Computer Simulation of the VASIMR Engine

Final Report

NASA Faculty Fellowship Program – 2004

Johnson Space Center

Prepared by:	David Garrison
Academic Rank:	Assistant Professor
University & Department	University of Houston-Clear Lake
	Physics Department
	Houston, TX 77058
NASA/JSC	
Directorate:	Space and Life Sciences
Division:	Advanced Space Propulsion Laboratory
Branch:	N/A
JSC Colleague:	John Shebalin
Date Submitted:	August 13, 2004
Contract Number:	NAG 9-1526

ABSTRACT

The goal of this project is to develop a magneto-hydrodynamic (MHD) computer code for simulation of the VASIMR engine. This code is designed be easy to modify and use. We achieve this using the Cactus framework, a system originally developed for research in numerical relativity. Since its release, Cactus has become an extremely powerful and flexible open source framework. The development of the code will be done in stages, starting with a basic fluid dynamic simulation and working towards a more complex MHD code. Once developed, this code can be used by students and researchers in order to further test and improve the VASIMR engine.

INTRODUCTION

Variable Specific Impulse Magneto-plasma Rocket

The Variable Specific Impulse Magneto-plasma Rocket (VASIMR) is a project at the Advanced Space Propulsion Laboratory (ASPL) at JSC [2]. The project is led from NASA JSC, and has contracts with several government research centers, industrial companies and universities. In addition, researchers from universities and institutes all around the world collaborate with ASPL.

The Magneto-plasma rocket engine provides propulsion by ionizing and heating neutral gases to high temperatures and then guiding them out of a magnetic nozzle in order to produce thrust, much like a chemical rocket engine. However, the essential difference between VASIMR and a chemical rocket engine is that VASIMR will produce very high specific impulse at relatively low thrust (*i.e.*, a low density, high velocity exhaust), while a chemical rocket engine produces high thrust at relatively low specific impulse (*i.e.*, a high density, low velocity exhaust).

The particular niche filled by VASIMR in the electric propulsion community is that of a relatively high-power plasma propulsion system that is focused on human space flight, rather than on less massive unmanned, robotic space flight missions. The efficiency of the engine permits a favorable ratio of payload mass to spacecraft mass, one that allows long-duration space exploration missions to be realistically contemplated.

In its research configuration, VASIMR utilizes four co-axial magnetic coils and two co-axial antennas to achieve its purpose. The first antenna is a so-called helicon antenna, which serves as a plasma generator in that it ionizes an injected neutral gas (typically hydrogen, deuterium or helium). The second antenna is known as the ion cyclotron resonance heating (ICRH) antenna and it boosts the energy of the plasma by feeding electromagnetic energy preferentially into the ions.

While the helicon antenna is primarily responsible for creating the plasma, the second antenna is used to increase the ion energy and exhaust velocity, and thus the specific impulse of the rocket engine. The magnetic coils work in concert to shape the strong axial magnetic field that guides the strongly magnetized plasma (*i.e.*, magneto-plasma). The final magnetic coil (or a smaller auxiliary coil) serves as a magnetic nozzle, by which the specific impulse and thrust of the plasma exhaust may be varied. When the components are operating together, the result is the Variable Specific Impulse Magneto-plasma Rocket, or VASIMR.

The magnetic nozzle gives VASIMR the unique ability to modulate the plasma exhaust so as to maintain maximum power and efficiency. This technique is termed "Constant Power Throttling" and is similar to adjusting the transmission on an automobile. The VASIMR engine (specific impulse, $I_{sp} \sim 15{,}000$ sec), is designed to run continuously, so that, although it has low thrust, any interplanetary transit time is considerably reduced. In contrast, a chemical rocket, such as the space shuttle main engine ($I_{sp} \sim 450$ sec), is designed to provide very high thrust, but only for about eight minutes. A traditional chemical rocket lifts a space ship off of a planet and gives it an initial velocity, after which it is in free flight towards its objective.

The role of VASIMR is to provide thrust during what would have been unpowered free flight, thereby shortening travel time. For example, using only a chemical rocket would give a transit time of about 300 days to reach Mars. Adding VASIMR for the interplanetary section of the journey (equipped with a nuclear power generation system) would reduce the trip to as little as 39 days carrying 20 tons of cargo, or 115 days for a larger 61-ton cargo load. Also, by minimizing transit time, physical stress and risk to the crew is also minimized.

Our goal in this project is to create a computerized model of the VASIMR system in order to understand the fluid dynamics and thermodynamics of plasma flow in the engine and in its exhaust [4,5]. This model will incorporate variations of such system parameters as magnetic coil current values and magnetic field structure. We found Cactus to be the best framework for developing these models.

Cactus

Cactus [1] is an open source problem-solving environment designed for scientists and engineers. The Cactus framework, which was originally developed for numerical relativity research, has become an extremely powerful and flexible tool. Cactus originated in the academic research community, where it was developed and used over many years by a large international collaboration of physicists and computational scientists. Its modular structure easily enables parallel computation across different architectures and collaborative code development between different groups.

The name Cactus comes from the design of a central core (or "flesh") that connects to application modules (or "thorns") through an extensible interface[1]. Thorns can implement custom developed scientific or engineering applications, such as computational fluid dynamics. Other thorns from a standard computational toolkit provide a range of computational capabilities, such as parallel I/O, data distribution, or checkpointing.

Cactus runs on many architectures. Virtually all Unix based systems as well as Windows NT are supported. Applications, developed on standard workstations or laptops, can be seamlessly run on clusters or supercomputers. Cactus provides easy access to many cutting edge software technologies being developed in the academic research community, including the Globus Metacomputing Toolkit, HDF5 parallel file I/O, the PETSc scientific library, adaptive mesh refinement, web interfaces, and advanced visualization tools.

[1] See Appendix A

We chose to use Cactus because it is flexible, modular and well documented. The Cactus development groups are quick to respond to questions and communication within the development community is freely available. Also, efforts by the Cactus organization as well as third party developers ensure that new features and bug fixes are constantly being developed [3].

GOALS

The goal of this Faculty Fellowship Program (FFP) project was to test the feasibility of using the Cactus framework to develop a magneto-hydrodynamic (MHD) code for use with the VASIMR project. There are many differences between the existing Cactus codes used in numerical relativity and the MHD codes used within the VASIMR project. These differences had to be addressed in order to develop VASIMR simulations within the Cactus framework.

An alternative to using Cactus would be to either develop a new MHD code from scratch or to modify existing codes. However, the main motivation for switching to the Cactus framework is to gain the support of existing documentation and a large development community. Unlike existing software, a program developed with Cactus will be relatively easy for short-term workers (such as students) to modify and use because of the well-designed structure of Cactus and its extensive support network and documentation.

The Physics Program at the University of Houston – Clear Lake (UHCL) focuses on a Masters degree in Physics. Our graduate students are required to complete a research project or thesis but typically only have about a year to work on such a project. Existing codes usually require several years to learn enough about the software to modify and use on original research projects and are therefore not useful for short-term student projects. This research program will provide a suitable vehicle for student theses because original work can be completed in just a few months. This will also provide a framework for the controlled evolution of software suitable for ASPL.

Development will be done in stages, starting with a basic fluid dynamic simulation and working towards a more complex MHD code. The fluid code is designed primarily to test the feasibility of installing and running Cactus on ASPL and UHCL machines. The fluid code will eventually evolve into a full MHD code but before that can happen several technological steps must be taken. These steps are outlined in the section titled "Development Thorn". Eventually, this code will then be used by students and researchers to further design and improve the VASIMR engine.

INSTALLATION

The first step of this project involved installing Cactus on each of the development machines and testing them using several existing sample thorns. The three development machines were a dual processor Macintosh G4 machine at UHCL, a Linux Beowulf cluster at ASPL and my personal Macintosh Powerbook G4. Each computer was already equipped with both Fortran and C compilers but I also added additional visualization tools (xgraph, ygraph and gnuplot) to the Macintosh machines.

The biggest challenge during the installation process was finding the correct configuration for Cactus for each different hardware/software setup. The only way to find the correct configuration for each operating system, compiler and software package was to review the documentation and search through the computer's directory structure for the right parameters. This involved some trial and error and in a few cases, we had to correct a few Unix login files. After a couple weeks of searching for the right configurations, all three machines were compiling and running the example codes well.

The test examples ranged for a simple "Hello World" screen printout to a scalar wave simulation that used Cactus' ability to steer computer simulations through a web browser. These tests proved that all the compilers and tools were working correctly so we could move on to the next step, developing an original thorn.

THE COMFLUID THORN

Instead of jumping right into the development of a full MHD thorn, we thought it would be a good idea to first develop a compressible fluid simulation code which has a similar geometry to the VASIMR engine. This involves using a cylindrical coordinate grid and a set of coupled differential equations representing the number density and velocity of particles in the fluid. This is effectively the same problem as in MHD except that the fluid is not charged and there are no magnetic fields. The comfluid thorn was then developed to further test to concept of using Cactus for fluid simulations. The equations, which it evolved, are given below:

$$\frac{\partial n}{\partial t} + \nabla \cdot \left(n\vec{V}\right)$$

$$\frac{\partial \vec{V}}{\partial t} + \vec{V} \cdot \nabla \vec{V} = -\frac{k_B T}{m} \frac{\nabla n}{n}$$

where all units are MKS, $\vec{V}$ is the particle flow velocity, n is the number density (particle/m^3), m is the atomic mass for the fluid under consideration, mass density $\rho = mn$, T is the fluid temperature (assumed to be constant here) and k_B is the Boltzman constant. In addition the energy and momentum is calculated at each grid point for use in our analysis of the code's performance. It should be noted that the above equations are relatively simple, but are suitable to start with.

The equations were evolved in two dimensions in cylindrical coordinates ignoring the angular direction. Because Cactus is based on a Cartesian grid, we had to write subroutines to calculate gradients and divergences in cylindrical coordinates. We also used periodic boundary conditions to "roll" the Cartesian grid into a cylindrical one. As soon as a cylindrical grid thorn becomes available for Cactus, we plan to implement it into our program.

The code compiled and ran on all three development machines without any platform specific modifications. Slices taken in the radial and axial coordinates where then used for data analysis[2]. The initial data for the system modeled a Gaussian distribution of particles with velocities pointing out towards the radial and axial directions. As time evolved the particle distribution dispersed and the particles disappeared out the edges of the simulation domain. Towards the end of the simulation, boundary value errors begin to appear.

This test revealed two problems with the way the simulation was designed. 1) Further work is needed to increase the stability of the code so that it can run longer before significant errors occur. 2) Customized boundary conditions need to be implemented so that we can make some boundaries reflective (example when the radial direction rho = 0) while others are absorbing. Also, the stability of a finite differenced numerical code such as this depends on several factors such as boundary conditions, grid spacing, time step size and other parameters choices. Future work will involve increasing the stability of the code as well as adding new features to make the simulation more realistic.

In order to coordinate the development of improvements to the code while not destroying the progress that we have already made, we split the code and began work on an advanced "development" version. The "stable" version was saved for later study while the development version is continuously changed to improve stability and experiment with new features.

DEVELOPMENT THORN

Time Integration

The first technique adopted in the development code is the Iterated Crank-Nicholson time integrator. By using a second or higher order time integration technique such as Iterated Crank-Nicholson or Runge-Kutta, we can further increase the stability of the code. These techniques work well in numerical relativity and should work well for our program. These systems work by correcting for small errors, which occur as we evolve the equations from the solution at one time to the next. Instead of the growth in errors depending directly on the time step, they depend on the time step squared. This can decrease error growth by several orders of magnitude without a significant decrease in computational speed.

Boundary Conditions

The stable version of our code currently depends on Cactus' built in boundary conditions. By developing our own boundary condition subroutines, we can reduce errors at the boundary by "tuning" the boundaries to our system. Eventually we can introduce absorbing boundary conditions, which eliminate computational artifacts such as unwanted reflections and further reduce boundary errors. Most importantly, we can choose where to apply reflective and absorbing boundary conditions in order to make our

[2] See Appendix B

simulation more realistic. If we are working in cylindrical coordinates, no information should leave the grid when it passes through rho equals zero.

Spectral methods

Cactus is currently designed to use finite differencing as a method of numerically calculating the derivatives of functions. Spectral methods have been shown to be much more accurate and stable than finite difference methods but more difficult to implement. There is currently an effort to develop a general spectral methods thorn for Cactus. Once it has been released, we can begin testing it and eventually add it to our code.

Adaptive mesh refinement

Adaptive mesh refinement (AMR) is a technique where the grid spacing can change depending on the dynamics of the code. This leads to greater accuracy in parts of the grid where it is needed and less accuracy where it is not. This increases both accuracy and computational efficiency. There is currently a third party Cactus Thorn that adds AMR to Cactus.

Other improvements

There are several other improvements that can be made to the code including improved initial conditions, the addition of dissipative terms, viscosity, temperature variations in the fluid and much more. These improvements can be added as needed, however, the focus of the code will be to test the concept of using Cactus for VASIMR research and then to develop a MHD code.

Add MHD equations

The long-term goal of this project is to add the MHD equations and turn this fluid dynamic code into a full MHD code [4,5]. This will involve adding a several more evolution equations to the list of coupled differential equations. These include equations for charge density, magnetic field, and the energies and momentum carried by both. There is an additional difficulty at this point in understanding the dynamics of how these equations are evolved and making them as stable as possible. Because of this it is to our advantage to develop a modular, well documented and easy to understand code so that future students can add equations with minimal intimidation.

APPENDIX A

Cactus program structure

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

APPENDIX B

Preliminary Numerical Results

The energy of the compressible fluid flows to the boundary and disappears, boundary errors develop. For both plots: Blue = early times, Red = late times, x = radial, z = axial. Both plot where produced with ygraph.

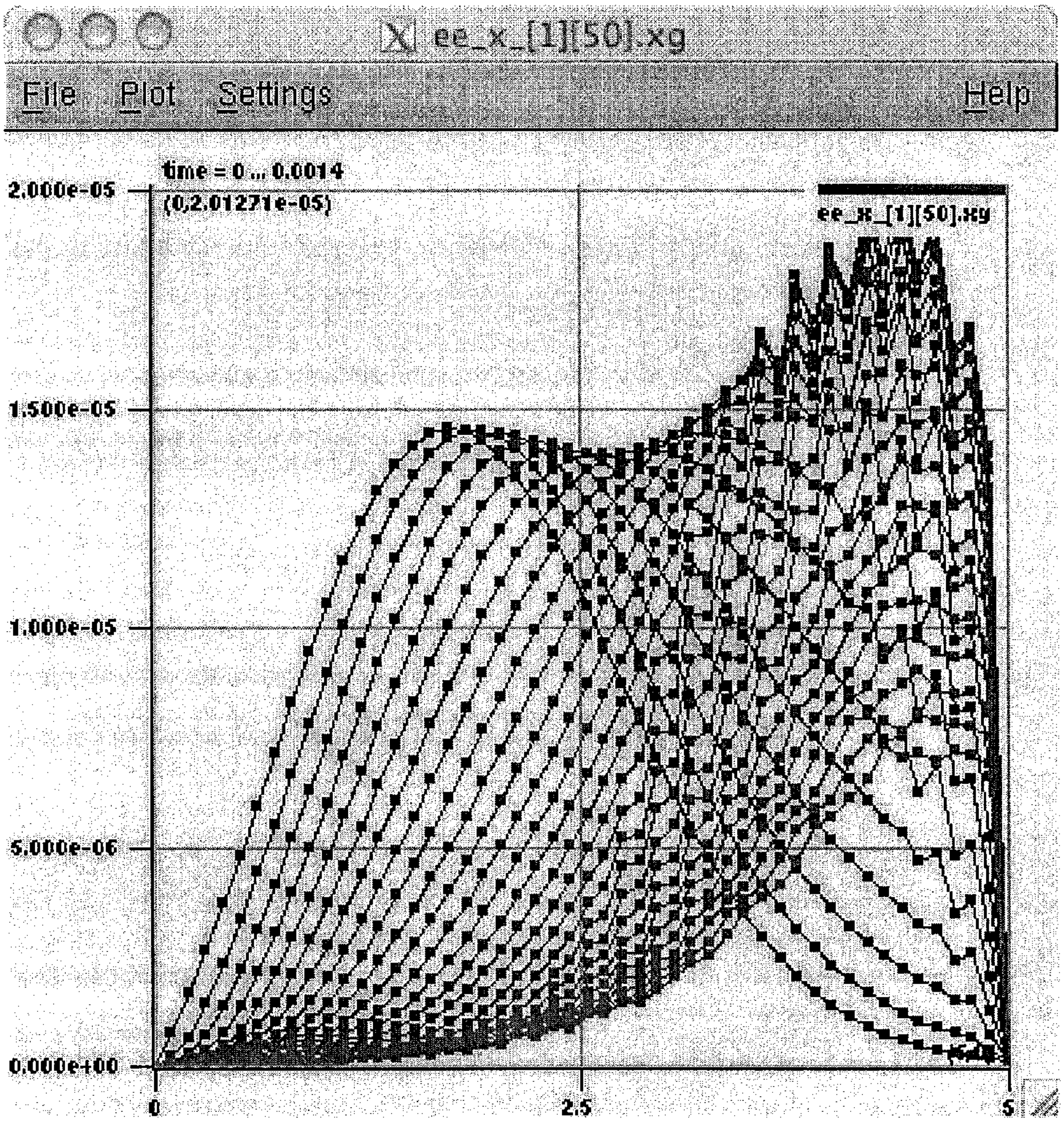

Figure 1 : Above is a plot of energy vs. radial position taken at several times. The y axis is energy amplitude while the x axis shows radial position.

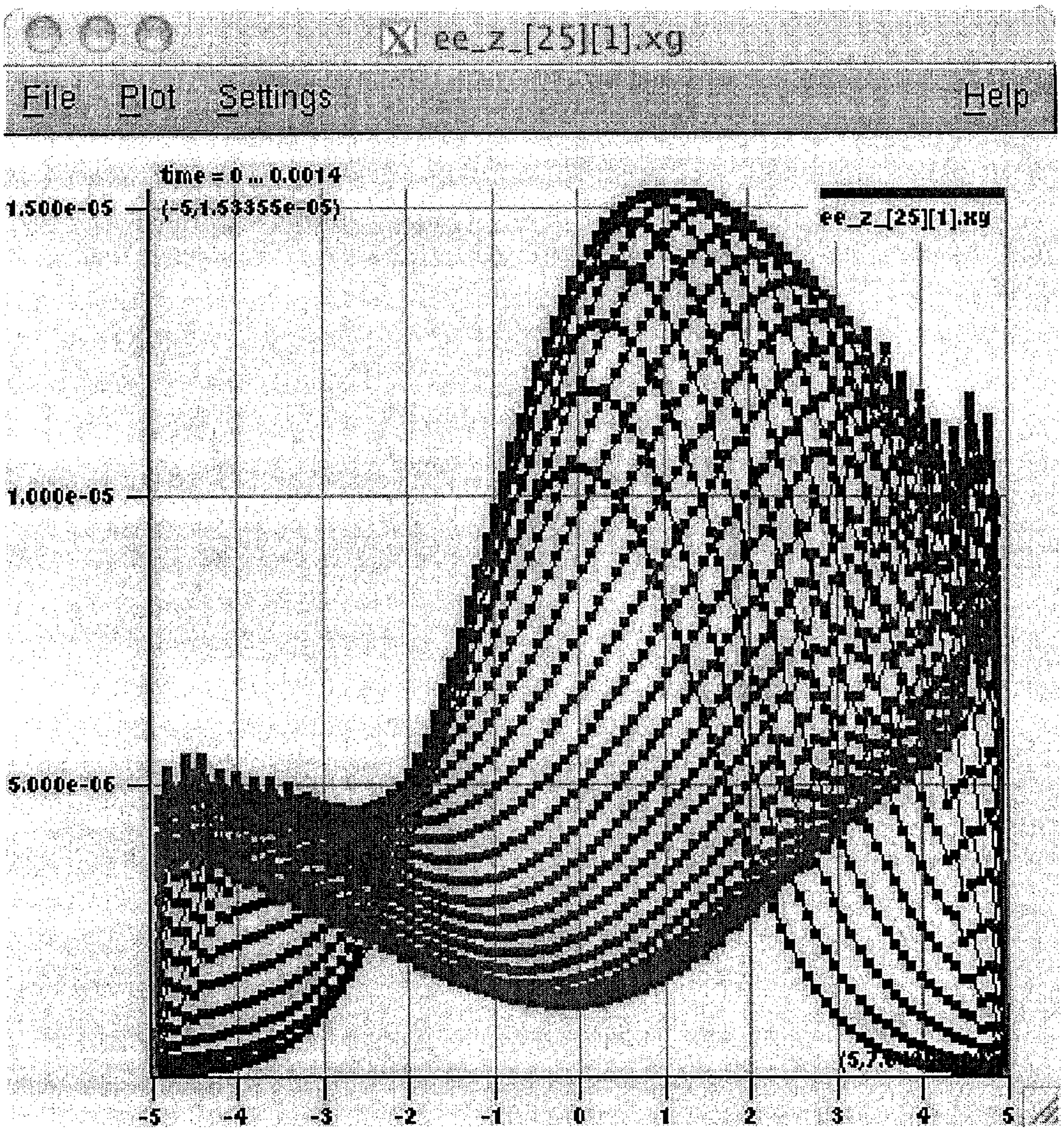

Figure 2 : Above is a plot of energy vs. axial position taken at several times. The y axis is energy amplitude while the x axis shows axial position.

REFERENCES

1. Cactus, **http://www.cactuscode.org**

2. Chang Diaz F. R. (2000) ***The VASIMR Rocket***, Scientific American, 283, (5), 90-97.

3. Goodale, Tom, ***Cactus 4.0: An Introduction and Perspectives On Future Plans***, PowerPoint presentation.

4. Tarditi, A. G. and J. V. Shebalin, ***MHD simulation of flow through the VASIMR magnetic nozzle***, APS Division of Plasma Physics Meeting, Oct. 2003.

5. Tarditi, A. G., J. V. Shebalin, and E. A. Bering, ***MHD simulation of the exhaust plume in the VASIMR advanced propulsion concept***, XXIII General Assembly of the International Union of Geodesy and Geophysics, IUGG2003, Sapporo, Japan, June 2003