

INTERACTION OF ENERGETIC SOLAR PLASMA WITH THE HERMEAN  
MAGNETOSPHERE DURING NOMINAL AND ICME CONDITIONS

A Thesis  
Presented to  
The Academic Faculty

by

Stoyan Ivanov

In Partial Fulfillment  
of the Requirements for the Degree  
Masters in the  
School of Earth and Atmospheric Sciences

Georgia Institute of Technology  
May 2016

COPYRIGHT© 2016 BY STOYAN IVANOV

INTERACTION OF ENERGETIC SOLAR PLASMA WITH THE HERMEAN  
MAGNETOSPHERE DURING NOMINAL AND ICME CONDITIONS

Approved by:

Dr. Carol Paty, Advisor  
School of Earth and Atmospheric Sciences  
*Georgia Institute of Technology*

Dr. James Wray  
School of Earth and Atmospheric Sciences  
*Georgia Institute of Technology*

Dr. Britney Schmidt  
School of Earth and Atmospheric Sciences  
*Georgia Institute of Technology*

Date Approved: 5/16/2016

# TABLE OF CONTENTS

	Page
LIST OF FIGURES	v
SUMMARY	vi
<u>CHAPTER</u>	
1 Introduction	1
MESSENGER, Mercury and the Hermean Magnetosphere	1
Hermean Magnetosphere	3
Magnetic Field	3
Coronal Mass Ejection	3
Studies of the Hermean Magnetosphere	3
2 Simulation, Boundary Conditions and Tools	6
Multifluid MHD Model	6
Simulation Boundary Conditions	7
Simulation Space	7
Nominal Solar Conditions	9
CME Conditions	9
Particle Tracker	10
3D-Linear Interpolator	12
Path Sampling Method	13
3 Simulation and Postprocessing Demonstration and Preliminary Results	15
Nominal Conditions	16
CME Conditions	20

4	Discussion and Future Work	22
	Discussion	22
	Future Work	22
	APPENDIX A: Input File	23
	APPENDIX B: Particle Tracker and Interpolator	25
	B.1: Particle Tracker Code	25
	B.2: 3D-Linear Interpolator Implementation	30
	REFERENCES	32

## LIST OF FIGURES

	Page
Figure 1: Overview of Hermean Magnetosphere	2
Figure 2: Overview of Simulation Space	8
Figure 3: XZ Projection of Multifluid MHD Pressure Output During Nominal Conditions	16
Figure 4: Magnetic Field Path Sampling Comparison	16
Figure 5: Particle Tracking at Nominal Conditions	17
Figure 6: XZ Projection of Multifluid MHD Pressure Output During CME Conditions	18
Figure 7: Particle Tracking at CME conditions	20

## SUMMARY

This dissertation combines work done in modeling plasma interactions in Mercury's near-space environment with MESSENGER observations to create new tools that can be used to further analyze the interactions between the solar wind and the Hermean magnetosphere and surface. This dissertation details the tools we developed, including a particle tracker, a 3-D linear interpolator and a path-sampling method and covers how they can be used to study the Hermean magnetosphere. The Particle Tracker is used to study the high-energy tail of the solar wind particle population and how it interacts with the Hermean magnetosphere by tracing the particle paths through the electric and magnetic fields generated by the multifluid MHD model. The path-sampling method is used to sample the multifluid MHD simulation output along the path of the MESSENGER probe, allowing us to verify the model accuracy.

# CHAPTER 1

## INTRODUCTION

### **MESSENGER, Mercury and the Hermean Magnetosphere**

Mercury is the closest planet to the sun and, since the re-categorization of Pluto as a dwarf planet, also the smallest. Yet despite this it possess a global magnetic field, a magnetosphere and a tenuous exosphere. Observations from Earth bound telescopes and flybys, by Mariner 10, were more recently supplemented by the dedicated Hermean probe MESSENGER. The probe, launched in 2004, reached orbit of the planet in 2011. It operated until April 2015, when it was de-orbited into the Hermean surface. During that time, it gathered an unprecedented amount of data regarding the surface features of the planet, the plasma environment inside and outside the magnetosphere, as well as data on surface composition and magnetic fields. These data have helped build a better understanding of the Hermean system, its interaction with the solar wind and the effects of solar radiation on the planet itself. This dissertation covers a number of points of interest relating to studies of the Hermean magnetosphere and its interaction with the solar wind. It includes descriptions of the modeling of the solar wind-magnetosphere interactions and tools developed to track sputtering sources and validate model output.

#### **Hermean Magnetosphere**

Since the Mariner flyby, Mercury has been known to possess a magnetic field and an associated magnetosphere. The more recent MESSENGER probe has revealed a great deal more about the conditions of the near-Hermean space environment (Johnson, 2012). First, while the planet does not possess a collisional atmosphere, it does possess a tenuous

exosphere. This exosphere is the result of sputtering of surface materials from impacting high-energy ions from the solar wind and, to a lesser degree, liberation of ions through photonic excitement. The principle constituents in the Hermean exosphere and magnetosphere are  $H^+$ ,  $Na^+$ ,  $Mg^+$  and  $Ca^+$  (Anderson, 2008, Benna, 2008, LeBlanc, 2003). The existence of this exosphere, along with the Hermean magnetic field, leads to the presence of a Hermean magnetosphere. However, unlike the magnetosphere of the Earth, Mercury's is much more compact due to the low magnitude of its magnetic field strength and proximity to the sun. Under nominal solar conditions, the bow shock of the Hermean magnetosphere is located about  $1.7 R_M$  sunward, with a magnetotail stretching back 10-100  $R_M$  and with the magnetopause located around  $1.4 R_M$  from Mercury (Winslow et al., 2013). The solar wind at this proximity, composed primarily of ionized hydrogen, has an energy around 1 KeV, though the high-energy tail of the distribution reaches up to 1 MeV and represents the most energetic solar wind particles.

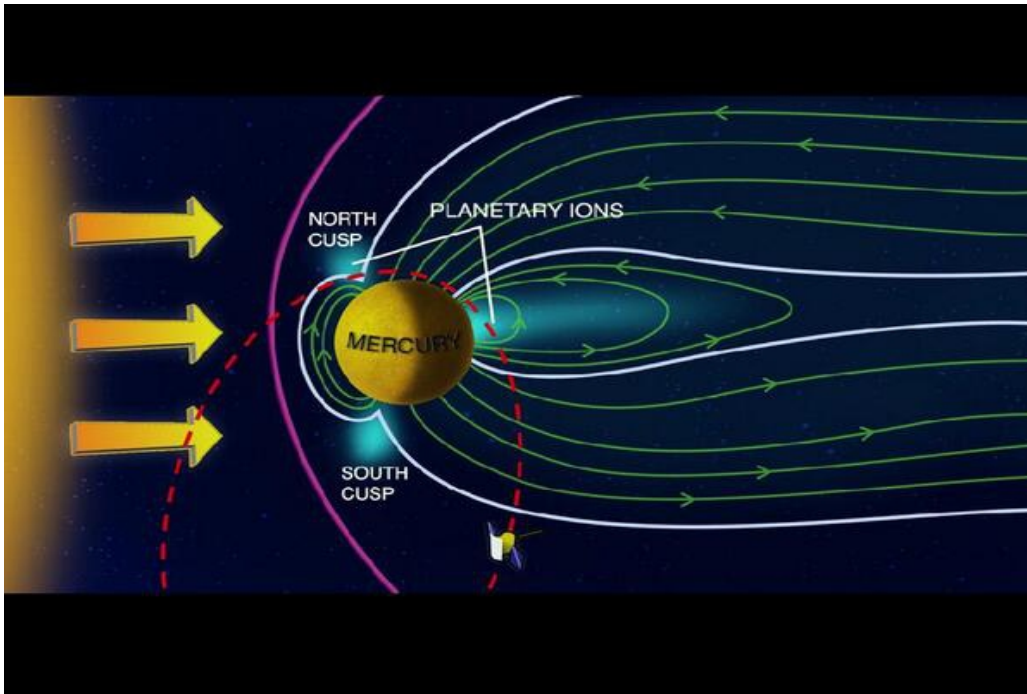


Figure 1: An image showing the overall Hermean magnetic field with respect to the solar wind and the Messenger orbit. Cusps are seen in the 'north' and 'south' of the magnetosphere where the magnetic field is open to the solar wind. Courtesy of J.A. Slavin, NASA Goddard Space Center

## Magnetic Field

The magnetic field of Mercury is thought to be driven by an internal dynamo the energy for which is provided by the heat of accretion as well as tidal stresses exerted on the planet because of its highly elliptical orbit in close proximity to the sun. The magnetic field itself is offset from the center of the planet, N 400 km, towards the northern pole of the planet, indicating the presence of higher order moments in the planetary dynamo (Anderson, 2009). The strength of the field is on the order of 400 nT, equatorially, roughly 1% of Earth's. The dipole central axis is nearly parallel with the rotation axis of Mercury, tilting only 4.5 degrees (McLintok 2008)

## **Coronal Mass Ejection**

Coronal Mass Ejections (CMEs), or Interplanetary Coronal Mass Ejections (ICMEs) occur at the surface of the sun, with a frequency related to the state of solar activity. They are characterized by an increase in particle velocity and density. Average velocity of solar wind inside a CME increases to 500 km/s and up, while density increases by at least a factor of two, compared to regular solar wind parameters. Given Mercury's proximity to the sun, it experiences an increased number of CME interaction events, where the main body of the CME flows over the Hermean magnetosphere. Under these conditions, the Hermean magnetosphere has been observed to contract towards the planet's surface as a result of the increased solar wind ram pressure.

## **Studies of the Hermean Magnetosphere**

A number of other researchers have conducted modeling and analysis of the Hermean magnetosphere to date, folding into these analysis information from the Mariner flyby's, ground observations and the most recent MESSENGER probe. Winslow, et al. 2013, established a location for the average bow shock and magnetopause locations using the data from MESSENGER's flyby and related the positions of the two to the solar wind ram pressure and the Alfvénic Mach number. The magnetopause and bow shock contract at higher solar wind ram pressures, their surfaces moving closer to Mercury. The analysis by Winslow et al. shows a subsolar standoff distance of  $1.45 R_M$ . The magnetopause varies from  $1.55-1.35 R_M$  and the bow shock from  $2.29-1.89 R_M$ , for solar ram pressures of  $8.8-21.6 \text{ nP}$  and  $4.12-11.8 \text{ nP}$  respectively.

Simulational studies conducted by Trávníček et al. 2007, 2009 and 2010 model the Hermean magnetosphere with three-dimensional and three-dimensional hybrid simulations, under high and low solar wind pressure. The results yield a bowshock and magnetopause that generally agrees with observations and the retreat of these boundaries towards the Hermean surface is seen in the simulational output. In their 2010 paper, Benna et al., modeled the Hermean magnetosphere using a bi-fluid magnetohydrodynamic (MHD) model. The model reproduced an asymmetry in the northern and southern magnetospheric cusp, and also indicated the existence of the stable drift belt around the planet.

This dissertation builds on the simulational foundation by incorporating several new techniques, described in greater detail below, in analyzing the Mercury-Solar wind interaction. The use of fluid and hybrid models allows for the simulation and visualization of global scale structures, i.e. bowshock and magnetopause, cusps etc., but doesn't immediately allow for the analysis of particle behavior. Kidder et al, utilized a 3-D multifluid simulation to model the Hermean magnetosphere, looking into magnetospheric erosion and flux-rope reconnection. This dissertation utilizes a particle tracker in tandem with a multifluid MHD model to study how energetic solar wind particles interact with the Hermean magnetosphere. Using the behavior and structures generated by the bulk component of the plasmas, the particle tracker allows for the study of how high-energy particles move through the magnetosphere, penetrate the magnetopause and interact with the surface of the planet.

## CHAPTER 2

### SIMULATION, BOUNDARY CONDITIONS AND ANALYSIS

#### Multifluid MHD Modeling

To simulate the interaction between the solar wind and the Hermean magnetosphere, a multifluid magnetohydrodynamic (MHD) model is used. In this simulation, the different particle species are treated as fluids moving through magnetic and electric fields. The changes in the electric and magnetic fields due to the motion of the charged fluids are taken into account between steps and used in the next step to calculate the motion of the particle fluids. Here, electrons are treated as a massless, conductive fluid. The ions, principally  $H^+$  as well as the heavier ion species  $Na^+$  and  $Mg^+$ , observed in the LeBlanc et al., 2003 paper, are also incorporated as fluids with an assigned mass depending on their atomic masses. The energies and masses of the constituent fluids represent the average population of the ions. The multifluid MHD model does not take into account the energetic tails of the population distribution, either high-energy ions or low-energy ions, only the bulk moment. The equations below are solved by the multifluid MHD code.

The equations used to model the multifluid MHD interactions are the same as those used by Kidder et al. 2008 and are listed below:

$$\frac{\partial \rho_\alpha}{\partial t} + \nabla \cdot (\rho_\alpha \vec{v}_\alpha) = 0 \quad (1)$$

$$\rho_\alpha \frac{d\vec{v}_\alpha}{dt} = n_\alpha q_\alpha (\vec{E} + \vec{v}_\alpha \times \vec{B}) - \nabla P_\alpha - \left( \frac{GM_M}{R^2} \right) \rho_\alpha \vec{r} \quad (2)$$

$$\frac{\partial P_\alpha}{\partial t} = -\gamma \nabla \cdot (P_\alpha \vec{v}_\alpha) + (\gamma - 1) \vec{v}_\alpha \cdot \nabla P_\alpha \quad (3)$$

Where  $\rho_\alpha$  is the mass density,  $v_\alpha$  the bulk velocity,  $n_\alpha$  the number density, and  $q_\alpha$  the charge.  $G$  is the gravitational constant,  $M_M$  the mass of Mercury,  $R$  the planetary radius,  $\mathbf{E}$  the electric field,  $\mathbf{B}$  the magnetic field.  $P_\alpha$  is the pressure for each ion species and  $\gamma$  is the ratio of specific heats (5/3).

Electron dynamics are treated as massless and their behavior is determined by a pressure equation where the electrons motion along the magnetic field lines is rapid enough to be treated as steady state.

$$\frac{\partial P_e}{\partial t} = -\gamma \nabla \cdot (P_e \vec{v}_{de}) + (\gamma - 1) \vec{v}_{de} \cdot \nabla P_e \quad (4)$$

Where  $P_e$  is the electron pressure and  $v_{de}$  the electron drift speed.

In this limit, Ohm's law becomes:

$$\vec{E} = - \sum_i \frac{n_i}{n_e} \vec{v}_i \times \vec{B} + \frac{\vec{J} \times \vec{B}}{en_e} - \frac{\nabla P_e}{en_e} + \eta(\vec{r}) \vec{J} \quad (5)$$

where  $n_e$  is the electron number density,  $e$  the fundamental charge,  $\mathbf{J}$  the current density and  $\eta$  the resistivity. Conductivity is finite only in the ionosphere, and zero elsewhere. Combining these terms with the momentum equation, the following equation is obtained for use in the multifluid MHD model:

$$\rho_\alpha \frac{d\vec{v}_\alpha}{dt} = \frac{q_\alpha n_\alpha}{en_e} (\vec{J} \times \vec{B} - \nabla P_e) - \nabla P_\alpha + q_\alpha n_\alpha \left( \vec{v}_\alpha - \sum_i \frac{n_i}{n_e} \vec{v}_i \right) \times \vec{B} + q_\alpha n_\alpha \eta \vec{J} - \left( \frac{GM_M}{R^2} \right) \rho_\alpha \vec{r} \quad (6)$$

### Simulation Boundary Conditions

#### *Simulation Space*

The simulation solves the multifluid MHD equations in a simulation space defined by a number of grid-points in X, Y and Z. The figure shows the number of grid points which define four nested, simulational boxes. The smaller boxes have a higher

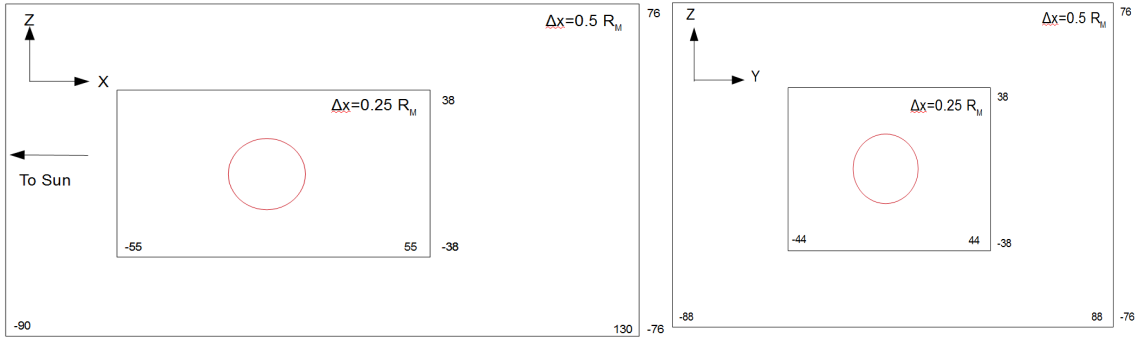


Figure 2: A diagram of the simulation space. Mercury is in the center relative to the YZ portions of the simulation space, but is off-center along the X axis, placing it closer to the sun. This is done to study the down-stream plasma effects in the simulation, since tail-ward plasma interactions are spread out over space. The simulation space is a series of four nested grids, two of which are shown here. The resolution becomes coarser as the grids cover more physical volume. The inner-most grid is of interest in this dissertation and has a physical resolution between grid points of 0.25 Mercury Radii.

resolution, but cover a smaller physical volume than the larger, more granular boxes which cover a greater physical volume. Mercury is located at the 'origin' of this coordinate space, with the sun located along the negative X axis. In this dissertation, motion will be described with respect to these axes. Thus solar wind propagating from the sun towards Mercury moves in the 'positive X' direction. The 'North Hermean Pole' is up in the positive Z direction and the Y axis completes the coordinate system.

Data is available only at these grid points, and is output by the simulation as a data cube. Based on the input parameters, the physical locations of these data points can be calculated and used to sample the magnetic and electric fields at those locations. The data cube is parsed utilizing a 'cutter' to sample a subset of all output data for post-simulation analysis.

#### *Nominal Solar Conditions*

Under nominal solar conditions the solar wind is modeled using the average particle energy. Incoming solar wind consists of  $H^+$  and moves at 400 km/s moving along the positive X axis. The solar wind is treated as coming from the negative X direction uniformly across the YZ plane representing the simulation box 'edge' at the negative X limit. The solar wind has a density of 60 particles  $cm^{-3}$ . These conditions correspond to a solar wind ram pressure of 15 nPa.

#### *CME Conditions*

Under CME conditions, the solar wind again consists of  $H^+$  at a velocity of 600 km/s with a particle density of 100 particles/ $cm^3$ . This represents the more energetic,

higher density plasma released during CME conditions. The solar wind is introduced into the simulation space in the same manner as for the nominal solar conditions. The increased velocity of the particles and the increased density translate into a higher solar ram pressure, thus simulating conditions under which the bowshock and magnetopause will be forced towards the surface of Mercury. These conditions correspond to a solar wind ram pressure of 60 nPa, a factor of four increase over the nominal pressure.

### *Input File*

The multifluid MHD simulation performs the calculations in nonphysical units designed to reduce the possibility of rounding errors that can occur when operations between very large and very small numbers are performed. As such, the solar wind and planetary plasma conditions are converted to these units. Appendix A.1 details the input data based on the physical values provided above. The values there are provided in simulation units.

### **Particle Tracker**

The Particle Tracker is a kinetic model of ion particle motion in specified electric and magnetic fields. It is used to model the behavior of the high-energy tail of the ion population in the solar wind and inside the Hermean magnetosphere. The tracker utilizes a fourth-order Runge-Kutta solver to calculate the Lorentz Force equation and solve for the particle motion. The magnetic and electric fields are those from the MHD simulation, and are determined, in principle, by the bulk flow of the solar wind and magnetospheric ions. However, since the fluids in the MHD simulation only represent the bulk flow, the

motion of higher-energy ions must be modeled independently. The Particle Tracker allows for the tail end distribution to be specified, in terms of individual ions of set velocities, and their behavior observed. Since the population of these energetic ions is always small compared to the bulk flow, they would not themselves effect the magnetic and electric fields through which they travel.

$$\vec{a} = (q/m)[\vec{E} + (\vec{V} \times \vec{B})] \quad (7)$$

The Particle Tracker solves eq. 7 in time with  $\vec{a}$ ,  $\vec{E}$ ,  $\vec{V}$  and  $\vec{B}$  being the acceleration, electric field, velocity and magnetic field vectors, respectively, with q and m being the scalar values of the particle's charge and mass respectively. The particle tracker solves for the accelerations of the ions and propagates those in time to calculate movement over time, thus modeling the travel of the high-energy tail particles through the simulation.

Each particle is tracked independently along with its velocity. When a particle would exit the simulation space, its last position is kept but it is not simulated in future time-steps. If a particle would impact the planet, its final position is stored as the surface of the planet and it is likewise not simulated in future time-steps. This provides a collection of positions showing the motion of each of the simulated particles (Appendix B.1 contains the particle tracker code.)

### Particle Tracker Setup

In order to study the interaction between the more energetic solar wind ions and the Hermean magnetosphere/surface, a series of populations are used. Particles, consisting of H<sup>+</sup> ions, at energies of 10 keV and 1MeV are placed along YZ boundary at

the negative X limit of the simulation space. They are distributed across the whole of the boundary in order to probe the behavior of these energetic particles throughout the simulation space, i.e. near and far away from the bow shock and magnetopause. The energy of the ions is determined with the following equation:

$$V_x=(2E/m)^{.5} \quad (8)$$

Where E is the energy of the particle, m is the mass and  $V_x$  is the velocity in the positive X direction.

### 3-Dimensional Linear Interpolator

The multifluid MHD simulation outputs a volume of electric and magnetic data over the course of its operation. However, due to the parameterization of the model, the electric and magnetic fields are only available at certain grid-points within the model volume. To be used with the particle tracker, which models the motion of the ions continuously throughout the simulation volumes, a method of interpolating the values between different grid-points is required. The Particle Tracker, therefore, makes use of a 3-Dimensional Linear Interpolator which takes into account the 6 nearest grid-points to the ion and linearly interpolates the X, Y and Z components of the magnetic and electric field. This provides the Particle Tracker with an effectively continuous magnetic and electric field with which to carry out the solution to the Lorentz force equation and therefore solve for the motion of the particles.

The interpolation algorithm is of the form:

$$\begin{aligned} x_d &= (x-x_0)/(x_1-x_0) \\ y_d &= (y-y_0)/(y_1-y_0) \\ z_d &= (z-z_0)/(z_1-z_0) \end{aligned} \quad (9)$$

$$\begin{aligned} c_{00} &= \mathbf{f}[x_0, y_0, z_0](1-x_d) + \mathbf{f}[x_1, y_0, z_0]x_d \\ c_{01} &= \mathbf{f}[x_0, y_0, z_1](1-x_d) + \mathbf{f}[x_1, y_0, z_1]x_d \end{aligned}$$

$$\begin{aligned} c_{10} &= \mathbf{f}[x_0, y_l, z_0](1-x_d) + \mathbf{f}[x_l, y_l, z_0]x_d \\ c_{11} &= \mathbf{f}[x_0, y_l, z_l](1-x_d) + \mathbf{f}[x_l, y_l, z_l]x_d \end{aligned} \quad (10)$$

$$\begin{aligned} c_0 &= c_{00}(1-y_d) + c_{10}y_d \\ c_1 &= c_{01}(1-y_d) + c_{11}y_d \\ c &= c_0(1-z_d) + c_1z_d \end{aligned} \quad (11)$$

Where  $x$ ,  $y$  and  $z$  indicate the position of interest between a set of grid-points and  $x_0$  and  $x_l$  indicate the grid-points immediately above and below  $x$ , respectively, with the same system used for  $y$  and  $z$ . Here  $\mathbf{f}$  indicates the value of interest, magnetic or electric field, at those points indicated. These values are taken from the simulational output. The final value of interest is calculated by  $c$ , giving the desired component of the electric or magnetic field integrated at the location of  $x$ ,  $y$  and  $z$ . That is, this method solves for each component of the electric or magnetic fields through these calculations.

The gradient of the magnetic and electric fields in this regime is small enough that a linear interpolation produces a good approximation. Furthermore, the scale over which the magnetic and electric fields exhibit changes is large enough that interpolating between vertices is acceptable (Appendix B.2 contains the implemented interpolation algorithm).

### **Path Sampling Method**

The Path-Sampling method uses the orbital parameters of the MESSENGER orbit in order to sample the simulation space. By specifying a path in the simulation space that follows the MESSENGER orbit the simulation output can be sampled at any arbitrary point along that orbit. This method is useful in confirming simulation accuracy by

comparing the results to MESSENGER observations. That is, the magnetic field strength as well as the components of the field and plasma species.

### Creating a Sampling Path

Using the orbital ephemeris of the MESSENGER craft, the positions that the craft moves through are converted into simulation coordinate space. Distance from Mercury is converted to an XYZ location in the coordinate space. The converted positions, now, can be used to sample the electric and magnetic fields, in conjunction with the 3-dimensional interpolator, and compared directly with MESSENGER magnetic fields over time.

$$C_m=[x,y,z] \quad (12)$$

$$C_b=C_m+C_p \quad (13)$$

Where  $C_m$  is the position of the MESSENGER craft in Mercury-Centered coordinates, and  $C_b$  is the position of MESSENGER in simulation box coordinates, with respect to to a corner of the box. Values are calculated in kilometers and converted to data-points when simulational output is sampled.

# CHAPTER 3

## SIMULATION AND POSTPROCESSING DEMONSTRATION AND PRELIMINARY RESULTS

This section goes over the preliminary simulation output for the two solar wind states of interest: the nominal and CME conditions. Furthermore, here we cover some of the application of the Particle Tracker, Path Sampling method and 3-D interpolator. These are only preliminary results showing some of the applications of these postprocessing tools and a case is made for their use in the future to better understand the Hermean magnetosphere.

### **Nominal Conditions**

#### Multifluid MHD Results

Figure 3 shows output from the multifluid MHD simulation code for the nominal solar wind conditions. The formation of a bow-shock and magnetopause is seen in the pressure profile in the figure. Additionally, two concentrations of pressure are seen to the north and the south of the planet. These pressure enhancements correspond to cusps which occur when particles of solar wind are 'funneled' towards of the surface of the planet by open magnetic fields.

The gross structures of the Hermean magnetosphere have formed, but an additional analysis is done using the path-sampling method. Using that method, the magnetic and electric fields output from the model are sampled in order to determine how

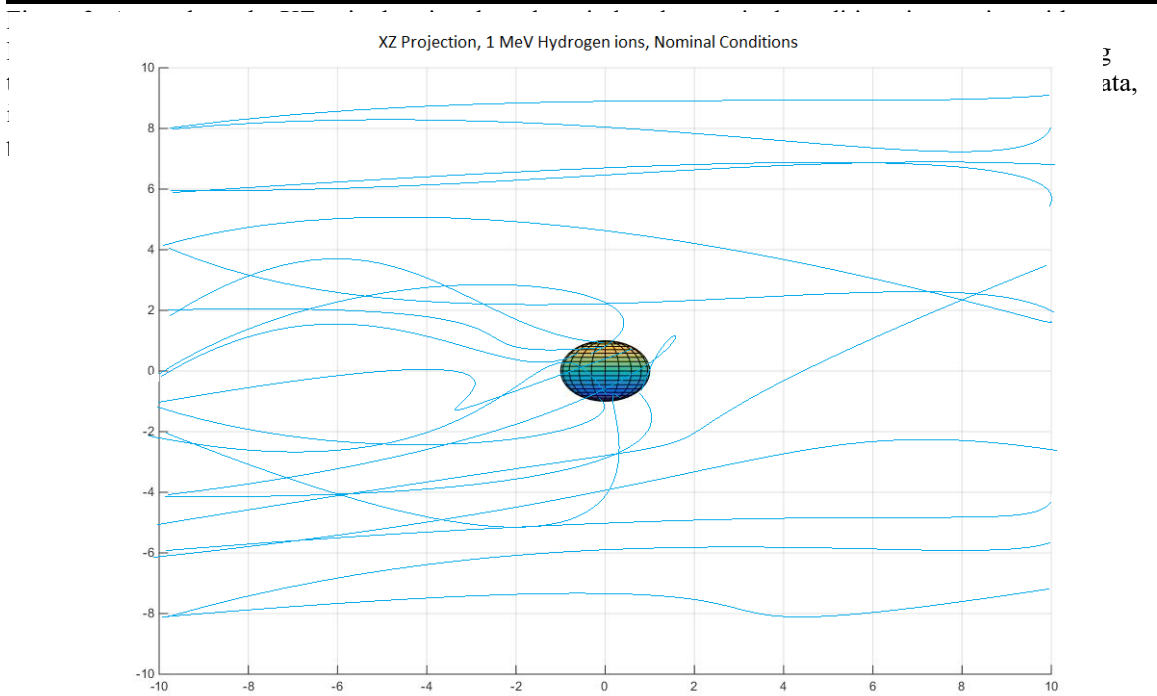
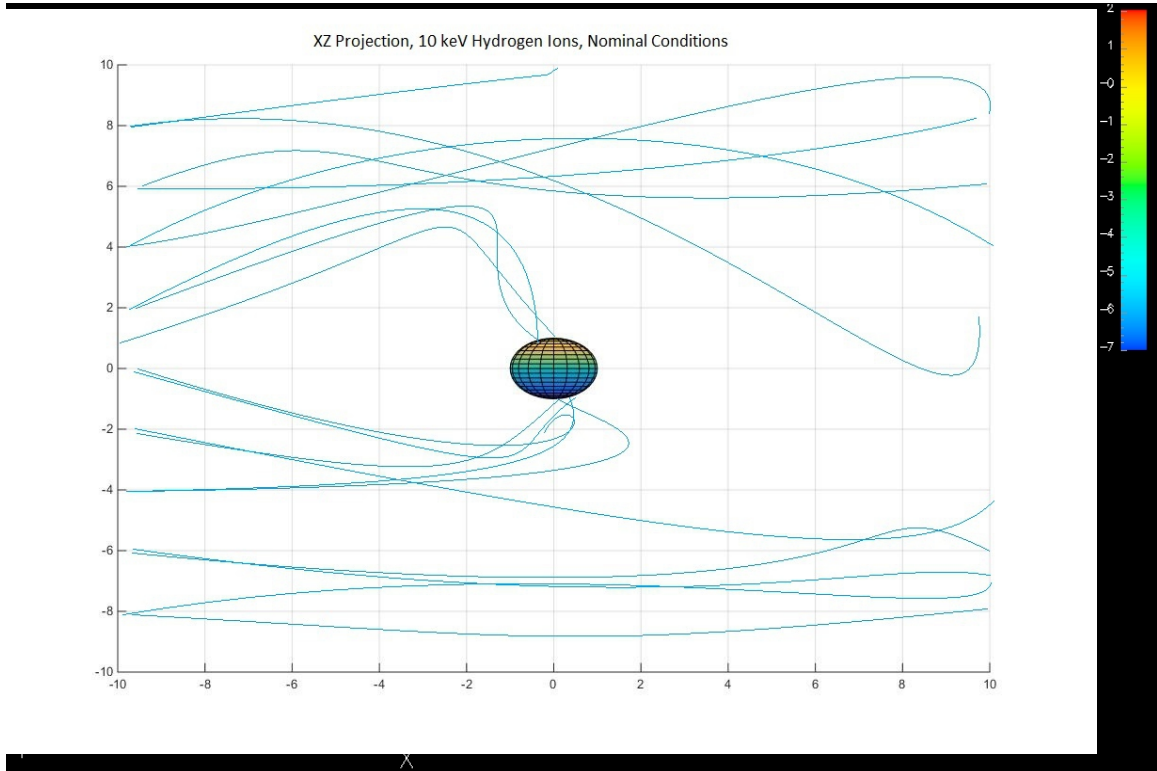


Figure 5: The two images show the particle tracker output projected onto the XZ axes for the 10 keV and 1 MeV particles during nominal solar conditions. The axes are in Mercury radii. Figure 4: The magnetic field sampled by MESSENGER over the course of one revolution compared to the data sampled along the same physical path from the multifluid MHD simulation under nominal solar conditions. The Distance of MESSENGER represents its distance from the center of the planet in Mercury Radii.

closely they predict the actual MESSENGER data. The comparison in figure 4 is made using data from the simulation shown above and comparing it to Orbit 638 of the MESSENGER craft. Visible are the magnetic fields 'outside' of the Hermean magnetosphere, the transition from the IMF to the Hermean magnetosphere and then the magnetic fields inside the magnetosphere. The 'transition' period is noisy and indicates the craft passing through the magnetopause from the IMF to the Hermean magnetosphere.

### Particle Tracking

Figure 5 shows two representations of particles tracked through the magnetic and electric fields produced by the multifluid MHD simulation under nominal conditions. The images project the particles path onto the XZ plane. The particle tracker can trace the paths of hundreds or thousands of particles, but here we show just ten for simplicity.

We found that the higher energy particles are less readily deflected by the Hermean magnetosphere. The lower energy particles tend to be deflected or move toward the cusps along the magnetic field lines open to the solar wind. Lower energy particles were not able to reach the surface which was not 'open' to the interplanetary magnetic field (IMF), however the higher energy particles were able to do so.

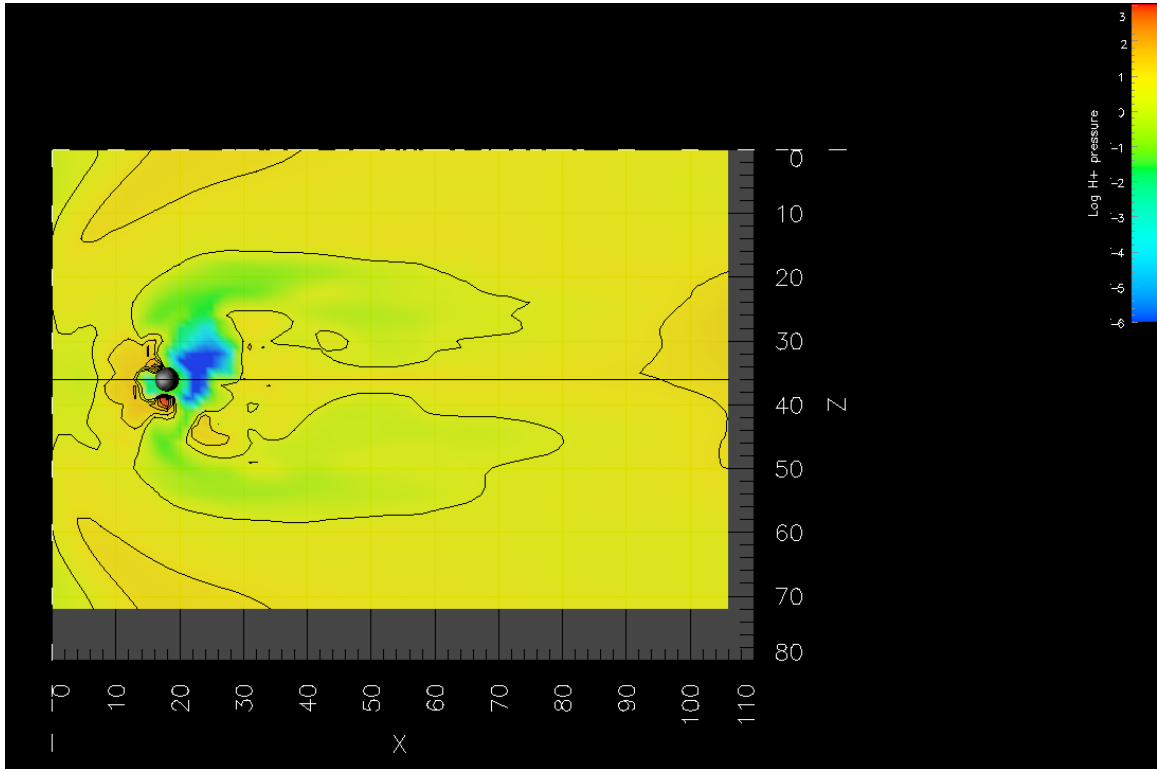


Figure 6: The results from the multifluid MHD simulation incorporating ICME conditions for the solar wind. The bow-shock and magnetopause have moved planetward due to the increased pressure, indicating a contraction of the Hermean magnetosphere.

## **CME Conditions**

### Multifluid MHD Results

The multifluid MHD simulations predicted a show a contracted bow-shock and magnetosphere, indicating that the increased solar wind ram pressure has pushed back the features of the Hermean magnetosphere towards the planet.

### Particle Tracking

The higher energy particles are still able to tunnel through the magnetosphere and are otherwise less strongly deflected. However, the lower energy particles achieve higher access to the surface of Mercury, due in part to the contraction of its magnetosphere.

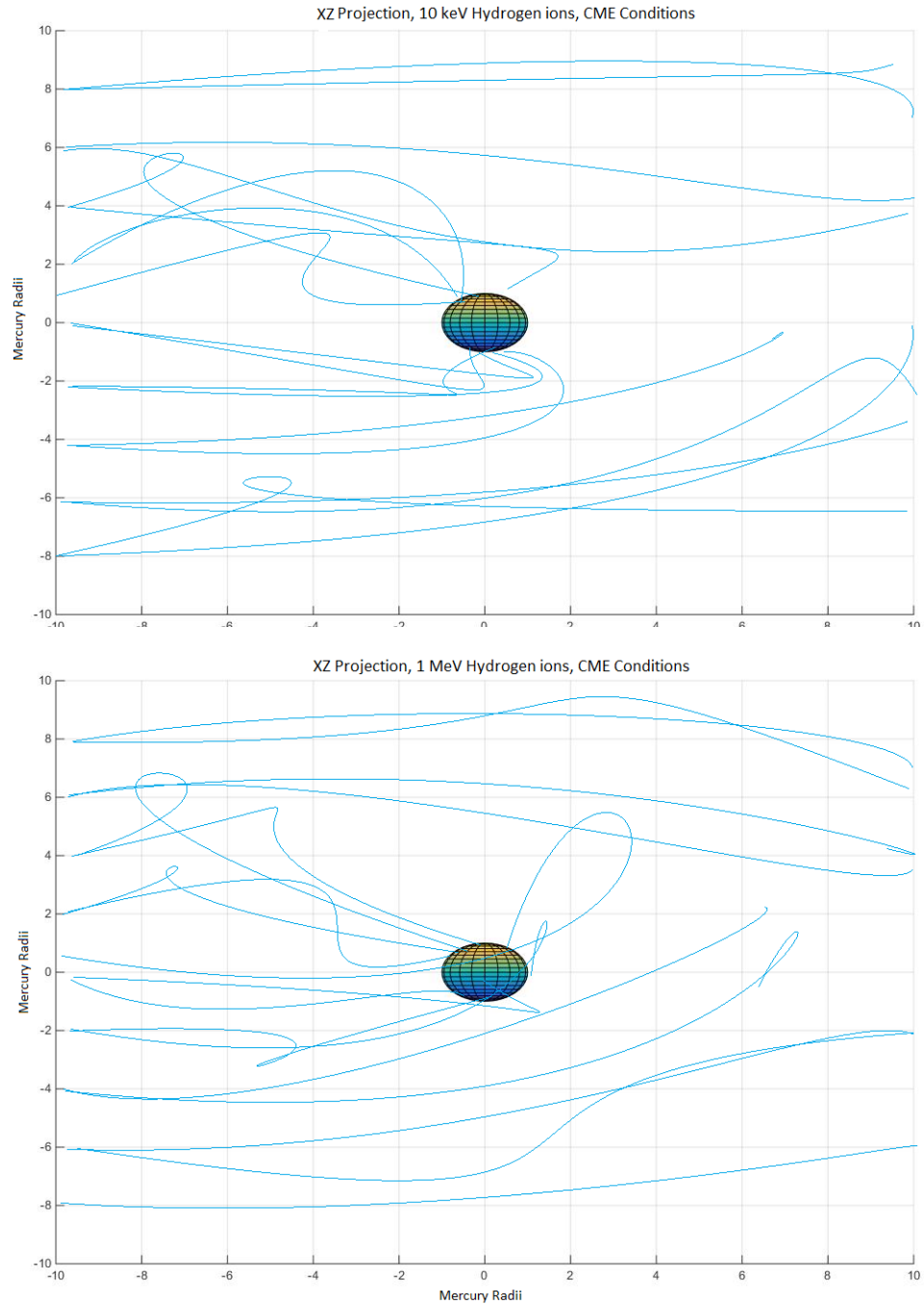


Figure 5: The two images show the particle tracker output projected onto the XZ axes for the 10 keV and 1 MeV particles during CME conditions. The axes are in Mercury radii.

Indicating that during CME events the surface of Mercury is more accessible to the solar wind plasma.

## CHAPTER 4

### DISCUSSION AND FUTURE WORK

#### **Discussion**

The work outlined in this dissertation developed simulations and coupled post-processing tools to better understand Mercury's complex interactions with the solar wind. These tools allow for a more specific study of energetic particles in the Hermean system. Utilizing these tools, the preliminary results indicate that the Hermean magnetosphere contracts under the higher solar wind ram pressure associated with a CME event. During these events the surface would experience increased weathering through ion sputtering. Evidence was also seen that indicates that higher energy particles under both nominal and CME conditions are capable of tunneling through the magnetosphere to the surface of Mercury. The high energy tail of the solar wind plasma distribution could therefore continuously be responsible for surface weathering even beneath the 'cusp' zones which we predict will see the greatest incidence of solar wind ions.

#### **Future Work**

Future work, building off of the development of these tools would include a quantitative analysis as well as integrated energy flux to the surface calculation. Currently, while the plasma environment has been simulated with expected values, based on observed field strengths of solar wind plasma conditions, a particle tracker assessment is warranted to gain an overall understanding of how these energetic particles would behave when interacting with the Hermean magnetosphere. So far, simulations have been limited to just a few energy ranges for demonstration purposes. While this enables us to

gain an overall understanding of the system, a more complete and quantitative assessment is warranted. Such an assessment would include:

- A more complete set of particle energies whose paths are simulated using the particle tracker.
- A comprehensive assessment of surface accessibility to solar wind ions, including a comparison of accessibility during nominal and CME conditions.
- Quantifying the proportion of solar wind ions which precipitate onto the Hermean surface during nominal and CME conditions.
- Relating the aforementioned points to surface weathering and exospheric generation rates.

## APPENDIX A

### INPUT FILE

#### A.1 Input File

The Input file is shown below, units are in normalized simulation units and specify the various quantities used by the model:

||Input File Start

&option

tmax=1200.01 ntgraf=15 stepsz=0.125 start=.t. tsave=400.0

/

&earth

xdip=0.00001 ydip=0.00000 zdip=0.000000 rearth=22.0

tilt1=00.0 tilt2=00.0 tilting =.f.

rmasq=1. rmassh=28. rmasso=40.

/

&speeds

cs\_inner=0.000129 alf\_inner1=0.0001 alf\_inner2=0.0001 alpha\_e=6.0

den\_earth=5.5 o\_conc=0.75

gravity=0.67 ti\_te=0.5

gamma=1.6666 ringo=.f. update=.f. reload=.f.

/

&windy

re\_wind=35. cs\_wind=0.10 vx\_wind1=0.45 vx\_wind2=0.45

vy\_wind1=0.000 vy\_wind2=0.000 vz\_wind1=-0.00 vz\_wind2=-0.00

alfx\_wind1=0.0000 alfx\_wind2=0.0000

alfy\_wind1=-0.000 alfy\_wind2=-0.000

alfz\_wind1=-0.043 alfz\_wind2=-0.043

```

den_wind1=0.005 den_wind2=0.005
reynolds=10.0 resist=40. rho_frac=0.01
bfrac=1.0 vfrac=1.0
/
&physical
  re_equiv=0.0455 b_equiv=65.75 v_equiv=1000. rho_equiv=2.
  spacecraft=.f. warp=.f. utstart=0.00
/
&smooth
  chirho=2.0 chipxyz=2.0 chierg=2.0
  difrho=0.002 difpxyz=0.002 diferg=0.005
/
-55. 55. -44. 44. -38. 38. 1. 2 0
-90. 130. -88 88. -76. 76. 2. 3 1
-120. 320. -176. 176. -152. 152. 4. 4 2
-160. 720. -352. 352. -304. 304. 8. 0 3
-280. 1480. -608. 608. -608. 608. 16. 0 4
                                ncore  nbdry
                                m = small  m=big
                                ncore = big  nbdry=small
|| Input File End

```

# APPENDIX B

## PARTICLE TRACKER AND INTERPOLATOR

### B.1 Particle Tracker Code

```
PROGRAM ParticleTracker1
IMPLICIT NONE

REAL (KIND=8), DIMENSION(4,3) :: aH,vH,pH

INTEGER, PARAMETER :: popSize=99, counts=320000 !Population size, iterations per particle per sim
INTEGER, PARAMETER :: nx=107, ny=85, nz=73 !Size of Cube (of E/B fields)
INTEGER, PARAMETER :: px=44, py=43, pz=37 !Location of moon
REAL (KIND=8), PARAMETER :: dt=.0005, r=11.5 !Length of timestep in seconds & radius in gridpoints
REAL (KIND=8), PARAMETER :: nano=10.0**(-9) !Nano conversion factor

REAL (KIND=8) :: bx,by,bz,ex,ey,ez !Value holders for magnetic and electric field components
REAL (KIND=8) :: gr=2640000, cx=9.73, cy=7.73, cz=6.64 !Physical conversion factors gridpoint to kilometers
REAL (KIND=8), DIMENSION(nx,ny,nz) :: efx,efy,efz,bfx,bfy,bfz !Holders of E/B data
INTEGER i,j,k,hold,counter !Indices for loop, simulation timestep, particle
LOGICAL, DIMENSION(popSize) :: con !True if the particle has not impacted a planet/exited simulation box, false otherwise
REAL (KIND=8), DIMENSION(popSize,2) :: qm !Holds the charge and mass of particles in the population
REAL (KIND=8), DIMENSION(popSize,counts,6) :: pop !Holds the particles and their positionsx3(m/s), velocitiesx3(m/s),
charge(C) and mass(kg)
!popSize indicates the particle, counts indicates which timestep in the simulation is accessed, and 8 holds the relevant information

!Imports the E/B field values
OPEN(101,FILE='EBfielddata_Case1_a',STATUS='old',ACTION='read',POSITION='rewind')
DO i=1,nx
  DO j=1,ny
    DO k=1,nz
      READ(101,*) efx(i,j,k),efy(i,j,k),efz(i,j,k),bfx(i,j,k),bfy(i,j,k),bfz(i,j,k)
      bfx(i,j,k)=bfx(i,j,k)*nano
      bfy(i,j,k)=bfy(i,j,k)*nano
      bfz(i,j,k)=bfz(i,j,k)*nano
    END DO
  END DO
END DO
CLOSE(101)

!E FIELDS
print *, efx(1,1,1)
print *, efx(10,10,10)
print *, efx(107,85,73)
!E FIELDS

!Reads in the population of particles, positions, velocities, charge, mass

OPEN(202,FILE='input.txt',status='old',action='read')
DO i=1,popSize
  READ(202,*) pop(i,1,1),pop(i,1,2),pop(i,1,3),pop(i,1,4),pop(i,1,5),pop(i,1,6),qm(i,1),qm(i,2)
  con(i)=.TRUE.
END DO
CLOSE(202)

OPEN(101,FILE='ParticleTracker1Out.txt',status='replace',form='formatted') !Opens an output file for results
!Begins loop iterating over each particle for duration of iterations
DO i=1,popSize
  print *, i
```

```

j=2
counter=80
DO WHILE ((con(i).EQV..TRUE.).AND.(j.LE.counts))
  vH(1,1)=pop(i,j-1,4)!VX
  pH(1,1)=pop(i,j-1,1)
  vH(1,2)=pop(i,j-1,5)!VY
  pH(1,2)=pop(i,j-1,2)
  vH(1,3)=pop(i,j-1,6)!VZ
  pH(1,3)=pop(i,j-1,3)
  CALL BFIELD(pop(i,1,1),pop(i,1,2),pop(i,1,3),nx,ny,nz,bfx,bfy,bfz,cx,cy,cz,gr,bx,by,bz)!Initial Position Magnetic Field
  CALL EFIELD(pop(i,1,1),pop(i,1,2),pop(i,1,3),nx,ny,nz,efx,efy,efz,cx,cy,cz,gr,ex,ey,ez)!Initial Position Electric Field
  CALL ACCELERATION(qm(i,1),qm(i,2),vH(1,1),vH(1,2),vH(1,3),bx,by,bz,ex,ey,ez,aH(1,1),aH(1,2),aH(1,3))!Initial
Acceleration

  vH(2,1)=vH(1,1)+dt/2.0*aH(1,1)!Intermediate vx-1
  pH(2,1)=pop(i,1,1)+dt/2.0*vH(1,1)+(dt**2.0)/8.0*aH(1,1)!Intermediate x-1
  vH(2,2)=vH(1,2)+dt/2.0*aH(1,2)!Intermediate vy-1
  pH(2,2)=pop(i,1,2)+dt/2.0*vH(1,2)+(dt**2.0)/8.0*aH(1,2)!Intermediate y-1
  vH(2,3)=vH(1,3)+dt/2.0*aH(1,3)!Intermediate vz-1
  pH(2,3)=pop(i,1,3)+dt/2.0*vH(1,3)+(dt**2.0)/8.0*aH(1,3)!Intermediate z-1

  CALL BFIELD(pH(2,1),pH(2,2),pH(2,3),nx,ny,nz,bfx,bfy,bfz,cx,cy,cz,gr,bx,by,bz)!Magnetic Field at Intermediate Point-1
  CALL EFIELD(pH(2,1),pH(2,2),pH(2,3),nx,ny,nz,efx,efy,efz,cx,cy,cz,gr,ex,ey,ez)!Electric Field at Intermediate Point-1
  CALL ACCELERATION(qm(i,1),qm(i,2),vH(2,1),vH(2,2),vH(2,3),bx,by,bz,ex,ey,ez,aH(2,1),aH(2,2),aH(2,3))!Acceleration at
IP-1

  vH(3,1)=vH(1,1)+dt/2.0*aH(2,1)!Intermediate vx-2
  pH(3,1)=pop(i,1,1)+dt/2.0*vH(1,1)+(dt**2.0)/8.0*aH(1,1)!Intermediate x-2
  vH(3,2)=vH(1,2)+dt/2.0*aH(2,2)!Intermediate vy-2
  pH(3,2)=pop(i,1,2)+dt/2.0*vH(1,2)+(dt**2.0)/8.0*aH(1,2)!Intermediate y-2
  vH(3,3)=vH(1,3)+dt/2.0*aH(2,3)!Intermediate vz-2
  pH(3,3)=pop(i,1,3)+dt/2.0*vH(1,3)+(dt**2.0)/8.0*aH(1,3)!Intermediate z-2

  CALL BFIELD(pH(3,1),pH(3,2),pH(3,3),nx,ny,nz,bfx,bfy,bfz,cx,cy,cz,gr,bx,by,bz)!Magnetic Field at IP-2
  CALL EFIELD(pH(3,1),pH(3,2),pH(3,3),nx,ny,nz,efx,efy,efz,cx,cy,cz,gr,ex,ey,ez)!Electric Field at IP-2
  CALL ACCELERATION(qm(i,1),qm(i,2),vH(3,1),vH(3,2),vH(3,3),bx,by,bz,ex,ey,ez,aH(3,1),aH(3,2),aH(3,3))!Acceleration at
IP-2

  vH(4,1)=vH(1,1)+dt*aH(3,1)
  pH(4,1)=pop(i,1,1)+dt*vH(1,1)+(dt**2.0)/2.0*aH(3,1)
  vH(4,2)=vH(1,2)+dt*aH(3,2)
  pH(4,2)=pop(i,1,2)+dt*vH(1,2)+(dt**2.0)/2.0*aH(3,2)
  vH(4,3)=vH(1,3)+dt*aH(3,3)
  pH(4,3)=pop(i,1,3)+dt*vH(1,3)+(dt**2.0)/2.0*aH(3,3)

  CALL BFIELD(pH(4,1),pH(4,2),pH(4,3),nx,ny,nz,bfx,bfy,bfz,cx,cy,cz,gr,bx,by,bz)!Magnetic Field at IP-3
  CALL EFIELD(pH(4,1),pH(4,2),pH(4,3),nx,ny,nz,efx,efy,efz,cx,cy,cz,gr,ex,ey,ez)!Electric Field at IP-3
  CALL ACCELERATION(qm(i,1),qm(i,2),vH(4,1),vH(4,2),vH(4,3),bx,by,bz,ex,ey,ez,aH(4,1),aH(4,2),aH(4,3))!Acceleration at
IP-3

  pop(i,2,1)=pop(i,1,1)+dt*vH(1,1)+(dt**2.0)/6.0*(aH(1,1)+aH(2,1)+aH(3,1))!New Pos x
  pop(i,2,2)=pop(i,1,2)+dt*vH(1,2)+(dt**2.0)/6.0*(aH(1,2)+aH(2,2)+aH(3,2))!New Pos y
  pop(i,2,3)=pop(i,1,3)+dt*vH(1,3)+(dt**2.0)/6.0*(aH(1,3)+aH(2,3)+aH(3,3))!New Pos z

  pop(i,2,4)=vH(1,1)+(dt/6.0)*(aH(1,1)+2*aH(2,1)+2*aH(3,1)+aH(4,1))!New VX
  pop(i,2,5)=vH(1,2)+(dt/6.0)*(aH(1,2)+2*aH(2,2)+2*aH(3,2)+aH(4,2))!New VY
  pop(i,2,6)=vH(1,3)+(dt/6.0)*(aH(1,3)+2*aH(2,3)+2*aH(3,3)+aH(4,3))!New VZ

  pop(i,1,1)=pop(i,2,1)
  pop(i,1,2)=pop(i,2,2)
  pop(i,1,3)=pop(i,2,3)
  pop(i,1,4)=pop(i,2,4)
  pop(i,1,5)=pop(i,2,5)
  pop(i,1,6)=pop(i,2,6)

  IF (counter.EQ.80) THEN
    counter=1
    WRITE(101,200) pop(i,2,1),pop(i,2,2),pop(i,2,3),pop(i,2,4),pop(i,2,5),pop(i,2,6)

```

```

ELSE
  counter=counter+1
END IF

CALL WITHIN_DIMENSIONS(pop(i,2,1),pop(i,2,2),pop(i,2,3),nx,ny,nz,cx,cy,cz,px,py,pz,gr,r,con(i))
IF (con(i).EQV..FALSE.) THEN
  hold=j
  !print *, pop(i,hold,1) !Debugging Prompts
  !print *, pop(i,hold,2)
  !print *, pop(i,hold,3)
END IF
IF (con(i).EQV..FALSE.) THEN

  DO k=j+1,counts
    IF (counter.EQ.80) THEN
      counter=1
      WRITE(101,200) pop(i,2,1),pop(i,2,2),pop(i,2,3),pop(i,2,4),pop(i,2,5),pop(i,2,6)
    ELSE
      counter=counter+1
    END IF
  END DO

  END IF
  j=j+1
END DO
END DO

200 FORMAT(6f24.8) !Formats data output
CLOSE(101)

END PROGRAM ParticleTracker1

SUBROUTINE WITHIN_DIMENSIONS(x,y,z,nx,ny,nz,rx,ry,rz,px,py,pz,rr,r,within)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: x,y,z,rx,ry,rz,rr,r
INTEGER, INTENT(IN) :: nx,ny,nz,px,py,pz
REAL (KIND=8) :: holdx, holdy, holdz,hr
LOGICAL, INTENT(OUT) :: within

holdx=x/((rx*rr)/DBLE(nx))+2.0
holdy=y/((ry*rr)/DBLE(ny))+2.0
holdz=z/((rz*rr)/DBLE(nz))+2.0

IF ((r**2).GE.((holdx-DBLE(px))**2+(holdy-DBLE(py))**2+(holdz-DBLE(pz))**2)) THEN
  within=.FALSE.
END IF

IF (holdx.GT.(DBLE(nx)+DBLE(.007289623))) THEN
  within=.FALSE.
END IF
IF (holdx.LT.(DBLE(2.0)+DBLE(.007289623))) THEN
  within=.FALSE.
END IF

IF (holdy.GT.(DBLE(ny)+DBLE(.007289094))) THEN
  within=.FALSE.
END IF

IF (holdy.LT.(DBLE(2.0)+DBLE(.007289094))) THEN
  within=.FALSE.
END IF

IF (holdz.GT.(DBLE(nz)+DBLE(.007287673))) THEN
  within=.FALSE.
END IF
IF (holdz.LT.(DBLE(2.0)+DBLE(.007287673))) THEN
  within=.FALSE.
END IF

```

END SUBROUTINE WITHIN\_DIMENSIONS

```
SUBROUTINE ACCELERATION(q,m,vx,vy,vz,bx,by,bz,ex,ey,ez,ax,ay,az)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: q,m,vx,vy,vz,bx,by,bz,ex,ey,ez
REAL (KIND=8), INTENT(OUT) :: ax,ay,az
ax=(q/m)*(ex+vy*bz-vz*by)
ay=(q/m)*(ey+vz*bx-vx*bz)
az=(q/m)*(ez+vx*by-vy*bx)
END SUBROUTINE ACCELERATION
```

```
SUBROUTINE VELOCITY(vx,vy,vz,ax,ay,az,dt,vxt,vyt,vzt)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: vx,vy,vz,ax,ay,az,dt
REAL (KIND=8), INTENT(OUT) :: vxt,vyt,vzt
vxt=vx+ax*dt
vyt=vy+ay*dt
vzt=vz+az*dt
END SUBROUTINE VELOCITY
```

```
SUBROUTINE POSITION(x,y,z,vx,vy,vz,dt,xt,yt,zt)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: x,y,z,vx,vy,vz,dt
REAL (KIND=8), INTENT(OUT) :: xt,yt,zt
xt=x+vx*dt
yt=y+vy*dt
zt=z+vz*dt
END SUBROUTINE POSITION
```

```
SUBROUTINE EFIELD(x,y,z,nx,ny,nz,efx,efy,efz,rx,ry,rz,rr,ex,ey,ez)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) ::x,y,z,rx,ry,rz,rr
REAL (KIND=8), DIMENSION(nx,ny,nz), INTENT(IN) :: efx,efy,efz
INTEGER, INTENT(IN) :: nx,ny,nz
REAL (KIND=8), INTENT(OUT) ::ex,ey,ez
INTEGER, DIMENSION(8,3) :: xyz
REAL (KIND=8), DIMENSION(8) :: v
REAL (KIND=8) :: holdx, holdy, holdz
holdx=x/((rx*rr)/DBLE(nx))+1.0
holdy=y/((ry*rr)/DBLE(ny))+1.0
holdz=z/((rz*rr)/DBLE(nz))+1.0
!Calculates the nearest neighbor vertices of the cube and calculates the x component of the electric field
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,efx,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,ex)
!Calculates the Y component of the electric field
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,efy,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,ey)
!Calculates the Z component of the electric field
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,efz,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,ez)
!print *, 'Electric Field Check'
ex=0
ey=0
ez=0
END SUBROUTINE EFIELD
```

```
SUBROUTINE BFIELD(x,y,z,nx,ny,nz,bfx,bfy,bfz,rx,ry,rz,rr,bx,by,bz)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) ::x,y,z,rx,ry,rz,rr
REAL (KIND=8), DIMENSION(nx,ny,nz), INTENT(IN) :: bfx,bfy,bfz
INTEGER, INTENT(IN) :: nx,ny,nz
REAL (KIND=8), INTENT(OUT) ::bx,by,bz
INTEGER, DIMENSION(8,3) :: xyz
REAL (KIND=8), DIMENSION(8) :: v
REAL (KIND=8) :: holdx,holdy,holdz
holdx=x/((rx*rr)/DBLE(nx))+1.0
holdy=y/((ry*rr)/DBLE(ny))+1.0
holdz=z/((rz*rr)/DBLE(nz))+1.0
!Calculates
```

```

the nearest neighbor vertices of the cube and calculates the x component of the magnetic field
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,bfx,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,bx)
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,bfy,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,by)
CALL NEAREST_NEIGHBOR(holdx,holdy,holdz,bfz,nx,ny,nz,xyz,v)
CALL TRILINEAR_INTERPOLATOR(DBLE(xyz),v,holdx,holdy,holdz,bz)
!print *, 'Magnetic Field Check'
bx=0
by=0
bz=0
END SUBROUTINE BFIELD

```

```

SUBROUTINE NEAREST_NEIGHBOR(x,y,z,vals,nx,ny,nz,vertices,values)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: x,y,z
INTEGER , INTENT(IN) :: nx,ny,nz
REAL (KIND=8), DIMENSION(nx,ny,nz), INTENT(IN) :: vals
INTEGER, DIMENSION(8,3), INTENT(OUT) :: vertices
REAL (KIND=8), DIMENSION(8), INTENT(OUT) :: values
vertices(1,1)=INT(FLOOR(x))
vertices(1,2)=INT(FLOOR(y))
vertices(1,3)=INT(FLOOR(z))
values(1)=vals(vertices(1,1),vertices(1,2),vertices(1,3))
vertices(2,1)=vertices(1,1)
vertices(2,2)=vertices(1,2)+1
vertices(2,3)=vertices(1,3)
values(2)=vals(vertices(2,1),vertices(2,2),vertices(2,3))
vertices(3,1)=vertices(1,1)+1
vertices(3,2)=vertices(1,2)
vertices(3,3)=vertices(1,3)
values(3)=vals(vertices(3,1),vertices(3,2),vertices(3,3))
vertices(4,1)=vertices(1,1)+1
vertices(4,2)=vertices(1,2)+1
vertices(4,3)=vertices(1,3)
values(4)=vals(vertices(4,1),vertices(4,2),vertices(4,3))
vertices(5,1)=vertices(1,1)
vertices(5,2)=vertices(1,2)
vertices(5,3)=vertices(1,3)+1
values(5)=vals(vertices(5,1),vertices(5,2),vertices(5,3))
vertices(6,1)=vertices(1,1)
vertices(6,2)=vertices(1,2)+1
vertices(6,3)=vertices(1,3)+1
values(6)=vals(vertices(6,1),vertices(6,2),vertices(6,3))
vertices(7,1)=vertices(1,1)+1
vertices(7,2)=vertices(1,2)
vertices(7,3)=vertices(1,3)+1
values(7)=vals(vertices(7,1),vertices(7,2),vertices(7,3))
vertices(8,1)=vertices(1,1)+1
vertices(8,2)=vertices(1,2)+1
vertices(8,3)=vertices(1,3)+1
values(8)=vals(vertices(8,1),vertices(8,2),vertices(8,3))
END SUBROUTINE NEAREST_NEIGHBOR

```

```

SUBROUTINE TRILINEAR_INTERPOLATOR(xyz,v,xr,yr,zr,vi)
IMPLICIT NONE
REAL (KIND=8), DIMENSION(8,3), INTENT(IN) :: xyz
REAL (KIND=8), DIMENSION(8), INTENT(IN) :: v
REAL (KIND=8), INTENT(IN) :: xr, yr, zr
REAL (KIND=8), INTENT(OUT) :: vi
REAL (KIND=8) :: xd,yd,zd,i1,i2,j1,j2,w1,w2
xd=(xr-xyz(1,1))/(xyz(8,1)-xyz(1,1))
yd=(yr-xyz(1,2))/(xyz(8,2)-xyz(1,2))
zd=(zr-xyz(1,3))/(xyz(8,3)-xyz(1,3))

i1=v(1)*(1-zd)+v(5)*zd
i2=v(2)*(1-zd)+v(6)*zd
j1=v(3)*(1-zd)+v(7)*zd
j2=v(4)*(1-zd)+v(8)*zd

```

```

w1=i1*(1-yd)+i2*yd
w2=j1*(1-yd)+j2*yd

vi=w1*(1-xd)+w2*xd
END SUBROUTINE TRILINEAR_INTERPOLATOR

```

The particle tracker takes in a set of particle positions. These particle positions include locations in space, X, Y and Z, as well as velocity values. This population is generated beforehand and represents the high-en particles of interest. Using these initial positions and velocities, the particle tracker then propagates these particles forward in time using a fourth-order Runge-Kutta to solve for the acceleration, velocity and position values of each particle. At each intermediate point used to calculate the final position and velocity of a particle during the course of one time step, the particle tracker solves the Lorentz Force equation.

Since the particles do not move solely between grid-points where the multifluid MHD simulation has output magnetic and electric field values, the particle tracker makes use of a 3-D interpolator to calculate the components of those fields between the grid-points.

## B.2 3D Linear Interpolator

```

SUBROUTINE NEAREST_NEIGHBOR(x,y,z,vals,nx,ny,nz,vertices,values)
IMPLICIT NONE
REAL (KIND=8), INTENT(IN) :: x,y,z
INTEGER , INTENT(IN) :: nx,ny,nz
REAL (KIND=8), DIMENSION(nx,ny,nz), INTENT(IN) :: vals
INTEGER, DIMENSION(8,3), INTENT(OUT) :: vertices
REAL (KIND=8), DIMENSION(8), INTENT(OUT) :: values
vertices(1,1)=INT(FLOOR(x))
vertices(1,2)=INT(FLOOR(y))
vertices(1,3)=INT(FLOOR(z))
values(1)=vals(vertices(1,1),vertices(1,2),vertices(1,3))
vertices(2,1)=vertices(1,1)
vertices(2,2)=vertices(1,2)+1
vertices(2,3)=vertices(1,3)
values(2)=vals(vertices(2,1),vertices(2,2),vertices(2,3))
vertices(3,1)=vertices(1,1)+1
vertices(3,2)=vertices(1,2)
vertices(3,3)=vertices(1,3)
values(3)=vals(vertices(3,1),vertices(3,2),vertices(3,3))
vertices(4,1)=vertices(1,1)+1
vertices(4,2)=vertices(1,2)+1
vertices(4,3)=vertices(1,3)
values(4)=vals(vertices(4,1),vertices(4,2),vertices(4,3))
vertices(5,1)=vertices(1,1)
vertices(5,2)=vertices(1,2)
vertices(5,3)=vertices(1,3)+1
values(5)=vals(vertices(5,1),vertices(5,2),vertices(5,3))
vertices(6,1)=vertices(1,1)
vertices(6,2)=vertices(1,2)+1
vertices(6,3)=vertices(1,3)+1
values(6)=vals(vertices(6,1),vertices(6,2),vertices(6,3))
vertices(7,1)=vertices(1,1)+1
vertices(7,2)=vertices(1,2)
vertices(7,3)=vertices(1,3)+1
values(7)=vals(vertices(7,1),vertices(7,2),vertices(7,3))
vertices(8,1)=vertices(1,1)+1
vertices(8,2)=vertices(1,2)+1
vertices(8,3)=vertices(1,3)+1

```

```

values(8)=vals(vertices(8,1),vertices(8,2),vertices(8,3))
END SUBROUTINE NEAREST_NEIGHBOR

SUBROUTINE TRILINEAR_INTERPOLATOR(xyz,v,xr,yr,zr,vi)
IMPLICIT NONE
REAL (KIND=8), DIMENSION(8,3), INTENT(IN) :: xyz
REAL (KIND=8), DIMENSION(8), INTENT(IN) ::v
REAL (KIND=8), INTENT(IN) :: xr, yr, zr
REAL (KIND=8), INTENT(OUT) :: vi
REAL (KIND=8) :: xd,yd,zd,i1,i2,j1,j2,w1,w2
xd=(xr-xyz(1,1))/(xyz(8,1)-xyz(1,1))
yd=(yr-xyz(1,2))/(xyz(8,2)-xyz(1,2))
zd=(zr-xyz(1,3))/(xyz(8,3)-xyz(1,3))

i1=v(1)*(1-zd)+v(5)*zd
i2=v(2)*(1-zd)+v(6)*zd
j1=v(3)*(1-zd)+v(7)*zd
j2=v(4)*(1-zd)+v(8)*zd

w1=i1*(1-yd)+i2*yd
w2=j1*(1-yd)+j2*yd

vi=w1*(1-xd)+w2*xd
END SUBROUTINE TRILINEAR_INTERPOLATOR

```

A subset of the particle tracker code contains the code for the 3D (trilinear) linear interpolator. First the code uses a subroutine “NearestNeighbor” to find the closest grid-points to the position of interest. These eight closest grid-points are then sampled from the data cube and the trilinear interpolator calculates the components of the magnetic and electric fields.

## REFERENCES

- Anderson, B. J., et al. (2008). "The structure of Mercury's magnetic field from MESSENGER's first flyby." *Science* 321(5885): 82-85.
- Benna, M., et al. (2010). "Modeling of the magnetosphere of Mercury at the time of the first MESSENGER flyby." *Icarus* **209**(1): 3-10.
- Delcourt, D. C., T. E. Moore, S. Orsini, A. Millilo, and J.-A. Sauvaud (2002). "Centrifugal acceleration of ions near Mercury." *GEOPHYSICAL RESEARCH LETTERS* 29.
- Delcourt, D. C., S. Grimald, F. Leblanc, J.-J. Bertherlier, A. Millilo, and A. Mura, A quantitative model of planetary Na<sup>+</sup> contribution to Mercury's magnetosphere, *Ann. Geophys.*, 21, 1723-1736, 2003.
- DiBraccio, G. A., J. A. Slavin, S. A. Boardsen, B. J. Anderson, H. Korth, T. H. Zurbuchen, J. M. Raines, D. N. Baker, R. L. McNutt Jr., and S. C. Solomon (2013), MESSENGER observations of magnetopause structure and dynamics at Mercury, *J. Geophys. Res. Space Physics*, doi:10.1002/jgra50123, in press.
- Johnson, C. L., et al. (2012). "MESSENGER observations of Mercury's magnetic field structure." *Journal of Geophysical Research: Planets* **117**(E12).
- Kabin, K. (2000). "Interaction of Mercury with the Solar Wind." *Icarus* **143**(2): 397-406.
- Kidder, A., et al. (2008). "Erosion of the dayside magnetosphere at Mercury in association with ion outflows and flux rope generation." *Journal of Geophysical Research* 113(A9).
- Leblanc, F., D. Delcourt, and R. E. Johnson, Mercury's sodium exosphere: Magnetospheric ion recycling, *J. Geophys. Res.*, 108(E12), 5136, doi:10.1029/2003JE002151, 2003.
- McClintock, W. E., et al. (2009). "MESSENGER observations of Mercury's exosphere: detection of magnesium and distribution of constituents." *Science* 324(5927): 610-613.

- Trávníček, P. M., Schriver, D., Hellinger P., Hercik, D., Anderson, B. J., Sarantos, M., Slavin, J. A. (2010). "Mercury's magnetosphere–solar wind interaction for northward and southward interplanetary magnetic field: Hybrid simulation results." *Icarus* 209.
- Trávníček, P., et al. (2007). "Structure of Mercury's magnetosphere for different pressure of the solar wind: Three dimensional hybrid simulations." *Geophysical Research Letters* **34**(5).
- Trávníček, P. M., et al. (2009). "Kinetic instabilities in Mercury's magnetosphere: Three-dimensional simulation results." *Geophysical Research Letters* **36**(7).
- Winslow, R. M., Anderson, B.J., Johnson, C.L., Slavin, J.A., Korth, H., Purucker, M.E.m Baker, D.N., Solomon, S.C. (2013), Mercury's magnetopause and bow shock from MESSENGER magnetometer observations, *Journal of Geophysical Research.*, doi:10.1002/jgra.50237.
- Johnson, C. L., et al. (2012). "MESSENGER observations of Mercury's magnetic field structure." *Journal of Geophysical Research: Planets* **117**(E12).