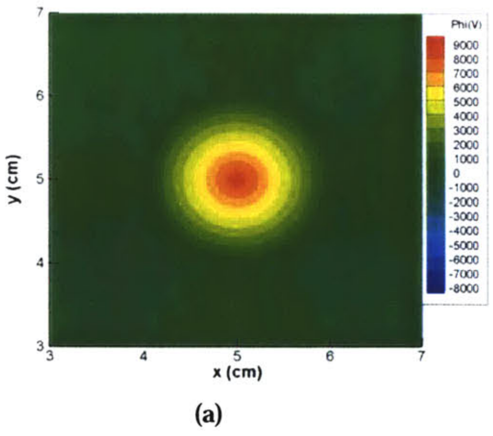
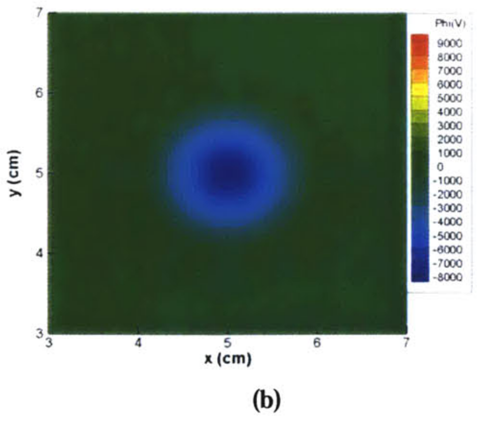
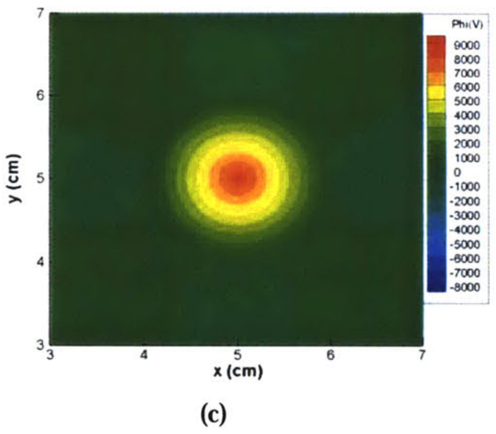
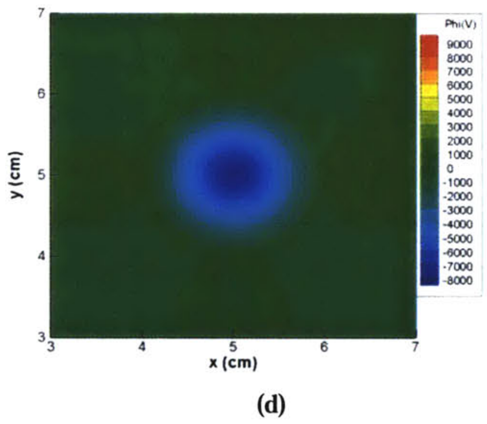





Figure 3-31: The potential of a trial with a maximum refinement of four levels. Notice that it is of course much smoother than the potential calculated with only two refinement levels. This smoothness leads to better interpolation and less error. The mesh in these images is not shown since its fineness would obscure the image.

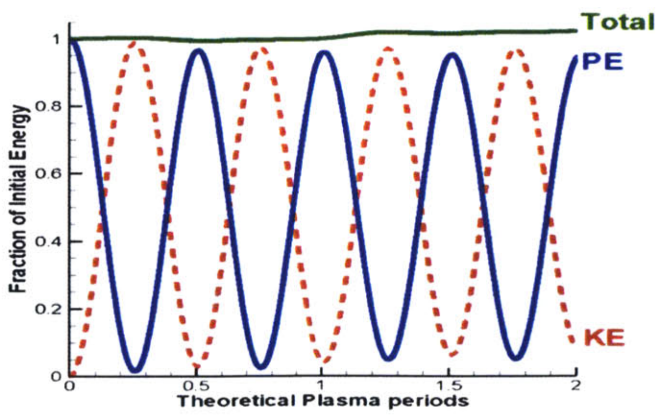


Figure 3-32: The simulation energies oscillate at the theoretical frequency, and the total energy increases only slowly over time.

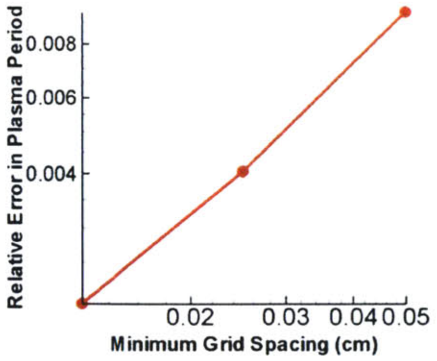


Figure 3-33: As the grid spacing is reduced, the plasma period becomes more accurate.

accuracy of the simulation arbitrarily. Given enough computational time and effort, we can produce results of arbitrary accuracy if we desire.

Having demonstrated that the two-dimensional solver and integrator are accurate on orthogonal meshes, we also wanted to test their accuracy on a non-orthogonal mesh such as the one shown below in Figure 3-34. The same set-up was used in this test as in those above. The test was conducted at a maximum refinement level of two. The energy results of this test are plotted below in Figure 3-35. The relative error between the calculated plasma period and the theoretical was approximately .0044, not significantly different from the error associated with an orthogonal mesh of similar refinement. The total energy also remained nearly constant, increasing less than 3% over two plasma periods, a rate comparable to the orthogonal trials as well.

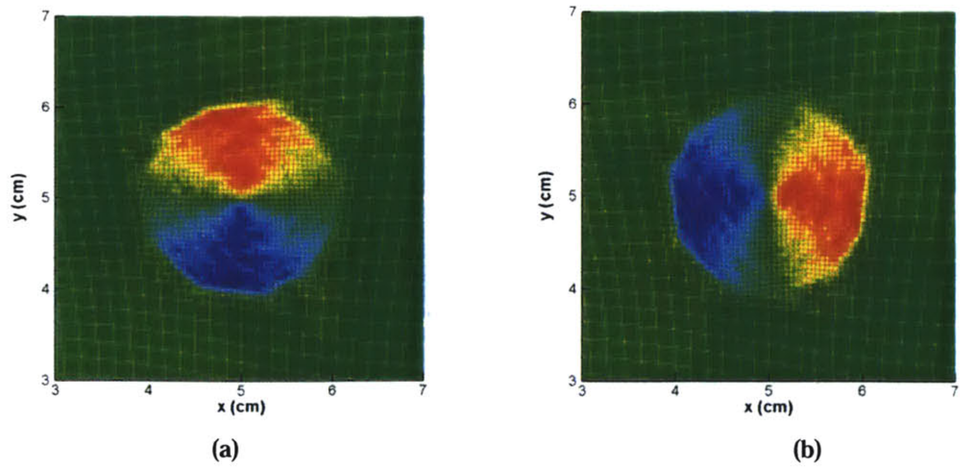


Figure 3-34: (a) E_x and (b) E_y for the non-orthogonal Langmuir oscillation tests.

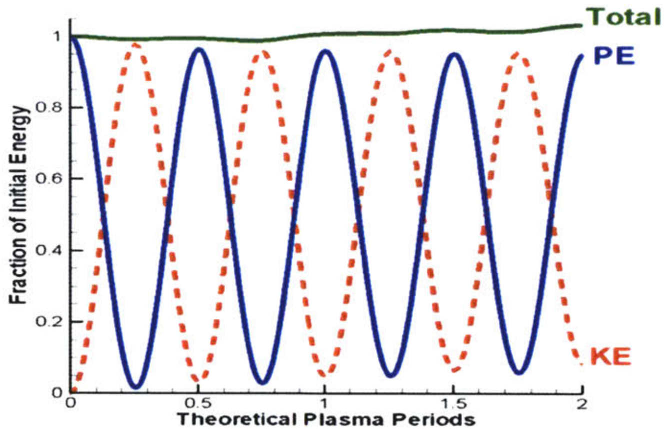


Figure 3-35: The total, kinetic, and potential energies over two plasma periods in the non-orthogonal two-dimensional Langmuir oscillation test.

3.3.3 RZ Langmuir Oscillation Tests

Similar to the test above, we now wish to test the particle integrator and field solvers together in an axisymmetric coordinate system. For this purpose, we created a simple system on the orthogonal, one-level-adapted mesh depicted in Figure 3-36 below.

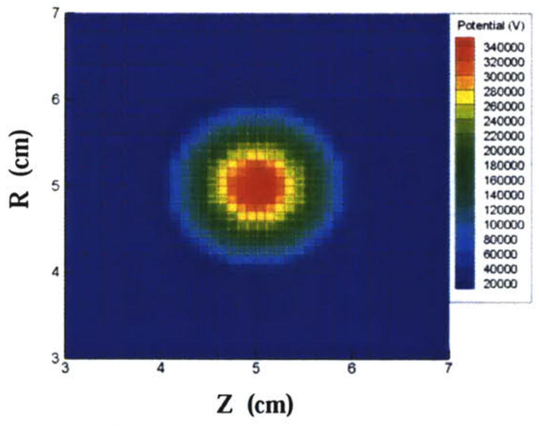
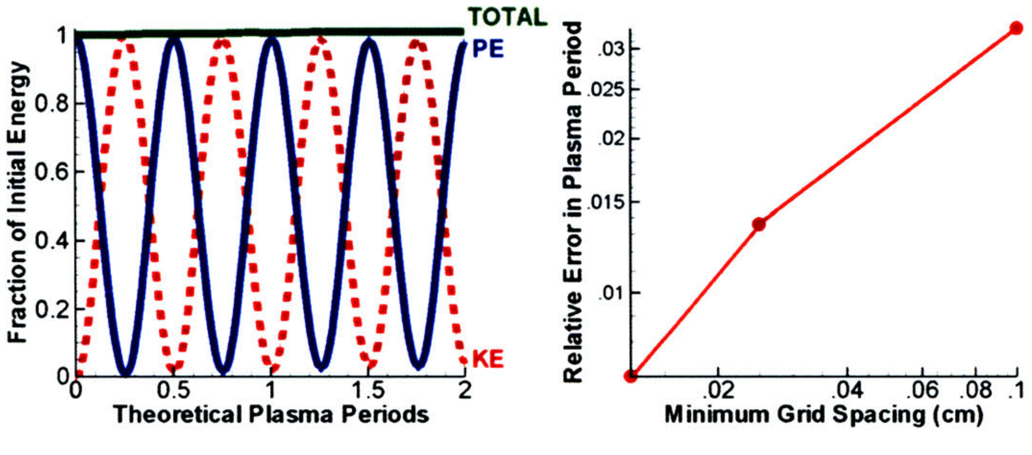


Figure 3-36: The initial potential for the RZ singly-adapted Langmuir oscillation test.

In this case, heavy (10^8 electron masses) positively-charged particles were again placed roughly in a circle centered at (5cm, 5cm) with a radius of 1cm. As a test that proper scaling is observable, the density of these particles was set to 4 in simulation density units. This should then increase the frequency of oscillations by a factor of 2, giving a theoretical plasma period of 16710 simulation units, or an energy oscillation period of 8355 simulation units. For each of the positive particles, an electron was also created at the same position, but shifted outward by 600 μm. All particles were given zero initial velocity. As before, the shifted position of the electrons created a restoring force that caused them to oscillate around the practically motionless positive particles at a frequency given by Equation *3.14* above. The boundary conditions for this trial were set to be periodic in the x-direction and the potential was held at 0V on the top boundary of the simulation. The bottom boundary of the simulation was, as usual, taken to be the centerline of revolution. The 0V boundary condition should not have appreciably affected the results since the potential for this system would in general be negligible at such a distance.

This system was allowed to oscillate and the potential and kinetic energies were again calculated. These, along with the total energy, are plotted in Figure 3-37(a) for the case of three levels of refinement. As before, the x-axis has been normalized to units of plasma frequency, and it is clear that the simulation accurately calculated the proper oscillation period. The energy in the simulation is again increasing slightly, rising by less than 2% over the two plasma periods for the three level-refinement case. Figure 3-37(b) demonstrates that as the refinement was increased, the error in the calculated plasma period was reduced.



(a) (b)

Figure 3-37: (a) The energy for a three-level refined RZ Langmuir oscillation test. (b) As the refinement was increased, the error in the calculated plasma period was reduced.

3.4 Neutral-Neutral Elastic Collision Testing

As described above in Section 2.8, our collisional techniques were designed virtually from scratch partially by choice and partly due to necessity. We chose to redesign them in the hopes of reducing the statistical noise inherent in Monte Carlo randomized collisions, but we would have been forced to adapt that technique quite severely anyway due to the fact that the particles in our simulation have various statistical weights. The following sections detail the testing we performed on the most basic of those collisional algorithms, the neutral-neutral elastic scattering module, in order to ensure that our implementation was indeed operating as we had anticipated and in agreement with theory.

3.4.1 Conservation Tests

Before any other tests of the neutral-neutral collision algorithm were undertaken, it was first necessary to ensure that the implementation accurately conserved the particle quantities required. The algorithm specified that the momenta, total temperature, and cell densities should not change during the collisional process. To test this, we placed a uniform density of neutrals in a simple box geometry, similar to the image shown in Section 3.4.2, Figure 3-39. We then allowed the particles to collide and examined the important quantities before and after the collision algorithm's operation. Figure 3-38 shows the relative change in these quantities plotted over the course of that simulation. All four quantities are conserved down to the level of machine accuracy, which is the best that could be expected considering that the collisional procedure does reorganize the particles and the order of operations on them. This reorganization results in the accumulation of different machine precision level floating point errors that are evidenced in the plots.

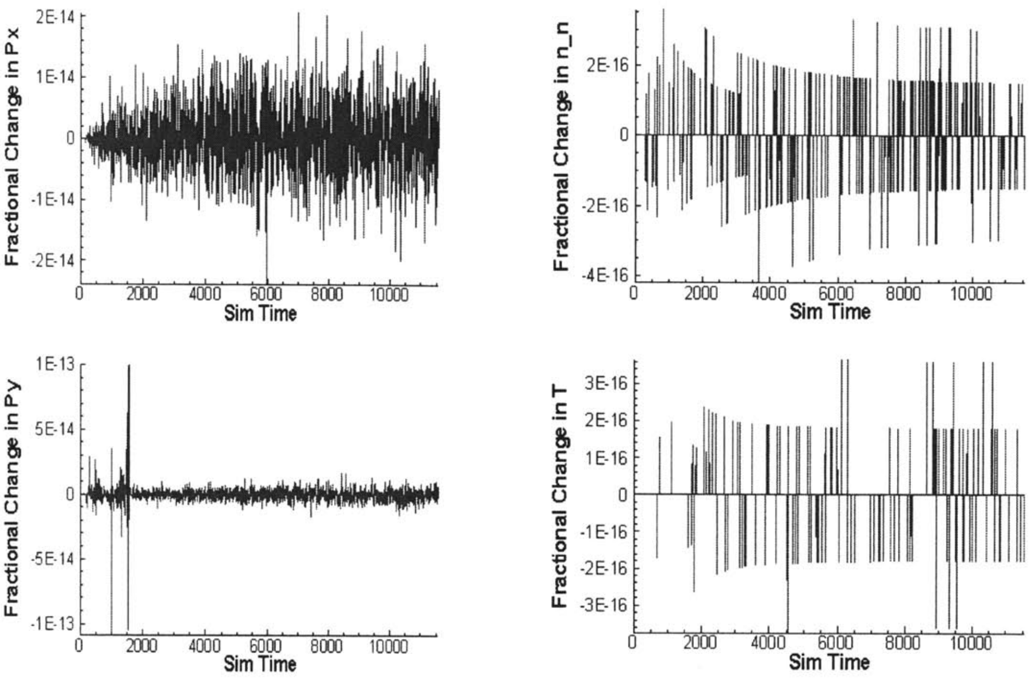


Figure 3-38: Conservation of quantities during the neutral-neutral collision test.

3.4.2 Neutrals in a 2D Box Test

Our next test of the neutral-neutral elastic collisions was performed to ensure that over a long enough period of time, a collisional distribution of neutrals would eventually approximate a Maxwellian distribution. To test this, we placed an initial neutral distribution of density .1 simulation units evenly spaced throughout the small two-dimensional 2cm by 2cm simulation domain pictured below in Figure 3-39. The particles were given an initially uniform random distribution, each particle having mass of 1 electron mass and initial x and y velocities uniformly chosen from the interval [-.00001c, .00001c]. The boundary conditions were set to be diffuse elastic walls, such that a particle impacting a boundary was reflected with the same energy, but a randomly-directed velocity. The particles were then released and allowed to collide with one another and the walls. In retrospect, it may have been better to perform the test with purely specular wall reflections so that the effects of wall collsions could have been entirely uncoupled from the particle-particle collisisons. However, the wall collisions did maintain the total energy of the particle while only randomizing its direction, and so while the isotropy of the particle-particle collisions might thus be affected, the decay to Maxwellian should not have been.

At any rate, resulting snapshots of the one-dimensional distribution functions overlaid with true Maxwellians for comparison are depicted in Figure 3-40, giving qualitative evidence that the distributions do approach Maxwellian. The relative L_1 error, computed by taking the sum of the difference between the calculated distribution and a Maxwellian at each of the velocity grid cells shown in Figure 3-42, is depicted over time in Figure 3-41. This shows the decay to Maxwellian somewhat quantitatively. However, since our method of collisions, as discussed in Section 2.8.1 above, does not guarantee the proper theoretical rate of relaxation, the axis of this plot is simply normalized by some arbitrarily selected t_f. If the rate of Maxwellianization does become important for a particular application, a scheme which is able to capture that physical property should be implemented as previously mentioned. To ensure that the distribution is not being created with a preferential direction, Figure 3-42 plots snapshots of the two-dimensional distribution function over time, showing its symmetry as it moves from a nearly flat uniform distribution to the Maxwellian.

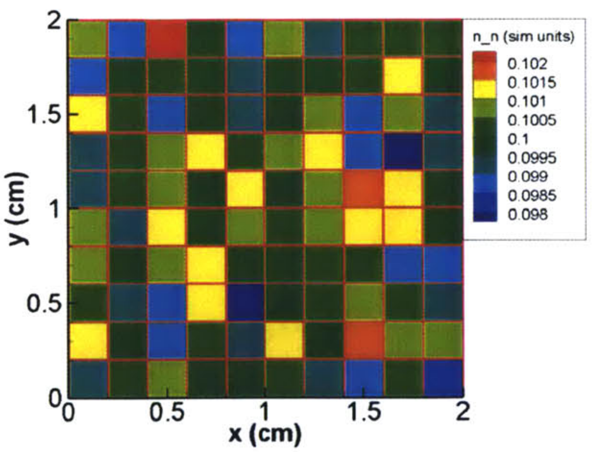


Figure 3-39: The grid for the neutrals in a 2D box test was a simple square. The density was roughly uniform at .1 simulation units.

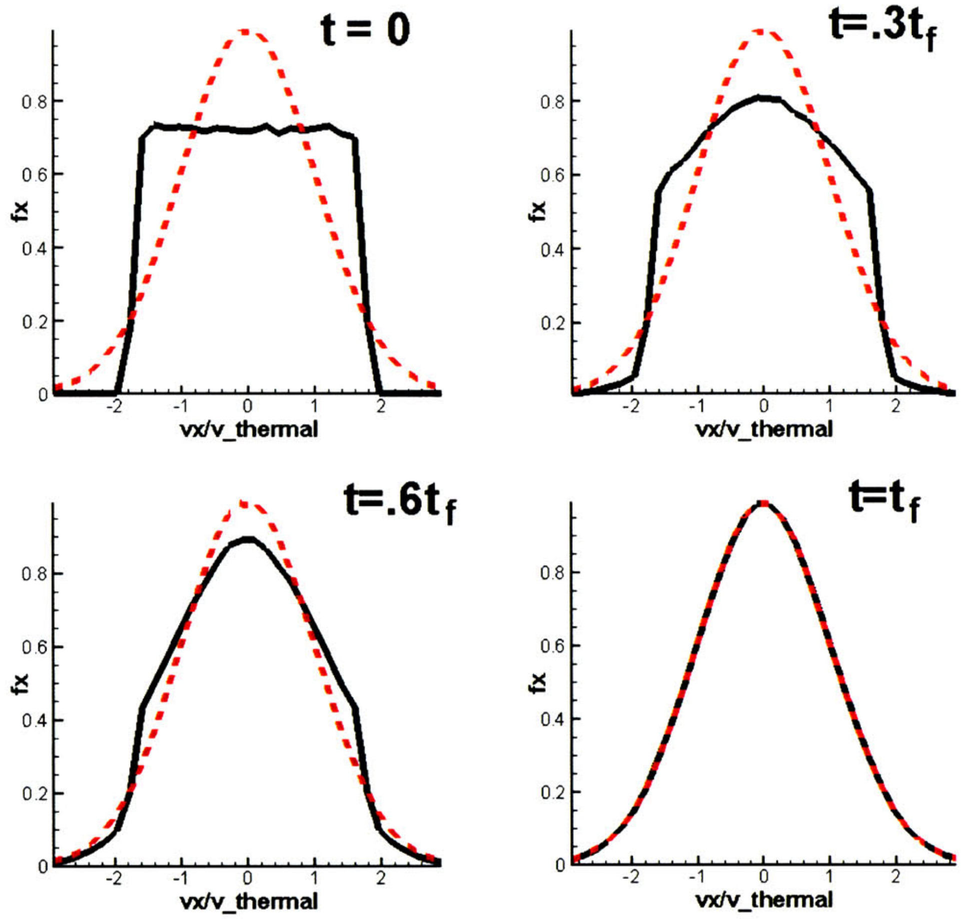


Figure 3-40: The x-distribution function overlayed with a Maxwellian distribution (dotted) as a function of time.

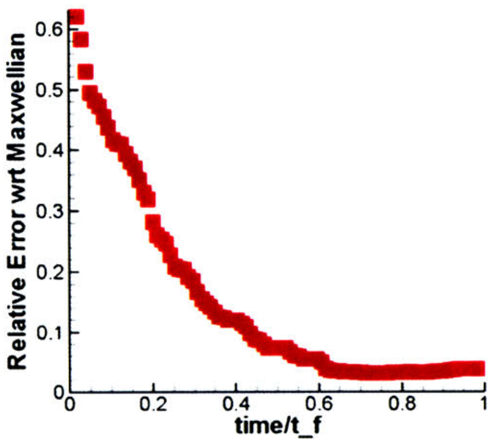


Figure 3-41: The relative L_1 error between the neutrals' distribution function and a true Maxwellian over time.

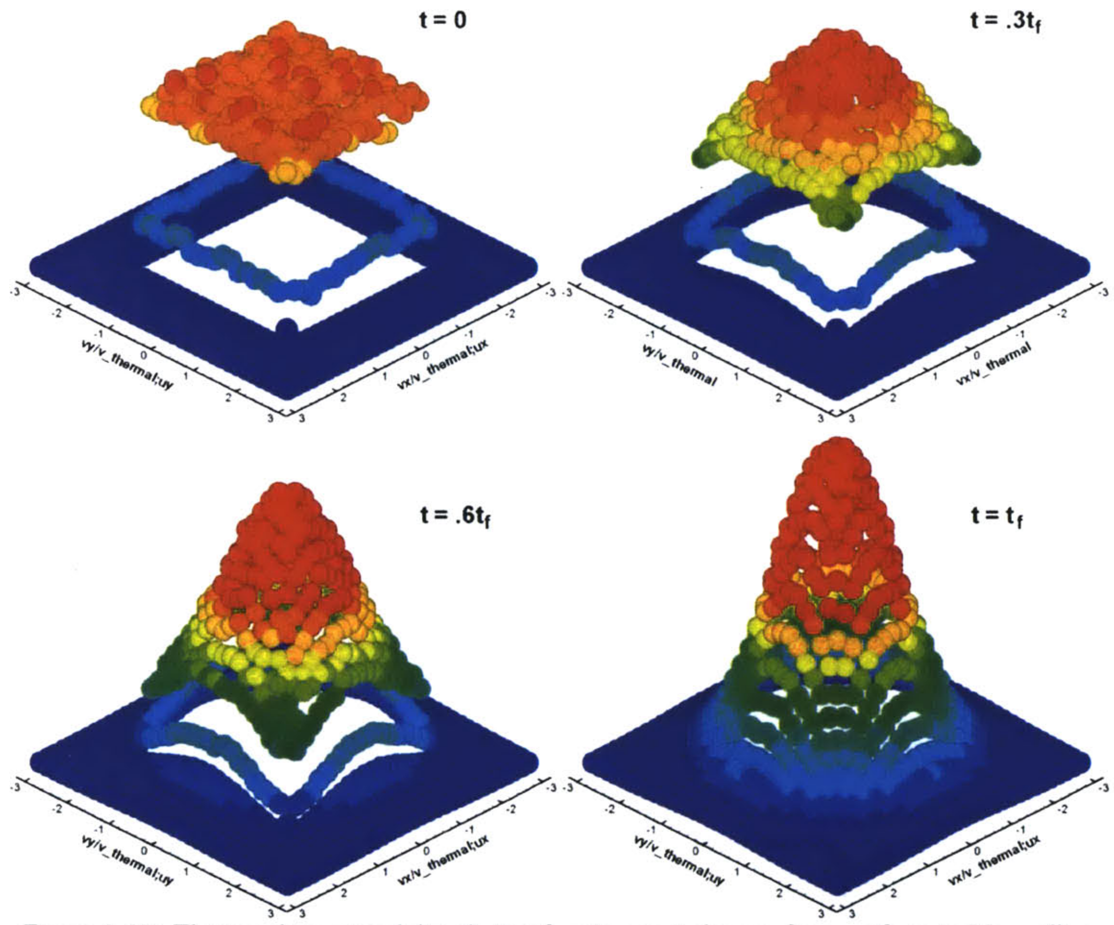


Figure 3-42: The two-dimensional distribution function as it changes from uniform to Maxwellian due to neutral-neutral elastic collisions.

3.4.3 Neutrals in an RZ Box Test

To be thorough and guarantee that our neutral-neutral elastic collision implementation was working in axisymmetric geometries as well, a second test was undertaken similar to that above. In this case, an initial neutral distribution of density .01 simulation units was evenly spaced throughout the small two-dimensional 2cm by 2cm simulation domain pictured below in Figure 3-43, the same as is pictured in Figure 3-39. Note that the density distribution is still accurate near the centerline where the volume goes to zero. The particles were given an initially uniform random distribution in velocity space, each particle having mass of 1 electron mass and initial x and y velocities uniformly chosen from the same interval as in the two-dimensional case, [-.00001c, .00001c]. The boundary conditions were set to be elastic walls as before.

The results of this test are depicted below in Figures 3-44 and 3-45. The distribution clearly changes from uniform to Maxwellian qualitatively in Figure 3-44 and quantitatively in Figure 3-45. This change occurs smoothly over time. Figure 3-46 illustrates that there is no preferential direction for the Maxwellianization of the distribution.

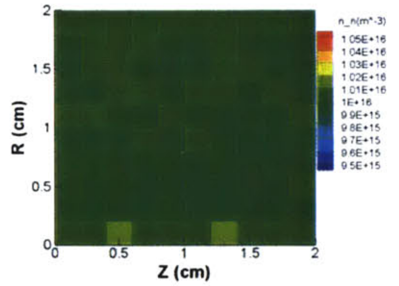


Figure 3-43: The initial density for the RZ neutrals in a box test. Note that the density is smoother than in Figure 3-39 since there are more particles per cell in this case to resolve the third dimension properly.

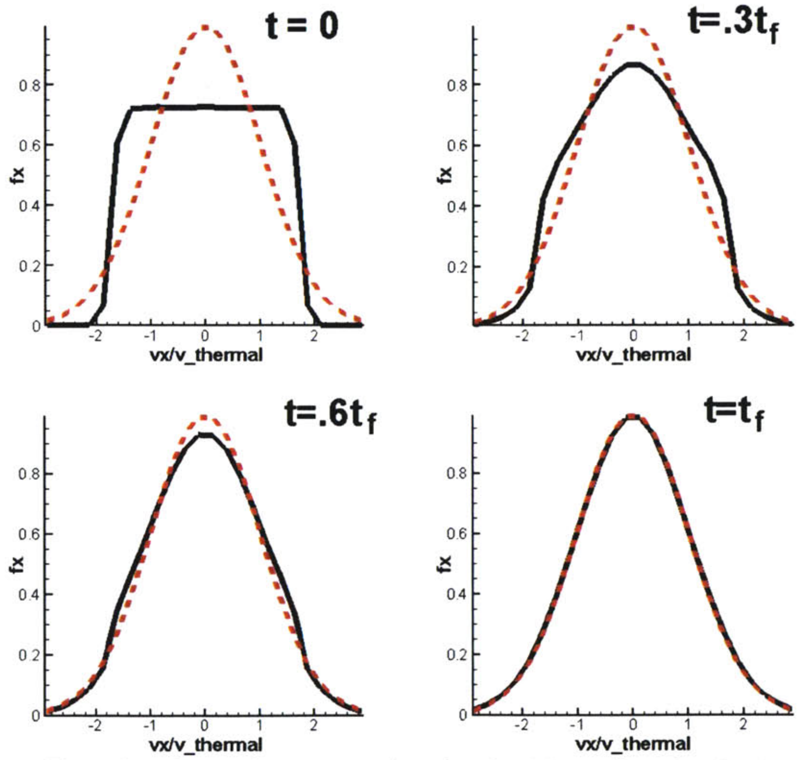


Figure 3-44: The x-distribution function overlayed with a Maxwellian distribution (dotted) as a function of time for the RZ case.

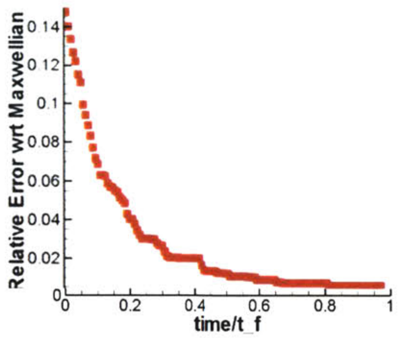


Figure 3-45: The relative L_1 error between the neutrals' distribution function and a true Maxwellian over time for the RZ case.

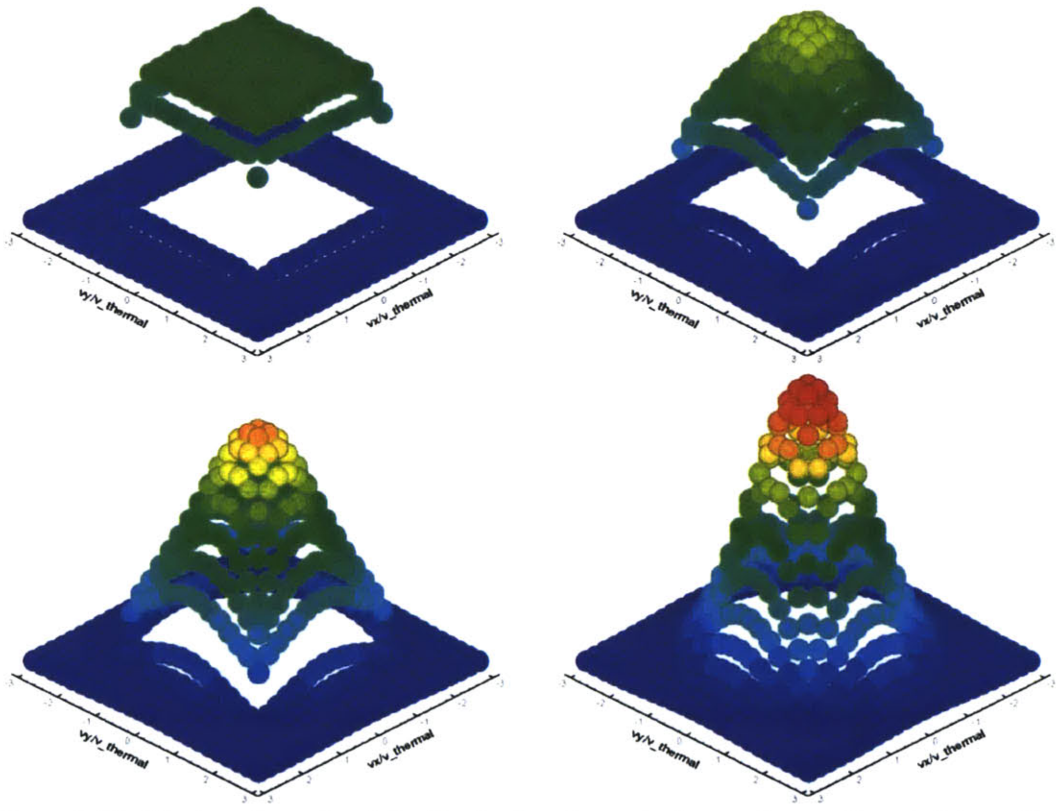


Figure 3-46: The two-dimensional distribution function as it changes from uniform to Maxwellian due to neutral-neutral elastic collisions for the RZ case.

3.4.4 Flow Past A Backward-Facing Step

In order to further test the correctness of the neutral collisional process, as well as to provide a further benchmark of the particle trajectory calculator, the situation of a neutral gas flowing over a backward facing step was modeled. Monatomic hydrogen gas was simulated being injected at a given density and temperature, T_0, from the left boundary of Figure 3-47 below. The far-right boundary was modeled as a free-space boundary while the remaining edges were treated as diffusively reflecting walls at a temperature of T_0.

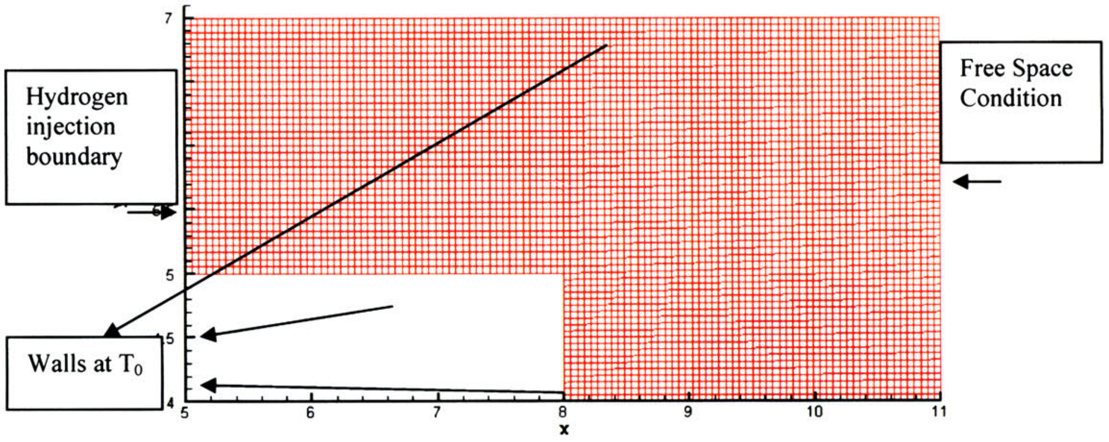


Figure 3-47: The geometry for the backward-facing step neutral-neutral elastic collision test.

Figure 3-48 gives a plot of the resulting time-averaged velocity calculated after a reasonably steady state had been reached. This velocity was normalized by an approximation to the neutral sonic velocity:

$$c_s \approx \sqrt{\frac{T_0}{m_H}} \tag{3.16}$$

where m_H is the mass of a hydrogen atom. In the narrow channel before the step, the figures show the bulging velocity profiles characteristic of a narrow pipe flow. The flow reaches sonic velocity at the point where divergence begins, approximately directly above the back step. After this point the gas continues to expand, speeding up as it does so and maintaining a constant mass flow. In the region of the backward-facing step, $y < 5$ and $8 < x < 9$, the vortices often documented in experiment can be seen, depicted in the figure by a black streamtrace in the region.

Also of note is Figure 3-49 which depicts how the distribution inside the channel region before the backward facing step attains a Maxwellian distribution over time.

Figure 3-49(a) shows that initially the distribution in the x-velocity is somewhat skewed. As collisions with other neutrals and with the wall occur, our collisional method ensures that this distribution does eventually reach an approximate Maxwellian as should be the case in a collisional flow. Figure 3-49(b) demonstrates the same concept for the y-velocity.

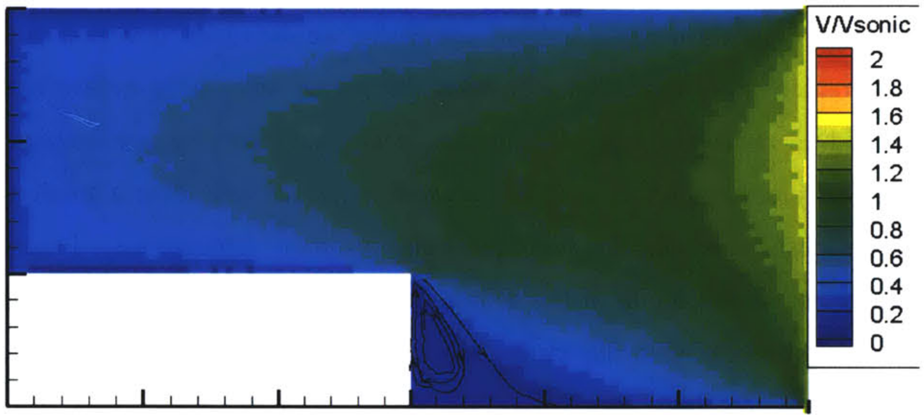


Figure 3-48: The time-averaged, normalized neutral flow velocity with backstep vortex streamline included.

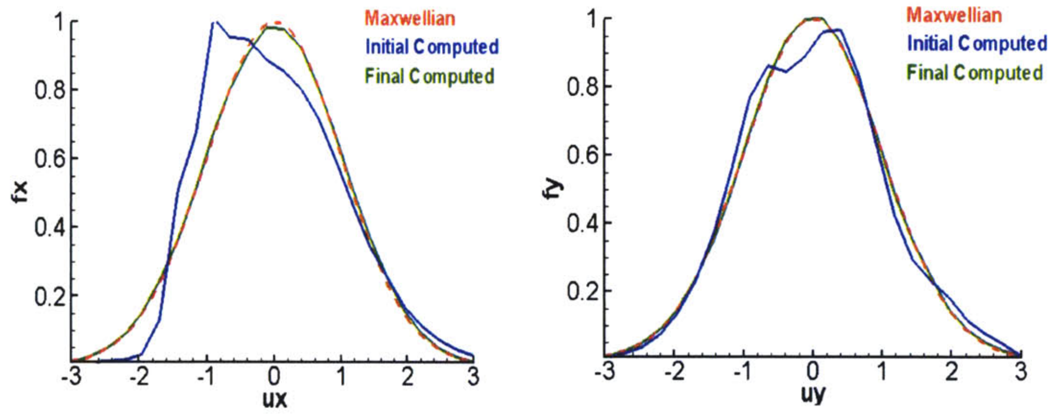
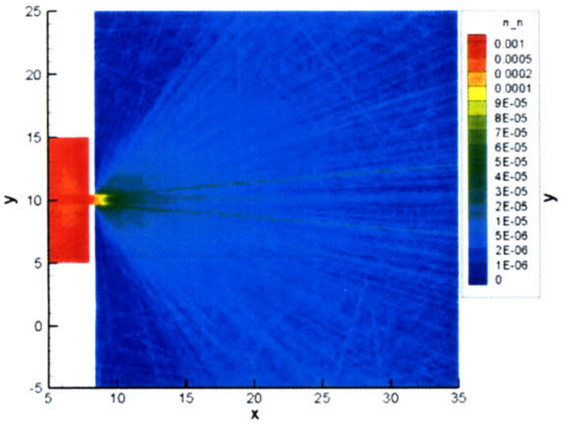


Figure 3-49: (a) The normalized velocity for the backward facing step case. Two streamlines are shown, one which is trapped in the vortical structure and the other which escapes. The reattachment point was approximately located at x=8.6 for this case.

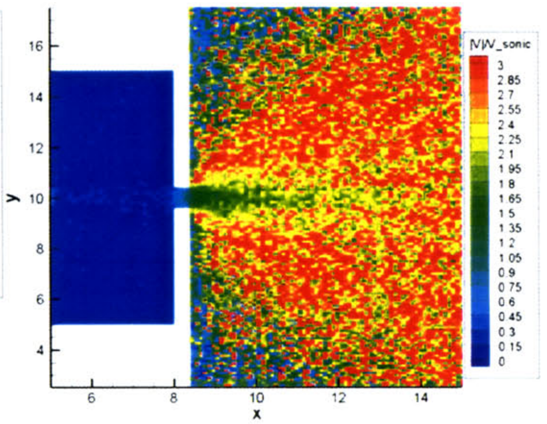
3.4.5 Flow Through A Choked Nozzle

One further benchmark was attempted demonstrating the code's ability to handle large simulation regions while testing the neutral-neutral collision and particle motion procedures. A neutral gas was again injected from the left side of the simulation region into an empty chamber. As the gas filled the chamber in which it was injected, it was allowed to leak out of a small hole in the right side of the chamber. We were then able to examine the properties of the throat region and the expanding plume. Figure 3-50(a) shows a plot of the neutral density in this experiment after some time had passed and the plume had begun to expand into the outside area. The chamber density had increased to an approximately uniform level, and one can easily see the rapid reduction in density in the plume region as it expanded. Outside the plume area, only a few high energy particles had reached the outer boundaries, accounting for the somewhat "streaky" trajectories visible. In addition, both figures were averaged only over a small number of simulation timesteps, adding to their relatively unsmooth appearance. Figure 3-50(b) shows the normalized velocity of the neutral particles in a slightly zoomed image of the throat area. The velocity in the throat accelerates through the sonic velocity, and the plume region is supersonic.

The simple benchmarks presented in the preceding sections have served to show that the neutral-neutral collisional model that we developed and implemented is able to capture a number of experimentally observable features of neutral gas flow.



(a)



(b)

Figure 3-50: (a) Chamber density is uniform while plume density rapidly decreases. (b) Flow accelerates to sonic velocity in the throat and is then further accelerated in the expanding plume.

3.5 Wall Sheath Test

It was important to test our boundary conditions to the greatest degree possible because they are complex and have in the past been one of the greatest sources of error for other simulations with which we have experience. To this end, we undertook to test that the wall conditions would indeed produce the plasma sheath foretold by theory.

The sheath is generally understood from a balance of flux perspective. At steady state, it is expected that the current to a wall is zero. This implies that the flux of ions is equal to the flux of electrons, or, ignoring pre-sheath effects for the ions:

$$\begin{aligned}\Gamma_i &= \Gamma_e \\ \frac{1}{4} n_i \bar{v}_i &= \frac{1}{4} n_e \bar{v}_e\end{aligned} \qquad (3.17)$$

where the ¼ arises from geometrical considerations and the "bar" notation denotes the average velocities of the two species. Assuming the electron density at the wall takes the form of the Boltzmann distribution for the electrons which have been accelerated by a sheath potential, ϕ_s, and that the ion density is roughly the same at the wall as it is in the bulk plasma, *n*, Equation *3.17* above becomes:

$$n\bar{v}_i = \bar{v}_e \left(n \exp\left[\frac{\phi_s}{T_e} \right] \right) \qquad (3.18)$$

where *e* is the fundamental electric charge, and T_e is the temperature of the electrons in electronvolts. Simplifying and solving for the sheath potential, we have:

$$\phi_s = T_e \ln \frac{\bar{v}_i}{\bar{v}_e} \qquad (3.19)$$

Finally, we use the fact that the average velocities of the two species generally differ by the square root of the ratio of their masses:

$$\frac{\bar{v}_i}{\bar{v}_e} = \sqrt{\frac{m_e}{m_i}} \qquad (3.20)$$

That fact allows us to reduce Equation *3.19* to our usable result:

$$\phi_s = \frac{1}{2} T_e \ln \frac{m_e}{m_i} \qquad (3.21)$$

The above expression has been attained through some approximations and is only perfectly valid in the most ideal of cases. However, in practice, it has been shown to be quite accurate [75]. At any rate, it gives us a general idea of how strong the sheath should be given a certain mass ratio between our computational ions and electrons. This, then, is a quantity that we can test.

Another quantity of note is the thickness of the computed sheath. The computation of this thickness is performed in other texts [75], and here we only quote an approximate value. In general, the sheath can be assumed to be on the order of four times as thick as a Debye length. That is:

$$t_s \approx 4\lambda_D \qquad (3.22)$$

3.5.1 Test Set-up

The results that will be shown are almost all pictured in normalized units. However, in case someone wishes to repeat these experiments, I here include the actual physical parameters that were modeled. It was desired to achieve a sheath thickness of approximately 1 cm for simplicity. Since, as noted above in Equation *3.22*, the sheath is generally four times as thick as the Debye length, we then need a Debye length on the order of .25 cm. To keep our values simple and round, we therefore chose a plasma density of 10^{14} m^{-3} and an electron temperature of 10 eV. Using the equation for the Debye length (given earlier as Equation *1.4*), the Debye length is then .2351 cm. This yields roughly the sheath thickness we were after.

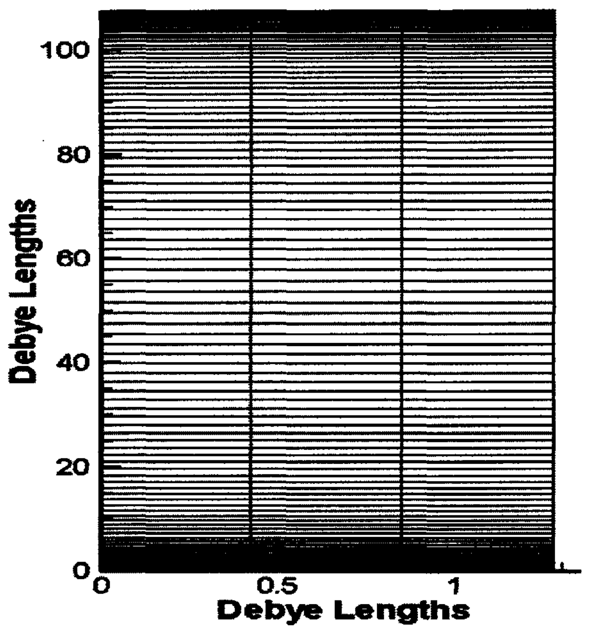


Figure 3-51: The mesh used for testing the sheath formation. Note that the scale on the axes is different; the mesh can actually be very long and narrow since it is periodic in the horizontal direction.

The mesh used for these tests is shown in Figure 3-51. It is a simple orthogonal, two-dimensional, rectangular mesh. The top and bottom boundaries were set to act as dielectric walls which would receive charge and then hold it in place. The left and right boundaries are periodic, so the mesh truly represents an infinitely wide space between two parallel plates. The height of the mesh was set to be approximately 25 times the expected sheath thickness, and was divided into 100 elements. To better resolve the sheath, the height of each element near the top and bottom was kept relatively small compared to the height of the elements in the center of the domain. The formula used to calculate these element heights was such that element j had a height:

$$h_j = \left[\frac{n_y}{2} - abs\left(\frac{n_y}{2} - (j-1) \right) \right] \qquad (3.23)$$

where n_y was the number of elements in the y-direction, in this case 100. The heights were then uniformly scaled to fit the length of 25 cm that was desired for the whole mesh.

3.5.2 Initial Results

The above arrangement was run using two different mass ratios, 1000 and 10000. A small table of mass ratios and their associated sheath potentials is shown in Table 3.1. This table also lists the sonic velocity of the ions, calculated by:

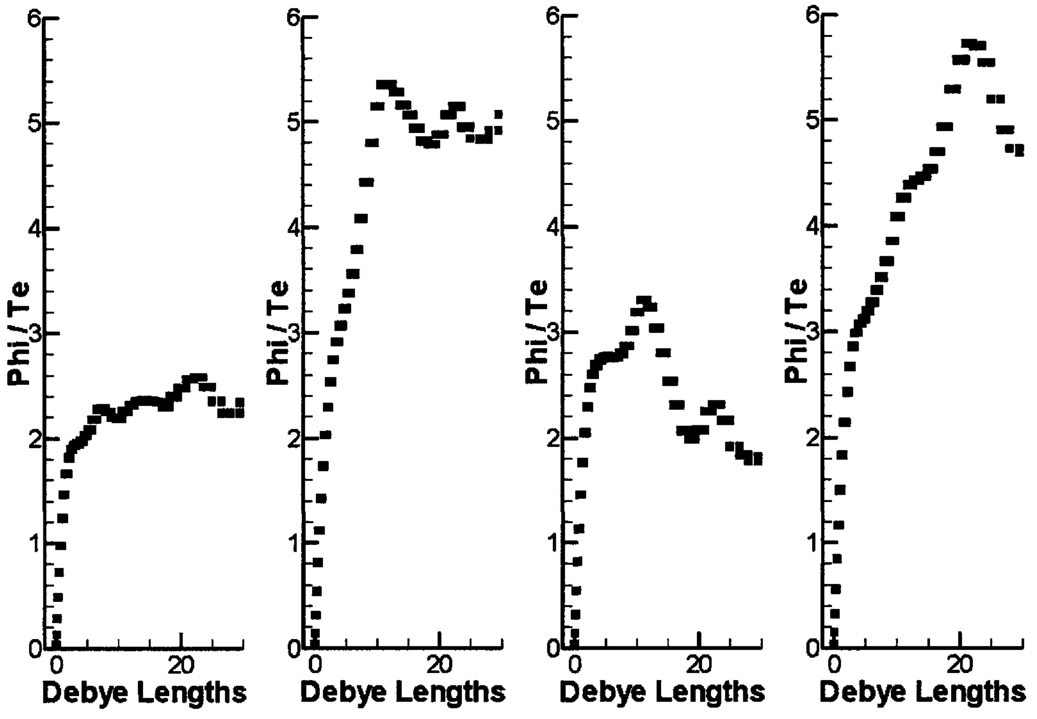
$$v_{sonic} = \sqrt{\frac{k_B T_e}{m_i}} \tag{3.24}$$

Sheath theory suggests that the ions must be accelerated to this sonic velocity at the edge of the sheath [75], giving us a simple way to locate the thickness of our sheaths in the trials. The particles are initially seeded in the domain uniformly randomly in position. They are given a Maxwellian energy distribution with a temperature, T_e, of 10 eV. For these tests, no neutrals are involved in the simulation, nor are there any collisions or Vlasov redistribution step. The particles are merely moved, the charge distribution is scattered to the nodes, a new electric field is calculated, and the process is repeated.

Mass Ratio (m_i/m_e)	Sheath Potential (ϕ_s/T_e)	Sonic Velocity (km/s)
1000	-3.45	41.9
10000	-4.60	13.3
100000	-5.76	4.19

Table 3.1: Depending on the relative mass of the ions to the electrons, the sheath will have a different potential and the ions will achieve different sonic velocities.

The sheath which was formed primarily exhibited the proper basic characteristics, but due to a numerically-induced oscillation, these initial tests were somewhat inconclusive. Figure 3-52 depicts the potential in the first 30 Debye lengths from the bottom of the simulation region. The images are taken at evenly-spaced intervals in time and clearly demonstrate the oscillatory nature of these tests. However, the order of magnitude of the sheath is seen to be approximately correct, as the oscillations appear to be centered around 3.5 normalized voltage units.



(a) (b) (c) (d)

Figure 3-52: The normalized sheath potential at several evenly-spaced intervals is highly oscillatory in this test. The mass ratio here was 1000.

This observation is further borne out by Figure 3-53. The first two images of this figure depict the time-averaged ion-velocity normalized by the sonic velocity for both the 1000 and 10,000 mass ratio cases. From these plots, which are adequately stable over time, we can deduce an approximate sheath thickness in both cases by locating the position at which the ion velocity reaches the sonic velocity. This thickness is calculated to be between 3 and 5 Debye lengths depending on the instant it is measured in the simulation. It is encouraging that the thickness agrees at least approximately with the estimate given by Equation *3.22* above. Figure 3-53 (c) and (d) present the time-averaged potential for both the 1000 and 10,000 mass ratio cases. In the region of about 5 Debye lengths from the boundary, the expected sharp potential drop is observed, followed by a slightly less steep pre-sheath drop in the region out to about 10 or 15 Debye lengths. The sheath potential is on the order of 3 normalized potential units in the mass ratio 1000 case and close to 3.8 units in the mass ratio 10,000 case. These potentials are slightly lower than expected from Table 3.1, but do show the correct trend of increasing potential with increasing mass ratio and are at least on the correct order of magnitude. Considering the severity of the numerical oscillations, such a result is encouraging, but of course not conclusive.

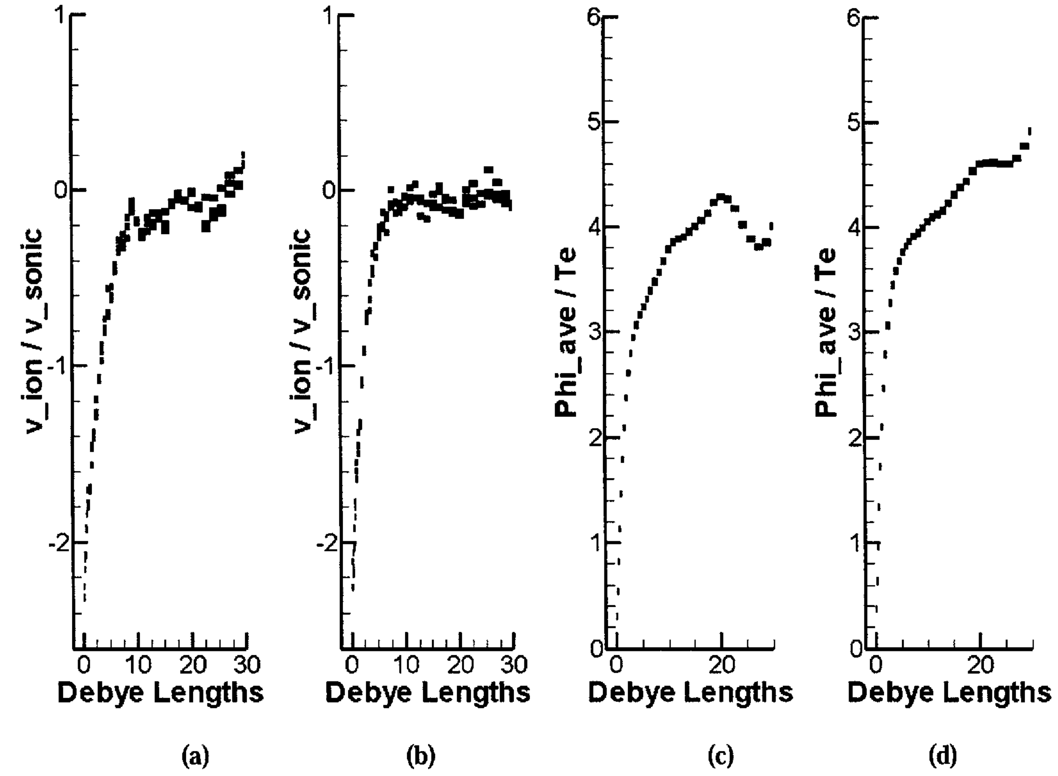


Figure 3-53: The normalized sheath velocities and the time-averaged potentials for the above sheath tests. The 1000 mass ratio case is shown in (a) and (c) while the 10,000 mass ratio case is shown in (b) and (d).

3.5.3 Addition of Quasi-Coulomb Collisions and Finalized Results

In order to reduce the oscillations noticed in the aforementioned tests, a simplified method of elastic Coulomb electron-electron collisions was implemented, solely for these tests. The general idea behind this addition is that the typical idealized plasma sheath model assumes that the bulk plasma maintains a Maxwellian or, at the very least, an isotropic distribution and is largely unaffected by the sheath region. Due to our small simulation domain and the numerics associated with a small number of particles, this was not entirely the case. So outside of the actual sheath region, we chose to approximate collisions by Maxwellianizing the distribution at regular intervals.

So, at specified intervals, the cells not within 1 cm, or approximately four Debye lengths, of the wall were examined individually. All of the electrons in a given cell were deleted while the other types of particles were left untouched. Then, a constant number of electrons was reseeded randomly spaced within the cell. These electrons were given an approximately Maxwellian velocity distribution centered at 10eV of kinetic energy. The total density in the cell was not changed, only the number of particles carrying that density and their velocities. This Maxwellianization approximates the process of collisions that would occur in a real plasma and dampens the worst of the oscillations which would otherwise be numerically induced.

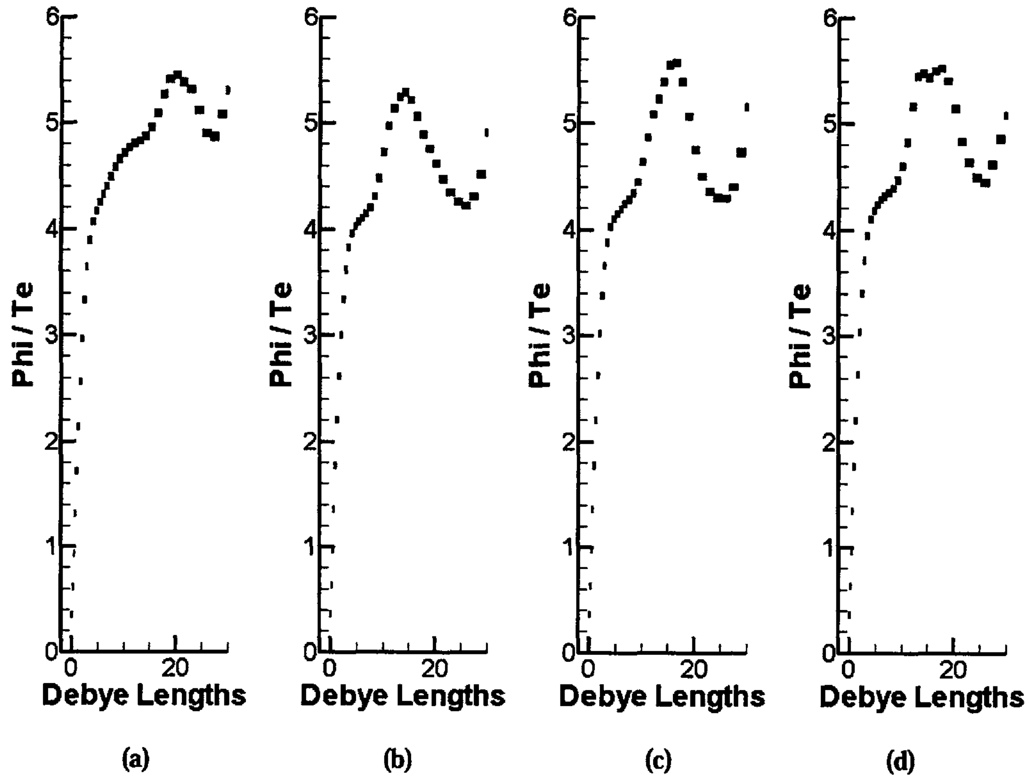


Figure 3-54: The normalized sheath potential at several evenly-spaced time intervals is much more stable in this test. This figure shows results of the mass ratio 10,000 test and can be compared to Figure 3-52 above.

The effect of this change is immediately noticed in the results of this test. Figure 3-54 shows the normalized potential for the mass ratio = 10,000 test. The sheath no longer significantly oscillates with time as before. There is still a significant spatial variation in the potential after the sheath region, but the addition of a collisional approximation has greatly damped these waves and smoothed the spatial distribution of particles. Further improvement could likely be made in the spatial distribution outside the sheath by increasing the frequency of the ad hoc collisions which were implemented, increasing the number of particles simulated, or decreasing the time step in the trial. However, our purpose here was to investigate the sheath, and the conditions in this experiment were accurate enough to maintain that structure's stability.

The images in Figure 3-55 are more or less comparable to those in Figure 3-53 of the previous section. Figure 3-55(a) and (b) show the ion velocity for the 1000 mass ratio and 10,000 mass ratio cases. Simply to demonstrate that both the top and bottom surfaces were functioning properly, the bottom sheath is shown for the 1000 mass ratio case and the top sheath is shown for the 10,000 mass ratio case. Note that, with the collisions implemented, the sheath thickness is approximately 4 Debye lengths as was expected from Equation *3.22*. The time-averaged potentials are again depicted in Figure 3-55 (c) and (d). The figures have been arranged purposely in such a way so that the heavy line indicating the edge of the sheath in figures (a) and (b) lies in the same position as the heavy line in figures (c) and (d). The potential values at these points then define the sheath potentials for their respective cases. The calculated values for these cases fell at approximately -3.5 and -4.6 normalized potential units, respectively, very near to the theoretical values expected from Table 3.1 above. It is also interesting to note the very clear presheath which develops out to a distance of approximately 20 Debye lengths from the wall. This presheath is about 20% as deep as the sheath itself and will also play a significant role in the acceleration of ions toward the walls and the containment of electrons from them.

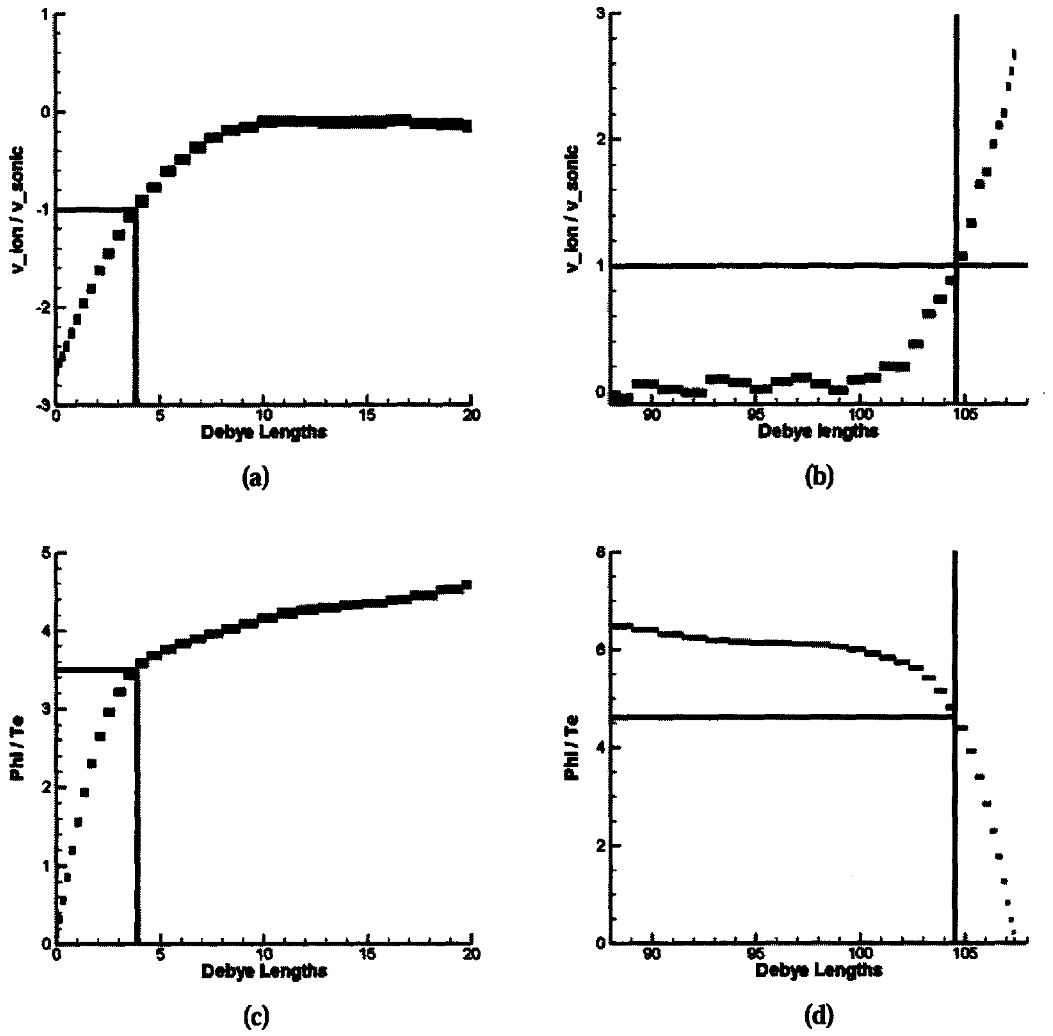
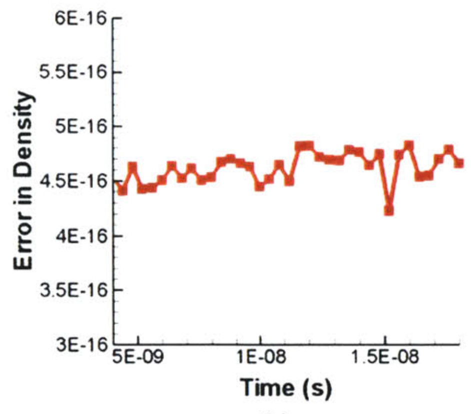


Figure 3-55: The normalized sheath velocities and the time-averaged potentials for the above sheath tests. The 1000 mass ratio case is shown in (a) and (c) while the 10,000 mass ratio case is shown in (b) and (d). The heavy lines indicate the sheath thickness is between 3 and 4 Debye lengths while the potentials calculated fall very nearly at the theoretical values expected.

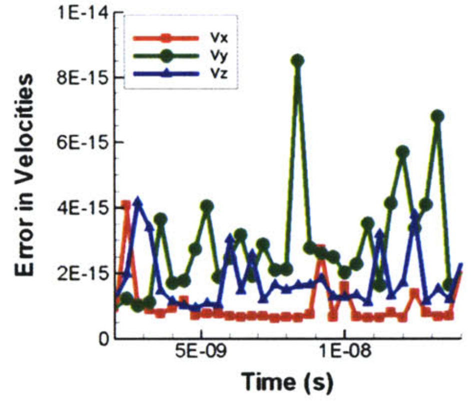
One final interesting note about these tests which might be worth mentioning is that while the tests seem to agree with the theoretical potentials indicated by the very approximate theoretical model outlined in Section 3.5, a more detailed physical model might have predicted lower potentials in the sheaths. Compared to such a model, our results would appear to have too high of a calculated potential. One conclusion that might be drawn from this experience is that our overall understanding of the actual kinetics and detailed physics of the wall sheaths are not yet complete. More detailed investigations into sheath formation in plasmas under various conditions such as those undergoing oscillations or large-scale turbulence or those with sources of non-Maxwellianized velocity distributions could prove valuable in furthering our perhaps too simplistic mental models of the wall-sheath interactions.

3.6 Velocity Redistribution Step Conservations

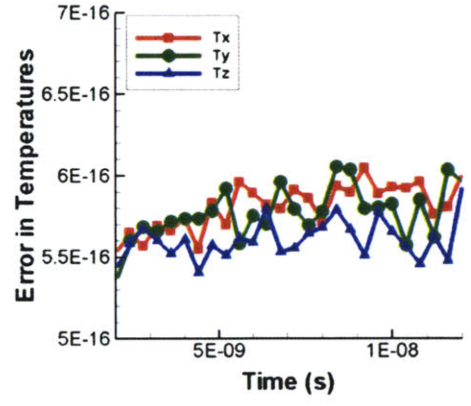
As we periodically reduced particle number through our velocity-gridded redistribution step, it was important that we ensured conservation of important quantities such as temperature, momentum, and density. Checking these conservations involved just a simple test. Since the results of the test would not depend on the grid used, we chose to use the mesh used for the P5 tests detailed in Section 4.1.1. We seeded this mesh with an initially uniform distribution of particles and then allowed the simulation to evolve as normal. Before and after each velocity redistribution, we calculated the local electron momenta, temperatures, and density. Figure 3-56 below shows the relative difference between these before and after values plotted as a function of time. As would be hoped, all seven of these values are conserved more or less to the level of machine precision. The before and after values are, of course, slightly different mostly due to the fact that altering the particles alters the overall order of operations and round off errors occurred during the arithmetic. It has thus been shown that our velocity redistribution implementation is not appreciably affecting the overall values of the temperature, momentum, or density within each cell other than at the level of machine round off errors.



(a)



(b)



(c)

Figure 3-56: Results of the velocity redistribution conservation tests. The (a) density, (b) velocities, and (c) temperatures are conserved to very fine precision even after the particles have been regridded and the particle number has been reduced.

3.7 Parallelization Benchmarking

Section 2.13 discussed the methodology used to parallelize the simulation. A great deal of effort went into constructing that parallel algorithm so that the implementation of it would be scalable and would exhibit high parallel efficiency, especially under the most computationally demanding conditions, namely applications requiring a large number of particles, elements, or both. For our purposes, the parallel efficiency will be measured as the "speed up" achieved for a given number of processors. This metric, here denoted by $\eta_p(i)$, is given by the following equation:

$$\eta_p(i) = \frac{time(1)}{time(i)} \tag{3.25}$$

where *time(1)* denotes the time required to complete the computation using one processor and *time(i)* denotes the time required for *i* processors working together in parallel. If the problem being studied was perfectly parallelizable and the parallelization implementation was exactly perfect, the speed up function would be linear with a slope of 1. In practice, this is never an attainable ideal. Overhead incurred in communication and in establishing the proper conditions for the parallel calculation is often significant. In addition, lack of scalability of the algorithm may be indicated by a negative second derivative in the calculated speed up curve. Thus, our hope must be to attain a speed up function which is as close to linear and slope 1 as possible, knowing that it will never be exact.

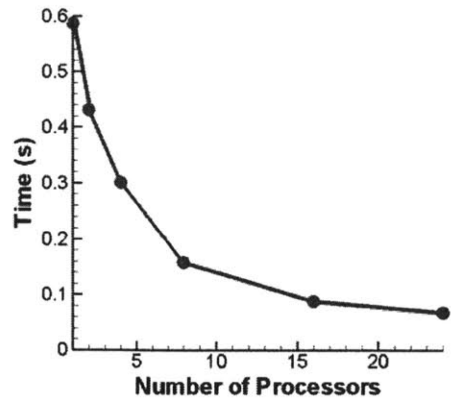
The tests that follow in the subsequent sections, excepting the final load balancing test, were all performed in a manner similar to each other. In order to determine a given data point, the test was conducted at least five times. Of the five elapsed times observed, the minimum time was recorded. This method attempts to account for the somewhat uncontrolled environment of the test. The parallel machines are not solely devoted to our use and may have other transient loads to process. Also, the state of the machine's cache or even the particular physical processors that are chosen for a given trial may vary from test to test. All of these variations and others can only add time to the "best-case" computation. As such, we accept the minimum value as being closest to the unattainable optimal scenario in which no performance reductions from outside influences were felt. This then is the time that is reported.

3.7.1 Timing the RZ Particle Movement Module

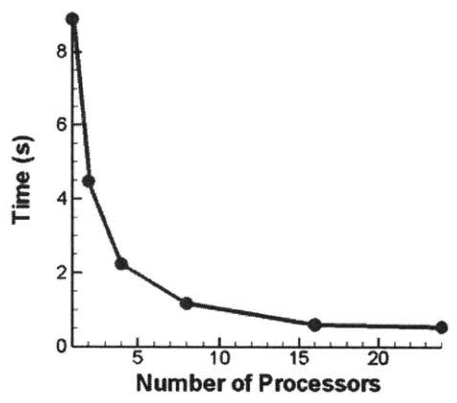
For some simulations, we may be moving more than 20 million particles each time step. Movement of the particles includes the calculation of their accelerations and velocity rotations as described in Section 2.5, resolution of boundary conditions including secondary electron emission, and finally the scattering of the particles' charge, current, and density back onto the computational grid. In addition, in order to maintain the proper distribution of particles between processors, a significant number of particles must actually be transferred between processors as described in Section 2.13.6. This transfer process is also included in the timing results that follow.

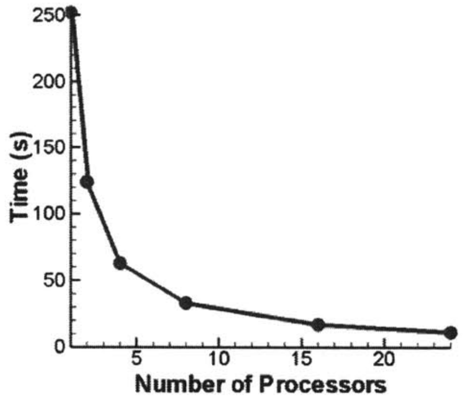
Shown in Figure 3-57 are the raw timing results from several tests of the parallel axisymmetric non-relativistic orbit integration function. Various numbers of particles were simulated in order to obtain a better picture of the function's parallel performance in multiple regimes. Figure 3-57(a) shows the raw results for a simulation incorporating 158,374 particles while Figures 3-57(b) and 3-57(c) depict the raw results for simulations having 1,904,085 and 20,324,568 particles, respectively. Thus, we covered approximately three orders of magnitude with these three tests. The general trend in the figures is quite similar in all three cases. As the number of processors is increased, the computation time decreases monotonically. This demonstrates that the parallelization is having at least some benefit and that even when there is a relatively small number of particles in the simulation (Figure 3-57(a)), the overhead involved in transferring particles and establishing the parallelization does not totally swamp the time required for the actual computation.

While trials with small particle number do exhibit some increase in speed with every processor added, Figure 3-58 demonstrates the implementation's level of parallel efficiency, or in other words how MUCH the speed increases with each added processor. The figure plots the speed ups calculated for each of three particle number conditions as a function of the number of processors used. Clearly, when the number of particles is



(a)





(b) (c)

Figure 3-57: The raw timing results for the axisymmetric particle motion module. Simulations with (a) 158,374 (b) 1,904,085, and (c) 20,324,568 particles were conducted.

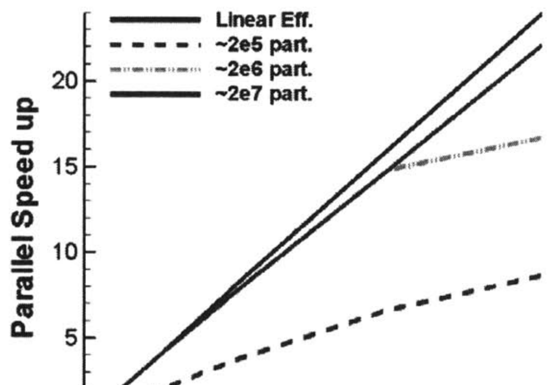


Figure 3-58: The parallel speed up of the axisymmetric particle motion module for three different numbers of particles.

small, on the order of $2*10^5$, the overhead involved in parallelization is seen to be significant when compared to the amount of computation needed to move the particles. As such, the speed up curve for this particle number is flat and far from the linear ideal. However, as the simulation becomes more complex and particles are added, the speed up moves closer and closer to the ideal linear efficiency, as expected and desired. Since our full-scale trials tend to run with more than two million particles and sometimes upwards of 10 million, we are securely in the regime that demonstrates high parallel efficiency.

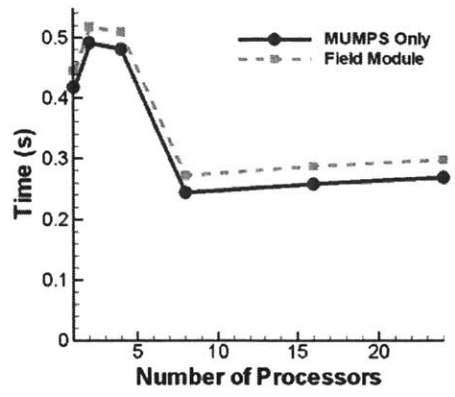
3.7.2 Timing the MUMPS Solver

The parallelization of iterative matrix solvers is often a difficult and time-consuming task, as my own Master's thesis demonstrated [33]. While the serial implementations of direct solution algorithms are sometimes more complex than their iterative cousins, parallelization of the direct solvers is often much more straightforward. Fortunately, the direct solver library we chose to use, MUMPS [86], was already parallelized for us, saving us from the laborious task. While the actual solver has been benchmarked in previous works [3][4][5], we wished to identify the parallel efficiency of our usage of that solver, the code written to convert our data structures into MUMPS format, and the subsequent broadcast operations required to synchronize the resulting potential and electric fields across processors.

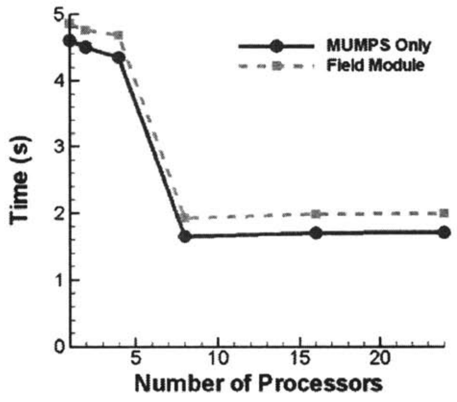
Thus, we tested both the MUMPS solver alone and the entire electrostatic field solution module using the P5 mesh depicted in later sections. Unlike other modules, the parallelization effort here scales with the number of grid elements, not the number of particles. Therefore, we refined this mesh using our automatic refinement techniques. The raw timing results plotted in Figure 3-59 were compiled for meshes of varying levels of refinement containing (a) 5,535 (b) 48,621 and (c) 457,827 elements. The data for the MUMPS solver alone is shown by the solid lines while the data for the entire field solution module, including transformation of our data into MUMPS format and broadcasting of the final result, is shown with a dotted line.

Clearly, Figure 3-59 shows that the MUMPS solution algorithm is not exceedingly efficient for the types of problems and the small numbers of processors we are generally employing. Indeed, the plot for 5,535 elements, Figure 3-59(a), actually increases when a second processor is added, indicating that the overhead required to communicate with that second processor is at that point more significant than the savings gained by employing it. For all three cases, the only real performance gain comes between the 4 and 8 processor level. For this one step, the computational time is approximately halved as it should be. Beyond this point, however, the time either remains stable or in some cases, actually increases at least out to 24 processors which is the maximum number that were tested. The vast majority of the computational time is spent within the MUMPS solver package. The remainder of the field solution module is mostly serial in nature and adds a generally constant with processor number 3-4% onto the MUMPS time.

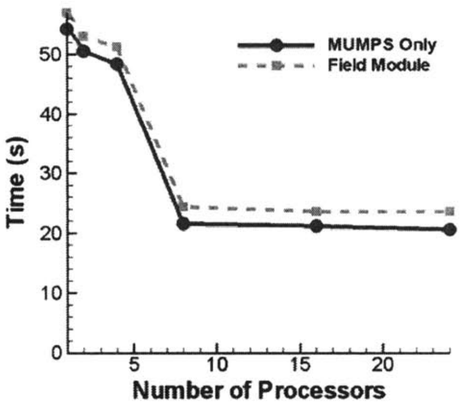
Figure 3-60 next depicts the parallel speed up calculated for the entire field module. Since the MUMPS solver accounts for the majority of the field module's time, the speed ups for the solver alone and the entire module are very similar and including both plots would have been redundant. This figure again demonstrates the relatively poor parallel efficiency of the MUMPS solver, with all three cases having very flat, low-sloped curves. In all three cases, the slope between 4 and 8 processors is acceptable and nearly 1. While this data indicates that MUMPS is generally not parallel efficient in the regimes that we are working, it is important to note that the solver is accurate and generally quite fast in serial comparisons to iterative solvers like our SOR algorithm. Direct solution of matrices is a very difficult problem in which a great deal of computational research has been invested, and the relatively low parallel efficiency of MUMPS must be taken in perspective to the difficulty of the problem which it is trying to solve. The package is by no means perfect, but it is an easily-used, viable alternative to homemade parallel iterative solvers.



(a)



(b)



(c)

Figure 3-59: The raw timing results for the MUMPS solver and entire field solution module. Simulations with (a) 5,535, (b) 48,621, and (c) 457,827 elements were conducted.

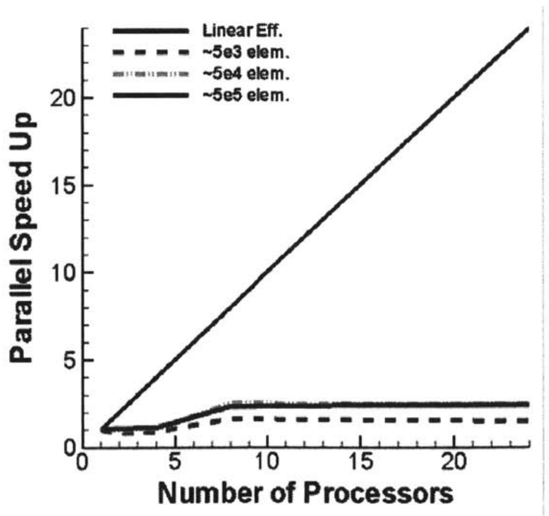


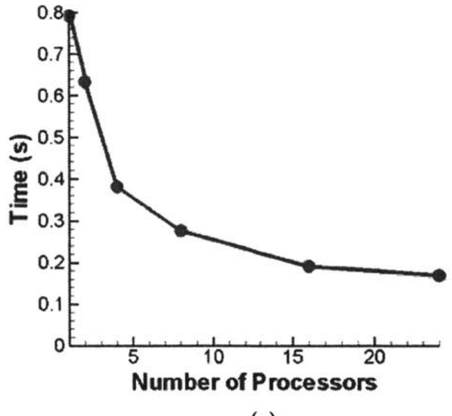
Figure 3-60: The parallel speed up of the complete field solution module for three different numbers of elements.

3.7.3 Timing the Collision Module

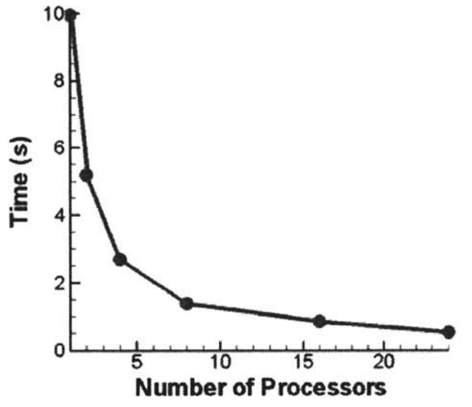
Every few iterations, the interactions of particles are calculated using the collision models presented in Section 2.8. Because these models are so complicated and require the algorithm to examine every particle in every simulation cell at least once, it is important that they be implemented efficiently, especially in regards to parallelization. Special pains were taken to ensure that this portion of the simulation would scale efficiently with an increased number of processors. These workarounds, which greatly complicated the algorithm's implementation and slightly decreased the parallel efficiency of the particle motion module, included the transferring of particles between processors and the reordering of particles by cell in the main particle array. The attention to detail was necessary, however, in order to maintain the level of parallel efficiency and scalability demonstrated in the results that follow.

The raw timing results from simulations containing 102,253, 1,503,263, and 10,364,653 particles are shown below in Figure 3-61(a), (b), and (c), respectively. Again, we attempted to span several orders of magnitude in particle number with these trials. The results are again all monotonically decreasing indicating that parallel overhead is being maintained at a manageable level. One interesting fact is seen when comparing the time axes of the three plots. The maximum time for collisional computation, that is the time when only 1 processor is used, does not scale exactly with the number of particles. At low particle number, the computation time is unusually high (Figure 3-61(a)) when compared with the moderate particle number case (Figure 3-61(b)). This difference in scaling is indicative of the fact that the collision calculations are not wholly dependent upon particle number but rather exhibit a significant amount of non-particle-related overhead. This overhead may be in part due to the collection of collision statistics for output that has been included in these time trials since it is part of the collision module.

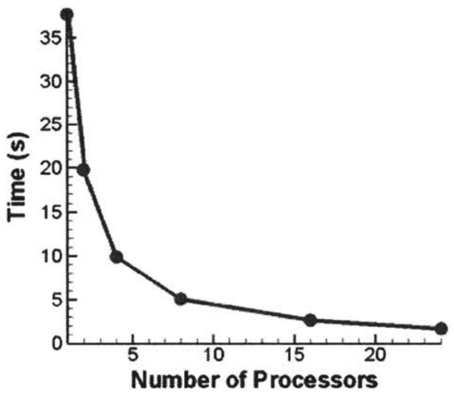
We know from the raw timing results that the parallelization is effective, but Figure 3-62 below shows just how effective it is given the number of particles under consideration. The typical trend is evident; trials with a small number of particles have relatively less parallel efficiency than those with a very large number of particles.



(a)



(b)



(c)

Figure 3-61: The raw timing results for the collision module. Simulations with (a) 102,253 (b) 1,503,263, and (c) 10,364,653 particles were conducted.

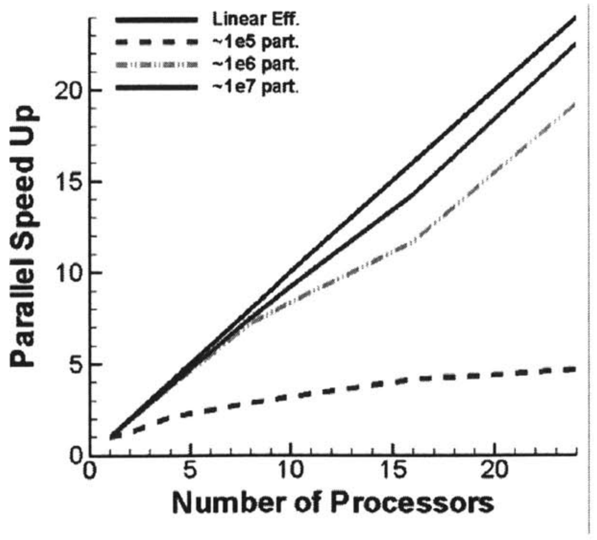


Figure 3-62: The parallel speed up of the collision module for three different numbers of particles.

3.7.4 Timing the Velocity Redistribution Module

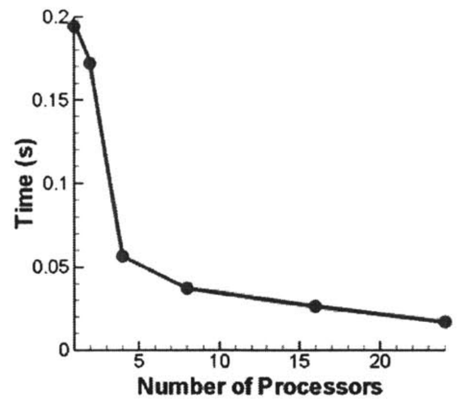
Very similar to the collision module, the re-organization of our implementation to efficiently parallelize the velocity-redistribution step was not simple. It is, then, very gratifying to see that those efforts did pay off in the end and we were able to achieve a relatively scalable, efficient design. The actual implementation details of this module are covered in detail in Section 2.9.

Figure 3-63 is again the raw timing results for the velocity-redistribution module using the same numbers of particles as in the collision module timing tests of the previous section. Those particle numbers were (a) 102,253, (b) 1,503,263, and (c) 10,364,653. The figures have the monotonically decreasing trend expected.

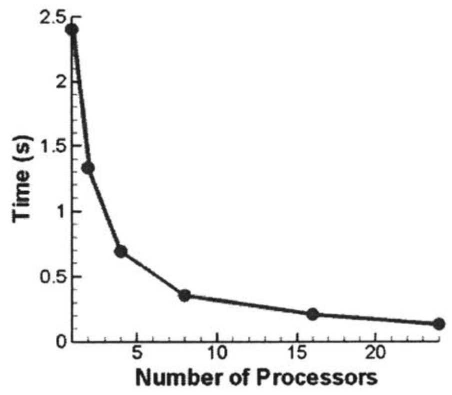
Figure 3-64 depicts, as in previous sections, the parallelization efficiency plotted as speed up versus number of processors. In this case, the low particle number test indicates greater parallel efficiency than was demonstrated for other modules. However, in general, the parallelization is again seen to be most efficient at simulated particle numbers in excess of $1*10^7$.

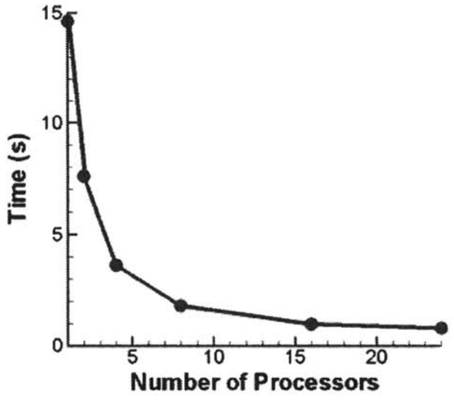
3.7.5 Load Balancing Test

Given that the particle-dependent modules above scaled so well with increased processors, it might already be clear that the particle load between processors is being efficiently balanced. To demonstrate this fact unequivocably, however, we chose to examine the percentage of particles residing on each processor as a function of time. The results of this experiment are shown in Figure 3-65. The images present only the maximum percentage on any of the processors at a given time step. Figure 3-65(a) presents results from a test with 8 processors while Figure 3-65(b) was performed with 16 processors. Thus, the optimum load balance would be at 12.5% in the first image and 6.25% in the second image. This optimum is achieved at some of the time steps, but periodically one processor does end up containing a slightly higher number of particles. This was due to the method used to divide the number of particles between processors.



(a)





(b) (c)

Figure 3-63: The raw timing results for the velocity-redistribution module. Simulations with (a) 102,253 (b) 1,503,263, and (c) 10,364,653 particles were conducted.

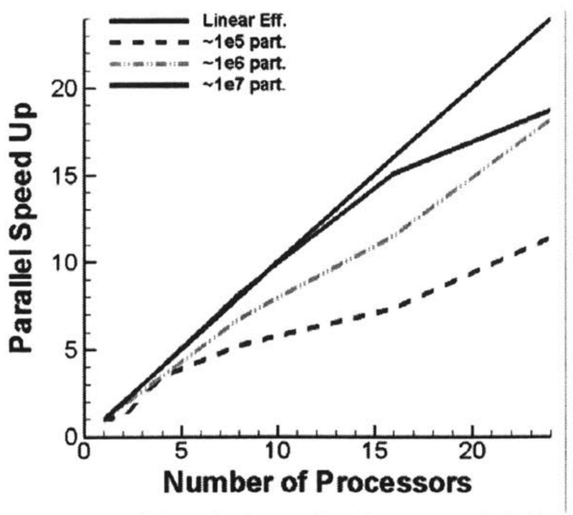


Figure 3-64: The parallel speed up of the velocity-redistribution module for three different numbers of particles.

We would not allow particles in the same simulation cell to be split between two different processors in order to make the velocity-redistribution and collision parallelization possible. The element division algorithm we employed was a simple greedy algorithm which fed a processor element after element until the next element that was fed to the processor would make its particle count become greater than the total number of particles divided by the number of processors. Thus, the first *np*-1 processors were given slightly less particles than their fair share. This meant that the final processor was assigned its share plus the leftovers from the others. Stopping before the processor has exceeded its share of particles has the benefit of guaranteeing that there are some particles remaining for the last processor. One could imagine a simplified situation in which there were two processors and two simulation cells. Cell 1 might contain 40 particles, and cell 2 might contain 60. In our method the first processor would take the first cell, but leave the second cell for its neighbor whereas using a greedy method that stopped once the processor's share had been exceeded would have assigned both cells to a single processor. The trade off, of course, is that our method can and, from the results in Figure 3-65, does leave too many particles for the last processor.

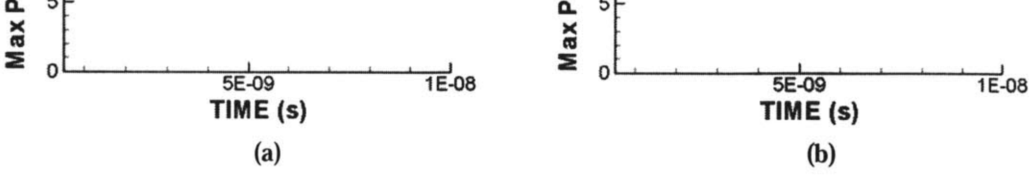


Figure 3-65: The maximum percentage of particles being carried at each time step by any processor. (a) 8 processor and (b) 16 processor tests were conducted. The balance is roughly achieved, but is imperfect due to our requirement that the particles within a given computational element not be split between processors.

Chapter 4

System-level Benchmarking Using the P5 Thruster

With the numerical model fully developed, implemented, and component-level tested, the next step was to see just how effective our new model could be in a real-world application. Other plasma systems, including ultrafast laser-matter interaction, were also studied using this simulation and results of that work can be found in the references [9]. For the present document, we include the most relevant real-world system-level benchmark that was undertaken, the simulation of the P5 thruster. This thruster was chosen by Blateau [18] when he extended the former PIC-MCC simulation to handle SPT type thrusters rather than just the simpler metallic-walled TAL thrusters. Thus, we have a computational model against which we can anchor our simulation. The physical thruster itself is also well-studied experimentally and was designed specifically for that purpose. As such, a significant amount of experimental data exists with which we could compare our results and thereby benchmark the simulation.

This chapter will first present a brief overview of the P5 thruster and a few of the numerous minor adjustments that were added to the code in order to model the thruster properly. We then present a discussion of our simulation results while comparing them with experimental data and the results of MIT's PIC-MCC simulation.

4.1 Details of the P5 Thruster

The P5 thruster was developed jointly by the University of Michigan and the Air Force Research Laboratory with the specific purpose of investigating the basics of thruster physics. [37] The thruster was not necessarily designed for optimal performance but rather for ease of adjustment, reconfiguration, and diagnostic measurement. Such qualities have made it a favorite among researchers in recent years, and a quantity of quality experimental data as well as some simulation results have been compiled for it. This, combined with the fact that the previous generation of MIT PIC simulation had already been configured to study this thruster [18][72], made it an excellent choice for the benchmarking and possible anchoring of the present simulation.

The P5 is an SPT-type thruster, with a ceramic-lined inside channel. This material requires slightly complex boundary conditions to be implemented which are described throughout this thesis and summarized again below in Section 4.1.3. The thruster is usually run at 500 Volts as a 5kW thruster or at 300V as a 3kW thruster. The latter configuration is the one used for all of the present simulation work. The basic experimental values for performance and design are included in Table 4.1.

Anode Voltage	300 Volts
Anode Current	10 Amps
Power	3 kW
Thrust	175 mN
I_{sp}	1670 s
Mass flow	10.7 mg/s
Maximum Magnetic Field	290.6 Gauss

Table 4.1: Basic operating parameters and performance of the P5 thruster based on the experimental results of Haas [37].

4.1.1 Geometry and Mesh

Shown below in Figure 4-1 is a simple schematic representation of the P5 thruster's shape and the various types of boundary materials used. The configuration of the thruster is almost entirely axisymmetric except for the positioning of the cathode. This forces axisymmetric, two-dimensional simulations, such as the present work, to model the cathode implicitly. Various techniques for this modeling have been attempted in the past, and the one chosen for the present simulations is described below in Section 4.1.3.4. The entire channel is approximately 3.81 cm in length, and it has an inner radius of 6.1 cm and an outer radius of 8.64 cm.

The structured mesh used by the previous MIT PIC simulation is depicted in Figure 4-2(a) alongside the unstructured mesh used for the present work in Figure 4-2(b). Effort was made to maintain similar element sizes between the two meshes so that comparisons between the two simulations would be more direct. However, one might immediately discern a benefit of the unstructured mesh. The element qualities in some regions of the structured mesh, especially at the channel exit and near the upper boundary between metal and dielectric, are somewhat low and improving them would be difficult and laborious. The element qualities for the unstructured mesh are generally very good, and its level of refinement could easily be increased or decreased with relatively little work.

The same distance into the plume was simulated in both cases, again to maintain the directness of comparisons. Since it is such a relatively short distance, however, the model used for the cathode may not be entirely accurate. If more accurate and detailed study of the P5 thruster was the goal of this thesis, a larger simulation region could easily have been used, especially since the use of RRC meshing techniques could reduce the number of elements required in low-density plume regions.

Finally, it should be noted that while the previous simulation grid does extend into the internal material boundaries, neither simulation actually calculated the field inside these boundaries and instead used a, perhaps suspect, boundary condition described in Section 4.1.3.1.

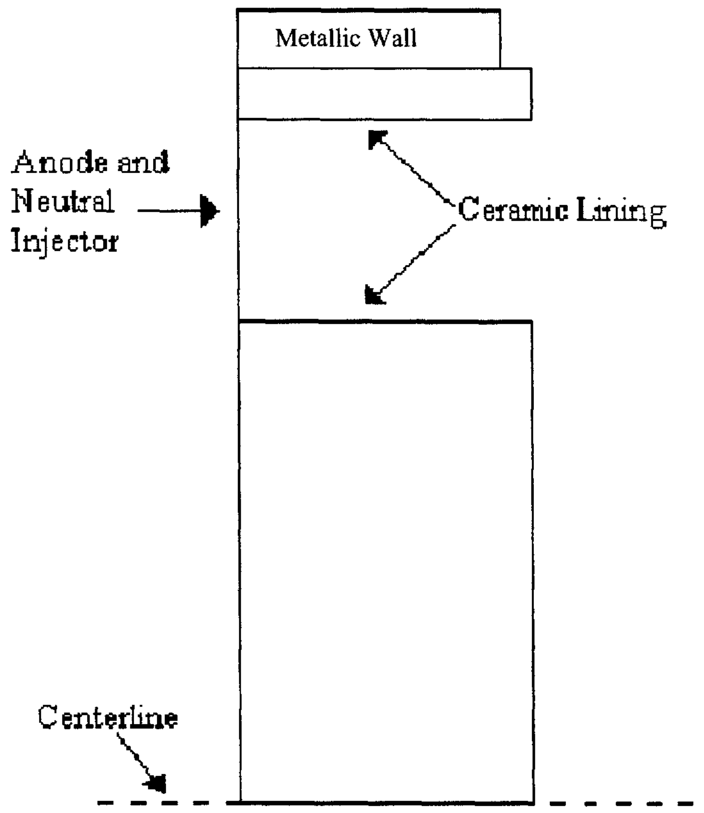
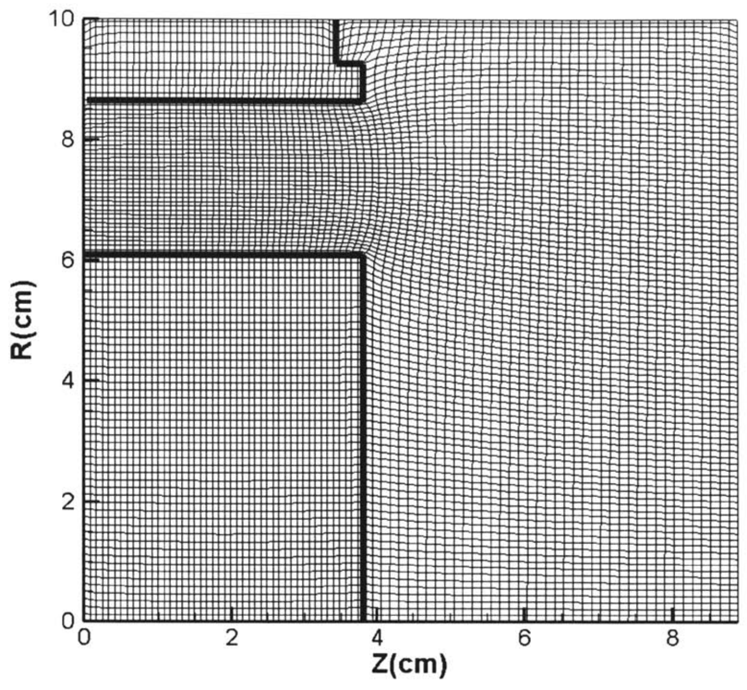
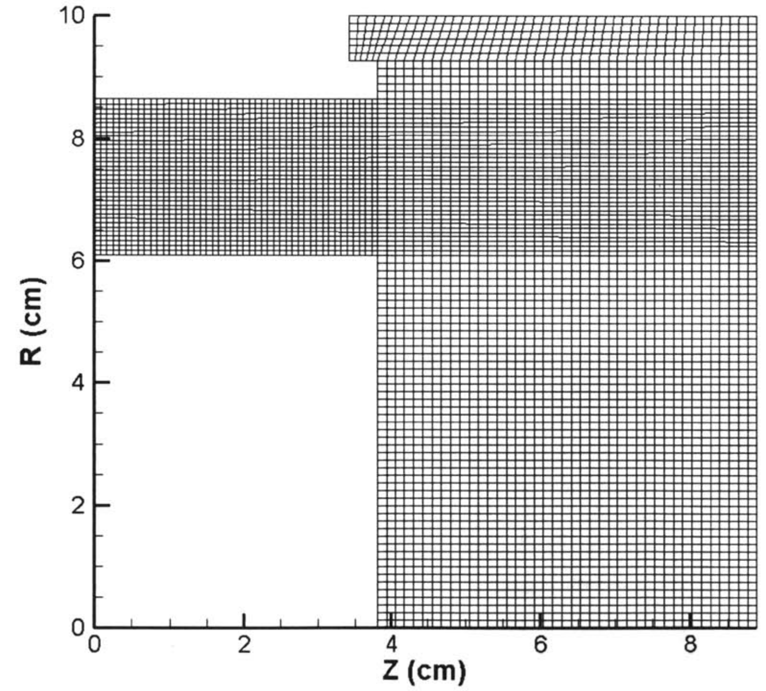


Figure 4-1: A schematic representation showing the basic configuration of the P5 thruster being simulated. The geometry is axisymmetric assuming that the cathode is implicitly modeled.



(a)



(b)

Figure 4-2: The computational meshes used in (a) the previous MIT PIC simulation and (b) the present work.

4.1.2 Magnetic Field

The magnetic field used to compute results for both simulations is shown in component form in Figure 4-3. This is the same magnetic field as reported in [18]. Sullivan [72] also used this same magnetic field for her simulations of the P5 thruster. While she initially reported the same maximum radial field of approximately 290 Gauss (for example [71 Figure 3-1]), however, she apparently rescaled that maximum field midway through her work to 190 Gauss (for example [71 Figure 5-1]). Exactly when that change was made is unclear, but it is certain that all of the anode and beam current results that she reported were found using this reduced magnetic field. This can be seen by examining the time scales of the discharge oscillations in her figures (for example [71 Figure 4-1]) and realizing that, after being scaled by $\sqrt{f}$, the artificial mass ratio, the frequency of these oscillations is only approximately 5.5kHz. This is about half the experimental frequency and half the frequency reported by Blateau for the PIC code with a 290 Gauss maximum radial magnetic field [18]. This reduction in Sullivan's magnetic field made direct comparison between her work and experiment or between her work and the present simulation difficult, prompting us to provide our own simulation results using the previous PIC simulation with the proper magnetic field. This unusual reduction in field may also have played a significant role in her findings regarding the optimal anomalous diffusion parameter for the MIT PIC simulation, as our results at full magnetic field appear to differ significantly.

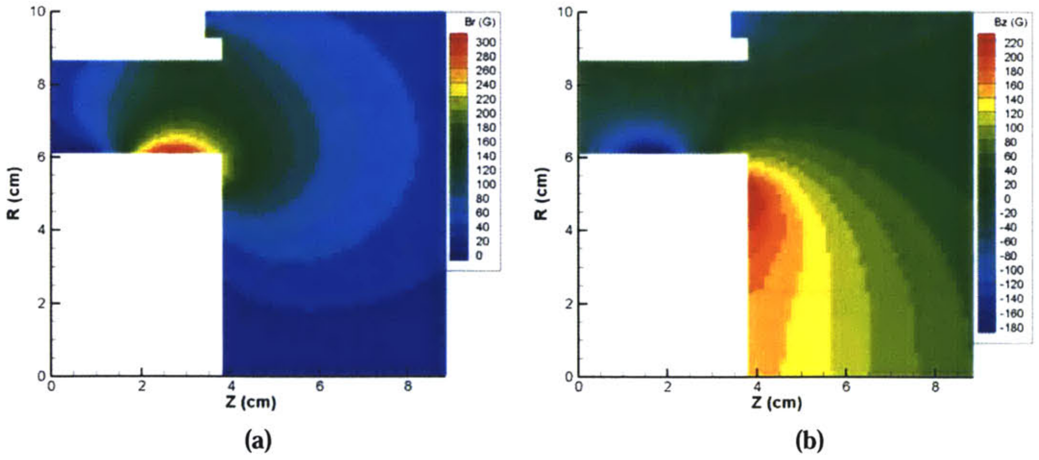


Figure 4-3: The (a) radial and (b) axial magnetic field contours used for the P5 thruster.

4.1.3 Special P5 Simulation Conditions

As mentioned above, the P5 required some additions to the present simulation in order to more accurately deal with some of the unique problems associated with this thruster type. These changes are reviewed in the following sections.

4.1.3.1 Dielectric Boundary and Potential Calculation

In Blateau's thesis [18], he discussed a method of self-consistently modeling the wall sheath formed by the ceramic walls of the P5 thruster. Since this material is an insulator, the basic principle to be adhered to is that any impacting charge is absorbed locally by the ceramic and then remains fixed in location. The method used to calculate the potential near the dielectric walls then is to keep a running tally at each grid cell within the simulation of the total amount of charge which has impacted the dielectric wall at that position. This charge is then directly incorporated into the potential calculation just as though it were free charge still within the cell volume. This same principle was implemented in the current simulation.

In terms of the electric field within the insulating dielectric, Blateau's calculations attempted to show that this field could be neglected relative to the much larger field within the plasma sheath. While in reality, this assumption may be tenable, we found that due to the implementation of the field solver and the numerical acceleration trick of increasing the vacuum permittivity, this assumption did have a fairly significant impact upon the former PIC-MCC simulation's results.

Blateau wished to show that within the channel:

$$E_{\perp}^{plasma} >> \varepsilon E_{\perp}^{dielectric} \tag{4.1}$$

with ε being the dielectric constant in the boron-nitride which has a value of about 4 in these units. For the value of the plasma's electric field, he assumed a sheath potential of about 40 Volts which would fall over a distance equal to the sheath thickness. As we have discussed previously, the thickness of the sheath is approximately 3 or 4 Debye Lengths. Blateau chose a minimum .02 mm value of the Debye Length within the chamber, citing Haas' experimental data which indicated a range of .02 mm to .05 mm. This choice of the minimum value already inflates his estimate of the plasma's electric field by a factor of about 2. All told then, Blateau arrived at an estimate for the plasma's electric field of:

$$E_{\perp}^{plasma} \approx \frac{40V}{3*.02mm} = 670{,}000\frac{V}{m} \tag{4.2}$$

For the field within the dielectric, Blateau assumed the remaining potential drop of 270 Volts over a distance equal to the thickness of the dielectric walls. He quotes this thickness to be about 2 cm. This gave him an estimated electric field in the dielectric of:

$$\varepsilon E_{\perp}^{dielectric} \approx 4\frac{270V}{2cm} \approx 54{,}000\frac{V}{m} \tag{4.3}$$

Since his estimate for the field in the dielectric was less than 10% that of the plasma's field, he claimed that the dielectric field could be neglected.

Unfortunately, we must take issue with a number of the assumptions made in Blateau's treatment. First of all, as mentioned above, the minimum value of the Debye Length in the chamber is .02 mm and in reality it may be as much as twice that number. In addition, the thickness of the sheath is generally closer to 4 times the Debye length, but we will ignore this adjustment since the thickness is only approximate anyway. The final physical inconsistency is the thickness of the ceramic which Blateau cited to be 2 cm. The outer ring of the P5's channel as it is simulated has only a .6 cm thick ceramic coating which is backed by conductive metal. It remains unclear from where Blateau's value of 2 cm originated.

If we now redo Blateau's calculation with the updated distances and thicknesses, we estimate the field within the plasma sheath to be:

$$E_{\perp}^{plasma} \approx \frac{40V}{3*.04mm} = 335{,}000\frac{V}{m} \tag{4.4}$$

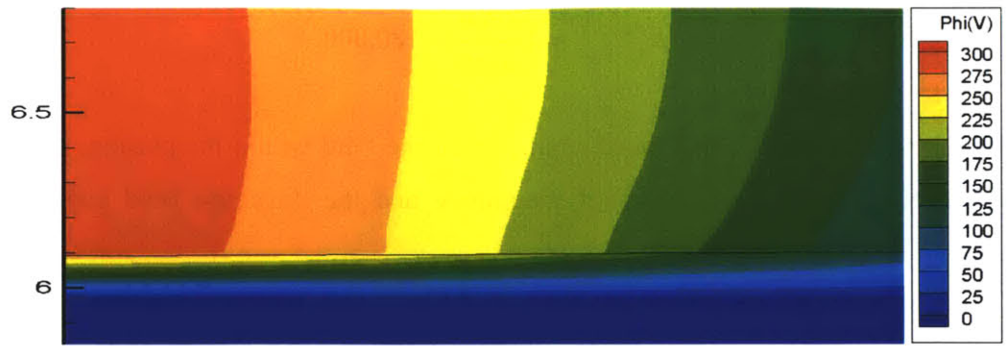
and the dielectric field now becomes:

$$E_{\perp}^{dielectric} \approx 4\frac{270V}{.6cm} = 180{,}000\frac{V}{m} \qquad (4.5)$$

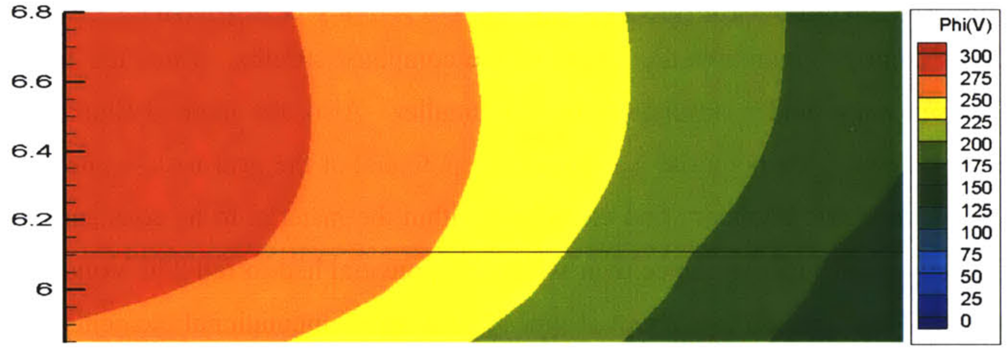
While the dielectric field is indeed smaller than the field within the plasma, the two are now at least on the same order of magnitude and the dielectric field can hardly be neglected in the general case.

To make matters worse, the implementation of the simulation further exacerbates the need to simulate the field within the dielectric. The addition of an artificial permittivity factor, used in both the present simulation and in Blateau's version of the PIC-MCC model, increases the length of the computed sheaths. Thus, the field within the plasma may, under simulation, be still smaller. Also, the method Blateau actually used to calculate the potential was to set the potential at the grid node's one step away from the dielectric boundary and elsewhere within the material to be constantly 0. This effectively reduced the distance over which the potential had to fall 270 Volts to just the spacing of one grid cell, or about .1 cm. These two computational elements combined actually make the computed electric field in the single cell within the dielectric boundary significantly LARGER than the field in the plasma just outside the boundary.

Having noticed this discrepancy, we undertook to discern just how much of an effect the neglect of the dielectric field would have on the overall simulation results. It turns out, that the effect is real and noticeable, but fortunately not exceptionally detrimental. Figure 4-4 below demonstrates the time-averaged calculated electric potential near to the inner ring of the P5 thruster for two cases computed with the PIC-MCC code that Blateau used. The first case in Figure 4-4(a) assumes the field is negligible and does not compute within the material. The second case, Figure 4-4(b) carries out the more expensive calculation of the potential within the dielectric. Sullivan [72] and others had attempted to understand why the PIC-MCC simulation predicted "reverse-sheaths" as demonstrated by the orange-yellow equipotential line in Figure 4-4(a). If the potential is calculated throughout the dielectric, the figure shows that these anomalies are no longer apparent. In addition, the computed sheaths are generally much fuller and deeper, agreeing more accurately with theoretical predictions.



(a)



(b)

Figure 4-4: The electric potential near the inner ring of the P5 thruster's channel. These results were obtained using the PIC-MCC simulation with (a) Blateau's original approximation and (b) the full calculation inside the dielectric. Much thought and confusion had been invested in trying to understand why the sheaths in the original simulation seemed to bend the wrong direction as the orange-yellow equipotential demonstrates in (a). This anomaly is no longer present if the full dielectric calculation is used.

The present PIC-Vlasov simulation, with the above analysis in mind, chose to take a middle-ground approach to the calculation of the dielectric potential. We collect charge at the location that it impacts just as Blateau did. However, instead of setting the potential of the element just one grid-spacing away to 0 as the PIC-MCC had done, we chose to instead, assume that this potential went to 0 over a distance of approximately .6 cm, or the minimum thickness of the dielectric. In this way, we do not assume that the field within the dielectric is negligible and calculate implicitly by approximating the distance we think it will require it to decline. It is not as accurate as meshing the entire dielectric wall, but is certainly more efficient. The method should also still prevent the worst of the reverse sheathing seen in the former simulation since the field calculated within the dielectric should fall off gradually over a longer distance instead of simply over one grid cell.

4.1.3.2 Secondary Electron Emission

Another complication caused by the addition of dielectric boundaries to the simulation is the emission of secondary electrons from electron impact collisions with the walls. Fortunately, this was studied and well-implemented in the PIC-MCC by Blateau and our only task was simply to understand the model and re-implement it in a manner that would jive with our data structures and assumptions about particle weights.

The data of Jolivet and Roussel [45] for boron-nitride emission was fit with a power-law given by:

$$\delta = .086 * E^{.57} \tag{4.6}$$

where here δ is the number of secondaries emitted and E is the impacting electron's energy in electronvolts.

Since the electron energy must be greater than about 74eV in order for a single secondary electron to be created per incident electron, the δ value is assumed to represent a probability. In the PIC-MCC code, if δ was not exactly 0, 1, or 2, the fractional part was compared to a random number and another whole electron would be created if this random number was less than the fractional part. Since our particles were already weighted, our method could remove the randomness from the process. When an electron impacts the dielectric, the proper yield is calculated. The electron's statistical weight is then multiplied by δ, and the particle is bounced back into the simulation. The energy of the particle is reset to a nominal value of 1eV and it is quickly reaccelerated by the sheath back into the bulk of the plasma.

It should be noted that the sputtering of material from the walls due to ion collisions also plays a prominent role in many studies [23], but was here neglected in the interest of simplicity.

4.1.3.3 Adjusting the Wall Sheath for Artificial Mass Ratio

As described in Section 2.4.3 above, both the PIC-MCC and our current simulation utilized an artificial mass ratio to make the heavy particle kinetics occur on a time scale much closer to that of the electrons. This numerical trick greatly reduces computation. However, it also has its own drawbacks in terms of the creation of non-physical effects. The reduction of the sheath potential is one of those effects.

Equation *3.21* in a previous chapter noted that the approximate potential in the sheath is highly dependent upon the ratio of the mass of the ions to the electrons. Since we reduced the ion mass, we have also reduced the calculated sheath potential. Again, Blateau [18] dealt with this problem in the PIC-MCC simulation, and our modus operandi was simply to purloin that method, rearrange it slightly to meet our needs, and re-implement it in our simulation.

The ion flux, both in terms of number and energy, and the electron number flux to the walls remain unchanged. However, the electron energy flux to the wall is not correct. Some of the electrons that the computation "chooses" to impact the wall would, assuming the sheath was at its actual strength, in reality be reflected before they ever reached the actual wall material. The solution to this problem is relatively simple, but not without downsides and approximations. Basically, when an electron in the simulation reaches the dielectric, a calculation is made. The maximum electron temperature along a grid line perpendicular to the wall is found. This is then used in Equation *3.21* to calculate the voltage that the actual sheath would have had at this location if the ratio of masses was realistic. This calculation is unfortunately a significant approximation and source of possible error since the electron temperature is not in reality constant all along the cross-section of the channel. Blateau took the maximum temperature he found to be the "local electron temperature." We chose to select the second greatest temperature point we found in the hopes of avoiding possible points of numerical noise. Neither approximation is exactly correct, of course.

If the impacting electron has an energy greater than the calculated actual sheath energy, then there is no problem. This particle should have made it to the wall, and so it is absorbed and secondary electrons are created after subtracting off the energy difference between the actual sheath and the computational one. This last step takes into account the fact that the electron should have been decelerated more by a realistic sheath than it was by the simulated one.

If, on the other hand, the impacting electron does not have an energy greater than the actual sheath energy, the particle is instead reflected from the wall specularly. The amount of energy in the sheath at that position is then recorded along with the statistical weight of the impacting electron. Later, after all particles have been moved, a separate function is called. This function scans the electrons in a neighborhood around the spots where electrons were reflected. A nearby electron with an energy greater than the energy of the actual sheath is then deleted from the simulation. In addition, secondary electrons are created using Equation *4.6* above and the energy of the deleted particle. This process continues until all of necessary energy has been removed or there are no more electrons available in the vicinity with a high enough energy to be deleted. In the latter case, a numerical factor known as the *sheath_coeff* is reduced while in the former it is slightly increased. This factor, which can be turned on or off within the code, attempts to absorb the inaccuracies of the sheath potential calculation through multiplication with the actual calculated sheath. The hope is that as the while the actual sheath potential is probably not being calculated perfectly accurately, the adjustment by the *sheath_coeff* will eventually balance out this fact.

Clearly, the correction that we add to the sheath in order to mitigate the effects of the artificial mass ratio is inaccurate and quite ad hoc. It is likely one of the primary sources of error within the simulation. Until such time as the computational hardware allows us to efficiently simulate these plasma systems without the use of numerical tricks like the artificial mass ratio and vacuum permittivity, we unfortunately have little choice but to accept some slightly unorthodox techniques in order to recover the physical solution. Fortunately, the Near Vacuum Hall Thruster, which was to be the main usage of our new simulation, has only metallic boundaries and so does not require this additional sheath correction, making its results a good deal more trustworthy.

4.1.3.4 P5 Cathode and Free Space Boundary Condition

It was mentioned in Section 2.10.3 that the electron-emitting hollow cathode crucial to Hall thruster operation is implicitly simulated in order to make viable our axisymmetric assumption and the spatially two-dimensional code which that assumption entails. Some details about the implementation of that cathode injection were provided in that Section, and here we present the few remaining bits which are unique to the P5 thruster and its geometry.

First of all, the cathode plane for electron injection purposes was specified to be the entire right-hand side free space boundary as well as the entire free space top boundary. The assumption then is that ions leaving through the top boundary would eventually be neutralized by cathode electrons downstream just as would their brethren exiting through the right boundary. Of course, a fraction of those ions or the neutralizing electrons may never actually reach the top boundary of the thruster, but such is simply a drawback of an incomplete simulation region. Whenever the charge in any of these cells becomes positive and the potential in that cell is not less than the user-specified cathode potential, electrons will be injected to neutralize the charge.

These outer boundaries also have some specific adjustments made to them in terms of their transparency to electrons. The top boundary specularly reflects any electrons which impact upon it unless the charge in that cell is less than 0. This is an attempt to account for those electrons that would have left the simulation domain, curled back toward the thruster on magnetic field lines, rebounded off of the thruster surface, and followed the field lines back down into the domain. Without simulating that region of the thruster directly, we use the simple assumption that the electrons re-enter at the same position and with the same energy. Szabo also used a similar approach [73].

In terms of the potential calculation, the upper-right hand corner of the simulation is assumed set at the user-specified cathode potential. This potential is generally either 0 or -10 Volts relative to the anode's 300 Volts. Finally, the potential of the "cathode plane" boundaries are not allowed to fall to less than 10 Volts below the cathode potential. This ensures that de-stabilizing negative potentials are not artificially introduced into the simulation due to the shortened domain. The PIC-MCC code employed much the same techniques.

4.2 Comparison of Current Work with PIC-MCC Simulation and Experiment

Having thus enabled our present simulation to model the P5's numerous unique features, we then compared our results to those of the past simulation and with the experiments of Haas [37]. The following sections each look at a separate facet of the simulation's output and find varying levels of agreement between the simulations and real world data.

4.2.1 Electric Potential

The first quantity we present, shown in Figure 4-5, is the electric potential in the thruster channel. As expected, the potential begins to fall off too early in both computational cases when compared with experiment. This is shown in Section 6.7 to be controlled by the Bohm coefficient assumption. The sheaths near the inner wall of the thruster in all three cases seem to be somewhat reversed. This issue was discussed above in Section 4.1.3.1, and it is still a point of debate exactly how much role the magnetic bottling plays in altering the sheaths relative to the approximate wall conditions our simulations employ. It is encouraging to note that the experimental results themselves do seem to indicate this reversal of the inner sheath in Figure 4-5(c). One final observation, this regarding the outer wall sheaths, is that the new simulation's wall condition does seem more effective than the PIC-MCC's. The sheaths near the outer wall appear to be somewhat deeper and rounder in Figure 4-5(b) than in Figure 4-5(a).

Of more interest, personally, was the relatively high-level of agreement between the present simulation and the former one. These plots represented the first real proof that our newly-developed model was fully-functional and properly operating during an actual, full-scale, system-level Hall thruster test. It is clear that the present model is able to accurately capture at least the same basic properties of the plasma that the PIC-MCC model could and that the many obstacles encountered during development have been sufficiently overcome.

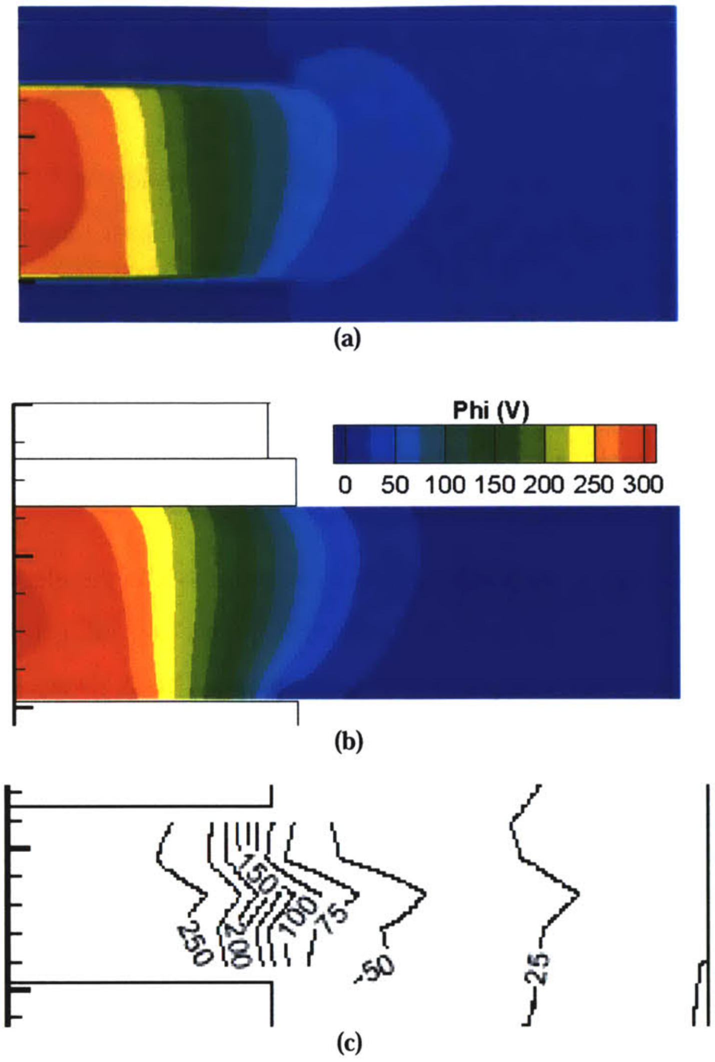


Figure 4-5: The electric potential from (a) the former PIC-MCC model, (b) the present simulation, and (c) the experiments of Haas.

4.2.2 Particle Densities

The figures below present particle density results from the present simulation, sometimes compared with the results of the PIC-MCC simulation. All results in this section were calculated under the assumption that the anomalous transport coefficient was a constant 1/400 everywhere in the thruster in order to allow direct comparison with the results of Sullivan [72].

Figure 4-6(a) gives the singly-charged ion density profile time-averaged over a single discharge oscillation. The peak density of $1.4*10^{12}$ cm^{-3} is slightly higher than the

$1.1*10^{12}$ cm^{-3} predicted by Sullivan and the former PIC-MCC code. This is thought to be due to the present simulation's more accurate coverage of velocity space. The high energy, statistically small particles that were being overlooked by the former simulation tend to increase the ionization levels in the present simulation, and bring them more into line with experiment, which predicts a peak ion density of approximately $2.5*10^{12}$ cm^{12} [37]. As in most cases, however, the constant Bohm coefficient assumption tends to produce temperatures and the plasma densities below that of experiment. Figure 4-6(b) shows the same trend in the electron density which more or less mimics the ion density as expected.

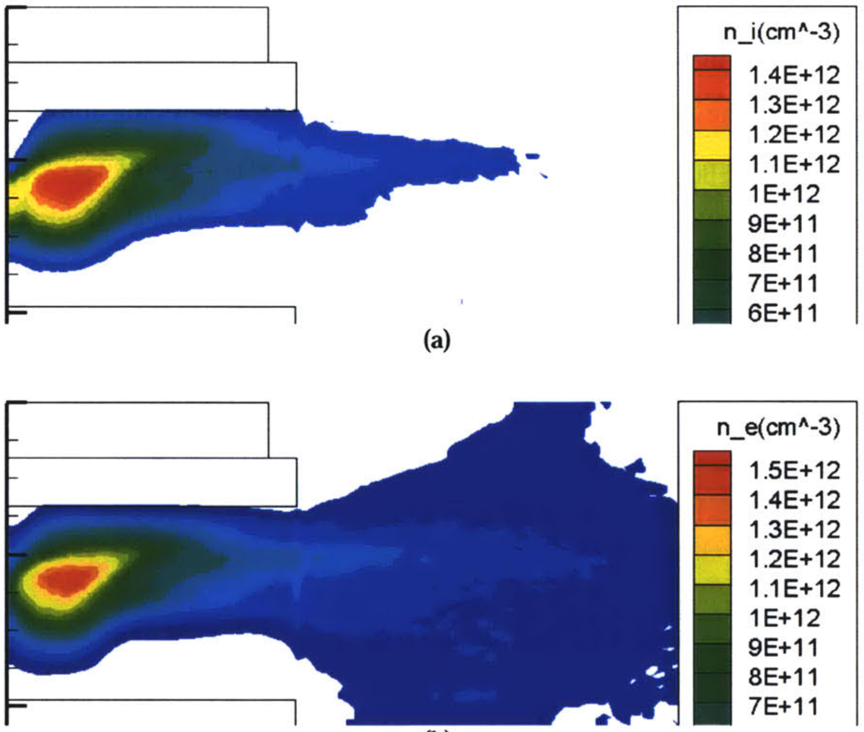
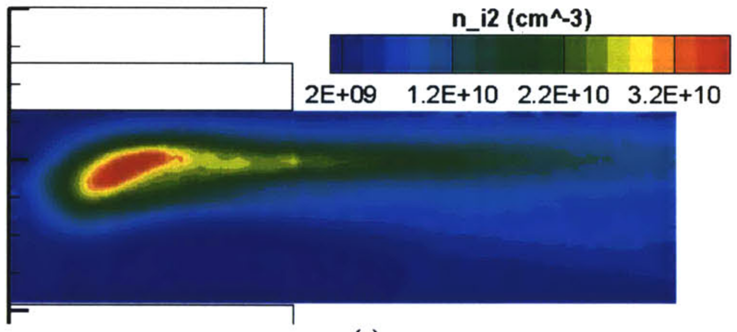


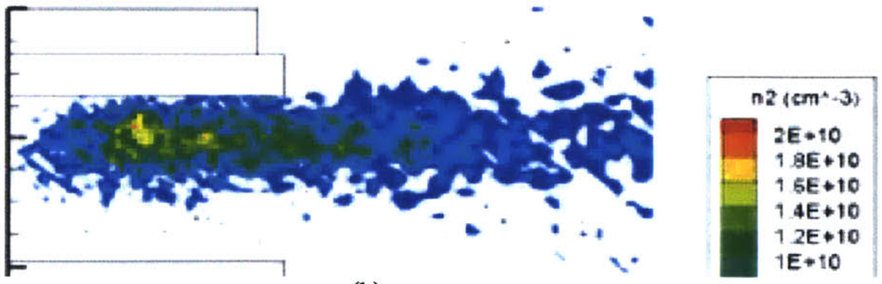
Figure 4-6: P5 Plasma densities calculated by the present simulation. The (a) ion density and (b) electron density are slightly increased relative to previous simulation results due to a better coverage of velocity space.

More impressive is Figure 4-7, comparing the present simulation results in Figure 4-7(a) to the PIC-MCC results of Sullivan in Figure 4-7(b) for the doubly-charged ion densities in the channel. Our new simulation is seen to be exceptionally smooth when compared to the noisy, hit-or-miss PIC-MCC results. This is clearly a benefit of our improved velocity space coverage and statistically more accurate number of particles per cell. In addition to the smoothness of these results, the new simulation predicts nearly double the density of doubly-charged ions that the former simulation did, though the ratio of double ions to single is still probably too low in the code. We predict that the double ion flux is perhaps 6-7% that of the single ion flux while experiments typically find 10% or more of the flux is doubly-charged [37]. Again, this is related to the improved velocity-space coverage and the increased energy available in the simulation in comparison to the PIC-MCC model.

The final plot in this section, Figure 4-8, simply plots the neutral density for the P5 thruster calculated using the present numerical model. The peak density is, of course, near the gas injection point and attained a value of about $1*10^{14}$ cm^{-3}. The figure also demonstrates that the neutrals near the top of the chamber are being ionized more than those near the bottom. This feature of the computation was identified by Sullivan to be caused by the significant magnetic bottling effect due to the strong B_z field near the inner wall of the thruster [72].



(a)



(b)

Figure 4-7: The doubly-charged ion density calculated by (a) the present simulation and (b) Sullivan's PIC-MCC code.

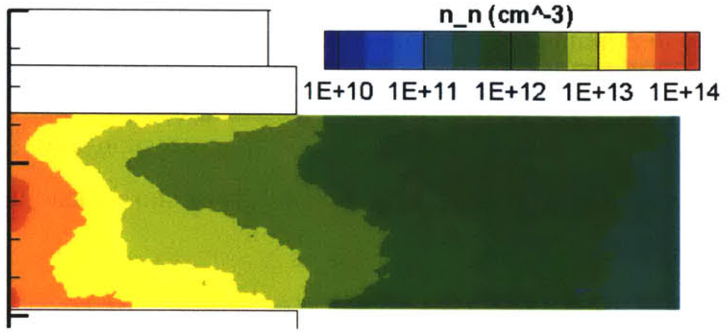


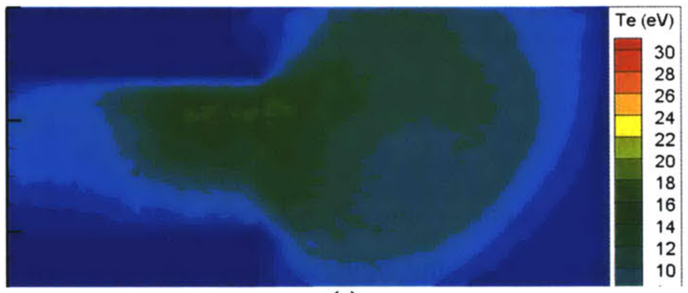
Figure 4-8: The neutral density in the P5 thruster calculated using the present numerical simulation.

4.2.3 Temperatures

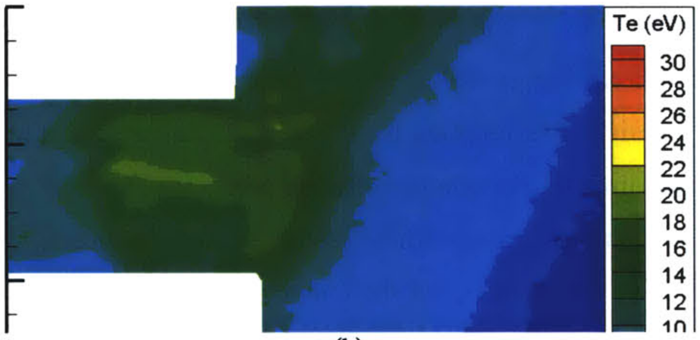
The electron temperature in the P5 has been determined using the PIC-MCC simulation, experimental probes, and now our new velocity-redistributing simulation. Figure 4-9 below compares the results of these three techniques. Figure 4-9(a) shows the computational results from the former PIC-MCC model with the constant Bohm coefficient assumption. Figure 4-9(b) shows the current model's results, also using the constant Bohm coefficient assumption for the sake of comparison. Finally, Figure 4-9(c) is again Haas' experimental electron temperature plot.

Both computational cases exhibit some similar features. The peak electron temperatures occur in the channel, but the temperature does not quickly drop off outside the channel as it does in the experimental case. Section 6.7 below indicates that this error in the computation is due to the Bohm coefficient assumption. The temperature near the anode tends to drop off in both computational cases as well. The experimental probe data does not reach this far into the channel. The peak temperature values in both computational cases are significantly less than the 30eV indicated by experiment, but again this is most likely due to the Bohm coefficient assumption.

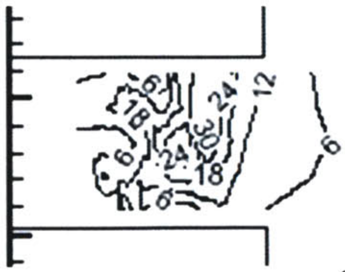
The current computational model also has some interesting new properties as compared to the old one. Figure 4-9(b) shows that the peak temperature for the new model case is on the order of 22 or 24 electronvolts while the former model predicted a peak of only 16 or 18 electronvolts. The new computational model has been designed specifically with the aim of more accurately simulating the velocity space, and it seems from these results that an improvement in agreement with experiments is the boon of that labor. Since the current simulation takes care to simulate the high energy particles in the tail of the velocity distribution, the calculated temperatures end up being higher and more accurate than the former model which was cutting off some of that tail, reducing its simulated temperatures. This 6eV difference is likely responsible for the increased amount of double ionization indicated in Section 4.2.2 and should lead to better estimates of thruster performance.



(a)



(b)



(c)

Figure 4-9: The electron temperature for the P5 thruster from (a) the former PIC-MCC simulation with constant Bohm approximation (b) the current simulation with constant Bohm approximation and (c) the experiments of Haas.

Chapter 5

Near Vacuum Hall Thruster Simulations

In this chapter we present an application of the numerical simulation developed and tested in the previous chapters of this thesis. The Near Vacuum Hall thruster is an exciting new concept which seemed the perfect subject for our computational model. It has received very little theoretical attention and future designers might significantly benefit from even the broadest overview of the thruster's plasma properties. In addition, as will be discussed, previous experimental results indicated that a severe design inefficiency was just waiting to be discovered which could drastically improve the engine's performance. With the objectives of demonstrating our simulation's capabilities, understanding the operation of the Near Vacuum Hall thruster, and hopefully improving the design of that device, we therefore turn our attention to the present chapter of this thesis.

5.1 Previous Experimental Work

The concept of a near vacuum Hall thruster has, to our knowledge, only been recently investigated by a small number of researchers here at MIT. It is, however, believed that earlier work, especially in the former Soviet Union, probably also exists that is unbeknownst to our group. Our first working prototype of an NVHT was designed and built by two undergraduate students, Tim Pigeon and Ryan Whitaker, working under the mentorship of members of MIT's Space Propulsion Lab [62]. Their work was treated simply as a proof of concept study. The prototype thruster was constructed and tested under several different near vacuum pressures, magnetic field settings, and anode voltages. The output the authors chose to measure was simply the total current collected by a metal plate downstream of the thruster. These data they then compared with a rudimentary theoretical model developed by Martinez-Sanchez. This model very accurately predicted the current-voltage characteristics of the thruster. Unfortunately, due to a small error in the manipulation of these equations, the authors believed they had obtained much poorer agreement with the theoretical levels of current predictions than they actually did. This minor update to that theoretical model is discussed below in Section 5.2.3.

Bouyed by the successful operation of that first NVHT, another team of undergraduate students attempted to improve upon the initial work of Pigeon and Whitaker. This pair, Regina Sullivan and Omar Bashir, used a thruster design very similar to their predecessors' with the major improvement being the inclusion of tunable electromagnets rather than the permanent magnets that had been used before [7]. More importantly, this team was able to secure an RPA probe and conduct a more detailed analysis of the ion energies in the thruster's plume. Their analysis indicated that, given their particular configuration and construction, the thrust produced by the NVHT was too small to compensate for a satellite's drag in LEO. The authors believed that this was largely due to the relatively low, 600-800 Volt, energy of the accelerated ions despite the 3-5 kV anode potentials used.

The discouraging computed thrust and ion energy results of this second study notwithstanding, the very concept of an electric propulsion device which would require no onboard propellant and therefore have, for all intents and purposes, an infinite specific impulse is very intriguing. Through the present work, we hoped to offer insight into how the current prototype NVHT could be redesigned in order to improve upon the thrust and efficiency measurements found in the above-mentioned studies. At the time of this writing, a third experimental endeavor is being undertaken by Lozano and an undergraduate, Aung, which incorporates the experience gained from our simulations to hopefully improve the NVHT's measured performance. More details about this effort can be found starting in Section 5.4.

5.2 Details of the NVHT

As described above, two separate NVHT configurations have been developed in the past. These experiments investigated various anode voltages, background pressures, and magnetic field settings. Table 5.1 gives estimated ranges for the nominal values of several important quantities from the experiments. Unless otherwise noted, we chose to simulate using Pigeon and Whitaker's "optimal" setting. This entailed an anode voltage of 3.172kV, an expected beam current of approximately 1.2mA, and a maximum magnetic field of 425 Gauss. We also chose to use those authors' maximum background pressure of $1.1*10^{-4}$ torr. In the Pigeon and Whitaker experiment, the background gas was Xenon while Sullivan and Bashir utilized Krypton. The experiments currently underway hope to use Argon. Since the P5 thruster simulations had already tested the Xenon branch of the code and associated cross-sections, we chose to use the Krypton branch for most of the NVHT studies in order to further demonstrate our model's capabilities.

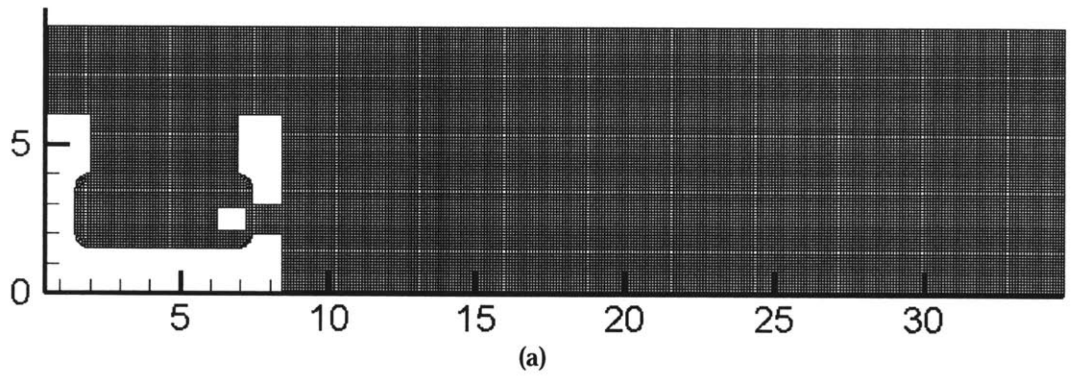
In the following sections, we further discuss the exact conditions that our initial NVHT simulations encompassed, along with the theoretical basis for those choices. Several modifications to the main simulation are also discussed that were incorporated for the specific purpose of more efficiently or accurately simulating this type of thruster.

Anode Voltage	1-5.5 kV
Beam Current	.1-1.2 mA
Maximum Magnetic Field	175-600 Gauss
Background Pressure	$3.3*10^{-6} - 1.4*10^{-4}$ torr

Table 5.1: Approximate ranges of important parameters tested in NVHT experiments.

5.2.1 Geometry and Mesh

The geometry used for the baseline tests of the NVHT was that of the prototype thruster built by Sullivan and Bashir [7] which was still available in the lab. The exact dimensions and schematic drawings of that thruster can be found in the reference. Our axisymmetric, unstructured quadrilateral mesh is shown below in Figure 5-1. The top figure depicts the entire simulation region while the bottom of the figure is a zoomed image of just the region around the anode and the acceleration channel. Note that both figures are in units of centimeters.



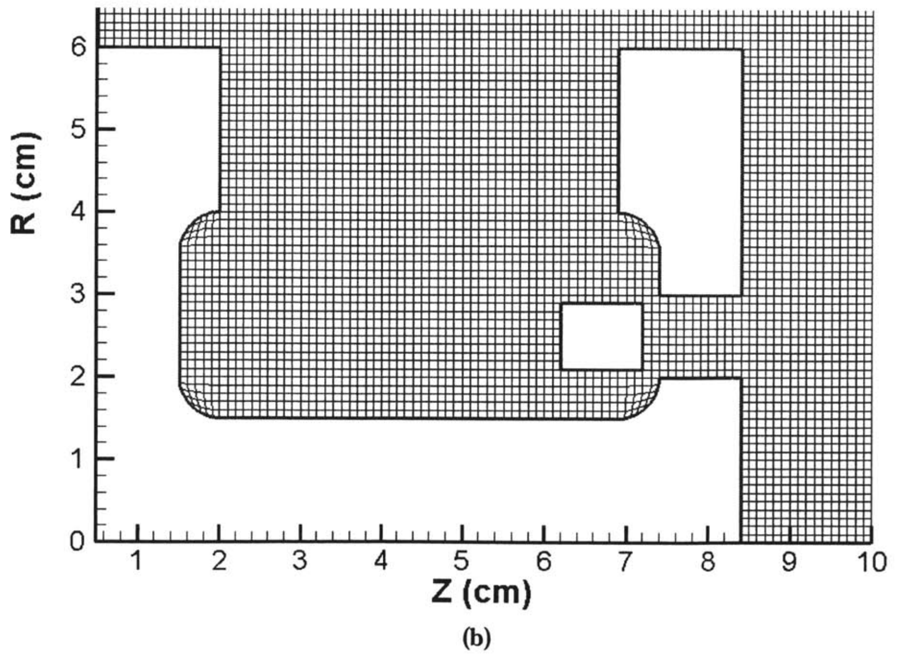


Figure 5-1: The base grid for the NVHT. Both (a) the whole simulation and (b) the zoomed region around the thruster are shown.

The reason that the grid was extended so far into the plume was in order to attempt to match the entire simulation domain that Pigeon and Whitaker's experiments encompassed, including the thruster and the metal collector plate which they placed approximately 30 cm from the thruster's exit. Great care was taken to ensure that the mesh cells remained as orthogonal as possible and of roughly similar size. This is not necessary given our non-orthogonal, unstructured solvers, but as mentioned in Chapter 3, better accuracy is usually obtained on simpler, orthogonal meshes. Where appropriate though, non-orthogonal elements were employed.

The unstructured nature of our gridding algorithm proved incredibly useful for modeling this sytem which has numerous void regions within the interior. A structured mesh would have been hard-pressed to simulate this geometry efficiently and could not have maintained the level of element quality that the unstructured mesh did. In addition, the large simulation region, with its highly varying plasma densities ranging from dense within the channel region to tenuous downstream near the collector plate, could only be efficiently handled through the use of a multi-gridded or adaptive meshing scheme such as the RRC technique we implemented. During trials, the region of high interest in and around the thruster's channel could be highly refined if necessary without requiring increased computation throughout the rest of the domain. Previous generations of MIT's Hall thruster simulations could not have effectively handled such an extensive and complex geometry.

5.2.2 Magnetic Field

The magnetic field for the near vacuum Hall thruster was modeled using the student version of Maxwell2D. This magnetic field may be considered to be very approximate as a number of assumptions were made in its creation. First of all, the electromagnetic solenoids were modeled instead as a permanent magnetic material. This made it possible to use the two-dimensional version of the Maxwell software. Also, in reality, there are six discrete solenoids, but the two-dimensional nature of our simulation forced us to approximate this system as being unchanging in the θ-direction. Thus, the magnets were modeled as a cylindrical shell, and the magnetic field was assumed to have no component out of the RZ-plane. This is not unusual for this type of simulation and is the same assumption commonly made in past models [73]. Finally, the field was scaled according to the maximum magnetic field data obtained experimentally by Sullivan and Bashir [7]. The instrument they used to take those measurements had very poor spatial resolution, giving their reported maximum magnetic field, and therefore our simulated one, wide error bars.

With these caveats in mind, the magnetic field used for the trials that follow is shown in Figure 5-2. Note that the field falls off rapidly outside the region of the channel and so only a zoomed, localized version of the entire field is depicted here. The field was calculated on a regular grid in Maxwell, and a utility was then written to interpolate that field onto our non-orthogonal, irregular mesh. Some errors, especially near the boundaries where steep gradients exist, may have been incurred in this step as well. The norm of the magnetic field is shown and the heavy black streamtraces are provided to give an indication of the field's direction within the channel. The lines, as modeled, are practically straight with only very small relative B_z at points within the channel's midsection. The modeled field lines also appear to bow inward rather than outward which would usually be the case in Hall thrusters. A better model of the field or an actual measured field would definitely be a future improvement to the simulation, but for now, the present structure will suffice.

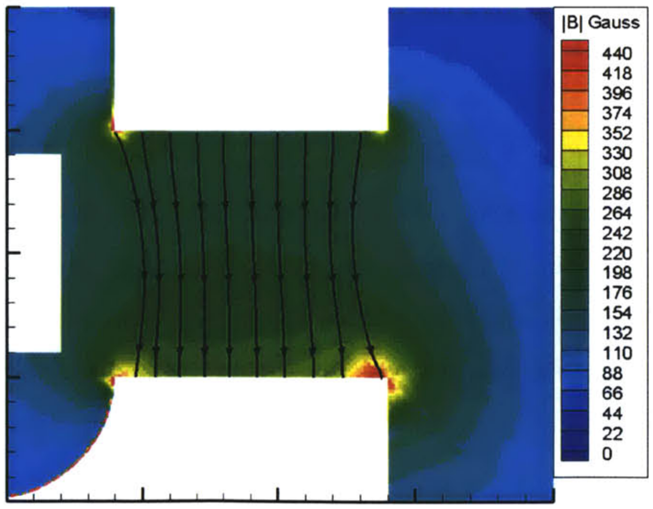


Figure 5-2: The norm of the magnetic field in the channel of the NVHT.

5.2.3 Update to the Theoretical Model for the NVHT

In order to gain a better understanding of some of the expected results of our simulations, it was helpful to re-examine the theoretical model detailed by Pigeon and Whitaker [62] which takes a relatively simplistic, one-dimensional view of the thruster and attempts to discern several important operating characteristics from these assumptions. We here present the basic steps of that model, including the small arithmetic correction, and that model's updated agreement with the previous experimental data.

The first step in the derivation is to assume that the electron density in the channel of the NVHT is constant, both in space and in time. We then also make the assumption that the electric potential only varies in the axial direction. With these two facts in mind, Poisson's Equation:

$$\nabla^2\phi = -\frac{\rho}{\varepsilon_0} \tag{5.1}$$

can be rewritten for the channel region as:

$$\frac{d^2\phi}{dz^2} = \frac{n_e e}{\varepsilon_0} \tag{5.2}$$

Integrating this equation twice, we can find the maximum electric potential allowable by this model given that the potential falls to 0 at the exit of the channel which has an assumed length of Δ:

$$\phi_{opt} = \frac{n_e e}{\varepsilon_0}\frac{\Delta^2}{2} \tag{5.3}$$

The next step is to try to relate the electron density to the magnetic field in the hopes of obtaining an expression for the optimal voltage given a particular magnetic field and channel length. This is accomplished by realizing that at steady state, the number of electrons inside the channel region should not be changing, and therefore the flux of electrons out of the volume must be equal to the production rate inside the volume. In other words:

$$\begin{aligned}\Gamma_e &= production \\ A n_e v_\perp &= n_e n_n Q_i \bar{c}_e A \Delta\end{aligned} \tag{5.4}$$

Here, A represents the area of the channel exit plane, $v_\perp$ is the velocity of electrons in the axial direction perpendicular to the magnetic field, Q_i is the cross-section for ionization, and $\bar{c}_e$ is the average electron velocity.

To find a value for $v_\perp$ we assume only classical cross-field diffusion and no anomalous diffusion for the present. Under this assumption, each collision with a heavy particle causes an electron to move about one Larmor radius under the influence of the electric field. Thus, the axial velocity is given approximately by:

$$v_\perp \approx \frac{E}{B}\frac{v_e}{\omega_c} \tag{5.5}$$

where the collision frequency is just:

$$v_e \approx n_n \bar{c}_e Q_{tot} \tag{5.6}$$

If we now return to Equation *5.4*, substitute in our value for the perpendicular velocity, and simplify, we see that:

$$E = \frac{\phi}{\Delta} = \frac{eB^2}{m_e}\frac{Q_i}{Q_{tot}}\Delta \tag{5.7}$$

and therefore:

$$\phi = \frac{eB^2}{m_e}\frac{Q_i}{Q_{tot}}\Delta^2 \tag{5.8}$$

Setting this expression for the potential equal to that derived in Equation *5.3*, we can solve for the electron density to find:

$$n_e = 2\varepsilon_0 \frac{Q_i}{Q_{tot}}\frac{B^2}{m_e} \tag{5.9}$$

Comparing this expression to Pigeon and Whitaker's Equation (2) which was supposedly derived under the same assumptions, we see that the only difference is the location of the constant factor of 2. Thus, many of the previous author's calculations were off by a factor of 4, a fact which led them to believe that the model was drastically under-predicting the thruster's current and caused them to select too low of an "optimal" voltage given their magnetic field setting, as we will see in a moment. We therefore proceed along lines similar to their analysis and attempt to derive the correct estimates for the NVHT's operating parameters.

So, as mentioned above, our initial goal is to obtain an expression for the optimal anode voltage setting given a particular magnetic field. Under the assumption that the magnetic field is constant in the thruster channel, we substitute Equation *5.8* into Equation *5.3* to find:

$$\phi_{opt} = \frac{e\Delta^2}{m_e}\frac{Q_i}{Q_{tot}}B^2 \tag{5.10}$$

For the sake of comparison, we accept the previous author's value of 2/5 for the ratio of cross-sections. The NVHT has a channel width Δ=1cm, which information allows us to create Table 5.2 below, similar to table (1) in [62]. The values for the optimal voltages in this table are essentially four times those calculated by Pigeon and Whitaker. Under this model then, it would appear that previous tests of the NVHT have been performed at sub-optimal voltages which is corroborated by Pigeon and Whitaker's own statement: "testing revealed that the current continued to rise beyond the optimal voltage" [62].

B (Gauss)	425	354	283	213	175
ϕ_{opt} (kV)	12.7	8.81	5.63	3.19	2.15

Table 5.2: Theoretical NVHT optimal voltages for given magnetic fields.

The next parameter that we can calculate is the expected anode current. For this, we assume that this current will be given in steady state by just the electron charge multiplied by the production rate. In equation form, this is just:

$$I = e(n_e n_n Q_i \bar{c}_e)(A\Delta) \tag{5.11}$$

If we assume that the average electron velocity is about half the maximum attainable velocity under the assumed potential:

$$\bar{c}_e \approx \frac{1}{2}\sqrt{\frac{2e\phi}{m_e}} \tag{5.12}$$

then we can substitute in our expression for n_e and find:

$$I = \left(\frac{e}{m_e}\right)^{\frac{3}{2}} \sqrt{2}\varepsilon_0 A\Delta \frac{Q_i^2}{Q_{tot}} n_n B^2 \sqrt{\phi} \tag{5.13}$$

Further, the form of the potential in equation *5.10* can be plugged into the above expression to find:

$$I \propto \left(\frac{e}{m_e}\right)^2 \frac{\varepsilon_0}{\sqrt{2}} A\Delta^2 \frac{Q_i^{5/2}}{Q_{tot}^{\ 3/2}} n_n B^3 \tag{5.14}$$

Using this expression, an exit plane area of $1.57*10^{-3}$ m^2, and a conversion of neutral gas pressure in torr to particle number density form, we can compute Table 5.3 below for the magnetic field settings and tank pressures evaluated by Pigeon and Whitaker. Note that these values are computed using not the optimal potential, but the potentials that Pigeon and Whitaker experimented with and believed were optimal.

	Background Neutral Gas Pressure (torr)				
		$3.3*10^{-6}$	**$8.8*10^{-6}$**	**$5.5*10^{-5}$**	**$1.1*10^{-4}$**
B (Gauss)	**425**	.037 mA	.099 mA	.62 mA	1.24 mA
	354	.021 mA	.057 mA	.36 mA	.72 mA
	283	.010 mA	.029 mA	.18 mA	.36 mA
	213	.0047 mA	.012 mA	.080 mA	.16 mA
	175	.0026 mA	.0068 mA	.0044 mA	.089 mA

Table 5.3: Theoretical currents for various neutral gas pressures and magnetic field settings. The potential used in these calculations is four times less than the optimal in order to allow comparison with the experiments of Pigeon and Whitaker.

If we compare these calculated values to the experimental results, as well as the currents predicted by the previous version of this theoretical model, we obtain Table 5.4. The agreement between the theoretical model and experiment has now been greatly improved. For better visualization, these results are also graphically represented in Figure 5-3.

	Pressure (torr)			
	$3.3*10^{-6}$	**$8.8*10^{-6}$**	**$5.5*10^{-5}$**	**$1.1*10^{-4}$**
Experiment	.0608 mA	.134 mA	.513 mA	1.191 mA
Previous Model	.0093 mA	.025 mA	.16 mA	.31 mA
Current Model	.037 mA	.099 mA	.62 mA	1.24 mA

Table 5.4: Comparison of the anode currents found by the former model, the current model, and experiment.

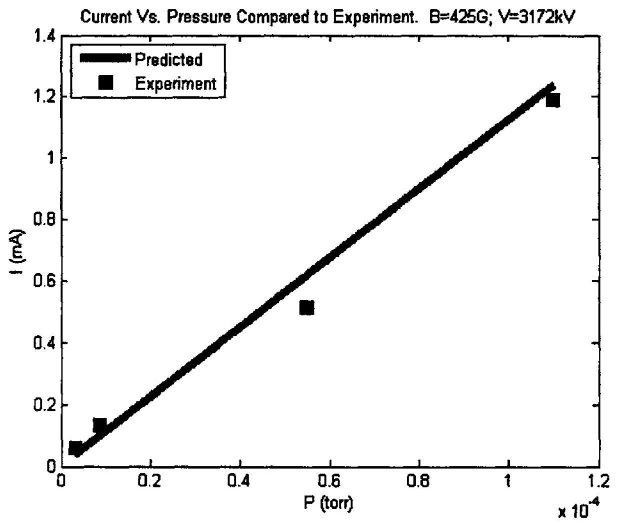


Figure 5-3: The updated theoretical model for the NVHT current variation with pressure agrees quite well with experiment.

The final performance parameter that we can derive is the expected thrust. For this, we assume ideal conditions in which each ion is created at the maximum potential. The thrust is then given by:

$$T = \dot{m}_i \bar{c}_i \approx \left[I \frac{m_i}{e} \right] \left[\sqrt{\frac{2e\phi}{m_i}} \right] = I\sqrt{\phi}\sqrt{\frac{2m_i}{e}} \tag{5.15}$$

This result indicates a square root dependence of the thrust on the potential, and describes an upper bound on the possible thrust given our assumption regarding the ion energies.

5.2.4 Mean Free Path Analysis

The rate of reaction between two particle species is defined to be:

$$R_{12} = n_1 n_2 v_{12} Q_{12} \tag{5.16}$$

where n_1 and n_2 are the species' densities, v_{12} is their relative velocity, and Q_{12} is the cross-section. If we assume that species 1 is colliding with a background density of species 2, then we have:

$$\nu_{12} = \frac{R_{12}}{n_1} = n_2 v_{12} Q_{12} \tag{5.17}$$

From here there are three different possible cases which lead to further simplifications.

If $v_1 >> v_2$, then $v_{12} \sim v_1$ and:

$$\lambda_{12} = \frac{v_1}{\nu_{12}} \approx \frac{1}{Q_{12} n_2} \tag{5.18}$$

If $v_1 \sim v_2$, then $v_{12} \sim 2v_1$ and:

$$\lambda_{12} = \frac{v_1}{\nu_{12}} \approx \frac{1}{2Q_{12} n_2} \tag{5.19}$$

If $v_1 << v_2$, then $v_{12} \sim v_2$ and:

$$\lambda_{12} = \frac{v_1}{\nu_{12}} \approx \frac{v_1}{Q_{12} n_2 v_2} \tag{5.20}$$

These three equations, along with the relevant data and approximations thereof, allowed us to calculate the mean-free-path values for the NVHT reported in Table 5.5.

Given the mean-free-path of a collisional process, we can also calculate the Knudsen number by simply:

$$Kn = \frac{\lambda}{L} \tag{5.21}$$

where L is taken to be the characteristic distance for macroscopic gradients or some other characteristic distance within the thruster. For our purposes, we will take L to be the approximate path length of a particle over its lifetime in the thruster. Since the general path length of an electron in a Hall thruster is more or less determined by collisions, the definition of Knudsen number is somewhat muddied by that definition of L, and so we will report Knudsen numbers only for the heavy species and simply examine mean free paths for the electrons. These values are all estimated in Table 5.5.

With this information at hand, we can begin to assess the importance of the various types of collisions which might occur in the Near Vacuum Hall thruster, and decide whether or not each type is necessary for accurate simulation.

e-n collisions: The peak values for the cross-sections of these types of collisions were used in the calculations shown in Table 5.5 which follows. The neutral density within the NVHT is approximated from the background vacuum chamber pressures reported by Pigeon and Whitaker [62]. Since the electron velocity is so much greater than the background neutrals, the first equation above, Eqn. *5.18*, was used to calculate the remaining relevant values.

e-i coulomb: The cross-section for this type of collision can be derived from the theoretical Rutherford differential cross-section. Szabo details the derivation of this cross-section [73]. An estimated value for a 30 eV electron is given in Table 5.5. The ion density for this type of collision is taken to be close to the electron density in the chamber which is assumed to peak at about 1e10 cm^{-3}, as given by the updated analytical model presented in Section 5.2.3. Again, Eqn. *5.18* above has been used since the electron velocity is so much greater than that of the ions.

e-e coulomb: These collisions have the same theoretical development as e-i coulomb collisions and so the cross-section is found in the same way, except that now the velocities are of the same order and so Eqn. *5.19* is now used to complete the calculation.

i-i coulomb: These collisions are most likely of little importance in the NVHT. This can be seen by realizing that the reduced mass for these collisions is the ion mass which is several orders of magnitude larger than in the electron cases. This makes the b_0 parameter from Rutherford scattering very small, and thus the cross-section. The cross-section presented here is taken from [85].

e-i non-coulomb: Other possible interactions between the plasma species have relatively very small cross-sections and so have been neglected here.

	Q_{12} cm^2	n_2 cm^{-3}	λ_{12} cm	L cm	Kn
e-n scat	4e-15	1e11	2e3	-	-
e-n ioniz	5e-16	1e11	2e4	-	-
e-n excit	4e-16	1e11	2.5e4	-	-
n-n scat	4.9e-15	1e11	2e3	1	2e3
i-n chX	5.9e-15	1e11	1.7e3	1	1.7e3
e-e coul	7.2e-16	1e10	1.4e5	-	-
i-n scat	1.8e-15	1e11	5.5e4	1	5.5e4
e-i coul	7.2e-16	1e10	1.4e5	-	-
e-i scat	2.5e-16	1e10	4e5	-	-
i-i coul	7e-21	1e10	1.5e10	1	1.5e10

Table 5.5: Summary of various collisional processes in order of increasing Knudsen number and therefore relative "importance."

From Table 5.5 and the analysis that created it, it appears that very few collisions are actually important for the modeling of the NVHT. A collision type can be seen as important if its Knudsen number is small, generally less than 1 or, in the case of electron collisions, if the mean free path is not too much larger than the mean free path of the electron-neutral collisions which generally cause the particle to move toward the anode.

This would imply that a particle generally would undergo more than one of these collisions during its lifetime in the simulation region. Therefore, the three types of collisions that are necessary for accurate simulation include the three electron-neutral interactions. Coulomb interactions between electrons and ions and self-interactions between electrons are also seen to be marginally significant in the NVHT. Based on this analysis, we chose to model only the electron-neutral interactions while ignoring the Coulomb interactions as a simplification of this first simulation attempt. The addition of Coulomb interactions to the model is probably a fairly simple addition that future generations of coders may wish to incorporate.

5.2.5 Special NVHT Simulation Conditions

Our computational model was developed to be as general as possible in order to enable its use for a wide range of plasma applications. In order to simulate the NVHT then, there were a number of small additions to the code that needed to be implemented to make our general plasma model specific enough to be efficient and accurate on this particular thruster. These changes are discussed below.

5.2.5.1 NVHT Implicit Cathode Condition

The basic methodology for implicitly simulating a beam-neutralizing cathode is discussed in Section 2.10.3 above. However, the NVHT, under the initial experimental conditions, does not utilize a hollow cathode as do most Hall thrusters. In fact, there was no intentional mechanism incorporated in the experiments in order to neutralize the beam. In space, of course, the cathode would be necessary in order to neutralize the charge on the spacecraft. However, with the thruster being fired inside the vacuum tank and grounded, it is thought that some sort of emission from the tank walls or, more likely, the metal collector plate placed downstream as a beam collector is functioning as a neutralizing source of electrons. The exact physics of these sources being unknown and unstudied, our most obvious course of action was to revert to that which we have done before. Thus, a condition for the implicit cathode very similar to the P5 condition was employed for the NVHT tests. The main concept that an electron source is neutralizing the beam at a downstream point is common to both applications, and so it was felt that this condition would be suitable.

The cathode plane was then chosen to be the right-hand side and top boundaries of the simulation, however those were defined for the particular test run. Initially, the entire simulation domain shown in Figure 5-1 above was used for trials, but when it was realized that most of the important physics we were interested in for re-designing the thruster happened close to the channel, the cathode plane was moved inward to the 12cm location on the mesh. Thus, any particle reaching this 12cm point was deleted and added to the beam current and/or thrust. If the charge in any of these cells became positive and the potential calculated for those cells was not less than zero, then electrons were injected to neutralize the beam. The potential was still calculated for the entire domain of Figure 5-1, with the collector plate downstream set to 0 Volts. This somewhat shortened domain helped reduce computational effort, and since the downstream physics were not thought to significantly affect the near-anode physics, it was felt to be an acceptable trade.

5.2.5.2 Background Neutral Density Approximation

Another savings in computation was garnered by this further NVHT-unique approximation. The neutral number density of the background gas at $1.1*10^{-4}$ torr is simulated to be approximately $3*10^{18}$ m^{-3}. By contrast, the maximum plasma density in the channel calculated using the updated theoretical model given in Section 5.2.3, is only about $1*10^{16}$ m^{-3}. Thus, it seemed wasteful and very time-consuming to model the neutral dynamics which would have also required a great deal of approximation anyway in order to model the vacuum tank's pumping actions, the introduction of new neutrals, and the removal of others. With the neutrals being so complicated to approximate anyway and under the assumption that their density would not be significantly altered by the thruster's operation, we opted to simulate a static background neutral density rather than individual neutral particles. This approximation was made for this thruster only and should certainly be reviewed depending upon the application one wishes to simulate.

The background density of neutrals is specified by the user in the *input.txt* file in units of torr. This allows simulations to be conducted at any of the experimental pressures investigated by Pigeon and Whitaker or Sullivan and Bashir. For the purposes of neutral-electron collisions, this background density is taken to be spatially and temporally constant. An optimally designed thruster for use in space would likely try to ionize a significant portion of the neutral gas it could collect though, and if future studies attempted to simulate a device at the lower neutral densities that could be expected on-orbit, the neutral population may be significantly depleted in certain locations and should then at least be modeled using a simple fluidic assumption. However, since the plasma density is so small compared to the neutral density in the laboratory cases, it is believed that we are not excluding any important physics via this assumption.

5.3 Initial Simulation Results

Given the above configuration of the NVHT, we first wished to see how well our simulation could reproduce the experimental results of the two teams of authors discussed above. The following sections present a subset of our computational results investigating some of the more important aspects of the NVHT and its nominal operation. The observations and analyses made during this process drove the re-design effort of the NVHT which is discussed further in Sections 5.4 and 5.5.

5.3.1 Electric Potential

Figure 5-4 below plots the calculated electric potential in the near-anode region of the NVHT. Note that the color scale on the figure is not linear, and there are many more contour levels between 0 and 1000 Volts than between 1000 and 3000 Volts. The plot was constructed this way because the potential falls so rapidly through the thruster's channel. The metal walls that make up the top and bottom of the channel float at very low, less than 50 Volt, potentials. This means that the 3.172kV anode potential is sitting just .2 cm from the practically 0 Volt metallic walls. This inexorably causes the very steep potential gradients that are observed near the anode and also near the edges of the thruster. While the strong "sheaths" are beneficial in that they very strongly confine the electrons and limit electron losses to the walls, they are so large that they tend to give any ions formed in that region very significant radial velocities. More importantly, the

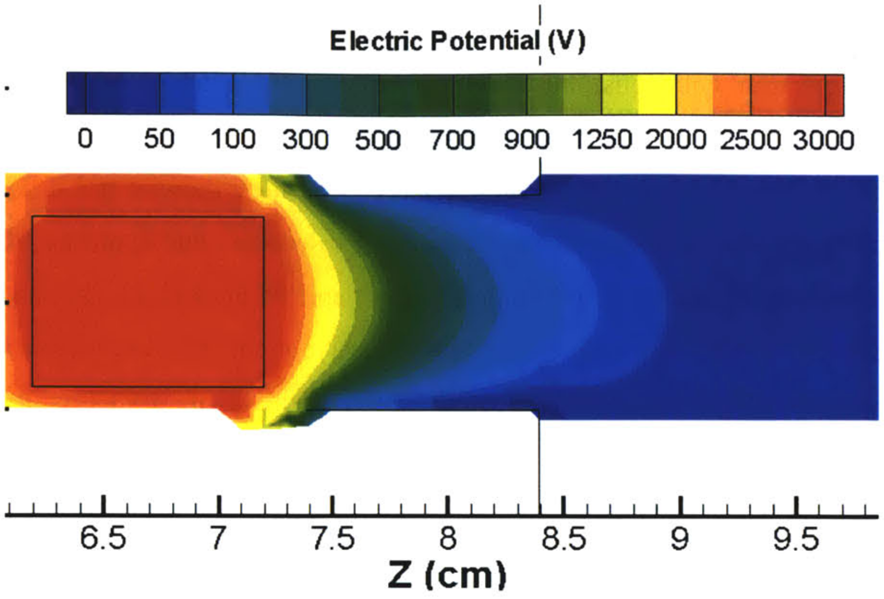
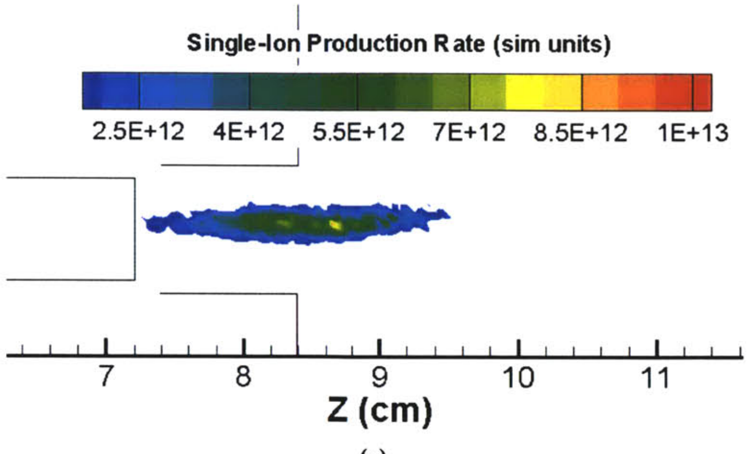


Figure 5-4: The electric potential in the near-anode region of the NVHT. Note that the scale in this figure is not linear.

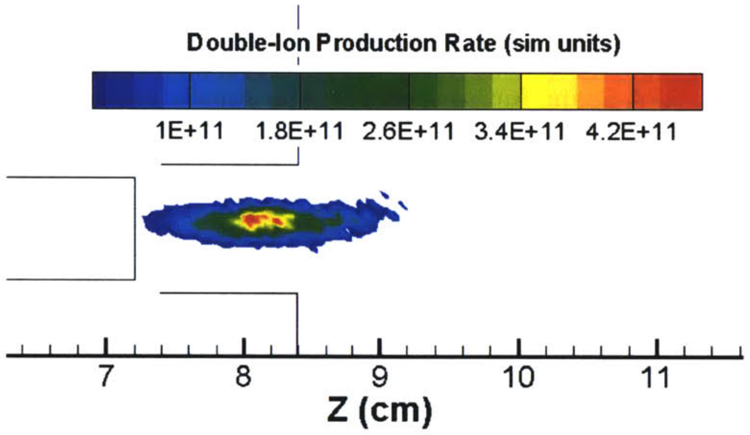
potential falls off so quickly in the channel that there is almost no chance that the ions could be created at the high anode potential. Instead, as we see in the next section and as was observed in experiment, the ions are created at a much lower energy. This observation of the poor electric potential structure played a non-trivial role in the re-design effort for the NVHT discussed later in this chapter.

5.3.2 Ionization Zone

One of the most interesting and useful results that is output by the simulation is the location and intensity of the various types of collisions occurring during the simulation. Specifically, we examined the location of the single and double ionization events. These values are plotted below in Figure 5-5, with Figure 5-5(a) being a plot of the neutral to singly-charged ion events and Figure 5-5(b) being a plot of the neutral to doubly-charged ion events. What is important to notice about these plots is how far down the channel and even outside the channel, the greatest portion of ions are being produced. Relatively few ions are produced very close to the anode at the high 3000 Volt potential. Instead, most of the ions are produced in a region where the potential has fallen to somewhere between 100 and 700 Volts, as can be seen when these figures are compared to Figure 5-4 above. This is not the way the thruster was anticipated to work given the



(a)



(b)

Figure 5-5: Plots of the regions of (a) singly- and (b) doubly-charged ion production in the original NVHT design.

design. It was hoped and believed that the ions would be produced inside the channel at a much higher potential, greatly increasing the thruster's specific impulse. The fact that the ions are being created at low potentials as observed in experiment and further verified by our computation is a very strong indication that the thruster may benefit from a re-design effort.

5.3.3 Ion Energies

The second MIT team which experimented on the NVHT, Bashir and Sullivan [7], conducted an RPA analysis yielding data on the energy of the ejected ions. Those data, while somewhat suspect in their own right due to a non-physical dip in the current-voltage derivative, indicated that the majority of the ions were being created at low potentials, on the order of 500-1000 Volts. Significantly, the ion energies remained the same even if the anode voltage was varied from 2 to 5 kV. This would seem to indicate that increasing the anode voltage above approximately 1000 Volts would no longer increase the output energy of the near vacuum Hall thruster's ions. The ion energy may be being limited by some physical process, such as the decrease in the ionization cross-section at high energies. Those experimental RPA data are reprinted here as Figures 5-6 and 5-7 for the sake of comparison with the computational data.

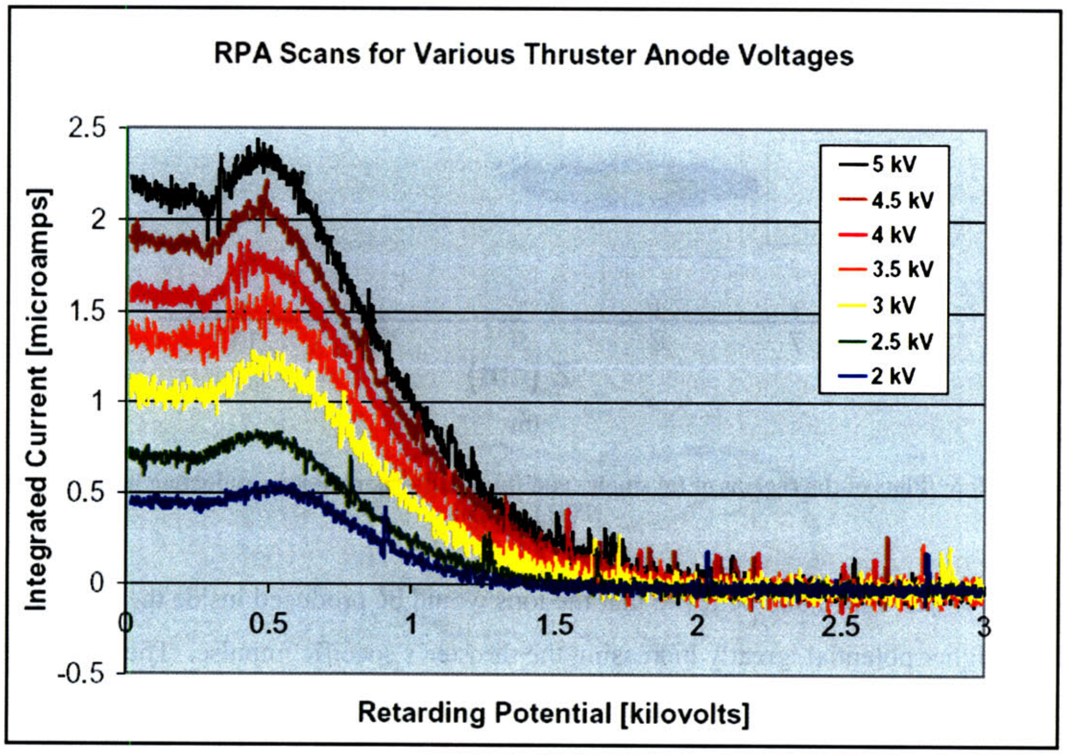


Figure 5-6: The raw RPA data reprinted from Sullivan and Bashir [7]. The region of highest derivative represents the most common energy of collected ions. Even with very high anode voltages, this high derivative region did not appreciably move to higher energies as expected. Note that these plots should be monotonically decreasing, but show an anomalous peak near 500 Volts.

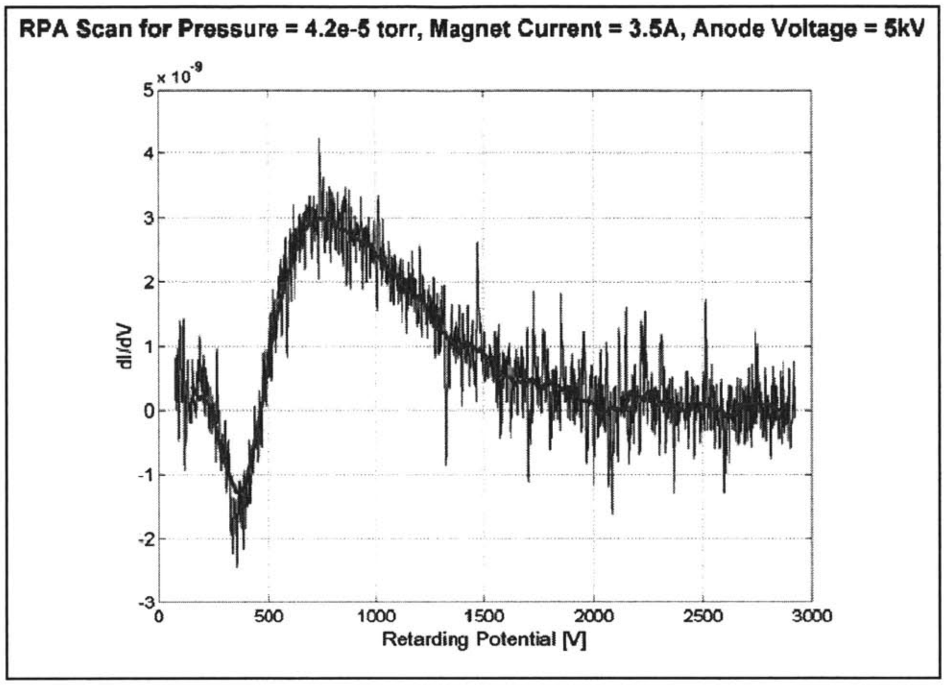


Figure 5-7: The derived current vs. voltage data from Bashir and Sullivan's RPA analysis [7]. The peak ion energy occurs at less than 1kV despite the 5kV anode potential.

Since these experimental results were so important, we very much wanted to see how well our simulation could reproduce them. Figure 5-8 below shows such an attempt. The plot is the derivative of the current that might have been collected by the RPA at each different voltage setting. This is the final output of an RPA analysis and is comparable to Figure 5-7 above reprinted from Bashir and Sullivan's paper. The two figures cannot be compared directly unfortunately since that published result had a much lower background tank pressure and a 5kV anode potential rather than the 3.172kV setting we used in order to match Pigeon and Whitaker's experiments. However, our Figure 5-8 does show similar features to the experimentally-derived one. There seems to be a large population of ions right around the 500-1000 Volt range as was seen in experiment. The major difference between our plot and the experiment was that below 500 Volts our plots showed that the derivative kept increasing, meaning that the majority of ions were being created with even less than 500 Volts of energy. This also agrees with our plots of the ionization zone and electric potential. Sadly, all of Bashir and Sullivan's data for less than 500 Volts was corrupted by some unknown effect believed to be the fault of the RPA itself [7]. Their data for integrated current seems to decrease when moving from 0 to 300 Volts before increasing, an artifact that cannot be physically indicative of the actual current in the beam. At some point, this experimental glitch should be re-investigated. Until then it is very difficult to tell whether their derivative might have continued to increase as our simulation predicts. At any rate, what is abundantly clear from both experiment and simulation is that the majority of the ions are not being created upstream of about the 1000 Volt point.

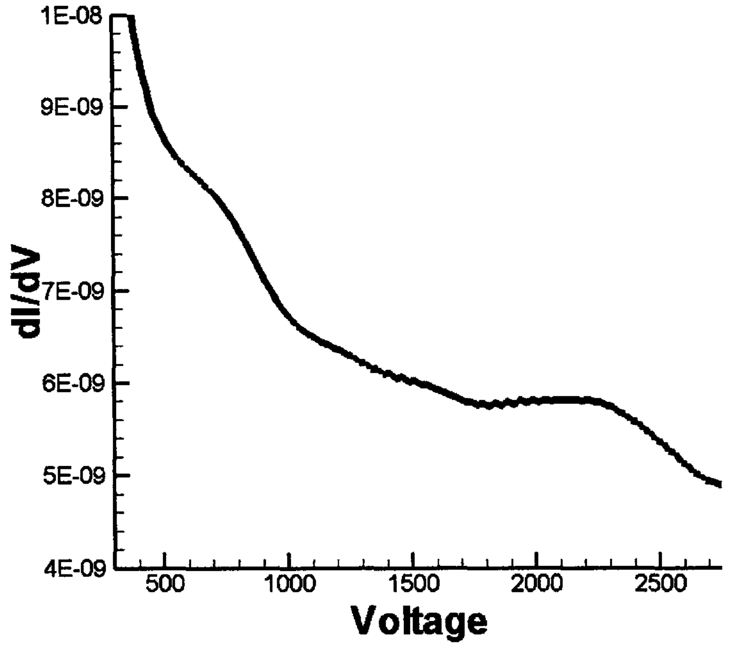


Figure 5-8: The derivative of the current with respect to voltage as the voltage on the simulated RPA was varied. This trial was conducted with a simulated tank pressure of 1.1*10^{-4} torr and an anode voltage of 3.172kV.

5.3.4 Particle Densities

The densities within the NVHT were theoretically approximated using the model given in section 5.2.3. That model predicted an electron density of 10^{10} cm^{-3}. Figure 5-9 below presents the densities of the three species kinetically simulated in the NVHT, the (a) electrons, (b) singly-charged ions, and (c) doubly-charged ions. The peak electron density is seen to be about 9.5*10^9 cm^{-3}, very near to the theoretically predicted value. The structures of all three densities are very interesting, however. The electron density in Figure 5-9(a) is very narrow. This is most likely an effect of the potential in the thruster's channel. The strong potential drop near the top and bottom of the channel create a field which strongly confines the electrons in the radial direction, even out into the plume. All three of the density peaks are located downstream of the actual channel and of the ionization zone. Examining the ion densities closer, we see that they are in general much less confined than the electrons. Both species have a tendency toward the positive radial direction, again most likely due to the field structure in the channel where they are produced.

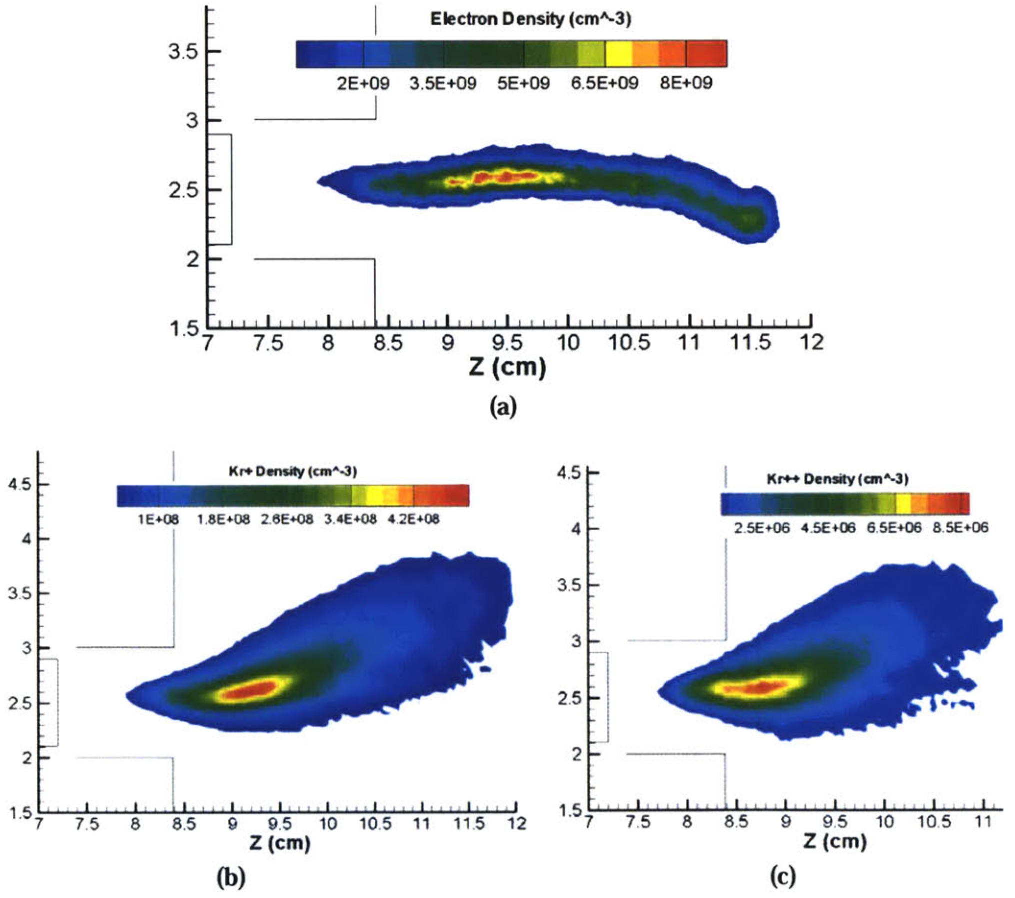
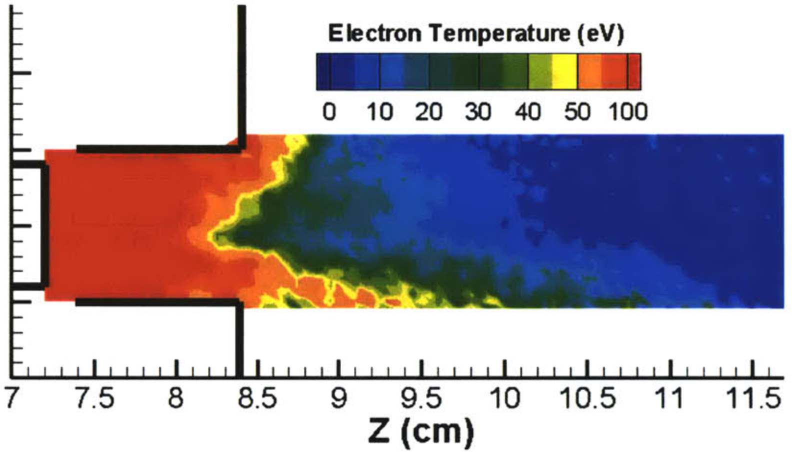


Figure 5-9: The (a) electron, (b) singly-charged ion, and (c) doubly-charged ion densities for the simulated near-vacuum Hall thruster.

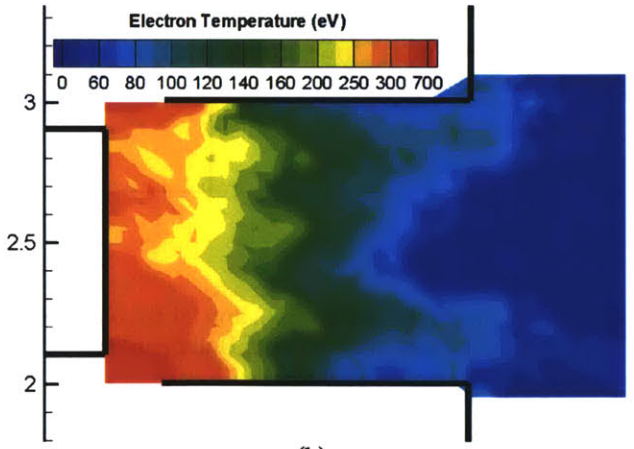
5.3.5 Electron Temperature

Figure 5-10 below presents the electron temperature calculated in the NVHT. The zoomed out case in Figure 5-10(a) has a scale which is non-linear and maxes out at

100eV. Figure 5-10(b) gives a blown up picture of the channel region with a more suitable scale. The temperatures of the electrons are calculated to reach as much as 500 electronvolts very near the anode. This is due to the 3 kV anode potential that experimentally was required. Other than the unusually high electron temperatures, the only other interesting feature of the temperature profiles is that the ionization region "peak" in temperature is nearly swamped and unobservable due to the high anode potential accelerations. In Figure 5-10(a), it does appear that the temperature has a 30-40 electronvolt "peak" in the center of the channel just before 9 cm, and this corresponds to the highest region of ionization.



(a)



(b)

Figure 5-10: The computed electron temperature for the near vacuum Hall thruster. (a) The temperature outside the channel is generally less than 100eV while (b) near the anode it can reach as much as 500eV.

5.4 Re-Design Recommendations

Throughout the previous result sections, a number of observations were made regarding the configuration of the NVHT and its possible impact on the thruster's overall performance. Those observations are first gathered and summarized here and a new configuration for the thruster is suggested.

The most important observation which was made both experimentally and computationally is that the energies of the output ions are a great deal lower than the anode potential. In the experimental case, Sullivan and Bashir's data [7] suggested that the major portion of ions was being created between 500 and 1000 Volts. Those data were unfortunately erroneous below 500 Volts. Otherwise they may have agreed with our simulation results that show that in fact the majority of ions are being created with even less than 500 Volts. We still see a significant number of ions between 500-1000 Volts and our data seem to match the trend of the experimental data above this point. Very few ions reach the beam with the full anode potential.

One possible reason for this was evidenced by our simulated plots of the electric potential. Due to the fact that the channel walls in the NVHT are metallic conductors, they tend to float at a generally low potential, on the order of 50 Volts. This low potential being situated so near to the very positively biased anode tends to quickly depress the potential in the channel. This depression was much more rapid and more influential than the theoretical model or the designers predicted. Due to this configuration, only a very small volume of the channel is actually even exposed to the high potentials caused by the anode, making it unlikely that the ions will be created at high energies.

The NVHT had been designed as a prototype and a proof of concept. As such it was designed quickly and without careful attention to a common distinction between the two types of Hall thrusters. The SPT type thruster, like the P5 discussed above, is usually characterized by a relatively long acceleration channel and ceramic, insulating walls. The ceramic helps keep the potential high through the channel. It is able to attain a high potential near the anode and a low potential near the exit because it is an insulator. The other type of Hall thruster, the TAL type, is constructed using metallic walls and a much shorter or even non-existent acceleration channel. The floating channel walls are kept short so as to not interfere as much with the high anode potential. Unfortunately, our NVHT was designed with the improper combination of these two traits. It has a long, metallic channel. This then seemed like the primary element that could benefit from a redesign.

While a number of other recommendations were also originally made, mostly concerning the thruster's magnetic field design, the alteration to the channel length was the first to be adopted by our experimentalists since it seemed to offer the most probable benefit for the least amount of work. MIT undergraduate Aung with the help and guidance of Lozano has, at the time of this writing, begun the process of adapting the NVHT with a shorter acceleration channel. These experimentalists decided that actually trimming material off of the front plate of the thruster would be destructive and more time-consuming. Instead, they opted to lengthen the anode. Thus, the anode was re-designed to be thinner in radius and protrude into the channel. Figure 5-11 shows a computational mesh of the proposed new design. The only change between this figure and Figure 5-1 above is the extra ring which has been attached onto the front of the anode.

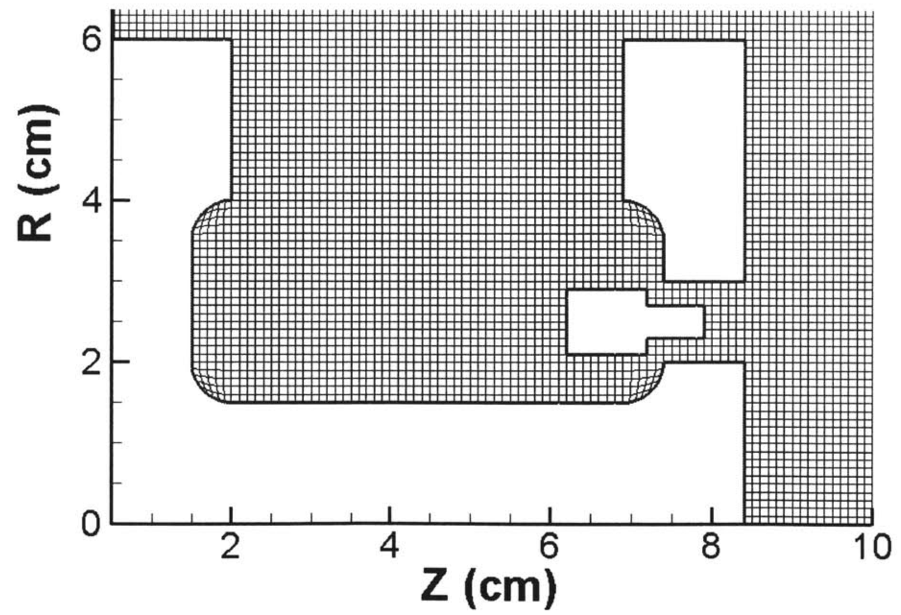


Figure 5-11: The mesh of the redesigned NVHT featuring an anode which protrudes farther into the channel.

The process of creating this new adaptation to the thruster and re-testing in the vacuum chamber is currently underway. There are, of course, some concerns that this simplistic implementation of our design recommendations may not produce the desired effects. The magnetic field has not been optimized properly for an anode protruding so far into the acceleration channel. There is a possibility that the anode will cross the region of strong magnetic field lines and electron loss to the anode will be increased. Also, the low voltage metallic walls are still present very near to the anode. This will still cause the potential to fall off quite rapidly away from the anode, though hopefully not as quickly as before. Despite these and other concerns, it is hoped that this relatively simple change to the thruster geometry will provide at least some improvement in performance or overall operation. With such evidence in hand, more complex and difficult alterations to the geometry could then be attempted.

5.5 Simulation Results After Re-Design

Of course, having made the above recommendations based on our original simulation results, it was only logical that we should also simulate the newly-designed thruster and attempt to determine what effect, if any, could be expected by the experimentalists. The results of those tests are given in the section which follows. Again, all tests were conducted with the settings described in previous simulations, an anode voltage of 3.172kV, background neutral pressure of $1.1*10^{-4}$ torr, and a maximum magnetic field of 425 Gauss. Lozano and Aung have discussed using Argon as their propellant gas and will likely be changing a number of other experimental conditions. As such, caution should be used if these simulation results are eventually compared with experiment. It will likely be more appropriate to simply compare them with the computational results provided above in Section 5.3.

5.5.1 Electric Potential

Figure 5-12 depicts the electric potential that was calculated for the newly designed NVHT. The protruding anode did accomplish its goal and has increased the potential near the exit plane of the thruster from the 100 Volts observed in the nominal case to about 800 Volts in this re-designed case. This is the region in which the former simulation predicted most of the ionization was occurring. Assuming that the ionization zone remained fixed under the re-design, the ions should then have been created at a much higher potential than before, increasing the specific impulse of the NVHT. As we will see in the next section, however, the static ionization zone assumption unfortunately does not seem to hold true.

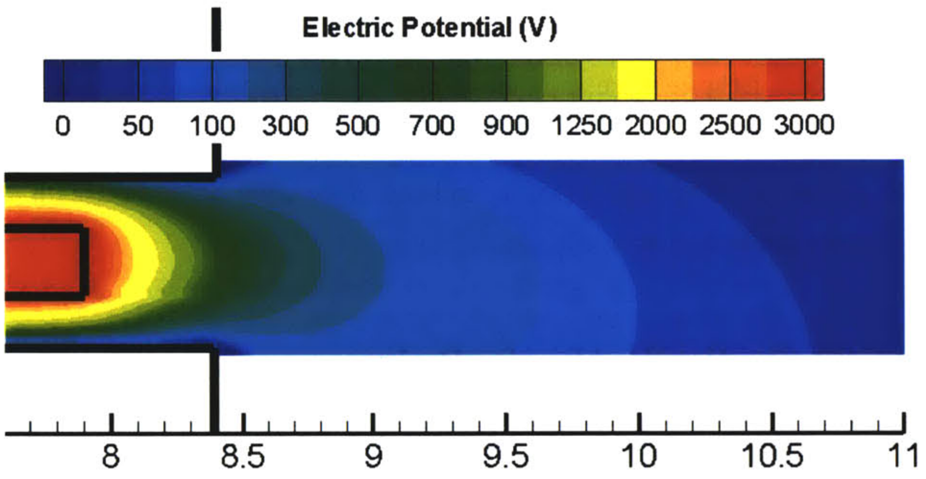


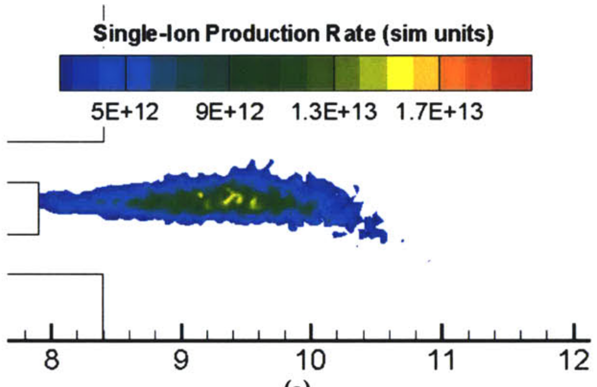
Figure 5-12: The electric potential in the re-designed NVHT.

5.5.2 Ionization Zone

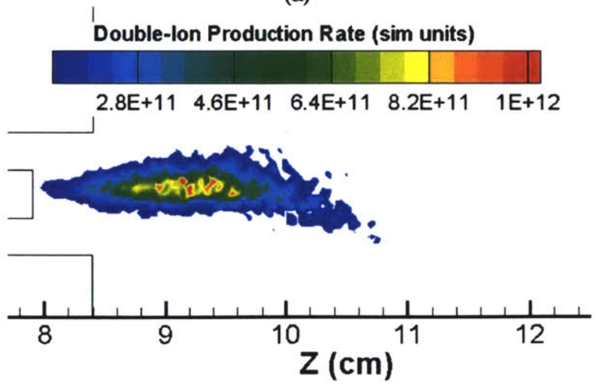
In the analysis that led to the thruster's first-cut re-design, it was assumed that the ionization region in the NVHT was located near the exit plane in the region of high magnetic field. Unfortunately, after some brief calculations of the Larmor radius in the thruster's channel given the magnetic field strength used, it was realized that the field was probably far too low to create good electron confinement. The electrons then do not remain long enough in the channel of the thruster, but simply zip on past to the anode. In addition, it appears that as soon as the electrons have gained a large enough energy, they have a high probability of ionizing the ambient neutrals. Therefore, by moving the potential forward, we seem mostly to have moved the ionization forward as well. This is demonstrated in Figure 5-13 below. The singly-charged ionization zone before re-design peaked just before 9 cm while in the protruding anode case the peak occurs at about 9.5 cm. A similar trend is also indicated in the doubly-charged region of ionization.

Apparently, the electrons are not likely to penetrate into the high potential regions without first undergoing some form of ionization collisions. At the same time, the cross-section of ionization, as shown in Section 2.8.3 for example, shows a sharp decline at higher energies. So electrons which do reach the anode region without colliding have so much energy that they may be whizzing right past the neutrals there, a fact that is exacerbated by their poor confinement in the magnetic field.

One relatively unexpected benefit of the re-design seems to be that the rate of ionization for both singly- and especially doubly-charged ions has been increased significantly. More ions being created means more thrust overall. This result is further borne out by Figure 5-14 later. It seems to indicate that the protruding anode design should be able to operate at lower anode potentials than the original design and will likely have noticeably improved performance at the same voltages in terms of total beam current, thrust, and also perhaps slightly increased ion energy, as the next section discusses.



(a)



(b)

Figure 5-13: The (a) singly-charged and (b) doubly-charged ion production regions for the newly-designed NVHT with protruding anode.

5.5.3 Ion Energies

The simulated RPA analysis for the re-designed NVHT is given below in Figure 5-14. The results from the previous design, Figure 5-8, were overlaid on the results for the new design in Figure 5-14 for ease of comparison. What is immediately observable is that the curve for the protruding-anode NVHT seems to be shifted out farther to the right by a small, but noticeable amount. This seems to indicate that the energy of the ions produced has been increased slightly as we had hoped the re-design would accomplish.

Despite this moderate increase, the majority of the current is still found in ions of less than 1000 Volts, though it seems the re-design has brought the energies somewhat closer to 1000 Volts. As discussed in the previous section in regards to the ionization zone placement, it appears our naïve adaptation of the thruster's anode will have to be rethought in greater depth before we can expect to see a more profound increase in the energy of the produced ions. However, even this slight change to the thruster design seems likely to slightly improve the ion energies in the thruster under experiment and should also produce a number of other benefits as well.

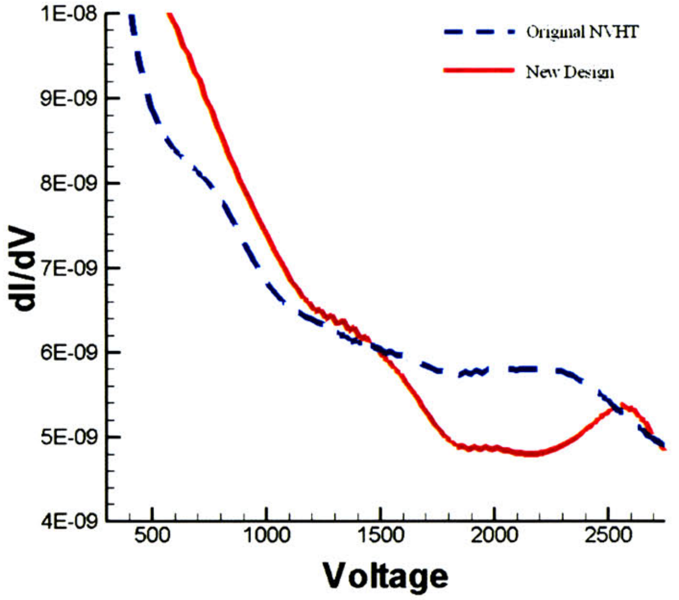


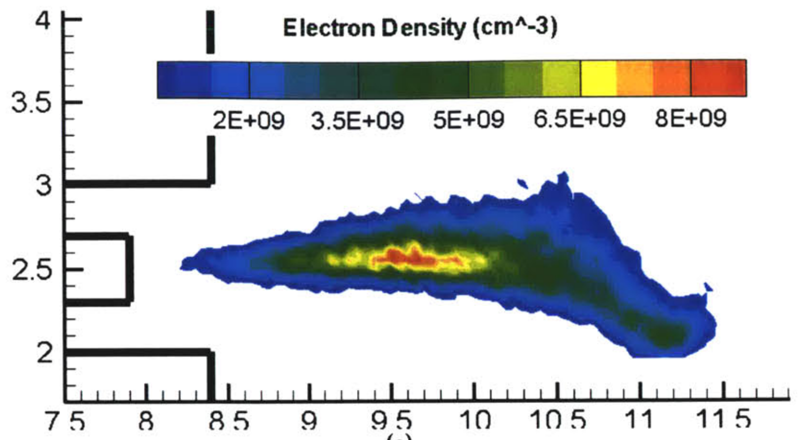
Figure 5-14: For the re-designed NVHT, this plot shows the derivative of the current with respect to voltage as the voltage on the simulated RPA was varied. The result from the original design is also overlaid with a dotted line.

5.5.4 Particle Densities

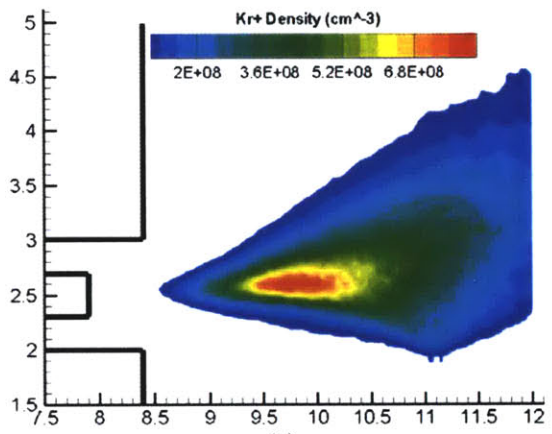
The densities of the electrons, singly-charged ions, and doubly-charged ions simulated are given in Figure 5-15. These results are basically in line with the observations already made regarding the relocation of the ionization zone. The peak density regions have been moved outward by between .5 and 1 cm due to the anode's repositioning. The electron density here seems slightly less confined than in the original case, probably due to the fact that the "sheaths" created in the channel are now having less effect on the plasma. More notable is the apparent increase in the doubly-charged ion density which previously peaked at about $1*10^7$ cm^{-3} but here reaches as much as $3*10^7$ cm^{-3}. The increase in total current could be the main cause of the right-shifted RPA curve shown in Figure 5-14. Both types of ions may be being created at nearly the same potential as in the original design case, but there seem to be more of both types being produced, leading to an increase in thrust. The ion densities in the re-designed case are also more compact and exhibit a noticeably reduced tendency to diverge radially. By moving the anode forward, we have caused fewer ions to be produced in regions of high radial electric field, improving the beam divergence. The simulation results seem to indicate that the re-designed NVHT should produce more thrust and a more focused beam than its predecessor when the experiments are conducted.

5.5.5 Electron Temperature

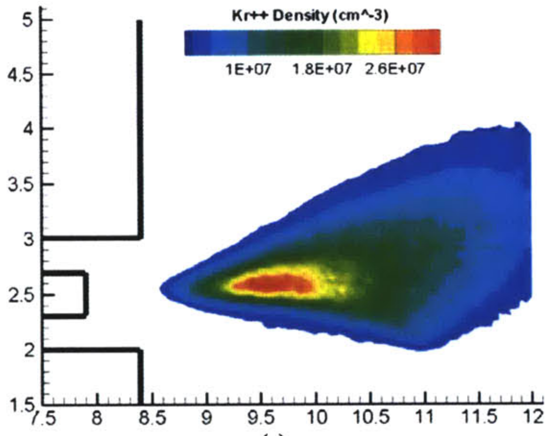
As seen in the previous simulation, the electron temperatures were computed to reach as much as 500eV near the anode. This is shown below in Figure 5-16. Figure 5-16(a) demonstrates that the region of high temperature has shifted with the shifting of the anode as might have been expected from the results above. A zoomed image of the anode region is provided in Figure 5-16(b). The temperature remains reasonably low until the exit plane of the thruster when it balloons to exceptionally high values.



(a)



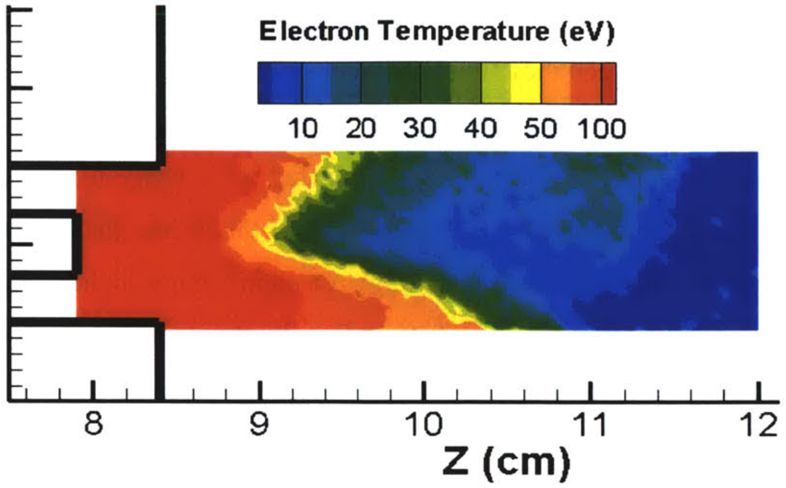
(b)



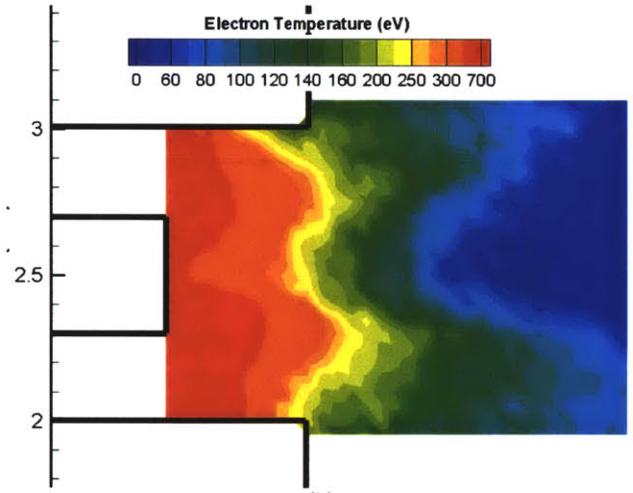
(c)

Figure 5-15: The (a) electron, (b) singly-charged ion, and (c) doubly-charged ion densities for the re-designed near-vacuum Hall thruster.

One possible explanation for the unexpectedly high temperatures in both cases is possibly more numerical than physical. Temperature in the simulation is merely a measure of the spread of the local velocities. If there are two separate populations of electrons in a given cell, for example highly energetic, uncollided primaries with kV energies and significantly less energetic secondaries that have collided or were produced during those collisions, our calculation of temperature does not discern between them. Thus, this two-peaked population may numerically exhibit a very high temperature when in reality, each population may have only a much smaller variation in velocity within itself. Given more time, it would be certainly be possible to investigate the electron distribution function in the channel more closely and examine this hypothesis.



(a)



(b)

Figure 5-16: The computed electron temperature for the re-designed near vacuum Hall thruster. (a) The temperature outside the channel is generally less than 150eV while (b) near the anode it can reach as much as 500eV.

5.5.6 Results at Lower Anode Potential

The above analyses of the simulation data, seem to indicate that the protruding-anode design of the NVHT should in theory produce more current due to the more effective use of the electric potential and the greater volume of space in which that potential remains at a high enough level to support ionization. Therefore, we theorized that if the anode potential was reduced, the original design of the NVHT may reach a point at which it no longer supported ionization while the protruding-anode design would still produce discernible emission. This is certainly something which should be explored by the experimentalists when they finish their construction.

In the meantime, we simulated the two NVHT designs now under a 2000 Volt anode potential. Figure 5-17 depicts the simulated RPA analysis results for both the original design and the new one. It is evident from this figure that the simulations are predicting the newly designed thruster to output a significantly higher amount of current than the old design at this lower voltage. Finally, Figure 5-18 depicts plots of the expected anode current and thrust over time for both of the two designs. Again, the new design clearly appears to produce more ions, yielding more thrust and higher anode current. The re-design effort may not produce the effect we had originally desired, but the simulations do indicate that it might still have some beneficial consequences.

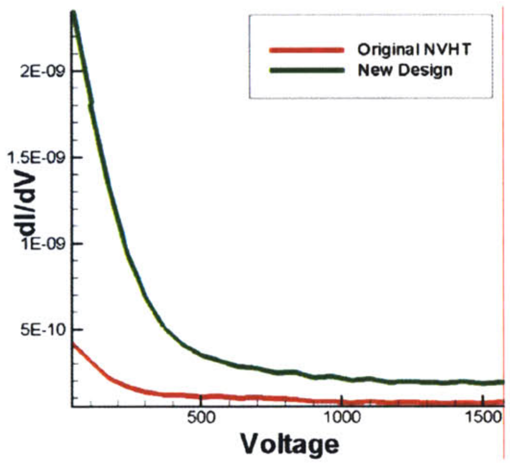
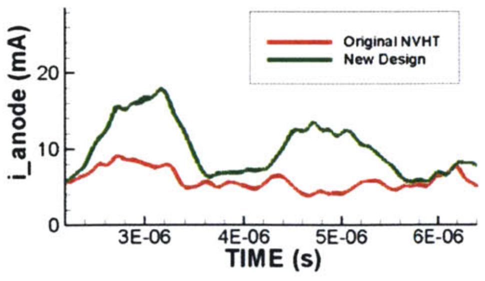
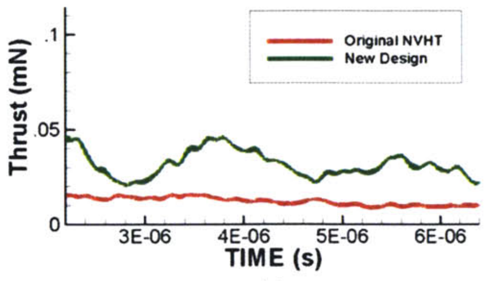


Figure 5-17: The simulated RPA analysis of the two NVHT designs at 2kV anode potential.



(a)



(b)

Figure 5-18: The predicted (a) anode current and (b) thrust for the two thruster designs at 2kV anode potential.

Chapter 6

Novel Anomalous Diffusion Model for Hall Thruster Simulations

One of the primary purposes of Hall thruster simulations is to aid the designers of new engines and to allow them to visualize the effects that their decisions will have on the performance and operation of a thruster before it is physically constructed or tested. Unfortunately, one of the biggest challenges facing these simulations has been that the amount of anomalous diffusion to include could not be known with any real accuracy or certainty without the inclusion of experimental data. Since this cross-field mobility plays a substantial role in predicting the location and value of practically every important plasma property in or around the channel, our simulations could never really be employed for the purposes of predictive modeling before with any true confidence of fidelity. This chapter of the thesis will explore the application of a particular model drawn from the confinement fusion literature to the problem of Hall thruster simulation. It will be shown that, without any reliance on prior empirical data, this "quench" model has been able to predict the exit plane transport barriers recently under intense scrutiny [33][2] and greatly improve a simulation's agreement with experiment, allowing us the ability to perform truly predictive modeling activities. The quench model is applied to simulations of the P5 thruster with promising success.

6.1 Explanation of Anomalous Diffusion

Almost since Hall-effect thrusters were invented, it was noticed that the cross-magnetic-field axial electron current to the anode was greater than could be accounted for through mere classical collisional diffusion processes. This phenomenon was also noticed in numerous other plasma applications, including the key research area of fusion and plasma confinement where the “anomalous”, or greater than classical, diffusion still plagues current researchers [81]. The exact causes of anomalous diffusion have been debated for many years, but recently the camp which claimed turbulent formations were responsible for the diffusion processes seems to have gained supremacy. Turbulent eddies in the RΘ plane like those described by Burrell [20] may help propel electrons across magnetic field lines and inward toward the anode under the prevailing electric field. Recent study also suggests that these turbulent structures can be formed by certain unstable oscillation modes prevalent in the operation regime of Hall thrusters. It appears that the theorists, especially those in the area of plasma confinement, are now well on their way to understanding how and why anomalous diffusion occurs. Unfortunately, for us, the cause seems to be a highly non-linear turbulent effect that involves azimuthal-variations which cannot be simulated directly using our axisymmetric simulation. Future computational studies interested in this topic should most definitely consider movement to a full three-dimensional model that might explicitly capture the effects of these out-of-plane phenomena. For the moment, however, we need to establish some method of approximation that can accurately predict the amount of anomalous transport in the thrusters under consideration, preferably a priori and without resorting to unreliable and incomplete experimental data.

6.2 Bohm’s Model of Diffusion

The MIT Hall thruster simulations up to now have relied upon the widely-accepted semi-empirical Bohm scaling to calculate the anomalous diffusion rate [73][72][29]. This method is based upon the experimental observation that the perpendicular diffusion coefficient, $D_\perp$, generally varies inversely with the magnetic field as:

$$D_{\perp} = D_B = \frac{1}{16}\frac{k_B T_e}{eB} \tag{6.1}$$

The backbone of this result can be derived in a very loose fashion by assuming the diffusion process results from a random walk. Each step in the random walk has a certain length, α, and a certain time, τ. The diffusion coefficient in such a situation is given by:

$$D = \frac{\alpha^2}{\tau} \tag{6.2}$$

In a collisional plasma, the random walk parameter, α, the distance between steps, is just the mean free path of an electron, and τ can alternatively be thought of as the inverse of the collision frequency, ν. When the plasma is magnetized, the collision frequency is usually small compared to the gyrofrequency, meaning that α will be on the order of the Larmor radius, r_L. This gives us:

$$D = {r_L}^2 \nu \tag{6.3}$$

If we assume that the electrons are moving at thermal velocity in between collisions, we can also write that:

$$D = v_T^2 \tau = \frac{k_B T_e}{m_e}\tau \tag{6.4}$$

In order for the diffusion to be at a maximum, the collision frequency would need to be equal to the gyrofrequency, allowing a particle to move a full Larmor radius at each collision step. Thus,

$$\tau = \frac{1}{\omega_c} = \frac{m_e}{eB} \tag{6.5}$$

And replacing this expression in Equation *6.4* above, we find that the diffusion coefficient is:

$$D_B \approx D = \frac{k_B T_e}{eB} \tag{6.6}$$

This then gives us the proper scaling, if not the exact empirical coefficient of the Bohm diffusion. In practice, many researchers have shown that the coefficient in the Bohm scaling is not necessarily constant from geometry to geometry, or indeed within a particular geometry itself. For this reason, we add an extra parameter to Equation *6.6* which allows us to choose the Bohm coefficient:

$$D_B = \frac{1}{\gamma}\frac{k_B T_e}{eB} \tag{6.7}$$

6.3 Previous Simulation Techniques

Bohm's model was incorporated into the previous generations of MIT Hall thruster simulations by assuming an increased neutral-electron collision frequency which only affected the perpendicular (to the magnetic field) component of an electron's velocity. To calculate this anomalous collision frequency, ν_{ano}, we start from the classical collisional diffusion coefficient:

$$D_{class} = \frac{kT_e}{m_e\omega_C}\frac{\beta_{class}}{1+\beta_{class}^2} \tag{6.8}$$

where here β_{class} refers to the Hall parameter and is given by:

$$\beta_{class} = \frac{\omega_c}{\nu_{class}} \tag{6.9}$$

The assumption is then made that the total diffusion coefficient, incorporating both anomalous and classical transport, can be written in the same form as Equation *6.8*, giving us:

$$D_{tot} = \frac{kT_e}{m_e\omega_c}\frac{\beta_{tot}}{1+\beta_{tot}^2} \tag{6.10}$$

with the analogous Hall parameter defined as:

$$\beta_{tot} = \frac{\omega_c}{\nu_{class} + \nu_{ano}} = \frac{\omega_c}{\nu_{tot}} \qquad (6.11)$$

Given this, we could set the total diffusion coefficient (Equation *6.10*) equal to Bohm's coefficient (Equation *6.7*) in order to solve for our newly-defined total Hall parameter in terms of known quantities. It turns out that the new Hall parameter is actually only a function of the scalar coefficient, γ, which was added as a tunable parameter to Equation *6.7*. The variation takes the form:

$$\gamma = \frac{1}{\beta_{tot}} + \beta_{tot} \qquad (6.12)$$

Under the low collisionality assumption wherein $\nu_{tot} << \omega_c$, this relation becomes simply:

$$\beta_{tot} \approx \gamma \qquad (6.13)$$

Using this result, we can know the total collision frequency and can calculate the anomalous collision frequency via:

$$\nu_{ano} = \nu_{tot} - \nu_{class} = \frac{\omega_c}{\gamma} - \sum_{collis} Q_{collis} n_n v_e \qquad (6.14)$$

This method has been inherited from the previous generations of MIT Hall thruster simulations and most definitely has some drawbacks and limitations. The limitations were so unmanageable that Blateau [18] even attempted to create his own method of Bohm diffusion for the simulations using a Brownian motion, random walk technique. However, Sullivan [72] found that method to be responsible for significant numerical heating of the electrons and to be ineffective at producing mobility. Consequently, she removed the method from the code and returned to Szabo's previously implemented technique as described above.

If one examines Equation *6.10* above, it should be clear that no matter what value β_{tot} takes, the coefficient:

$$\frac{\beta_{tot}}{1+\beta_{tot}^2}$$

which is also equal to 1/γ cannot be more than ½. This would seem to limit our choice of γ to being greater than 2 in order that the total diffusion coefficient retain the same form as the classical one. However, there is no physical reason why the total diffusion should have to take this form at any rate; that form was merely an assumption made in order to make our lives easier. Therefore, this does not alone constrain γ, so long as our implementation is altered slightly from the older methods to now use γ directly rather than β_{tot} as had previously been done.

The constraint which does limit the value of γ we can select stems from the fact that the collision modules in both the old simulation and our new one were not algorithmically implemented to handle the case where a particle collides twice in one time step. To account for this case would add further complexities to the implementation of the modules and an abundance of difficult bookkeeping as well, since the relevant densities of species might change drastically after the first collision events and would need to be updated very carefully. Foregoing this extra complexity, our simulations instead require the probability of such double collisions be small, less than 10% under the strictest criteria [73]. Sullivan quotes that due to the magnetic field strength in the thruster and some other factors, the single-collision restriction limits the Bohm coefficient to be greater than approximately 20.9, assuming that we take the maximum time step otherwise acceptable [72]. When the artificial mass ratio is also factored in, which requires multiplication of these frequencies by $\sqrt{1000}$ in the typical case, this condition becomes that $\gamma \geq 660$. Despite this admonition, Sullivan herself reported numerous results from runs with γ set to 64, 100, 200, or 400. It remains unclear how much effect the possible inaccuracy caused by the single-collision assumption and too small of a Bohm coefficient may be having upon the both the former simulation and the present simulation's results. Of course, reducing the time step further could alleviate this particular limitation.

The final, even more significant, limitation of the past simulations was their assumption that the Bohm coefficient, γ, was spatially and temporally constant throughout the entire simulation. A value for γ was chosen before each trial which would best allow the simulation to match the experimental results under consideration, and this value became a knob by which the simulation could be tweaked to fit the proper circumstances. The knob became further complicated and allowed the simulator even greater freedom to match experiments when I demonstrated in my own Master's thesis [33] that the Bohm coefficient could and should vary spatially along the channel and near-plume of the thruster. By hand-selecting a Bohm coefficient that was empirically shaped like the measurements collected by Stanford [2] or by Haas [37], the simulation's agreement with experiment, especially in its calculation of such spatially-varying quantities like electron temperature and electric potential, could be greatly improved. Figure 6-1 demonstrates some results from this procedure compared to the original spatially constant Bohm coefficient model. Note that the peak temperature in Figure 6-1(b) is only 16eV while in Figure 6-1(c) it reaches 28eV. Also, the ionization zone in the spatially constant case, indicated by the region of high temperature, was located farther back in the channel than in experiment and was less sharp and distinct. In contrast, the zone is properly located in Figure 6-1(c) and very sharp and well-defined. Finally, the electron temperature outside the channel exit is experimentally observed to fall off very rapidly down to only a few electronvolts. The spatially constant model does not predict this rapid drop in temperature after the channel exit while the hand-selected case does.

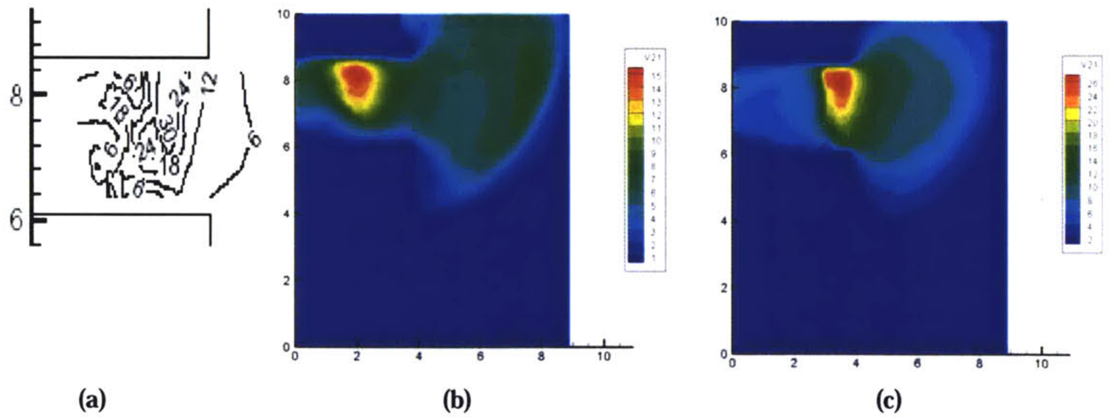


Figure 6-1: The dependence of the former MIT simulation's calculated electron temperature on the spatial profile of the Bohm coefficient is very strong. (a) An experimentally acquired plot (courtesy of Haas [37]) of the electron temperature with a peak temperature of roughly 30eV occurring near the exit of the channel. The former MIT simulation's calculation of the electron temperature is shown using (b) a spatially constant Bohm coefficient recommended by Sullivan and (c) an expertly-selected Bohm coefficient profile.

6.4 Goal of the Present Technique

Recognition of how great a role the spatial variation of the Bohm coefficient played in localizing the ionization region of the thruster and improving agreement with experiments was indeed a significant step forward. However, the method described above in which the proper profile needed to be expertly selected and generally relied upon previous experimental evidence of the Bohm coefficient's variation for a given thruster was not entirely satisfying. As Figure 6-2 and Figure 6-3 illustrate, the exact choice of this spatial profile was difficult and could easily lead to over-fitting of the simulation to a given thruster geometry.

Our present goal, therefore, was to discern some method by which we could analytically calculate the spatial and temporal variation of the Bohm coefficient without experimental evidence for the given thruster. The obvious application of this model

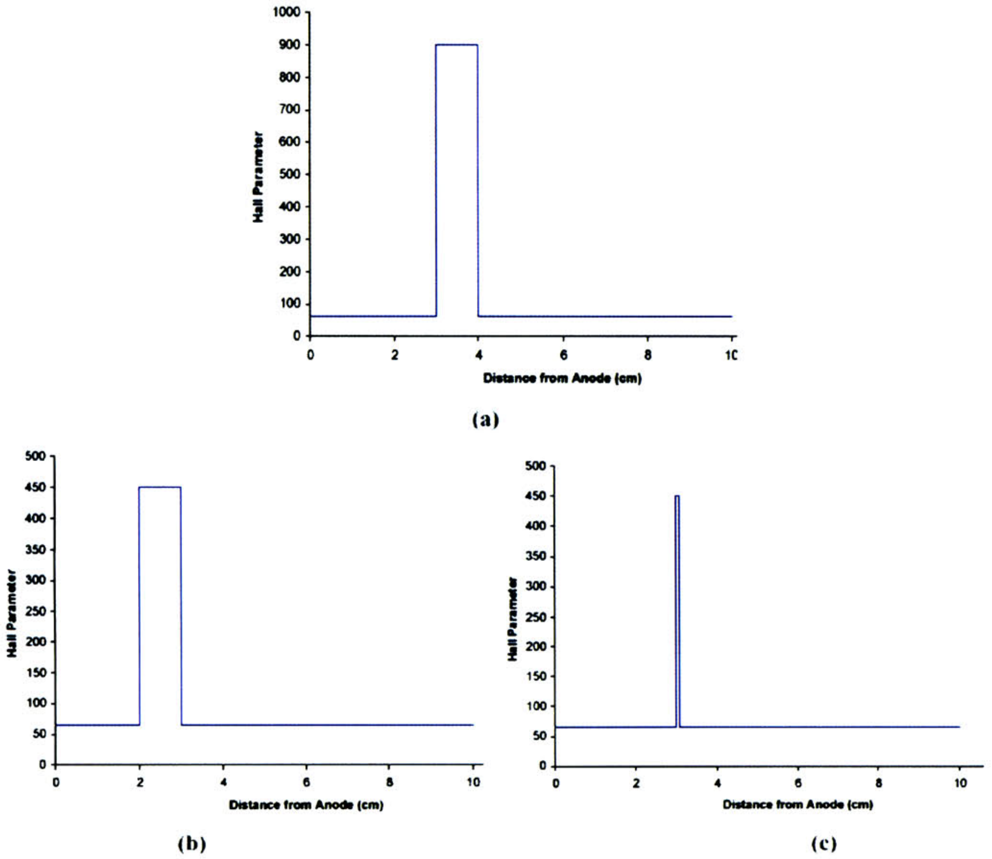


Figure 6-2: The three Hall parameter profiles used to compute the results shown in Figure 6-3.

would be to the Near Vacuum Hall thruster under consideration, for which no data currently exist indicating the Bohm coefficient's likely profile. Of course, in constructing such a model, we were only interested in those techniques that could be applied to our axisymmetric system and did not explicitly rely on the azimuthally-varying physics that are in all likelihood responsible for the anomalous transport. Future near-anode Hall thruster simulators are strongly encouraged to move into the realm of three-dimensional simulations which, while computationally difficult to the extreme, are the only definitive way to really explore and understand the fundamental causes of anomalous diffusion. For the present, our goal is less lofty and seeks only to find a way by which we can

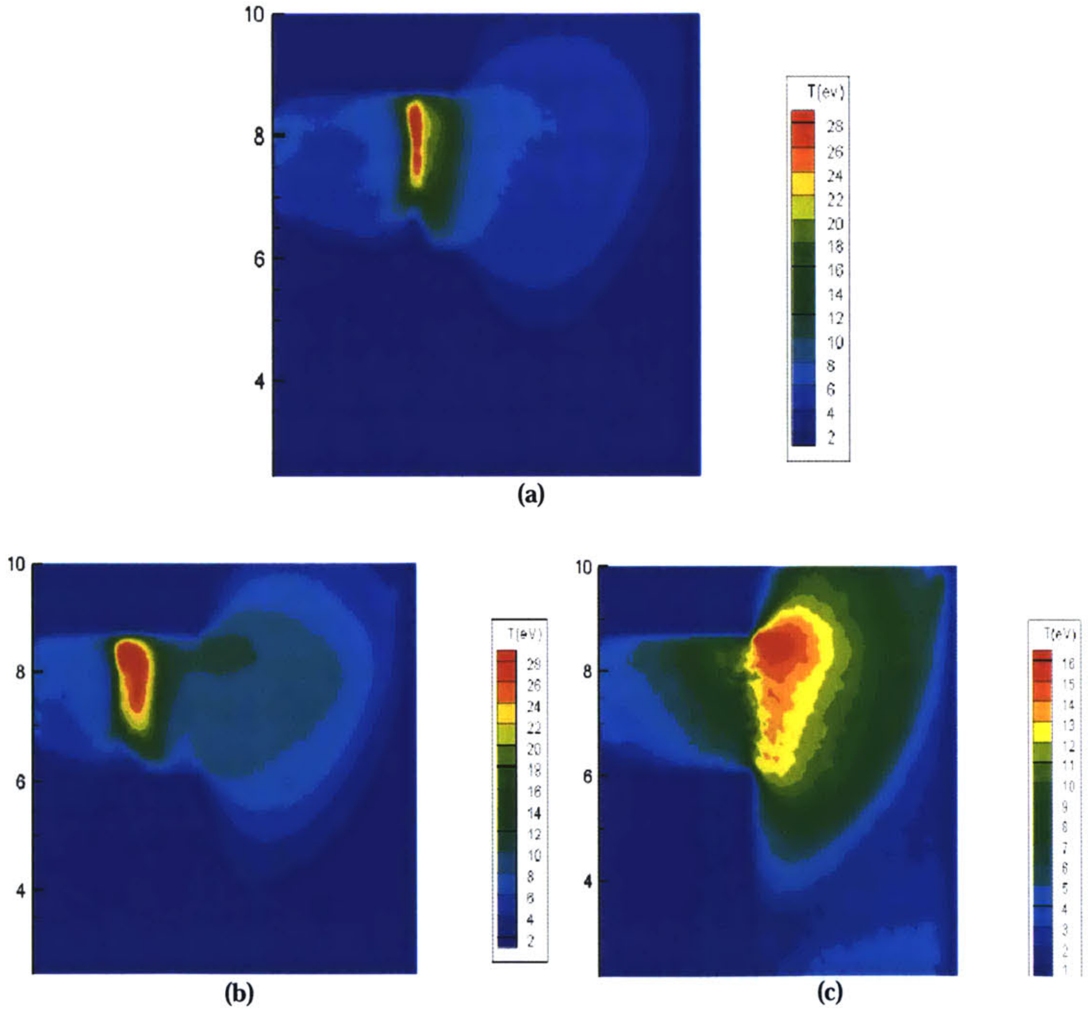


Figure 6-3: The results computed by the former MIT simulation using the spatial Hall parameter profiles depicted in Figure 6-2. The profiles shown in Figure 6-2(a), (b), and (c) match the results shown here in (a), (b), and (c), respectively.

approximate how much anomalous diffusion exists in the channel based entirely on the two-dimensional data we have available.

6.5 Correlation of ExB Shear and Transport in Hall Thrusters

One of the most promising results in recent years giving us hope that we could identify some two-dimensional characteristic of the thruster's operation to which we could correlate the anomalous transport evolved from the work of Cappelli's group [21] and, in the fusion community, a number of other researchers [20][27]. Basically, these works indicated that in numerous different geometries, including Hall thrusters, anomalous transport seemed to be controlled by the shear in the ExB drift velocity. Whenever the shear in the velocity was high, the anomalous transport would be significantly reduced and classical transport would dominate, and outside of the region of high shear, the transport attained values comparable to those predicted by Bohm. This phenomenon was very nicely predicted by Choueri [24], whose work showed how vortical turbulent structures in the plasma flow could be decorrelated by the velocity shear, reducing the effectiveness of the anomalous transport. These results, indicating that the anomalous transport should be strongly reduced near the exit plane of thrusters like the P5 whose axial gradient of azimuthal velocity, were supported not only by empirical evidence, but also by the predictions of the MIT simulation as described in the previous section.

6.6 Quench Model

Recognizing the importance of this effect and faced with a problem nearly perfectly analogous to our own in which the anomalous transport for an application needed to be predicted a priori, Waltz and his group [81][80][79] invented a computational model to predict the level of anomalous transport given the shear in the ExB velocity. They called this a quench model, and its original background and the analogy we drew to our own Hall thruster application are discussed in the sections that follow.

6.6.1 History and Background

The problem of anomalous transport reduction is critical to the successful operation of tokamak fusion reactors, and as such has been well-studied in that body of literature. As mentioned previously, much of the fusion community is convinced that the anomalous diffusion present in their system originates from the convection of turbulent eddies in the plasma [20]. In the 1980's, it was discovered that the addition of either a shear in the ExB velocity or a negative shear in the magnetic field could locally mitigate the effects of these eddies and caused the creation of ETB's (Edge Transport Barriers). Later on, ITB's (Internal Transport Barriers) were also discovered. These boundaries caused the plasma confinement to jump from a low to a high mode (L-H transition), greatly increasing the confinement efficiency of present-day fusion devices. It was these anomalous transport barriers' close similarity to the recently-identified exit plane "barrier" in Hall thrusters that caught our attention.

Of course, with such an amazing leap in experimental performance, the fusion theorists were eager to understand how these transport barriers worked and diligently set about inventing models to predict them. In particular, Waltz and his group at General Atomics had implemented hybrid-fluid and kinetic simulations, somewhat similar to those developed by MIT, to study instabilities in their plasmas [81][80][79]. While doing so, they found that they could accurately predict the level of cross-field transport due to ion thermal gradient (ITG) turbulence. They accomplished this by keying the assumed level of transport to the ExB shear rate through the use of a linearly progressive quench rule. The beauty of this rule was its lack of reliance on experimental data and its ability to be purely analytically calculated. Such a principle seemed to offer exactly the theoretical model for which we had been searching.

6.6.2 The Quench Model in a Hall Thruster Geometry

The Quench model as proposed by Waltz and his group reads as:

$$
\begin{aligned}
&if\left(\gamma_{E\times B} \le \gamma_{\max}\right) \longrightarrow \nu_{tot} = \nu_{class} + \left(1 - \frac{\gamma_{E\times B}}{\gamma_{\max}}\right)\nu_{ano} \\
&if\left(\gamma_{E\times B} > \gamma_{\max}\right) \longrightarrow \nu_{tot} = \nu_{class}
\end{aligned}
\qquad (6.16)
$$

where γ_{ExB} is the rate of shear in the ExB velocity and γ_{max} is the maximum linear growth rate of the turbulence responsible for the anomalous transport. This latter quantity will be discussed in Section 6.6.4. The equation above, however, basically states that where there is no change in the ExB velocity, the anomalous diffusion mechanism is in full effect. Eddies roam freely, and the anomalous transport is at a maximum. However, as the rate of shear increases, the eddies become sheared and decorrelated, "quenching" the anomalous transport in a hypothetically linear fashion. Once the shear rate has reached some high level, the eddies are no longer effective at producing transport and the entire diffusion mechanism must rely on purely classical, collisional means.

For our Hall thruster geometry, the quantities discussed by Waltz in his tokamak toroidal geometry have fairly simple interpretations. First of all, the important rate of the ExB shear velocity can evidently be given by the axial derivative of the expected azimuthal drift velocity, or:

$$\gamma_{E\times B} = \frac{d}{dz}\frac{E_z}{B_r} \qquad (6.17)$$

This rate is numerically challenging to produce as discussed in Section 6.6.5, but is otherwise requires no extra analytical work to produce. The other two quantities that we can quickly identify are the transport frequencies, ν_{ano} and ν_{class}. The latter of these two, ν_{class}, is simply given by the collision statistics as defined in previous sections. The anomalous frequency, ν_{ano}, is somewhat more of an enigma. It is unclear whether we know for sure what the maximum amount of anomalous transport should be for a Hall thruster geometry not under the influence of any ExB shear. However, the most logical value to use for this frequency would be the frequency predicted by Bohm-scaling and given previously in Equation *6.14*. This choice is certainly debatable, however, and more research could easily be done, both experimentally and computationally, into finding a better, more accurate estimate of this quantity. For our present purposes, however, we employed this format for ν_{ano} testing with a γ of 16 as originally predicted by Bohm and with a γ of 64 as was indicated by Sullivan, Blateau, and others [18] [72][73][37] as being an empirically more reasonable choice.

6.6.3 Selecting the Oscillatory Mode

The one quantity which remains to be defined in order for our transfer of Waltz's quench model to the realm of Hall thrusters to be complete is the maximum linear growth rate of the turbulence responsible for transport, denoted γ_{max}. In the tokamak fusion literature, again more work has been conducted investigating this problem, and they have determined that the source of their most prevalent anomalous transport is the Ion Thermal Gradient (ITG) turbulent mode. We similarly needed to understand which of a Hall thruster's turbulent modes might be most responsible for the transport in our case.

Here may be the most difficult step of the analogy: choosing the proper turbulence which has a calculable maximum growth rate and is most responsible for the anomalous transport. However, as later sections will show, having made that choice and tested it, we can feel quite confident afterward that it was indeed the correct choice. In fact, the agreement of this model with experiment provides even greater evidence that the turbulent mode we selected is in fact the major cause of anomalous diffusion in our thrusters.

How did we select this turbulent mode in the first place, however? Fortunately, we were not the first to consider this particular question, though apparently were the first to apply its answer in this particular manner. In 1972, the Russian scientists Esipchuk and Morozov, two of the fathers of Hall thrusters or "closed electron drift accelerators" as they were then called, conducted experiments to examine if the azimuthally asymmetric turbulent modes they had detected might be responsible for the anomalous electron conductivity that had been noticed [26]. In particular, they focused on the so-called transit-time oscillations which are named because of their strong frequency dependence on the ion transit time. Through their investigations they concluded that these transit-time modes did exhibit the proper correlated azimuthal asymmetry to produce greater-than-classical axial electron current and that the transient electric fields produced by these asymmetries were of the proper magnitude to be responsible for the anomalous electron transport to the anode. These results were found through a not very in-depth study of the literature, but had largely been forgotten or overlookoed by much of the present community. If nothing else, this exercise should stress to future students or even teachers in any field the importance of a strong, complete background in the fundamental literatures of the topic. Many questions already have known answers if we are simply determined enough or fortunate enough to find them.

This is not the only piece of evidence that led us to select the transit-time mode of Hall thrusters as the most likely candidate for being the primary contributor to anomalous diffusion. In [24], Choueiri completed a detailed analysis of the growth rates of numerous turbulent modes and published several figures which demonstrated a very strong correspondence between the electric potential in a thruster channel and the stability frequency map of the transit time oscillations. In particular, his results indicate that the sharp gradient region of the plasma potential which indicates the presence of the ionization region has almost precisely the same spatial location and structure of the sharp gradient region also associated with the transit-time oscillations' stability map. Taken together with our previous computational and experimental evidence, such as that demonstrated by Figure 6-1 above, this result seems to hint that the location of the transit-time instabilities is strongly keyed to the location of the ionization zone which, in turn, is strongly correlated with the profile of the anomalous transport. Given one piece of this triad, in our case information about the location and strength of the transit-time oscillations, the other two pieces appear to be generally predictable. No other mode exhibited this strong correlation.

A third criterion we used to narrow the field of possible diffusion-causing oscillatory modes was that the mode must be prevalent and significant during nominal Hall thruster operation. Several modes have been discovered and studied when a thruster was operated at less than optimal conditions, but only a small handful of modes can be considered significant in the typical operating regime of a thruster. The transit-time mode is one of that select few according to [24], and none of the rest of that small cadre of modes exhibited the same stark localization and correlation with the ionization zone that the transit-time mode did.

6.6.4 Maximum Linear Growth Rate of Transit-Time Oscillations

Having thus decided that the leading candidate for anomalous diffusion causing turbulent mode was the oft-cited transit-time oscillations, we were fortunate to find that Choueiri [24] had already performed a very complete analysis of these waves. In fact, he had even computed the linearized growth rate which was therefore ready for our immediate use. The dependence as he reports it is:

$$\gamma \approx k_x u_i \frac{\sqrt{b}}{b+1} \tag{6.18}$$

Here, k_x is the axial wave-number of the mode, u_i is the local average ion velocity, and b is a parameter defined by:

$$b \equiv \frac{u_B}{u_{ExB}} \tag{6.19}$$

where u_B is a magnetic drift velocity whose exact representation is unimportant here and can be found in Choueiri's work. It is interesting to note, however, the dependence of this mode on the ExB velocity, u_{ExB}, which further strengthens our connection to the ExB shear.

The exact value of b is actually unimportant also for our calculations. All that is necessary is to realize that γ has a maximum when b is equal to 1, and that maximum is:

$$\gamma_{\max} = \frac{1}{2} k_x u_i \tag{6.20}$$

Interestingly enough, Choueiri also computed the resonant frequency of these oscillations using similar parameters, finding that:

$$\omega_R = k_x u_i \frac{b}{b+1} \tag{6.21}$$

The maximum frequency of the oscillations has been observed experimentally to be on the order of 500kHz [24]. Since we are looking for the maximum growth rate, we assume that b is equal to 1 and then choosing the greatest frequency will maximize our solution for $k_x u_i$:

$$\begin{aligned} &500{,}000 * 2\pi \sim \omega_R^{\max} = (k_x u_i)^{\max} \frac{b}{b+1} \\ &\Rightarrow (k_x u_i)^{\max} \sim 2\pi * 1e6 \end{aligned} \tag{6.22}$$

This would give us an approximate maximum growth rate of:

$$\gamma_{\max} = \frac{1}{2}(k_x u_i)^{\max} \sim \pi * 1e6 \tag{6.23}$$

Note that the above value is only an order of magnitude estimate, and a more detailed and accurate method of computing the maximum growth rate is further discussed in Section 6.6.6. In the figure presented by Cappelli's group in [21] that plots the shear rate versus the anomalous transport for their particular Hall thruster, they note that at the point where the shear rate exceeds approximately 1e6 s^{-1}, the anomalous transport "barrier" begins to form. The quench model, of course, predicts this admirably. When the shear rate reaches about the same order of magnitude as the maximum growth rate, or about 1e6, the quenching begins and grows very strong in the narrow region near the exit plane where the shear rate is above 1e6. It is indeed satisfying that these order of magnitude estimates appear to line up nicely, and this gives us further confidence in our choice of the transit-time modes as the main cause of the anomalous diffusion barrier for Hall thruster exit planes.

6.6.5 Averaging the Electric Field

It was mentioned above that calculating the rate of shear in the ExB drift velocity presented a difficult numerical challenge. It is a typical well-known problem in numerics that an attempt to differentiate discrete, noisy data will often leave one with, in technical terms, pure junk. The noise is amplified in the derivation to the point where it often dwarfs the signal. Such was our discovery when we first started trying to calculate the axial derivative of the drift velocity, as given above by Equation *6.17.* Figure 6-4 demonstrates how noisy the computed derivative appeared. This figure was created by examining the grid line at R = 7.5 cm, roughly in the center of the channel. The horizontal axis represents the Z-coordinate along that line. The vertical axis is the computed shear rate at the individual grid nodes. For simplicity, we used only a simple first order differencing scheme to calculate the derivative, which also did not help the accuracy or smoothness of the calculation.

Clearly, these noisy data could not be considered indicative of the actual shear rate in the simulation. When tests were run using these data, the shear rate would oftentimes be very large at a few singular points in the domain causing the diffusion rate to also be very noisy. Our solution to this problem was simply to keep a running time-average of the electric field for the last, say, *n* iterations. As *n* was increased, Figure 6-5(a) shows that the noise in the calculated axial derivative of the shear was greatly reduced, improving the overall accuracy and viability of the model.

It is important to note that the electric fields in Hall thrusters are highly transitory and time-dependent. As such, it is not advisable for us to time-average the electric fields over any significant amount of time in the simulation, or else we would risk trying to smooth away not just noise but time-dependencies as well. This explains why it was necessary to use a rolling average and not just a simple total time average and why we did not continue to increase *n*, the number of iterations in the average. Figure 6-5(b) below illustrates this point graphically.

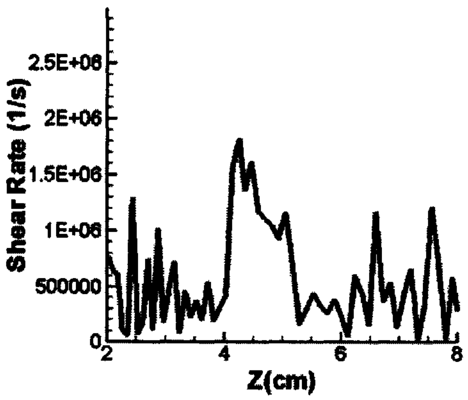


Figure 6-4: The noise in the derived shear rate is very high if we do nothing.

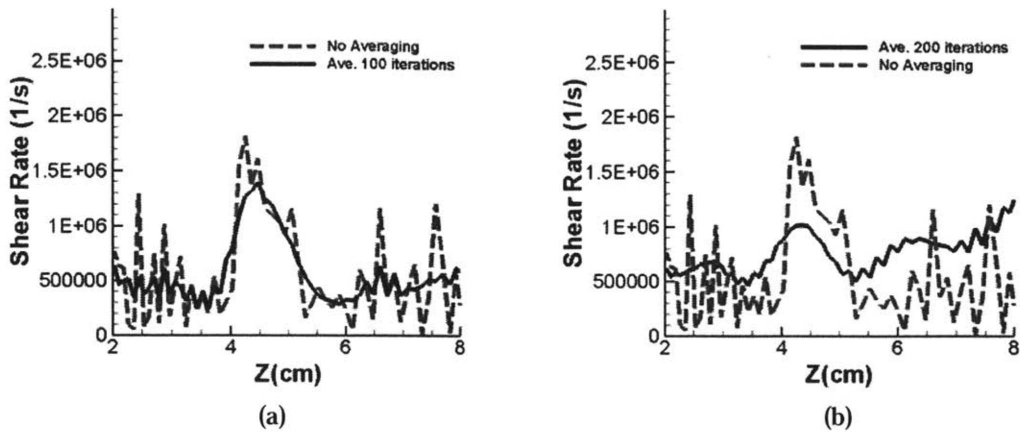


Figure 6-5: The noise in the shear rate data was reduced as the number of iterations in the rolling average was increased. (a) When 100 iterations are used, the result is smooth and maintains the proper structure. (b) If too many iterations are used, the structure of the shear rate is still relatively smooth, but tends to take in too much information from past time steps and the structure becomes distorted.

6.6.6 Putting It All Together – Implementation in the Code

The quench model was inserted into the former MIT simulation originally written by Szabo [73] and further added to by Blateau [18], Sullivan [72], and myself [33]. This model was used for this portion of the thesis due to the fact that it had already been extensively studied and benchmarked; we wanted to test the quench model in a "known" environment first. As discussed above in Section 6.3, that simulation approximated anomalous diffusion by increasing the collision frequency of electrons with the additional collisions acting only on the perpendicular (to the magnetic field) component of electron velocity. This method was easily able to be adjusted to utilize the quench model. Wherever the Bohm coefficient was previously used in the code, a quick calculation was instead performed. This calculation involved finding the local ExB shear rate, calculating γ_{max}, and adjusting the local Bohm coefficient according to the quench model ratio as described above.

Two methods of calculating γ_{max} were initially implemented. The first method simply used the calculation outlined above in Section 6.6.4 to compute a spatially constant maximum growth rate everywhere in the thruster. This method did achieve quite acceptable results. It was later decided, however, that we could easily be a little more accurate. Thus, the second and final method of computing the maximum growth rate involved a technique which yielded a different value at each spatial location, more closely resembling reality. For this method, we used Equation *6.20* and substituted the proper values. For the average local ion velocity, u_i, the instantaneous average ion velocity computed by the code was used which varied from point to point. For k_x, we chose to use simply $2\pi/L$, with L being the length of the discharge chamber, since it was experimentally determined that these waves are more or less axial in nature and extend the full-length of the thruster chamber. In earlier tests we did attempt to explore the effect of this choice, trying various values for k_x, but it was decided that the exact value did not seem to have a very appreciable effect on the resulting solutions. The change in the shear rate is so rapid, in fact, that the exact value of γ_{max}, for the P5 thruster anyway, did not seem to be very important. So long as it was greater than the shear rate's magnitude in the areas of low shear and less than the shear rate in the areas of high shear, the "quenching" for this thruster actually acted more like a bang-bang switch than a smooth transition. To within a scalar factor, our approximation to k_x was acceptable. With the quench model inserted into the PIC/MCC simulation, our next step was to test its effectiveness on a well-studied thruster.

6.7 P5 Results Demonstrating Quench Model Performance

The P5 Hall thruster developed by the University of Michigan seemed a perfect choice to begin our evaluation of the quench model as developed in the previous sections. This thruster was developed with the specific purpose of enabling easy access for research and study, and as such a copious body of experimental and previous simulation results exist to which we could compare. The following sections discuss the results we observed when the quench model was used along with their comparison to reality and to other simulation conditions.

6.7.1 Transient Diffusion Barriers

One of the first pieces of evidence that indicated to us that the quench model might be performing well was the observation of anomalous transport barriers being formed through the quench models predictions. As stated previously, it is generally accepted that thrusters like the P5 exhibit some form of anomalous transport barrier near their exit plane and it is the inclusion of this barrier which has in the past improved simulation agreement with experiment [33]. We were very pleased, therefore, to observe the trend shown in Figure 6-6. These figures depict the calculated Bohm coefficient profile over time as the simulation progresses, with the blue regions indicating areas of low anomalous transport and the red regions indicating high anomalous transport. The base ν_{ano} used in these tests had a γ coefficient of 64. The quench model, with no expert prompting and with no a priori experimental data, very clearly predicts that near the exit plane of the thruster the anomalous diffusion is reduced significantly. An unexpected demonstrated by this figure, however, was the periodic motion of the transport barriers. Indeed, as the barriers moved down the channel, they were correlated with pulses of high electron density that traveled along with the barriers. Both of these phenomena were found to be periodic, and even more interestingly, they showed a frequency of approximately 400-500 kHz, roughly that of the transit time oscillations that we believe to be the cause of the anomalous diffusion turbulence.

Apparently these transient barriers tend to slow as they reach the exit plane and spend more time there than elsewhere in the channel. This observation is supported by Figure 6-7 below, showing the time-averaged anomalous transport level in the P5 under the quench model assumption. Comparing this profile to the empirically derived ones shown in Figure 6-2 above, one immediately sees the similarities. The sharp barrier in this case is just outside the exit plane of the thruster which is slightly farther forward than we had expected. One other unusual feature of this plot is the low anomalous transport that is being calculated near the anode of the thruster. This may be contrary to empirical measurements, though the accuracy of those measurements this near to the anode is suspect at best [37]. Assuming the quench model is correct, the diffusion in this region should be reduced for two main complementary reasons. First of all, the ion velocity in this region is relatively small. This makes the calculated maximum growth rate very small for the transit time waves. In addition to this fact, the magnetic field in this region is generally very small and in some regions approaches zero. The shear rate in the ExB (or really E/B) velocity given by Equation *6.17* above becomes very large then in this region. These two trends tend to quench the diffusion completely.

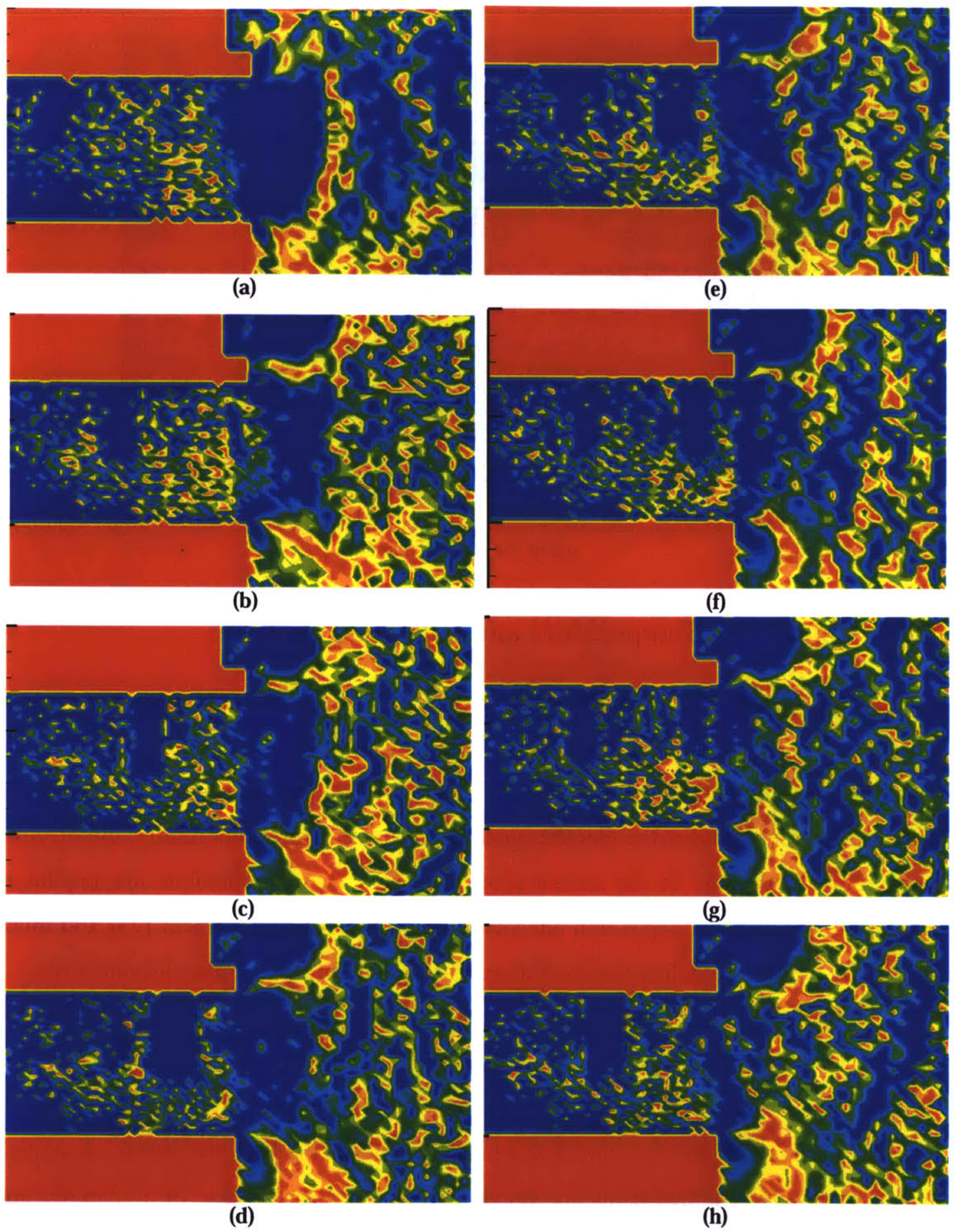


Figure 6-6: The Bohm coefficient in the channel of the P5 calculated using the quench model. Areas of low transport are indicated in blue while high transport areas are indicated in red. Time elapses moving from (a) through (h). The initial exit barrier is seen in (a). A transient barrier forms and then moves down the channel in (b)-(d). This barrier reaches the exit plane and another is formed in the remaining figures (e)-(h).

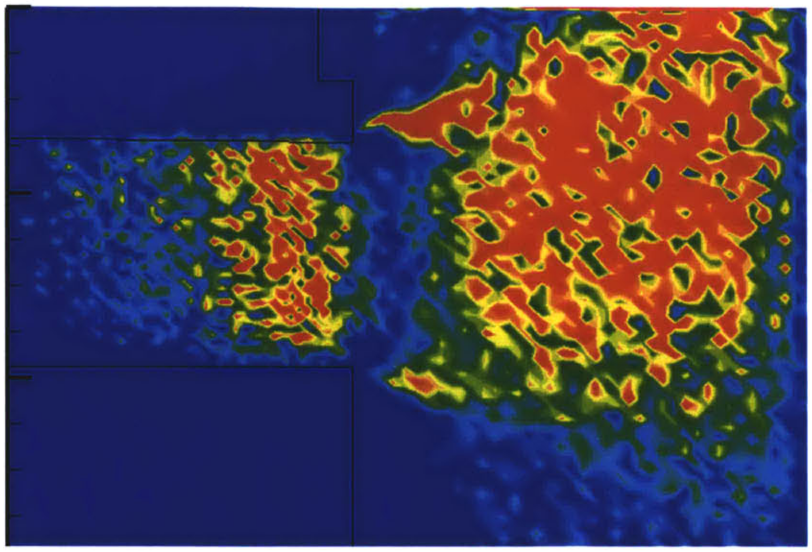
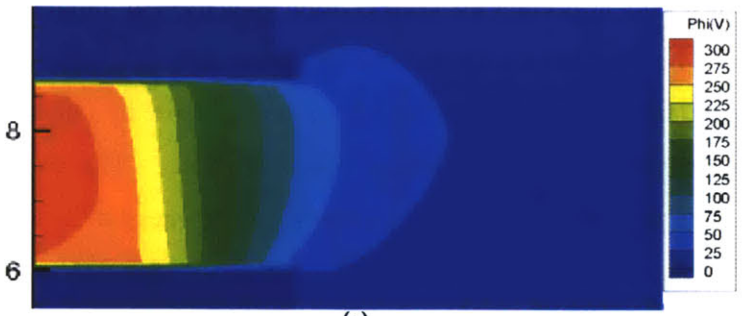


Figure 6-7: The time-averaged anomalous transport analytically calculated with no empirical input using the quench model.

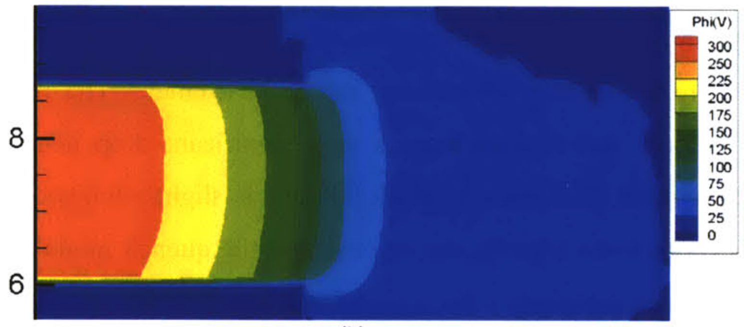
6.7.2 Comparing Temperature and Potential Profiles

One of the primary purposes of a Hall thruster simulation is to allow us to visualize how various quantities that would otherwise be difficult to measure are changing spatially and temporally within the thruster. Specifically, we would like to be able to accurately predict the electric potential and, even more importantly, the electron temperature over time in the acceleration channel. These predictions are crucial to sputtering and material deposition analyses currently under development [23] and much needed by industry to defray the costs of expensive and time-consuming lifetime tests.

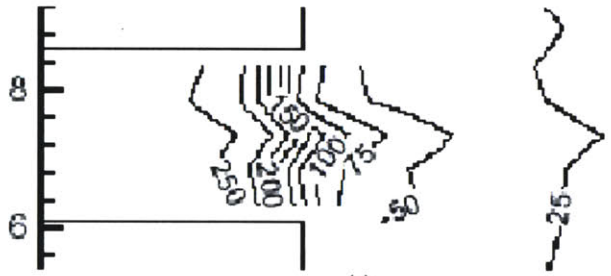
Figure 6-8 is a comparison of four results for the electric potential of a P5 thruster running in the 3kW mode with a mass flow of 10.7 mg/s and an anode voltage of 300 Volts. Figure 6-8(c) depicts the experimental results obtained by Haas [37] during his extensive probing of the thruster. Figure 6-8(a) presents the MIT PIC/MCC simulation results that were obtained using the previously-accepted assumption of a spatially and temporally constant anomalous transport level throughout the thruster. Figure 6-8(d) shows the potential calculated using the same simulation, but with an expertly-peaked, still constant-in-time Bohm coefficient profile like those shown above in Figure 6-2. Finally, Figure 6-8(b) is the result as computed using the quench model.



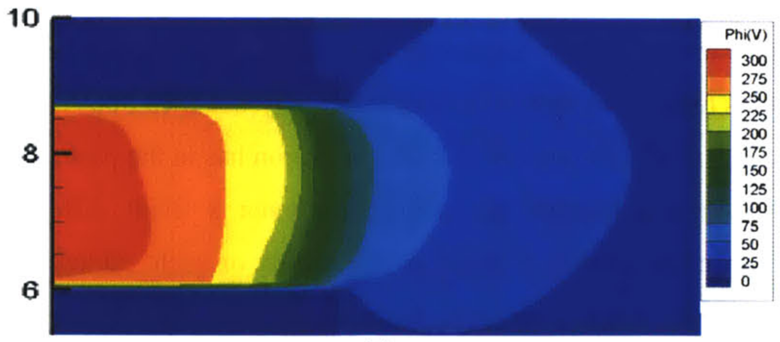
(a)



(b)



(c)



(d)

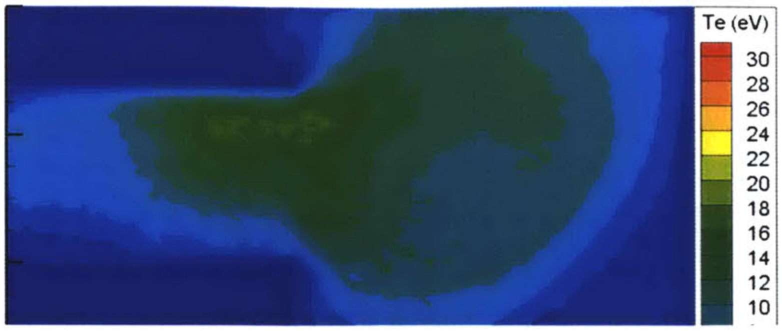
Figure 6-8: Electric potentials from the P5 thruster. Simulations with a constant Bohm coefficient assumption, (b) the quench model, and (d) an empirically-derived profile are here compared with (c) the experimental profile reported by Haas.

The wide range of resulting profiles in this figure again gives one a distinct impression of just how important the correct implementation of anomalous diffusion is to the simulation. The constant anomalous transport assumption, Figure 6-8(a), exhibits the very smooth gradient in potential that has been noticed before in a number of simulations [33][48] with simple anomalous diffusion approximations. The experimental results, Figure 6-8(c), and the simulations with an empirically "guessed" transport profile both have very sharp potential drops right near the exit of the channel. This potential drop corresponds to the location of the ionization zone and demonstrates that the anomalous diffusion plays an important role in localizing that zone. The simulation results using the quench model are somewhere in between the two extremes. The potential gradient is certainly not smooth and it does have a very significant drop near the thruster exit. However, the potential does seem to begin falling just slightly too soon in the channel as compared to experiment. Clearly, the results from the quench model simulation are not perfect, but they most definitely offer a significant improvement over the constant profile assumption and are not really much worse than the empirically-derived profile results. The fact that such good experimental agreement was obtained without the use of any empirical data is indeed a significant advance.

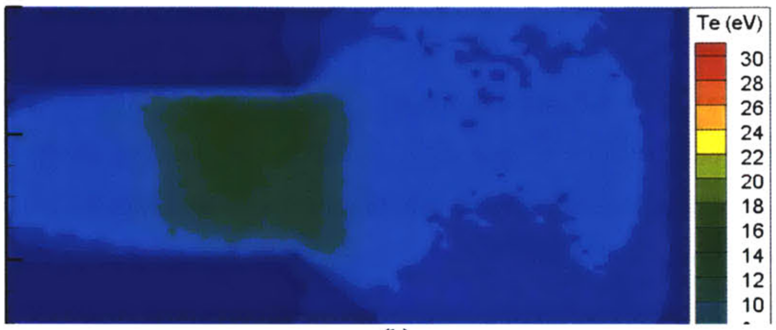
The next figure, Figure 6-9, compares the electron temperature results of the same cases, with the constant profile case, the quench model case, the experimental results, and finally the empirically-derived profile case being presented in Figure 6-9(a)-(d), respectively. The constant in time and space anomalous transport profile case in Figure 6-9(a) disagrees in two distinct ways with experiment. First of all, the peak temperature in the channel is only about half of the experimentally-obtained 30eV. This very striking disagreement is one of the main reasons the simulation has in the past provided very poor quality sputtering and lifetime predictions, for even a small difference in electron temperature can cause drastic changes in the results of such studies [23]. The hand-picked case in Figure 6-9(d) does seem to overcome this difficulty admirably and raised the electron temperature to the proper level. However, it only partially solved the second of the major problems that plague the constant-profile case. Experimentally, the electron temperature just outside the exit plane of the thruster is seen to fall off rapidly to very low levels of 5 or 6 electronvolts. The constant-transport case shows no clear sign of a temperature cliff, and the empirically-chosen case exhibits more of a slow, gradual descent than an abrupt drop. Here, however, the quench model seems to shine. Unfortunately, the peak temperature in the channel did not reach the hoped for 30eV, though it is a few electronvolts warmer than the constant-profile case. However, the localization of this temperature is much better than either of the other two cases. The region of elevated temperature begins at the proper point within the channel and then immediately and abruptly declines immediately after the exit plane, very much like experiment.

One important caveat to the preceding section involves the extremely time-dependent nature of Hall thruster discharges. The results given above for the simulations were time-averaged over one or two main discharge oscillation periods. The simulated temperature, for example, during these oscillations may have ranged from a peak of 70eV to a minimum of only a few electronvolts even in the hottest part of the channel. It is very difficult to know exactly how Haas' experimental results should compare to these experiments. His probes must have measured some form of time-averaged temperatures and potentials as well, but due to the actual physics of the probe, were those averages different than ours in some way? Future investigators may wish to ponder this question and should take great care before comparing simulation results computed in one manner with experimental results compiled in another. With that said, matching these results is currently the best method we know of benchmarking our simulations and must be stomached for the time being.

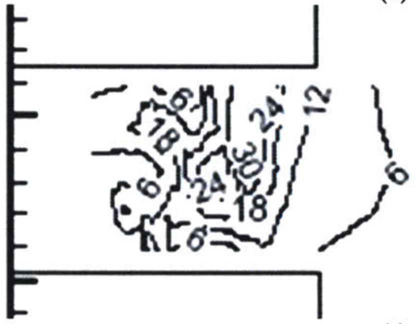
Such warnings regarding the difficulty of direct comparison between experiment and simulation aside, it appears that barriers created by the quenching model as implemented, while better than the constant coefficient model, are not drastic enough to perfectly reproduce the sharp potential drops or the high temperatures of empirically defined barriers. Several difficulties in implementation still remain to be overcome, however, giving hope to future improvement in the model's results. Simulating an accurate, noiseless shear rate profile is chief among these difficulties, and could go a long way toward improving the resolution and stability of the quenching model's transport barriers.



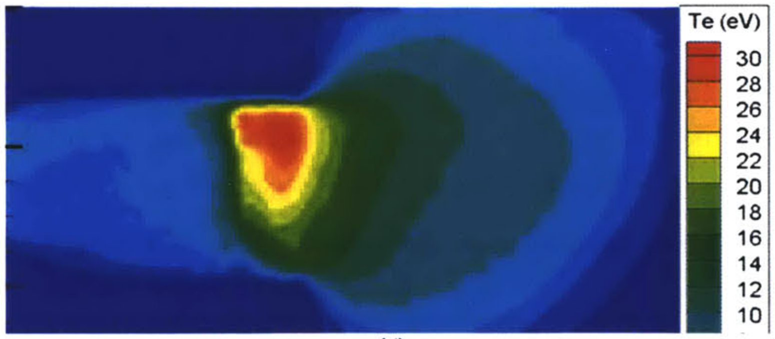
(a)



(b)



(c)



(d)

Figure 6-9: Electron temperature results for the P5 thruster. . Simulations with a constant Bohm coefficient assumption, (b) the quench model, and (d) an empirically-derived profile are here compared with (c) the experimental profile reported by Haas.

6.7.3 Comparing Current and Thrust Predictions

Another aspect of predictive modeling involves being able to describe the overall performance parameters of a thruster using simulation, preferably before a great deal of time and resources are wasted prototyping and testing an inefficient thruster design. For this reason, we also examined how well the quench-model based PIC/MCC simulation could predict the performance of the P5 thruster. First, we reprint here a figure from Haas' incredibly detailed and invaluable thesis. Our Figure 6-10 is a slightly adapted portion of Haas' Figure 4.18. The figure shows a plot of the experimental discharge current in the P5 thruster operating at the 3kW and 10.3 Amp nominal current setting, which value is marked by the horizontal red line. Of note in this figure is the amplitude and frequency of the current oscillations. In general, the maximum current does not exceed 16 Amps and does not fall below 6 Amps. The frequency of these oscillations is on the order of 11 kHz. Another interesting conundrum that again highlights the difficulty of comparing experiment with simulation is that the direct average of this current plot seems to fall below the stated nominal 10.3 Amps. It would appear then that even in experiments, sometimes the performance does not achieve the optimal, stated values. Thus, the exact performance numbers quoted might rather be imagined as a range. It is important then to keep in mind that if our simulations fall within that range and appear, through the application of common sense and intuition, to be creating accurate predictions, then over-fitting them to obtain perfect agreement with imperfect experimental values may, in fact, be counterproductive.

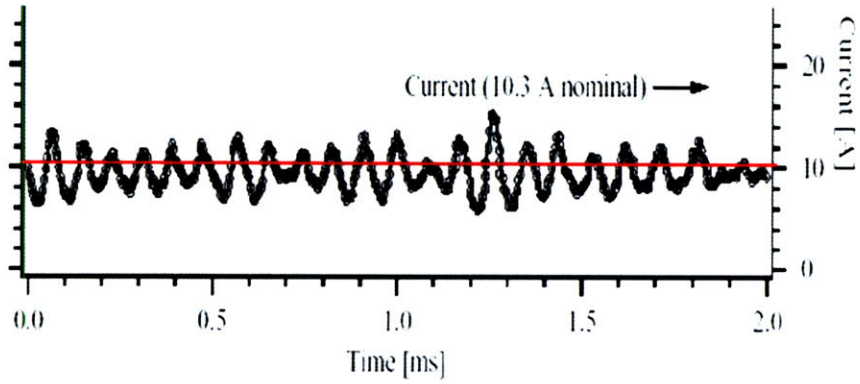


Figure 6-10: A plot of the experimental anode current as a function of time in the P5 thruster. Data reprinted from the thesis of Haas [37] for the sake of comparison.

With these caveats in mind, we turn our attention to Figure 6-11. This figure gives the simulated anode current found using the PIC/MCC code using two different assumptions for the anomalous transport rate. The green, sharply-peaked case was calculated under the assumption of a temporally and spatially constant Bohm coefficient. The case with lower amplitude, drawn in red, was found by inserting the quench model into that same simulation. Comparing first the amplitudes of these two cases with the amplitude of the oscillations observed experimentally, the difference is at once striking. The constant transport case has an amplitude between 2 and 50 Amps while the quench model brings the simulation much more in line with experiment and exhibits and amplitude range of only about 4 to 20 Amps. Such a drastic difference just by more accurately predicting the anomalous diffusion! Examining also the frequency of the two oscillations, we can approximate that the constant transport overestimates the frequency at about 14.3 kHz while the quench model case predicts a slightly more reasonable 10.7 kHz. It is evident that the quench model has significantly improved our simulation's agreement with the time-dependent view of the anode current, and all without the use of empirical data to predict the anomalous transport.

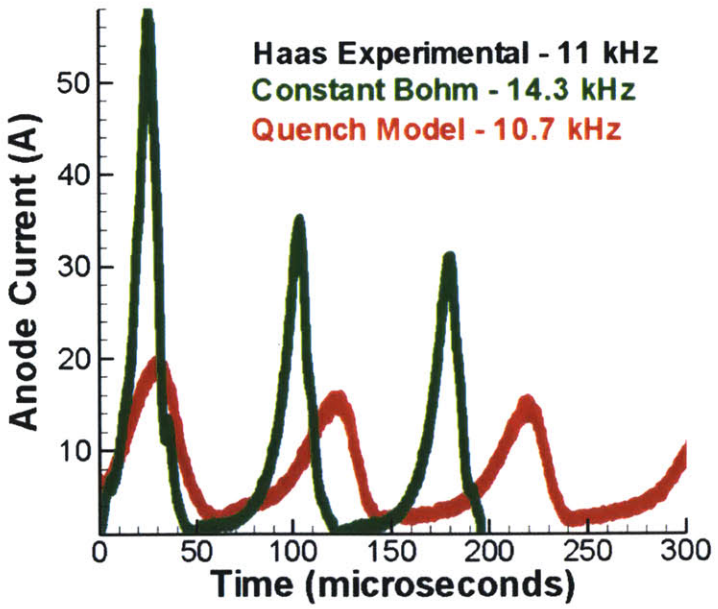


Figure 6-11: Two anode current versus time plots for the PIC/MCC simulation. The green, high-amplitude case was found using the constant transport assumption and the red, low-amplitude case was found using the quench model.

Strangely, when we compared our time-averaged results of performance quantities with experiment, as shown in Table 6.1, the trend was not as we would have expected. The thrust calculation did improve slightly when the quench model was used, though given our previous warnings regarding the exact accuracy of results, both experimental and computational, this difference is not really very significant. What was significant, however, was the change that the quench model made to the estimated average DC anode current. Comparing Figure 6-11 above to the experimental results in Figure 6-10, it would be expected that the quench model was drawing the simulation closer to experiment. However, the simulation was already, according to the numbers, under-predicting the time-averaged DC current. So by reducing the peaky, high-amplitude nature of the discharge oscillations and slowing their frequency to the more reasonable level, the time-averaged data indicate that we have reduced the effectiveness of the simulation.

There are a number of ways to rationalize this disagreement. Our first thought, after noticing that the experimental current plot did not have a 10 Amp direct average, was that possibly the experimental time-averaged current values for the P5 were in fact reported as AC-averaged, root-mean-square values. This could have occurred if, for example, the average current was reported using an AC setting on an oscilloscope. The RMS current results for the simulations are also given in Table 6.1. Under this assumption, the original constant transport assumption was actually strongly over-predicting the average current due to its exorbitantly high amplitude peaks. The quench model, though, well-predicts the average current in the RMS sense since it much more accurately traced the proper amplitudes and frequencies of the oscillations.

	Thrust (mN)	**DC Current (A)**	**RMS Current (A)**
Experiment	175.0	10.3 ?	10.3 ?
Quench Model	176.12	6.95	9.83
Constant Bohm	172.41	8.10	15.72

Table 6.1: Performance results for the P5 thruster under experiment and two simulated conditions.

In communications with Haas, however, it was clear that the 10 Amp current measurement was taken using the reading from the power supply used. A request to the power supply manufacturer for details on the power supply's measurement mode was inconclusive. There exist certain RC filters in the power supply that may or may not be effective in compensating for the frequency of oscillations present in the thruster, depending on their exact construction.

The expert experimentalists around me, however, all indicated that they believed the current measurement must indicate a DC value, and so we are left with the conclusion that the PIC/MCC simulation as a whole tends to under-predict the average DC current. This under-prediction was further exacerbated by the reduction in the amplitude of the discharge oscillations whose excessive peaks had previously compensated for the simulation's overly long recovery period after each oscillation peak. Under this assumption then, it would seem that the quench model did do an effective job of increasing the time-dependent agreement of the simulation with experiment, but sadly was unable to correct the underlying disagreement of the code with reality.

6.8 Evaluation of the Quench Model's Viability

The preceding sections have demonstrated that the quench model could indeed improve several aspects of simulation performance. The exit plane anomalous diffusion barrier was seen to be accurately predicted without a priori knowledge of its location or structure. The electric potential and electron temperature profiles calculated using the quench model showed improvement over simpler "guesses" of the thruster transport profile. Finally, the structure of the main discharge oscillations in terms of both frequency and amplitude was also better predicted through use of the quench model.

However, we have been unable to show definitively that the quench model perfectly predicts the rate of anomalous diffusion in the thruster, as we were unable to discern what portion of the final disagreement with experiment was due to the simulation as a whole and what portion was due to the anomalous diffusion model alone. We have demonstrated, though, that the quench model does provide a definite improvement over wild guesses to the anomalous transport profile. It is then an invaluable tool for the predictive simulation of unbuilt and untested Hall thrusters for which there may exist no empirical data to aid in the definition of accurate anomalous transport. In addition, we have highlighted the strong dependence of anomalous diffusion on the transit-time mode oscillations and the possible amelioration of this transport through the use of strong ExB shearing.

Chapter 7

Conclusions and Future Work

This thesis has described the development and implementation of an efficient and accurate parallelized simulation for Hall-effect thrusters. Computational modeling of these thrusters is a powerful and economic means of investigating integral properties of their plasmas. Numerical tools like ours will continue to be employed to more easily investigate important performance-altering phenomena such as anomalous diffusion and the effects of ion-wall interactions.

7.1 Unique Numerical Advances

The simulation that was herein developed incorporated a number of new and unique features that previous Hall thruster simulations had not. These features included:

- The use of unstructured, non-orthogonal meshes for greater freedom in domain choice and higher quality grid elements.
- Electrostatic solvers capable of operating efficiently and accurately on the novel meshes we employed.
- Semi-analytic integration of the particle motion equations
- Varied weighting of computational particles in order to achieve full statistical coverage of even the low probability tails of the velocity distribution.

- Periodic redistribution and re-weighting of particles to ensure efficient particle usage.
- Non-Monte Carlo, largely deterministic collision algorithms designed to operate on refined meshes with particles of various weights.
- A non-empirical, predictive calculation of the anomalous diffusion coefficient was introduced. This "quench model" was shown to greatly improve the agreement between simulation and experiment. It eliminates the need for the ad hoc and controversial expert adjustment of the anomalous transport coefficient which previously had been either assumed constant or empirically guessed.

7.2 Results from Application of the Simulation

To demonstrate the utility of our new numerical algorithm, we applied it to two real-world Hall thrusters. The following summarizes some of the more interesting results obtained via those computational experiments:

- Due to our more accurate coverage of the velocity space, our model predicted significantly higher electron temperatures in the channel of the P5 thruster than the previous PIC-MCC model. Our temperatures more closely resembled those found in experiments. This improvement also led to better estimates of the singly- and doubly-charged ion densities.
- The use of the quench model strongly altered a number of predicted aspects of the P5 thruster's performance. Calculated peak electron temperatures were increased to more experimentally-agreeable levels and were also better localized than previous models had predicted. The gradient in potential was moved farther down the channel as it is observed to be in experiments. Finally, predictions of the frequency and amplitude for the primary discharge oscillation were made much more reasonable through this more correct calculation of the anomalous transport.

- When our simulation was applied to the initial configuration of the NVHT, we immediately recognized a severe oversight in that thruster's design. The electric potential in the thruster's channel appeared to be highly sub-optimal.
- Simulation of the NVHT was able to capture the same trends in ion energy as an experimental RPA analysis. Better yet, our simulation filled in the missing information below 500 Volts which corrupt measurements had previously left unknown.
- Our simulation was also applied to the modified version of the NVHT that was partially inspired by our observations of the original electric potential and thruster performance. We showed numerically that this new configuration, which is now under investigation experimentally, is not likely to significantly improve the energy of the ions produced. However, the simulations predicted some other minor benefits of the new configuration, such as higher thrust, a reduced beam divergence and the possibility of plasma ignition at significantly lower potentials.

The algorithm developed in this thesis has already proven a useful tool for the designers of tomorrow's Hall thrusters. The code is ready and waiting to face whatever new challenges that might some day await it. We hope sincerely that our efforts will continue to bear fruit and that this model will be applied by future researchers, as well, to investigate whatever innovative configurations they might conceive.

7.3 Recommended Future Tasks

While we have accomplished much, science is a never-ending pursuit, and there is plenty more waiting to be done by those who wish to pick up the chase where we must needs leave it.

- **Three-Dimensions** – Much of the physics, especially of anomalous transport, requires the simulation of all three dimensions of the thruster simultaneously. While a 3-D simulation with the complexity of the code presented here is possibly still out of the question computationally, simpler models could be created that might be run in three-dimensions given enough computational power and patience.

- **Physical Parameters** – One of the most troubling aspects of this simulation and its predecessors is their reliance on the two acceleration tricks of an artificial mass ratio and an altered free space permittivity. While these methods do significantly speed up the simulation and make it viable computationally, they can confound the results and may be causing inaccuracies. Future researchers would do well to explore alternative methods of acceleration.
- **Multi-Scale Simulations** – One of the most promising of such alternative acceleration techniques would be the implementation of a truly multi-scale technique. RRC provides an initial attempt at this, but still requires a very small time step if the grid is well-refined. A method which could run the simulation first on the sheath and other fine structures, then update the bulk channel plasma and move the heavy particles, then calculate the sheath again, and so on, if done correctly, could greatly improve the efficiency and power of our current simulation techniques.
- **Improved Boundary Conditions** – The set of boundary conditions handled by this simulation is enough to model Hall thrusters, but could easily be extended to handle other plasma phenomena as well. In addition, many of the conditions are overly-simplified such as the secondary electron emission calculations and the lack of material sputtering by ions.
- **Erosion Estimations** – The "holy grail" of Hall thruster simulations, it would be highly beneficial if lifetime analyses could be accurately conducted via simulation. To achieve this goal, sputtering must be included, accurate remeshing must be performed on the fly as material is sputtered away from the walls, and the efficiency of the simulation must be drastically increased to allow for runs simulating hours of operation instead of milliseconds.
- **Excitation Levels** – Presently, the simulation lumps all of the numerous excitation collision types into a single cross-section. A more detailed treatment of this important process may lead to more accurate estimates of energy usage and

localization. Comparisons with spectroscopic data would also require this more detailed model.

- **Parallelized SOR** – The successive over-relaxation algorithm used to solve the Gauss's Law matrix equations in many trials is the only remaining serial portion of the algorithm. The MUMPS direct solver is parallel, but its performance was shown to be less than exceptional for our application. The parallelization of this SOR should not be terribly difficult and detailed discussion of that problem can be found in [33].
- **Neutral-Neutral Collisisons** – As mentioned in the thesis, the neutral-neutral elastic collision model implemented does not guarantee the proper rate of convergence to a Maxwellian. For Hall thruster applications, this is generally an acceptable imprecision, but a more accurate model could be developed in the future if required.
- **Dielectric Sheath** – The sheath patch used to deal with the problems arising from the artificial mass ratio originated with Blateau. It is unfortunately very ad hoc and may be creating unexpected inaccuracies in the simulation. If the use of the artificial mass ratio is to be continued, this patch should probably be readdressed.

Appendix A

Usage of Tecplot Mesh Generator Add-on

A.1 Brief Description

Tecplot's mesh generator add-on is a powerful, relatively inexpensive tool for creating complicated multi-zone geometries and various types of meshes to cover them. While the mesh generator is capable of outputting mesh files in a number of different formats, our only focus for this thesis will be on its capability to convert multi-zone unstructured or structured, quadrilateral or triangular meshes into a single quadrilateral unstructured mesh which is then output in a simple to parse ASCII text format.

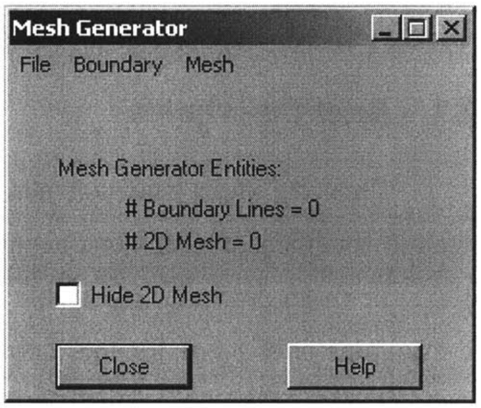
Tecplot allows for the creation of multi-segment boundaries having practically any shape, including poly-lines, circular arcs, and conic sections. The number and spacing of nodes along these boundaries can be specified through a clear, simple to learn graphical interface. Once a region's boundary outline has been created, the program contains algorithms for creating a number of different types of meshes, including algebraic structured, hyperbolic structured, and triangular unstructured meshes. Since all meshes can eventually be converted to a single quadrilateral unstructured zone, it is often easiest and most fruitful to create multi-zoned algebraic structured meshes of each natural region in a simulation domain and then combine the results into a single output. This is the technique that will be demonstrated in the following section which consists of a step-by-step example of creating a mesh with Tecplot's mesh generator add-on.

A.2 Example Step-By-Step Mesh Creation

This step-by-step tutorial demonstrates the subset of features from Tecplot's Mesh Generator tool that would be most useful to a future student continuing the present work or endeavoring to use Mesh Generator for a project of similar scope. Two structured algebraic meshes connected along a boundary will be created. These will then be saved as a single quadrilateral unstructured mesh capable of being used for the simulations detailed in this thesis or other such numerical simulations. For a more detailed and complete discussion of Mesh Generator's features, the Mesh Generator User's Manual may be consulted.

A.2.1 Starting Mesh Generator

Once installed, mesh generator appears under the **Tools** menu of the typical Tecplot screen. To begin this tutorial, first go to the **File** menu and click on **New Layout**. This brings up a fresh workspace. Next, start Mesh Generator by locating it under the **Tools** menu and clicking. The small mesh generator window should open containing its own menu list, some information about the mesh currently being created, and the standard **Close** and **Help** buttons. For reference, an image of this window is shown in Figure A-1.



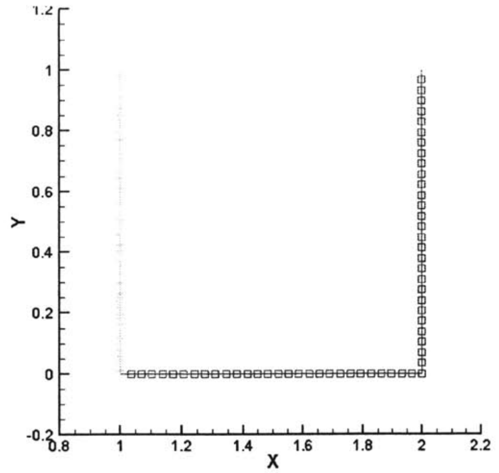
Mesh Generator Entities:
Boundary Lines = 0
2D Mesh = 0

Figure A-1: Mesh Generator's Main Window

A.2.2 Creating Mesh Boundaries

We will begin by creating a simple square boundary. To do this we will create four poly-lines which will correspond to Tecplot's Imin, Jmin, Imax, and Jmax boundary inputs. For details on the specific requirements of these boundaries, please consult the Mesh Generator User's Manual. For now, it is enough to think of Imin and Imax as the bottom and top boundaries, respectively and Jmin and Jmax as the left and right boundaries, respectively. If this logic is extended regardless of the actual mesh shape, it will ensure the proper ordering of elements and nodes.

1. In the Mesh Generator main window, click on the **Boundary** menu item and select **Create Polyline.** The polyline creation window should now open. In this menu, find the input text fields for X and Y control point values. The text fields should currently read 0 and 0. We will make our square's first corner start at the point (1,0), so change the value in the X text field to 1 and leave the Y field at 0.
2. Next click the **Insert Before** button. The point (1,0) should now be displayed as a control point in the list on the left side of the Polyline window.
3. We will have our first line end at the point (2,0), so now change the value in the X text field to 2, still leaving the Y field set to 0. Find the **Insert After** button and click it. This should insert the control point (2,0) beneath the point (1,0) in the window on the left.
4. Now we will determine the spacing of nodes along the boundary. Click on the **Node Distribution** button to bring up the Node Distribution dialog box. This box contains a number of useful features which control where Tecplot will create nodes along the boundary. For now, we will just find the Distribution drop-down box, and select **Even Spacing**. Then in the Number of Nodes text field, enter 30. Accept the remaining default values and close this window by clicking **OK**.



5. Back in the polyline creation window, locate the Line Label text field. The label entered here will henceforth designate a name for this boundary for use in the creation of meshes or for a label in the output mesh file. Enter METAL1 for this boundary.
6. Finally, click the **Create** button to complete the creation of the bottom boundary.
7. To create the other three sides of the square, simply repeat steps 2 through 6, substituting the appropriate control point values and boundary labels. That is,

make METAL2 the right side of the square by selecting control points (2,0) and (2,1). The left side of the square we will call METAL3 and should have control points (1,0) and (1,1). Finally, the top will be METAL4 and have control points (1,1) and (2,1). Choose 30 evenly spaced nodes for all four boundaries for simplicity. To view the resulting boundary more clearly, close the polyline creation window, click on **View** in the Tecplot menu bar, and select **Nice Fit to Full Size**. The resulting boundary should look something like that shown in Figure A-2.

8. Now we will create a three-part semi-circular mesh to stick onto the right side of our square. First click on the **Boundary** menu item and select **Create Circular Arc**. The Circular Arc creation window should open. There are a number of ways to create circular arcs in mesh generator. We will use the Center, Starting Point, and Arc Angle method, so check this box in the top of the window if it is not already checked.
9. Our semi-circle will be centered at (2,.5) so enter 2 in the Center-X text field and .5 in the Center-Y text field. We would like this boundary to start at the top right corner of the square, so enter 2 and 1 in the start X and Y fields, respectively. Finally, we would like our arc to travel 60 degrees, so enter 60 in the angle text field.
10. We will once again use 30 evenly spaced nodes along our arc boundaries, so click on the **Node Distribution** button. In the dialog box that appears, find Even Spacing in the distribution drop-down menu, set the Number of Nodes text field to 30, and click **OK**.
11. Label this boundary METAL5 in the Line Label text field, and press the **Create** button to complete creation of this 60 degree arc.
12. Create two more arc boundaries by repeating steps 9-11 with the appropriate values for center points, starting points, and arc angle. For METAL6, the center will again be (2,.5), the starting point will be (2,0), and the arc angle will be -60 degrees. Note the negative arc angle used to denote clockwise arcs. To create METAL7, use again the center of (2,5). To fill in the starting point text fields, simply click on the METAL6 boundary that we just created, somewhere close to

unconnected end. Next click the **Select Endpt** button next to the Start X and Y text fields. The values 2.433012701 and .25 should appear for X and Y respectively. Once again, use a -60 degree arc angle to finish METAL7.

13. View your work, by clicking the **View** button in the Tecplot main menu and once again selecting **Nice Fit to Full Size**. Your completed boundaries should look like those in Figure A-3.

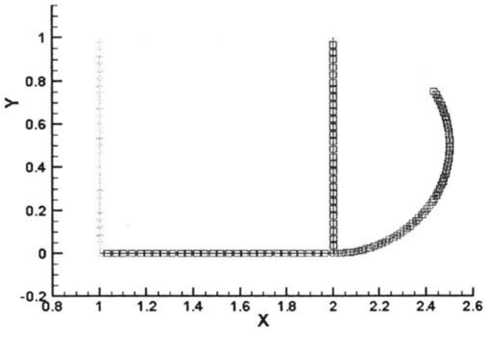


Figure A-3: The completed boundaries

14. Now we will create the meshes to go with these two separate zones. If the create circular arc window is still open close it first. Then find the **Mesh** toolbar item in Mesh Generator's main window. Click **Create Algebraic/Elliptic Structured**.

15. Let us create the square mesh first. In the Mesh creation window, click on the **Add From List** button beside the text field for IMin. In the dialog box that pops up, select boundary METAL1. Beside IMax, click **Add From List** and select the boundary METAL4. Notice that this is the bottom and top boundaries. Again for JMin add boundary METAL3 from the list, and finally for JMax add boundary METAL2 from the list. Name this mesh SQUARE in the Mesh Label text field and click the **Create** button.

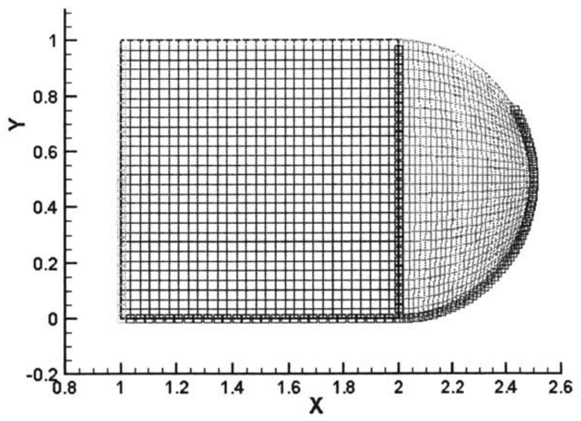


Figure A-4: Completed Meshes

16. We will create the semi-circular mesh in a similar fashion. For the JMin text field, add the boundary METAL2 from the list. For JMax add METAL7, for IMin select METAL6, and for IMax select METAL5. Note that the JMIN boundary corresponds to its

neighbor's JMAX boundary. This ensures, for example, that an element's "east" neighbor will have that element as its "west" neighbor. Name this mesh CIRCLE in the Mesh Label text field and click the **Create** button. Your Meshed boundaries should look something like those shown in Figure A-4.

17. Our final step in this tutorial will be to save the two-zone structured mesh we have created as a single unstructured quadrilateral zone. To do this, click on the **File** menu in the Mesh Generator's main window and select the option **Write Mesh File**. We want to write our file in the Tecplot format so choose this option in the Format drop-down list. We also want our file to be a readable ASCII format, so check the box next to this option if it is not already. Lastly, we want to write our mesh as a Single Quadrilateral Zone, so check the box next to this option if it is not already checked. Leave both of our created meshes highlighted and click **OK**. Tecplot will then prompt you for a file name and a directory in which to save the created file, after which you will have created an example mesh and completed this tutorial.

A.3 Incorporation of Mesh Generator Output Files Into Simulation

A.3.1. Mesh Output File Format

Tecplot Mesh Generator was designed to be simple and easy to use. The output file format is therefore fairly simple to understand and relatively easy to parse for most common languages. An example of a very simple output file along with the mesh it represents is shown in Figures A-6 and A-5, respectively. We will walk though the various sections of this output file in order to illustrate the output file format.

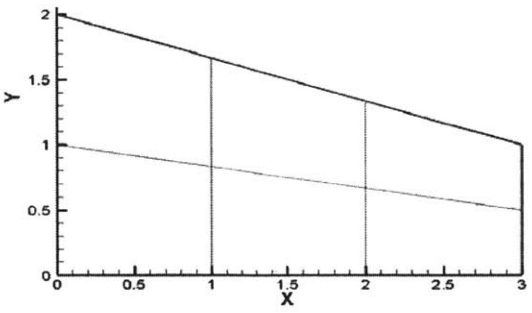


Figure A-5: A simple unstructured quadrilateral mesh which yields the output file shown in Figure A-6.

```
TITLE = "Mesh Generator" VARIABLES = X, Y
USERREC = "DBPATCH 1 METAL1 3"
USERREC = "1 4 4 7 7 10 "
USERREC = "DBPATCH 1 METAL2 2"
USERREC = "10 11 11 12 "
USERREC = "DBPATCH 1 METAL3 2"
USERREC = "1 2 2 3 "
USERREC = "DBPATCH 1 METAL4 3"
USERREC = "3 6 6 9 9 12 "
ZONE T = "Converted grid" N = 12 E = 6 F = FEBLOCK ET = QUADRILATERAL
0 0 0 1 1 1 2 2 2 3 3 3
0 1 2 0 0.8333333 1.666667 0 0.6666667 1.333333 0 0.5 1
1 2 5 4 2 3 6 5 4 5 8 7 5 6 9 8 7 8 11 10 8 9 12 11
```

Figure A-6: The output file for the simple unstructured quadrilateral mesh consisting of 6 elements and 12 nodes as shown in Figure A-5.

As can be seen from Figure A-6, the first line of the output file presents information about this output file for use by Tecplot. In particular, the title of the file is given which can be used to identify the origins of this mesh, how it was created, or details about its construction. The second part of the line defines how many variables will be written and what to name each particular variable. In the case of our unstructured quadrilateral meshes, this line should always correspond exactly to the one in Figure A-6 and need not be changed or parsed.

The next section of output file is denoted by the comment tags, "USERREC." These statements define the boundaries and the nodes which lie along them. The token "USERREC = 'DBPATCH 1" signals that a new boundary is about to be defined. The next symbol, in the case of Figure A-6, "METAL1", defines the name of the boundary. For our Fortran program, we currently presuppose that boundaries are labeled with five-letter prefixes corresponding to the boundary type and then a positive, unique integer greater than zero. This integer currently must be less than 1000. "METAL1" translates to a metallic boundary while "SPACE5" would be a free space boundary and so on. The final token in the boundary definition line is another integer. This integer denotes the number of edges which lie along this particular boundary. Thus, in the case of the second line of Figure A-6, there are 3 edges which lie along boundary METAL1.

The statements following a boundary definition statement also begin with the tag "USERREC." However, these statements list the nodes that reside on the boundary which has just been defined. For example, in Figure A-6, the line "USERREC = '1 4 4 7 7 10 '" tells us that the edges between nodes 1 and 4, between nodes 4 and 7, and between nodes 7 and 10 lie along boundary METAL1. Any number of lines beginning with the "USERREC" tag and containing lists of nodes may be necessary to ensure that all boundary nodes are included. So if the maximum line length is reached, but there are still edge nodes remaining to be printed, a new line will be started with the "USERREC" tag and the remaining nodes in a continuing list.

The signal that the boundary definition section has been completed is the line beginning with the "ZONE" tag. In the case of our quadrilateral unstructured meshes, this line will always have the same basic format. It begins with the "ZONE" tag which is followed by the grid type tag, "T = 'Converted grid.'" The next token begins with "N =" and defines the number of nodes in the mesh. For example, Figure A-5 has 12 nodes and so Figure A-6 shows the output "N = 12." Following this, the number of elements is defined using the "E =" tag. In the case of the simple mesh in Figure A-5, there are 6 elements, so Figure A-6 contains the phrase "E = 6." The remainder of this "ZONE" line will be the same for all of our meshes, and simply defines what type of element scheme we are using.

After the "ZONE" line is completed, the X positions of the node points are printed. The X values are printed in nodal order, from 1 to the number of nodes, and are separated by spaces. If there are more X values than can fit on one line, the list will be simply continued to the next line. After the last X value, however, a new line will always be started before starting to write the Y values.

Following the X values, the Y positions of the nodes will next be written, following the same ordering convention as the X values. Again, multiple lines may be used.

The final section of the output file, which corresponds to the last line of Figure A-6, is the element connectivity list. For quadrilateral meshes, this will be multiple lines containing quadruplets of node numbers which define the four corners of an element. Thus, the first quadruplet in Figure A-6, "1 2 5 4" defines the four corners of element 1 and the third quadruplet "4 5 8 7" defines the four corners of element 3. The corners are generally output in a clockwise direction starting with the lower-left node in the element. Depending on exactly how the mesh is created, this property need not hold, however. The element connectivity list can be continued for multiple lines until all 4*(number of elements) corner nodes have been listed.

Appendix B

Tables of Compiled Experimental Cross-Sections

Krypton

T (eV)	Q_T (10^{-16} cm^2)
0.175	5.03
0.2	4.83
0.225	3.77
0.25	3.22
0.275	2.67
0.3	2.31
0.35	1.75
0.4	1.3
0.45	1.04
0.5	0.828
0.55	0.649
0.6	0.57
0.65	0.495
0.7	0.455
0.72	0.45
0.74	0.441
0.76	0.451
0.8	0.458
0.85	0.482
0.9	0.53
0.95	0.6
1	0.672
1.1	0.853
1.2	1.07
1.3	1.32
1.4	1.54
1.5	1.84
1.75	2.23
2	3.02
2.25	3.83
2.5	4.68
3	6.36
3.5	8.24
4	10.1
4.5	12.04
5	14.08
6	18.03
7	21.67
8	24.65
9	26.2
10	27.04
11	27.27
12	27.29
13	26.57
14	26.14
15	25.4
16	24.59
18	23.34
20	22.29
30	17.42
40	14.92
50	13.64
100	10.71
150	8.68
200	7.3
300	6.1
400	5.23
500	4.5
1000	3.05
2000	1.91
3000	1.52

T (eV)	Q_{EXC} (10^{-16} cm^2)
20	0.728
30	0.983
40	1
50	0.978
60	0.949
70	0.927
80	0.907
90	0.899
100	0.882
150	0.756
200	0.622
300	0.479
400	0.398
500	0.33
600	0.291
700	0.255
800	0.23

T (eV)	Q_+ (10^{-16} cm^2)
15	0.102
16	0.367
17	0.529
18	0.732
20	1.17
22	1.48
24	1.76
26	2.14
28	2.34
30	2.55
35	2.99
40	3.24
45	3.38
50	3.45
55	3.49
60	3.45
65	3.49
70	3.45
75	3.45
80	3.43
90	3.4
100	3.33
110	3.25
120	3.2
130	3.11
140	3.05
150	2.98
160	2.91
170	2.86
180	2.78
190	2.72
200	2.68
225	2.54
250	2.42
275	2.29
300	2.19
350	2.01
400	1.85
500	1.6
600	1.43
700	1.28
800	1.16
900	1.07
1000	0.984

T (eV)	Q_{++} (10^{-16} cm^2)
45	0.0247
50	0.0825
55	0.137
60	0.187
65	0.226
70	0.253
75	0.28
80	0.298
90	0.307
100	0.308
110	0.304
120	0.302
130	0.292
140	0.273
150	0.267
160	0.255
170	0.25
180	0.236
190	0.229
200	0.221
225	0.207
250	0.19
275	0.179
300	0.166
350	0.147
400	0.137
500	0.115
600	0.102
700	0.0912
800	0.0814
900	0.0754
1000	0.067

T (eV)	Q_{+++} (10^{-16} cm^2)
90	0.0035
100	0.0069
110	0.0119
120	0.0163
130	0.02
140	0.0209
150	0.0225
160	0.0249
170	0.0259
180	0.0275
190	0.0281
200	0.0291
225	0.0308
250	0.033
275	0.0333
300	0.0349
350	0.0363
400	0.0374
500	0.0376
600	0.0367
700	0.0364
800	0.0345
900	0.0335
1000	0.0318

Xenon

T (eV)	Q_T (10^{-16} cm^2)
0.73	1.1
0.91	1.28
1.09	1.9
2	9.58
2.18	11.69
2.66	16.01
2.85	18.28
3.23	22.13
3.41	23.91
4.59	35.57
4.77	36.47
5.28	39.09
5.46	40.86
6.55	42.33
7.22	41.38
7.78	40.54
9.14	39.27
10	38.8
15	35.6
20	33.3
30	19.8
50	13
100	10.7
200	9.5
300	8.1
500	6.4
1000	4.4

T (eV)	Q_{EXC} (10^{-16} cm^2)
8.32	0
8.5	0.026
9	0.126
9.5	0.131
10	0.18
10.5	0.24
11	0.42
11.5	0.62
12	0.84
12.5	1.05
13	1.28
14	1.7
15	2.14
16	2.55
18	3.35
20	3.73
25	3.85
30	3.57
40	2.85
50	2.4
60	2.1
70	1.85
80	1.66
90	1.52
100	1.38
150	1
200	0.8
300	0.568
400	0.465
500	0.395
600	0.344
700	0.302
800	0.277
900	0.252
1000	0.231
2000	0.132
3000	0.095
4000	0.075
5000	0.063
10000	0.036

T (eV)	Q_+ (10^{-16} cm^2)
12	0.17
14	0.73
16	1.37
18	2.01
20	2.43
22	2.9
24	3.33
26	3.62
28	3.8
30	4.01
35	4.48
40	4.59
45	4.6
50	4.58
60	4.62
70	4.67
80	4.64
90	4.53
100	4.44
110	4.33
120	4.19
130	4.05
140	3.97
150	3.89
160	3.78
170	3.67
180	3.58
190	3.51
200	3.44
225	3.25
250	3.06
300	2.77
350	2.48
400	2.31
500	2
600	1.75
700	1.58
800	1.42
900	1.32
1000	1.21

T (eV)	Q_{++} (10^{-16} cm^2)
40	0.0688
45	0.174
50	0.257
60	0.363
70	0.386
80	0.398
90	0.461
100	0.503
110	0.534
120	0.53
130	0.517
140	0.491
150	0.465
160	0.439
170	0.415
180	0.397
190	0.39
200	0.373
225	0.345
250	0.325
300	0.305
350	0.294
400	0.267
500	0.24
600	0.22
700	0.204
800	0.19
900	0.175
1000	0.17

T (eV)	Q_{+++} (10^{-16} cm^2)
80	0.01
90	0.0324
100	0.0764
110	0.122
120	0.158
130	0.183
140	0.188
150	0.187
160	0.177
170	0.172
180	0.166
190	0.166
200	0.158
225	0.15
250	0.142
300	0.14
350	0.147
400	0.129
500	0.118
600	0.107
700	0.102
800	0.0921
900	0.0844
1000	0.0825

Appendix C

Simulated Distribution Functions for the P5 Thruster

C.1 Motivation

We present in this section a number of distribution functions which were calculated using our simulation. These functions were computed for a typical run of the P5 thruster at 3kW operation. The use of fully-kinetic codes was in part motivated by the hypothesis that the shape of the distribution functions for electrons was not Maxwellian [33][73][74]. Such shapes would complicate or perhaps even invalidate the use of hybrid-fluid models such as Fife's HPHall [28]. Due to our extremely accurate coverage of velocity space, we were able to take precise measurements of the distribution functions comparable to purely continuous methods [15] at various locations within the simulation, including sheath regions and the ionization zone. The results presented in the next section clearly demonstrate the non-Maxwellian nature of the electron distribution functions and accent the necessity of fully-kinetic simulations where the near anode region of Hall thrusters are concerned.

C.2 Distribution Function Results

Before examining the results, we first note that the distributions will be compared with one of two Maxwellian distributions. The first distribution is the distribution of the parallel velocity vector given by:

$$f_M\left(\vec{v}_{\|}\right) \sim \sqrt{\frac{1}{2\pi}}\frac{1}{v_T}\exp\left(-\frac{v_{\|}^2}{2v_T^2}\right) \tag{C.1}$$

This is computed simply by defining bins in the $v_{//}$ space and placing each particle into one of those bins based on its particular parallel velocity.

The second distribution used for comparison is the distribution of the perpendicular speeds as given by:

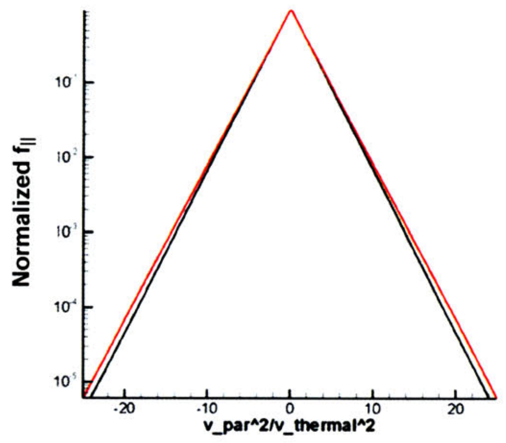
$$f_M\left(\left|v_{\perp}\right|\right) \sim \frac{\left|v_{\perp}\right|}{v_T^2}\exp\left(-\frac{v_{\perp}^2}{2v_T^2}\right) \tag{C.2}$$

Computing this distribution again simply requires calculating and binning the non-directional perpendicular speed of each particle. In both cases, the thermal velocity, v_T, is just given by the simple formula:

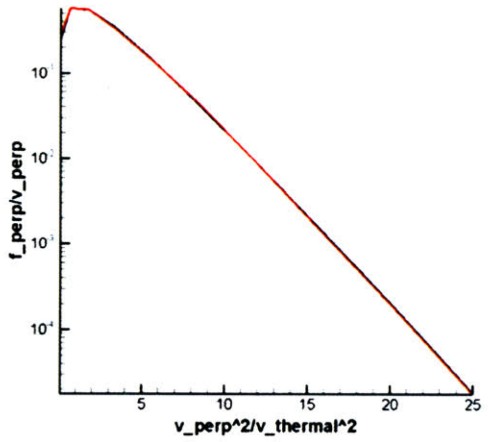
$$v_T = \sqrt{\frac{kT}{m}} \tag{C.3}$$

The parallel and perpendicular temperatures are of course different and are calculated separately.

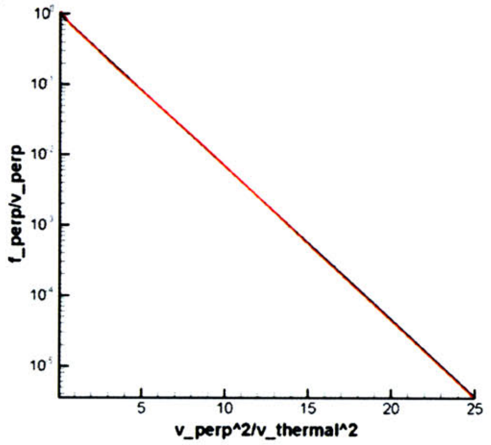
Before compiling data on the distribution functions during thruster operation, we thought it prudent to first make sure that upon initial seeding, our electrons did in fact begin with a Maxwellian distribution. Figure C-1 provides two example images from this initial testing. The structure of this and the following distribution function figures is similar and may require a brief explanation. The first plot of each figure shows the distribution of the velocity vector parallel to the magnetic field. The horizontal axes of these plots have been set to the square of the parallel velocity normalized by the square of the thermal velocity all multiplied by the sign of the parallel velocity. This transforms the Maxwellian distribution into a triangle shape with straight lines centered at 0 on the x-axis. The y-axis in these first plots is simply the calculated distribution. We normalized both the Maxwellian and the calculated distribution by dividing each y-value by the maximum of the distribution given by *(C.1)*. Thus, the plotted Maxwellian then peaks at 1. The remaining plots of this figure represent the distribution of perpendicular speeds. Since it is a distribution of speeds and not the velocity vector, the x-axis only shows positive values. Again this axis has been plotted as the square of the perpendicular speed normalized by the square of the thermal velocity. In Figure C-1(b), the y-axis has been left to represent simply the calculated distribution. However, since the distribution of speeds given by *(C.2)* includes a multiplication by the perpendicular speed, the last



(a)



(b)



(c)

Figure C-1: Distribution functions for the initially seed electrons. (a) The distribution of the parallel velocity vector, (b) the unnormalized perpendicular distribution of speeds, and (c) the distribution of speeds normalized by the velocity are depicted.

plot, Figure C-1(c), has had its y-axis transformed by dividing the distribution function by the perpendicular speed. This transforms the Maxwellian distribution into a straight, negatively-sloped line. Since the human eye is much more capable of discerning deviations from a straight line than it is at discerning deviations in curvature or slope, these transformations were made for easy, by-eye comparisons of the distribution functions. The perpendicular speed plots were normalized like the parallel plots by dividing each y-value by the distribution shown in *(C.2)*. In all of the distribution function plots, the Maxwellian distributions are depicted in black and the computed distributions are shown in red.

Returning our attention briefly to the actual plots in Figure C-1, we can see that the initial distributions do approximately represent Maxwellians. The small discrepancy, most noticeable in the parallel distribution function, may be due to a number of minor factors. One of the most significant of these is the slight discretization error. When the distribution is initially seeded, the particles are given a three-dimensional Maxwellian distribution by being situated on a cube-like velocity grid in the R,Z, and Θ directions. These grid nodes, of course, do not sit exactly on the grid nodes of the parallel and

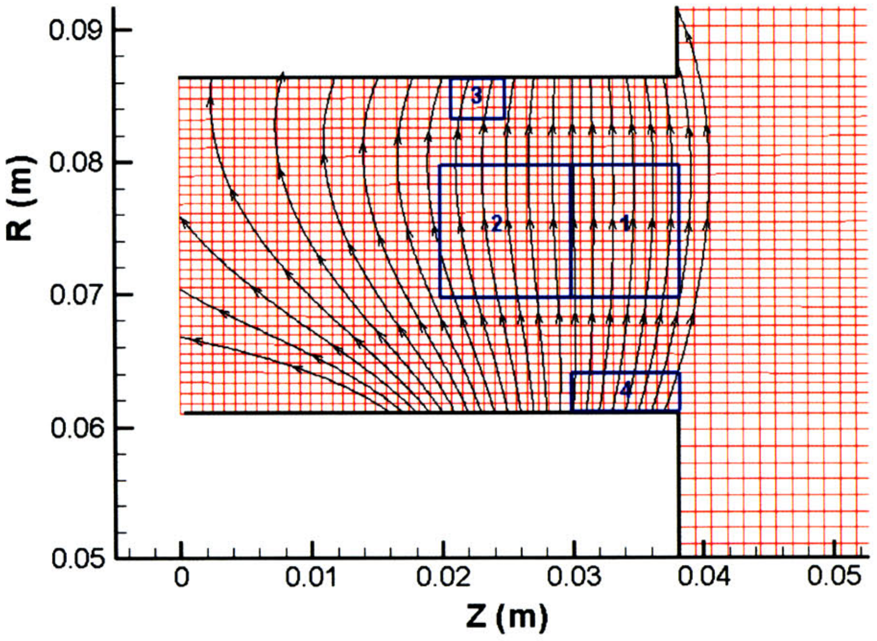


Figure C-2: The regions of the P5 thruster's channel in which distribution functions were calculated are shown by the numbered boxes. The direction of the magnetic field lines are also displayed.

perpendicular velocity bins which are then used to compute the distribution function plots. Due to the discrete nature of the calculation, some error is incurred. Clearly though, this error is small and should not drastically effect the conclusions we may draw from the remaining distribution function plots.

With the initial distributions checked, we next proceeded to collect usable data. Figure C-2 indicates the five regions of the P5 thruster's channel in which distribution functions were calculated, overlaid with the direction of the magnetic field lines for those zones as reference. The first zone represents a "nominal" region near the exit plane of the thruster before significant ionization has occurred and before the electrons have been substantially accelerated. The magnetic confinement in this region is significant. The distribution functions for this zone are shown in Figure C-3. The figure shows that the parallel and perpendicular distributions in zone 1 remain nearly Maxwellian with only a slight depopulation of the high energy perpendicular tail probably due to a small amount of ionization collisions.

Figure C-4, representing zone 2's distribution functions, tells a much different story. In this figure, both functions have significantly depleted high energy tails and an overabundance of particles with lower energies. This zone incorporates the main ionization region of the thruster and clearly suffers from the tail depopulation predicted by Batishchev [14][15].

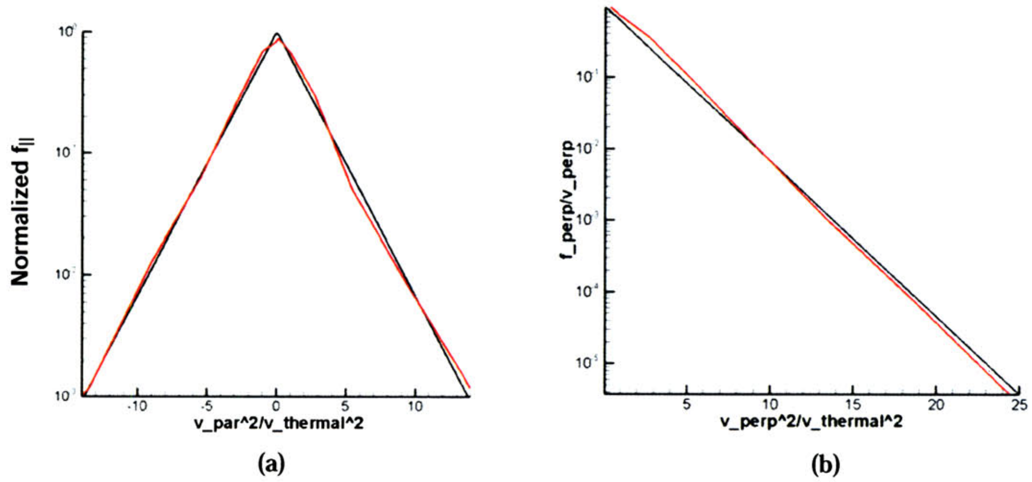


Figure C-3: The (a) distribution of the parallel velocity vector and (b) distribution of the perpendicular speed for zone 1 of Figure C-2.

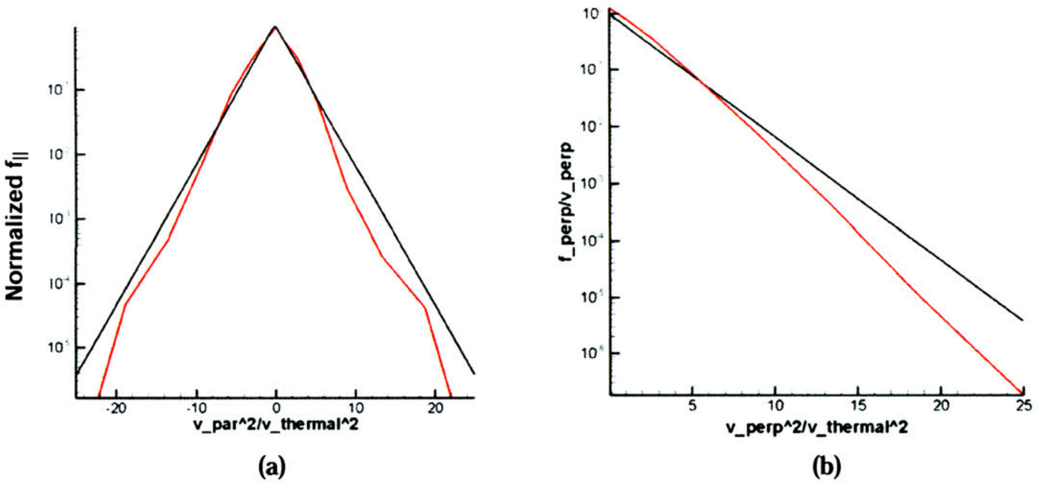
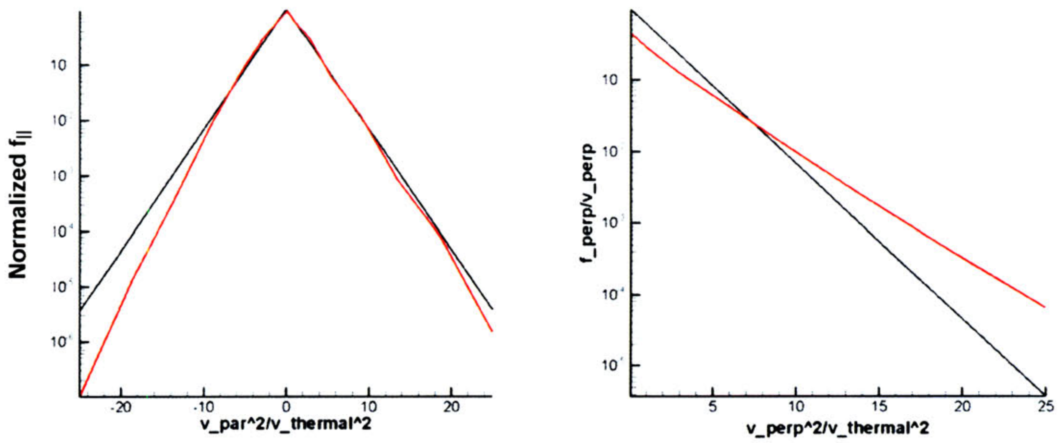


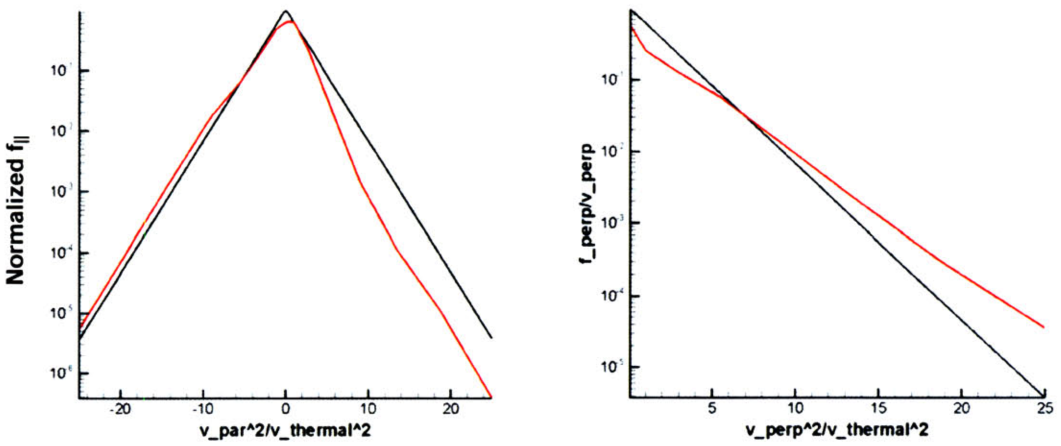
Figure C-4: The (a) distribution of the parallel velocity vector and (b) distribution of the perpendicular speed for zone 2 of Figure C-2.

Figures C-5 and C-6, representing zones 3 and 4, respectively, also demonstrate the non-Maxwellian nature of the electron distributions of a Hall thruster. These zones incorporate the outer and inner wall-sheath regions of this SPT-type thruster. One can see that on the upper wall (Figure C-5(a)), the parallel component rebounding back off the top of the wall has been depleted while at the lower wall (Figure C-6(a)), the opposite parallel tail has instead been depopulated. It was also noted that the wall-directed parallel populations in these regions appear to be nearly Maxwellian. This would seem to indicate that the randomization in the regions outside the wall sheaths is very strong, and the depopulated tail due to wall effects does not survive into the main channel region, as was also demonstrated by Figure C-3(a). Figures C-5(b) and C-6(b) indicate a striking over-population of the high velocity perpendicular tail for which we can provide scant explanation. This over-population could be a result of the reflection of colder particles out of the sheath region or it could perhaps be an artifact of secondary electron emission. One conclusion is definitely clear, however. Representing these regions with simple Maxwellian approximations would be to overlook important properties of the distributions.



(a) (b)

Figure C-5: The (a) distribution of the parallel velocity vector and (b) distribution of the perpendicular speed for zone 3 of Figure C-2.



(a) (b)

Figure C-6: The (a) distribution of the parallel velocity vector and (b) distribution of the perpendicular speed for zone 4 of Figure C-2.

References

[1] Ahedo, E., P. Martinez-Cerezo, and M. Martinez-Sanchez. *One-dimensional model of the plasma flow in a Hall thruster.* Phys. Plasmas 8, 3058, 2001.

[2] Allis, Michelle K., Nicolas Gascon, Caroline Vialard-Goudou, Mark A. Cappelli, and Eduardo Fernandez. *A Comparison of 2-D Hybrid Hall Thruster Model to Experimental Measurements.* 40th AIAA Joint Propulsion Conference, Ft. Lauderdale, FL, 2004.

[3] Amestoy, P. R., I. S. Duff and J.-Y. L'Excellent, *Multifrontal parallel distributed symmetric and unsymmetric solvers,* in Comput. Methods in Appl. Mech. Eng., 184, 501-520, 2000.

[4] Amestoy, P. R., I. S. Duff, J. Koster and J.-Y. L'Excellent, *A fully asynchronous multifrontal solver using distributed dynamic scheduling,* SIAM Journal of Matrix Analysis and Applications, Vol 23, No 1, pp 15-41, 2001.

[5] Amestoy, P. R. and A. Guermouche and J.-Y. L'Excellent and S. Pralet, *Hybrid scheduling for the parallel solution of linear systems.* Accepted to Parallel Computing, 2005.

[6] Bartlett, Philip L. and Andris T. Stelbovics. *Calculation of Electron-Impact Total-Ionization Cross Sections.* Physical Review A, Vol. 66, 2002.

[7] Bashir, Omar and Regina Sullivan. *Near-Vacuum Hall Thruster for Drag Compensation.* Internal MIT report, 2004.

[8] Batishcheva, Alla and Oleg Batishchev. *Multi-Scale Kinetic Model of Ultrafast Laser-Matter Interaction.* UP1.85, 48th APS DPP, Philadelphia PA, 2006; Bulletin APS, 51 (7) 282, October 2006.

[9] Batishcheva, Alla, Oleg Batishchev, and Justin Fox. *Hybrid Kinetic Method For Flows With Sharp Spatial Gradients.* Third MIT Conference on Computational Fluid and Solid Mechanics, 2005.

[10] Batishcheva, A.A. and O.V.Batishchev. *RRC Adaptive Grid Method For Magnetized Plasma.* Proceedings of the 17th Int. conference on the Numerical Simulation of Plasmas, Banff, Canada, May 22-24, 2000, p.29-32.

[11] Batishchev, Oleg. *Adaptive RRC Algorithm for SOL Plasma Simulation.* Bull. APS Vol.43, No.8, 1754, 1998.

[12] Batishchev, O.V. *Hybrid Vlasov/Fokker Planck - PIC method.* Proceedings of the 17th Intl. Conference on the Numerical Simulation of Plasmas, Banff, Canada, May 22-24, 2000, p. 24.

[13] Batishchev, Oleg, Alla Batishcheva, and Justin Fox. *Hybrid Kinetic Model for Ultrafast Laser-Matter Interactions.* Fourth Annual High Intensity Ultrafast Laser Symposium, 2005.

[14] Batishchev, Oleg and Manuel Martinez-Sanchez. *Charged Particle Transport in the Hall Effect Thruster.* 28th IEPC, paper 188, Toulouse, France, 17-21 March, 2003.

[15] Batishchev, O. V., M. M. Shoucri, A. A. Batishcheva, and I. P. Shkarofsky. *Fully Kinetic Simulation of Coupled Plasma and Neutral Particles in Scrape-Off Layer Plasmas of Fusion Devices.* Journal of Plasma Physics. Cambridge University Press, Vol. 61, Part 2, pp. 347-364, 1999.

[16] Bird, G.A. *Molecular Gas Dynamics.* Clarendon Press, Oxford, 1976.

[17] Birdsall, C. K., and A. B. Langdon. *Plasma Physics via Computer Simulation.* McGraw-Hill, New York, 1985.

[18] Blateau, Vincent. PIC Simulation of a Ceramic-lined Hall-effect Thruster. MIT Master's Thesis. 2000.

[19] Buckman, Stephen J. and Birgit Lohmann. *The Total Cross Section For Low-Energy Electron Scattering From Krypton.* Journal of Physics B: At. Mol. Phys., 20, 1987.

[20] Burrell, K.H. *Effects of ExB Velocity Shear and Magnetic Shear on Turbulence and Transport in Magnetic Confinement Devices.* Phys. Plasmas, **4** (5), 1499-1518, 1997.

[21] Cappelli, M. A., N. B. Meezan, and N. Gascon. *Transport Physics in Hall Plasma Thrusters.* 40th AIAA Aeropsace Sciences Meeting and Exhibit, AIAA-2002-0485, 2002.

[22] Celik, Murat, et. al. *Hybrid-PIC Simulation of a Hall Thruster Plume on an Unstructured Grid with DSMC Collisions.* IEPC-03-134, 31st International Electric Propulsion Conference, 2003.

[23] Cheng, Shannon. Ph.D. Thesis, Massachusetts Institute of Technology, 2007.

[24] Choueiri, E. Y. Plasma Oscillations in Hall Thrusters. Physics of Plasmas 8. 2001.

[25] Dias da Cunha, Rudnei. *Parallel Overrelaxation Algorithms for Systems of Linear Equations.* Transputing, Volume 1, 1991.

[26] Yu. B. Esipchuk, A. I. Morozov, et al. Plasma Oscillations in Closed-Drift Accelerators With An Extended Acceleration Zone. Zh. Tekh. Fiz. 43. 1973.

[27] Esposito, et. al. *Correlation Between Magnetic Shear and ExB Shear on JET ITBs.* 29th EPS Conference on Plasma Phys. And Controlled Fusion, Vol. 26B, 2002.

[28] Fife, Mike. *Hybrid-PIC Modeling and Electrostatic Probe Survey of Hall Thrusters.* Ph.D. Thesis, Massachusetts Institute of Technology, 1998.

[29] Fife, J. M., *Two-Dimensional Hybrid Particle-In-Cell Modeling of Hall Thrusters.* Master's Thesis, Massachusetts Institute of Technology, 1995.

[30] Fife, J. M., M. Martinez-Sanchez, and J. J. Szabo. *A Numerical Study of Lowfrequency Discharge Oscillations in Hall Thrusters.* 33rd AIAA Joint Propulsion Conference, Seattle, WA, 1997.

[31] Fon, W. C., K. A. Berrington, and A. Hibbert. *The Elastic Scattering of Electrons from Inert Gases: IV. Krypton.* Journal of Physics B: At. Mol. Phys., 17, 1984.

[32] Foster, John E. Compact Plasma Accelerators For Micropropulsion Applications. IEPC 2001.

[33] Fox, Justin. *Parallelization of a Particle-in-Cell Simulation Modeling Hall-Effect Thrusters.* Master's Thesis, Massachusetts Institute of Technology, 2005.

[34] Fox, Justin M., Alla A. Batishcheva, Oleg V. Batishchev, and Manuel Martinez-Sanchez. *Adaptively Meshed Fully-Kinetic PIC-Vlasov Model For Near Vacuum Hall Thrusters.* 42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit, 9 - 12 Jul 2006, Sacramento, CA, USA.

[35] Fox, Justin, Oleg Batishchev, and Manuel Martinez-Sanchez, *Kinetic Model of Anomalous Transport for Hall Effect Thrusters.* VP1.159, 48th APS DPP, Philadelphia PA, 2006; Bulletin APS, 51 (7) 328, October 2006.

[36] Gallimore, et. al. *Review of EP Activities of US Academia.* 37th Joint Propulsion Conference, AIAA-2001-3227, 2001.

[37] Haas, James M. *Low-perturbation Interrogation of the Internal and Near-Field Plasma Structure of a Hall Thruster Using a High-Speed Probe Posititioning System.* Ph.D. Thesis, University of Michigan, 2001.

[38] Hayashi, Makoto. *Determination of Electron-Xenon Total Excitation Cross-Sections, From Threshold to 100eV, From Experimental Values of Townsend's* . Journal of Physics D: Applied Physics, 16, 1983.

[39] Hirakawa, M. and Y. Arakawa. *Particle Simulation of Plasma Phenomena in Hall Thrusters.* IEPC-95-164, 24th International Electric Propulsion Conference, Moscow, Russia, 1995.

[40] de Heer, F. J., R. H. J. Jansen, and W. van der Kaay. *Total Cross Sections For Electron Scattering by Ne, Ar, Kr, and Xe.* Journal of Physics B: At. Molec. Phys., 12, 1979.

[41] Hockney, R. W. and J. W. Eastwood. *Computer Simulation Using Particles.* McGraw-Hill, New York, 1981.

[42] Ichimaru, Setsuo. *Plasma Physics.* Reading, MA: The Benjamin/Cummings Publishing Co., Inc., 1986.

[43] Jancel, R. and Th. Kahan. *Electrodynamics of Plasmas.* New York: John Wiley & Sons Ltd., 1966.

[44] Jeffrey, Alan and Daniel Zwillinger. *Table of Integrals, Series, and Products.* Elsevier, Inc., 2007.

[45] Jolivet, L. and J. F. Roussel. *Effects of the Secondary Electron Emission on the Sheath Phenomenon in a Hall Thruster.* Third International Conference on Spacecraft Propulsion, Cannes, France, 2000.

[46] Kallenrode, May-Britt. *Space Physics.* New York: Springer, 1998.

[47] Khayms, V. and M. Martinez-Sanchez. Design of a miniaturized Hall thruster for microsatellites. JPC 2006.

[48] Koo, Justin and Ian Boyd. Computational Modeling of Stationary Plasma Thrusters. 2003.

[49] Koo, Justin. Personal Communication. 2006.

[50] Lentz, C. A. *Transient One Dimensional Numerical Simulation of Hall Thrusters.* Master's Thesis, Massachusetts Institute of Technology, 1993.

[51] L'Excellent, Jean-Yves. Personal Communication. December 2006.

[52] Lewis, B. R., I. E. McCarthy, P. J. O. Teubner, and E. Weigold. *The Elastic Scattering of Electrons From Krypton, Neon, and Xenon.* Journal of Physics B: Atom. Molec. Phys., 7, 1974.

[53] McCormick, S.F., ed. *Multigrid Methods.* New York: Marcel Dekker, Inc., 1988.

[54] McEachran, R. P. and A. D. Stauffer. *Elastic Scattering of Electrons From Krypton and Xenon.* Journal of Physics B: At. Mol. Phys., 17, 1984.

[55] McIntyre, Neil, et al. *Numerical Efficiency in Monte Carlo Simulations—Case Study of a River Thermodynamic Model.* J. Envir. Engrg., Volume 130, Issue 4, pp. 456-464, 2004.

[56] Martinez-Sanchez, Manuel. Personal Communication. May 2007.

[57] Mimnagh, D. J. R., R. P. McEachran, and A. D. Stauffer. *Elastic Electron Scattering From the Noble Gases Including Dynamic Distortion.* Journal of Physics B: At. Mol. Opt. Phys., 26, 1993.

[58] *Naval Research Laboratory Plasma Formulary* (Ed. J.D. Huba), published by ONR, revised 2004.

[59] Noguchi, R., M. Martinez-Sanchez, and E. Ahedo. *Linear 1-D Analysis of Oscillations in Hall Thrusters.* IEPC-99-105, 26th International Electric Propulsion Conference, 1999.

[60] Opal, C. B. , W. K. Peterson, and E. C. Beaty. *Measurements of Secondary-Electron Spectra Produced by Electron Impact Ionizationof a Number of Simple Gases.* J. Chem. Phys, 55(2):4100-4106, 1971.

[61] Parra, F. I., E. Ahedo, J. M. Fife, and M. Martinez-Sanchez. *A Two-Dimensional Hybrid Model of the Hall Thruster Discharge.* Journal of Applied Physics, Vol. 100, 1, 2006.

[62] Pigeon, Tim and Ryan Whitaker. *Analysis of a Near-Vacuum Hall Thruster.* 42nd Aerospace Sciences Meeting. Reno, NV, AIAA-2004-127, 2004.

[63] Potter, D. *Computational Physics.* Wiley & Sons, London, 1973.

[64] Rejoub, R., B. G. Lindsay, and R. F. Stebbings. *Determination of the Absolute Partial and Total Cross Sections for Electron-Impact Ionization of the Rare Gases.* Physical Review A, 65, 2002.

[65] Roy, Subrata and Birendra Pandey. *Development of a Finite Element Based Hall Thruster Model for Sputter Yield Prediction.* IEPC-01-49. 29th International Electric Propulsion Conference, 2001.

[66] Schie, Eddie and Jan Middelhoek. Two Methods to Improve the Performance of Monte Carlo Simulations of Ion Implantation in Amorphous Targets. *IEEE Transactions on Computer-Aided Design*, Vol. 8, No. 2, February 1989.

[67] Serway, Raymond A. and Robert Beichner. *Physics for Scientists and Engineers.* 2003.

[68] Sin Fai Lam, L. T. *Relativsitc Effects in Electron Scattering by Atoms III. Elastic Scattering by Krypton, Xenon, and Radon.* Journal of Physics B: At. Mol. Phys., 15, 1982.

[69] Snir, Marc, et al. *MPI—The Complete Reference.* Cambridge, MA: MIT Press, 1998.

[70] Sorokin, A. A., et. al. *Measurements of Electron-Impact Ionization Cross Sections of Argon, Krypton, and Xenon by Comparison with Photoionization.* Physical Review A, 61, 2000.

[71] Subramanian, K. P. and Vijay Kumar. *Total Electron Scattering Cross Sections for Argon, Krypton, and Xenon at Low Electron Energies.* Journal of Physics B: At. Mol. Phys., 20, 1987.

[72] Sullivan, Kay. PIC Simulation of SPT Hall Thrusters: High Power Operation and Wall Effects. MIT Master's Thesis. 2004.

[73] Szabo, James. Fully Kinetic Numerical Modeling of a Plasma Thruster. Ph.D. Thesis. Massachusetts Institute of Technology. 2001.

[74] Szabo, James, Noah Warner, Manuel Martinez-Sanchez, and Oleg Batishchev. *A Full Particle-In-Cell Simulation Methodology for Axisymmetric Hall Effect Thruster Discharges.* Journal of Propulsion and Power (under review), 2007.

[75] Tajima, T. *Computational Plasma Physics: With Applications to Fusion and Astrophysics.* Addison-Wesley Publishing Company, Inc., Redwood City, CA, 1989.

[76] Thompson, W.B. *An Introduction to Plasma Physics.* Reading, MA: Addison-Wesley Publishing Company, Inc., 1964.

[77] Trajmar, S., S. K. Srivastava, H. Tanaka, H. Nishimura, and D. C. Cartwright. *Excitation Cross Sections for Krypton by Electrons in the 15-100eV Impact-Energy Range.* Physical Review A, Vol. 23, No. 5, 1981.

[78] Wagenaar, R. W. and F. J. de Heer. *Total Cross Sections for Electron Scattering from Ne, Ar, Kr, and Xe.* Journal of Physics B: Atom. Molec. Phys., 13, 1980.

[79] Waltz, R.E., J. Candy & M.N. Rosenbluth. *Gyrokinetic turbulence simulation of profile shear stabilization and broken gyroBohm scaling.* Phys. Plasmas, vol. 9, page 1938, 2002.

[80] Waltz, R.E., R.L. Dewar & X. Garbet. *Theory and simulation of rotational shear stabilization of turbulence.* Phys. Plasmas, vol. 5, page 1784, 1998.

[81] Waltz, R.E., G.M. Staebler, W. Dorland, G.W. Hammett, M. Kotschenreuther & J.A. Konings. *A gyro-Landau-fluid transport model.* Phys. Plasmas, vol. 4, page 2483, 1997.

[82] Williams, George J., et al. <u>Characterization of the FMT-2 Discharge Cathode Plume.</u>

[83] Williams, J. F. and A. Crowe. *The Scattering of Electrons From Inert Gases II. Absolute Differential Elastic Cross Sections for Neon, Krypton, and Xenon Atoms.* <u>Journal of Physics B: Atom. Molec. Phys.</u>, Vol. 8, No. 13, 1975.

[84] http://fti.neep.wisc.edu/~jfs/neep602.lecture30.plasmaProp.96

[85] http://fusionenergy.lanl.gov/Documents/MTF/Why_MTF/Why-MTF-Comments.html

[86] http://graal.ens-lyon.fr/MUMPS/

[87] http://mathworld.wolfram.com

[88] http://ocw.mit.edu/OcwWeb/Nuclear-Engineering/22-611JIntroduction-to-Plasma-Physics-IFall2003/CourseHome/index.htm

[89] http://www.brainyquote.com

[90] http://www.tecplot.com

[91] <u>http://www-unix.mcs.anl.gov/mpi/</u>