Verification, validation, and performance optimization of magnetohydrodynamics simulation using OpenFOAM code

Griffin Kowash

14 October 2021

Abstract

We report progress on verification and validation of a new magnetohydrodynamics code developed for liquid metal breeding blanket fusion applications in the open-source CFD library OpenFOAM. Two test cases with analytical solutions derived by J.A. Shercliff and J.C.R. Hunt are described, and in this paper simulation results for the former at a Hartmann number of 500 are presented; papers by fellow interns Isabella Marshall and Vincent Galvan detail results at higher Hartmann numbers. We conclude by discussing efforts to optimize solver parameters and meshing procedures for computation and workflow efficiency.

1 Introduction

Given the limited abundance of tritium in nature, commercial-scale fusion reactors utilizing the deuterium-tritium process will require a mechanism to generate their own tritium for sustainable fueling. Liquid metal breeding blankets are a proposed technology designed to meet this need. Consisting of a layer of liquid lithium circulating around the reactor core, they generate tritium when high-energy neutrons produced by fusion reactions interact with lithium nuclei to trigger fission into tritium and helium. With sufficient knowledge of the flow behavior in the liquid lithium, the tritium can be safely extracted and used as fuel for the reactor. In some designs, liquid metal breeding blankets also play a secondary role by transferring heat away from the core, either as the primary coolant or as a supplement to a traditional water or helium coolant.

Accurately modeling the flow behavior within liquid metal breeding blankets is critical for safely collecting tritium, mitigating flow instabilities, and optimizing the thermal efficiency of the reactor [1]. However, the presence of strong magnetic fields in the region around the reactor core combined with the conductive nature of liquid metal introduces complex electromagnetic effects which requires the use of magnetohydrodynamics (MHD).

Our team at PPPL, consisting of Dr. Andrei Khodak and fellow interns Isabella Marshall and Vincent Galvan, worked with Daniel Suárez Cambra of the Polytechnic University of Catalonia from June through August of 2021 to perform verification and validation procedures on a new MHD code optimized for liquid metal breeding blanket applications. The code was created by Dr. Elisabet Mas de les Valls [1] using the computational fluid dynamics (CFD) library OpenFOAM to provide an open-source alternative to commercial CFD software.

This paper will focus on efforts to optimize computation and simulation workflow. Papers by fellow PPPL interns Isabella Marshall and Vincent Galvan provide a more comprehensive presentation of our final simulation results at multiple Hartmann numbers.

2 Background

2.1 Qualitative description of key MHD effects

The necessity of MHD in breeding blanket simulation arises from the movement of the liquid metal's conduction band electrons through the strong magnetic fields surrounding magnetic confinement fusion reactors. The resulting Lorentz force acting on the electrons will generate electric currents in the fluid perpendicular to both the magnetic field and the direction of motion. In turn, these currents will experience a Lorentz force opposite the velocity of the fluid, which exerts a damping force on the liquid metal flow. The exact distribution and magnitude of this damping force depends on the nature of the electric currents, which in turn depend on the magnetic field strength, flow velocity, fluid and wall conductivities, and channel geometry. In principle, the electric currents will also affect the magnetic field via induction; however, the conditions found within liquid metal breeding blankets make inductive effects insignificant, allowing an inductionless quasi-static approximation to be used that substantially simplifies computation [1].

To understand these effects in more detail, we will consider a long, rectangular, insulated duct with liquid metal flowing down the longitudinal axis. We assume that the magnetic field is oriented perpendicular to one set of walls; these are referred to as the Hartmann walls, while the walls parallel to the field are referred to as the side walls. The Lorentz force induces a current across the direction of flow toward the side wall, as shown by the horizontal current streamlines on the cross-section on the right-hand side of Figure 1. In a narrow region along the side walls called the side layer, charge conservation forces the current to redirect 90 degrees toward the Hartmann walls, as shown by the vertical streamlines at the edges of the diagram. At the corners of the duct, the current is again redirected to pass along the Hartmann walls in a narrower region called the Hartmann layer. Finally, it returns along the opposite side wall and circulates back through the bulk region.

The magnetic field strength is an important factor in determining the nature of the velocity and current profiles. With fluid properties held constant, the magnetic field strength can be represented by the Hartmann number Ha:

$$Ha = BL\sqrt{\frac{\sigma}{\nu\rho}} \tag{1}$$

where B is the magnetic field, L is the half-width of the duct in the direction of the field, σ is the fluid conductivity, ν is the fluid kinematic viscosity, and ρ is the fluid density. The Hartmann number has a strong effect on the width of the side and Hartmann boundary layers, which are given by the following expressions:

$$w_{side} = \frac{L}{\sqrt{Ha}} \tag{2}$$

$$w_{Hartmann} = \frac{L}{Ha} \tag{3}$$

Both boundary regions grow narrower with increasing magnetic field strength, but the Hartmann layer does so at a faster rate. Meshes must sufficiently resolve these areas, as their electric currents are important to overall flow behavior, while avoiding discontinuity in cell size at the interface with the lower-resolution bulk region. Meshing procedures and associated challenges are addressed later in this paper.

Scenarios at high Hartmann numbers (5000–15000) also present a challenge for computation. While this paper only utilizes simulations at a relatively low Hartmann number of 500, the performance of the code at higher Hartmann numbers is discussed in papers by fellow PPPL interns Isabella Marshall and Vincent Galvan.

2.2 Verification and validation cases

Verification and validation is an essential part of evaluating the performance of CFD codes. Verification is the process of determining whether the mathematics of the code have been implemented correctly by comparing simulation results with analytical solutions. Validation compares simulation results with experimental measurements to determine whether the code accurately models physical reality. We follow the standardized verification and validation procedure described in Smolentsev et. al. 2015 [2] to evaluate the performance of the new code.

To perform the verification and validation procedure, we examine two test cases with analytical solutions given in Shercliff 1953 [3] and Hunt 1965 [4]. Both cases share the same basic rectangular geometry described above, but with different conductivities for the walls of the duct. In Shercliff's case, all walls are electrically insulating, whereas in Hunt's case, the Hartmann walls (top and bottom walls in the diagram in Figure 1) are electrically conductive.

In both cases, the central region in which the current flows perpendicular to the magnetic field experiences a Lorentz force opposite the direction of the fluid flow, which has the effect of depressing the middle of the otherwise parabolic velocity profile. Along the side walls, the currents run parallel to the magnetic field, so little magnetic damping occurs.

The exact nature of the resulting velocity profile depends strongly on the wall conductivity, which is where the Shercliff and Hunt cases differ. With non-conducting walls, as in the Shercliff case, the currents are modest enough that the center of the velocity profile is merely flattened, creating the plateau shown on the left side of Figure 1. However, the conductive walls present in Hunt's case reduce electrical resistance, which allows for higher current. With sufficient current, the liquid metal flow in the center of the channel can be restricted to almost zero, causing most flow to occur along the side walls where it is relatively unopposed by the Lorentz force. This results in the M-shaped velocity profile shown on the left side of Figure 2.

2.3 Algorithm outline

This section presents a simplified description of the algorithm used by the code, which is depicted as a diagram in Figure 3 below.

The algorithm begins with an initial state provided by the user; we set pressure and electric potential both to zero and velocity to a uniform value consistent with our desired Reynolds number. The code computes an approximate solution to the governing MHD equations for momentum, performing multiple

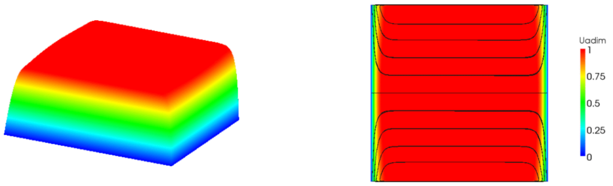


Figure 1: Velocity profile and electric current lines of the Shercliff case, with magnetic field oriented upward. The left side shows a three-dimensional velocity map of the duct's cross section, while the right side shows a two-dimensional view of the velocity field with circulating electric current lines overlaid in black. The central region of the velocity field is flattened by the magnetic force acting on the horizontal current. (Figure from Mas de les Valls 2011 [1])

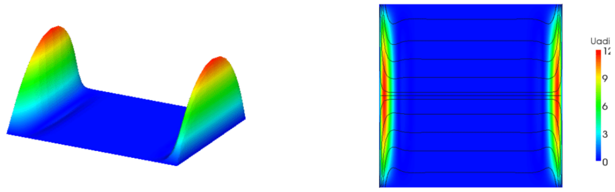


Figure 2: Velocity profile and electric current lines of the Hunt case, similar to Figure 1. The conducting walls permit a higher current to flow, resulting in a greater magnetic force impeding fluid motion in the central region. The regions along the side walls are relatively unimpeded due to the redirection of the current lines, which leads to the formation of velocity jets that support most of the fluid flow. (Figure from Mas de les Valls 2011 [1])

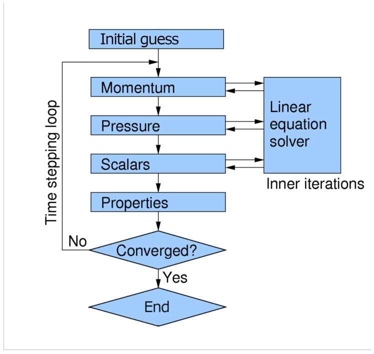


Figure 3: Simplified diagram depicting the procedure followed by the MHD solver. Starting from an initial state, momentum, pressure, electric field, and other properties are calculated iteratively over multiple time steps until the desired convergence is achieved. (Diagram modified from Perić 2016 [5])

iterations with a linear equation solver to refine the result. The same is done for the pressure field (using a PISO-like pressure-velocity coupling algorithm), and then for other scalar fields such as the electric potential. Other properties like electric current are updated, and the convergence is evaluated by comparing the updated solution to the previous state. If it has not reached the desired convergence, the process is repeated for another time step. The fields ideally will eventually reach a state of sufficient convergence, at which point the simulation will be complete.

3 Preliminary results and GCI analysis for Shercliff case

Our work from June through August of 2021 focused on the Shercliff case described in the Background section, with some limited results for Hunt's case obtained toward the end of the internship period. This section will present simulation results and a grid convergence analysis for the Shercliff case at a Hartmann number of 500.

Simulation accuracy is evaluated using the dimensionless flow rate $\widetilde{Q}$. This quantity is corresponds to the rate of fluid flow through the channel, normalized to be independent of fluid properties, channel geometry, wall conductivity, and magnetic field strength. It is calculated from the simulation output according to the following integral over the duct's cross-section [2]:

$$\widetilde{Q} = \int_{-1}^{1} \int_{-1}^{1} U \ / \ [L^2 \nu^{-1} \rho^{-1} (-dP/dx)] \ dy \, dz \qquad (4)$$

where U is fluid velocity, L is the half-width of the duct in the direction of the magnetic field, ν is the fluid dynamic viscosity, ρ is the fluid density, and dP/dx is the pressure gradient. To evaluate code performance, the computed $\widetilde{Q}$ values are compared with the analytical solutions derived by Shercliff and Hunt for their respective cases.

Figure 4 shows the convergence of the dimensionless flow rate $\widetilde{Q}$ over iterative time steps for simulations run on a coarse, medium, and fine mesh.

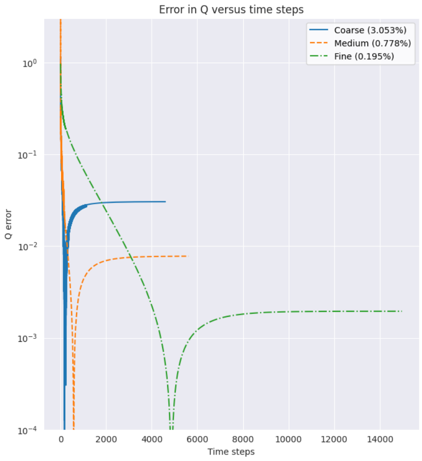


Figure 4: Absolute error in dimensionless flow rate plotted against number of time steps for coarse, medium, and fine meshes (Shercliff case, $Ha = 500$). The final errors are 3.053%, 0.778%, and 0.195%, respectively.

The results, with final errors of 3.053%, 0.778%, and 0.195% for the coarse, medium, and fine meshes, respectively, indicate that the error decreases with increasing levels of refinement.

The grid convergence index (GCI) is a metric that characterizes a mesh's discretization error. An ideal mesh will tend to converge to zero error with increasing refinement. The analysis utilizes Richardson extrapolation with the results of three simulations with increasingly fine meshes to predict the expected result with an infinitely fine grid. This extrapolated value is then compared with the simulation results and the known analytical solution to evaluate the discretization error and its apparent order. The mathematical details of the calculation can be found in Celik 2008 [6].

Simulating the Shercliff case with a Hartmann number of 500, we obtained an extrapolated $\widetilde{Q}$ error of 0.0207%, a discretization error order of 1.9656, and a fine grid convergence index of 0.2508%.

The following section discusses the procedures used to generate and refine the meshes used in the GCI analysis.

4 Mesh generation and refinement

4.1 Mesh generation

The simple rectangular geometry of the Shercliff and Hunt cases lends itself well to algorithmic mesh generation. Meshes are created on an orthogonal grid aligned with the walls of the duct. The simplest option is to create a mesh of uniformly sized cells; however, the need to resolve the narrow side and Hartmann boundary layers would result in a prohibitively large cell count in the bulk region. A straightforward solution is to make cells in the bulk region larger, as less resolution is needed there, but this introduces large discontinuity in cell size at the interface between bulk and boundary as shown in Figure 5, which is known to introduce error to CFD simulations.

To address these concerns, we implement a graded mesh in which cell widths increase geometrically toward the center of the mesh. We select the desired ratios between adjacent cell widths in each region (the cell-to-cell ratios) and the number of cells desired in the boundary layers, and those values are used to calculate the required number of cells in the bulk region to ensure continuity in cell size at the boundary-bulk interface.

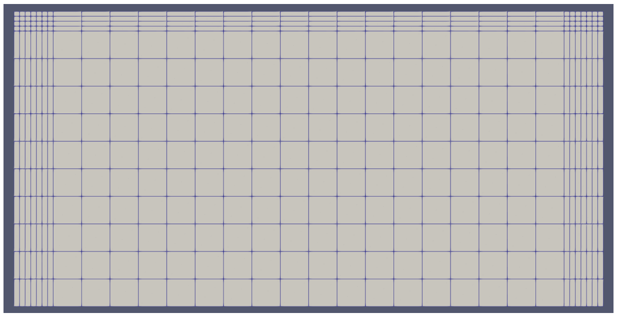


Figure 5: Example of a mesh with discontinuities in cell size at the boundary-bulk interfaces. Resolving the narrow boundary regions at the top, left, and right edges of the image requires much smaller cell sizes than are needed for resolving the more uniform central region.

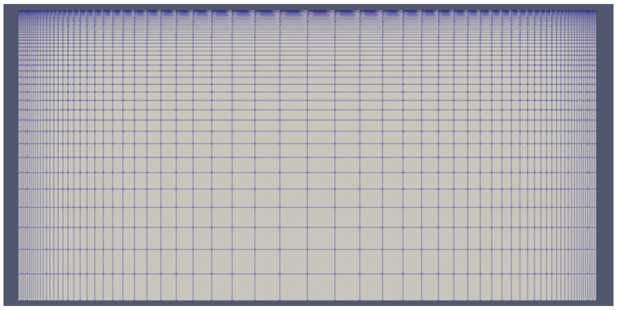


Figure 6: Coarse graded mesh used for $Ha = 500$ simulations. The side layers are visible at the right and left edges as regions of denser cells, and the narrower Hartmann layer is similarly visible at the top. The cells in the bulk region grow larger toward the bottom-center of the image due to the geometric grading.

This information is used to determine a quantity called concentration for each region, which is a required input for the built-in grading function in OpenFOAM:

$$C = \frac{1}{(c2c)^{N-1}} \tag{5}$$

where C is concentration, $c2c$ is the cell-to-cell ratio, and N is the total number of cells along the relevant axis in the region. Physically, this quantity is the ratio of the smallest cell width to the largest cell width within the given region.

An example of a typical graded mesh is shown in Figure 6. Note that the mesh comprises only the top half of the square duct; the bottom half is computed during simulation using a symmetry plane.

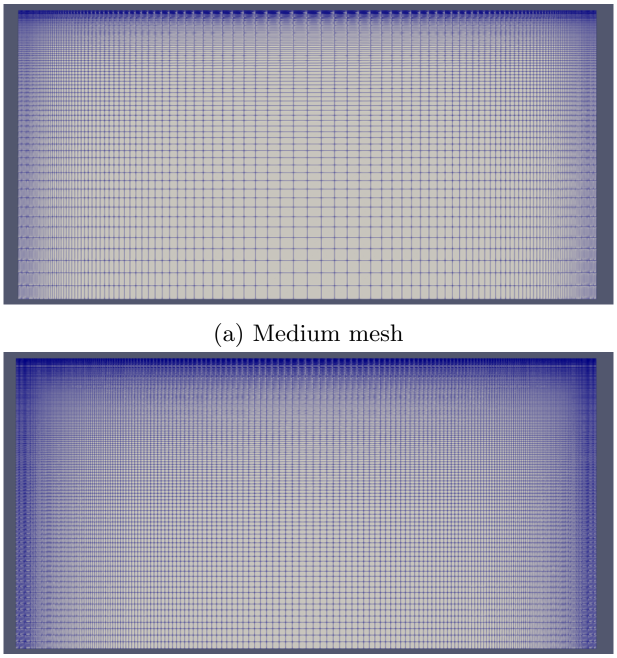
4.2 Mesh refinement

For the GCI analysis described above, it is necessary to generate three meshes with increasing levels of refinement. To simplify the calculation, it is useful to ensure that the grid refinement factor, or the ratio of average cell widths between a mesh and its refined counterpart, is constant for each stage of refinement (refer to Celik 2008 for details). For example, if we select a grid refinement factor of two, and the average cell width in our coarsest mesh is 2mm, then the respective quantities for the refined meshes should be 1mm and 0.5mm.

In the absence of mesh grading, refinement is a straightforward task of scaling the number of cells in each region by the grid refinement factor. However, the geometric nature of the mesh grading results in discontinuity at the bulk-boundary interface when this technique is applied. To avoid the complexity of this effect, we investigated a simple workaround; to create a mesh with a refinement factor of two, we double the number of cells in each region while maintaining the same concentrations from the previous mesh. This procedure produces meshes with the correct grid refinement factors, and by rough visual inspection there is no obvious discontinuity in cell size. However, this process does in fact introduce some discontinuity at the boundary-bulk interfaces, which presents a concern for simulation accuracy.

To evaluate the significance of this discontinuity, we consider the following expressions, which give the minimum and maximum cell widths within a geometrically graded region:

$$w_{min} = L * \left(\frac{1 - C^{\frac{-1}{N-1}}}{1 - C^{\frac{-N}{N-1}}} \right) \tag{6}$$



(a) Medium mesh

(b) Fine mesh

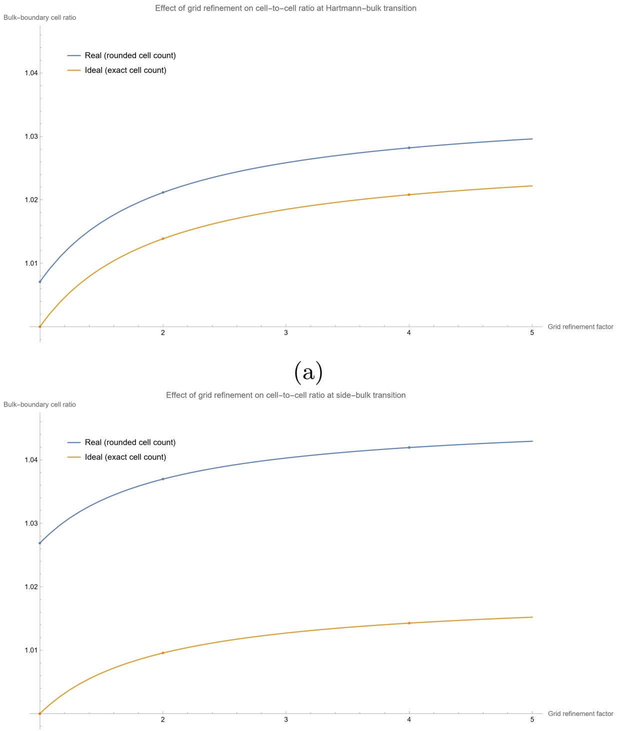
Figure 7: Refined meshes generated from the mesh in Figure 6 with refinement factors of a) two and b) four.

$$w_{max} = \frac{w_{min}}{C} \tag{7}$$

where L is the total length of the region, C is the concentration, and N is the number of cells along the relevant axis.

To measure the degree of discontinuity, we can find the cell-to-cell ratio across the boundary-bulk interface by dividing w_{min} for the bulk region by w_{max} for the boundary region. The resulting cell-to-cell ratios for the mesh shown above are plotted in Figure 8 for grid refinement factors ranging from one to five.

The lower orange curves show the theoretical values obtained by allowing fractional cell counts during mesh calculations, as mentioned in the previous section; the higher blue curves show the actual values resulting from rounding cell counts to the nearest integer.



(a)

(b)

Figure 8: Plots of cell-to-cell ratios across the boundary-bulk interface against grid refinement factor for (a) Hartmann layer and (b) side layer. The lower orange curves show the theoretical ratios allowing for fractional cell counts, while the higher blue curves show the actual ratios using rounded integer cell counts.

The plots confirm that our simplified grid refinement process does introduce discontinuity at the boundary-bulk interfaces, and the magnitude increases with higher levels of refinement. However, even at the highest grid refinement factor of four used in our GCI calculation, the greatest cell-to-cell ratio observed is only slightly above 1.04. Considering that cell-to-cell ratios in the bulk region typically range between 1.05 and 1.15, we conclude that this simplified mesh refinement procedure does not introduce significant error to the calculation.

5 Algorithm optimization

Given the importance of computational efficiency in CFD, where simulations can take hours, days, or even weeks to reach convergence, it is worthwhile to optimize the parameters of the solver. We examined the pressure and electric potential loops described in the Background section to optimize the number of iterations of the linear solver performed per time step. We used the Shercliff case with a Hartmann number of 500 as our test case, independently varied the pressure and electric potential iteration counts, and evaluated their performance by the rate of convergence of the dimensionless flow rate.

Results for varying the electric potential iterations are shown below in Figure 9. The plot on the left shows the convergence error against the number of time steps for each iteration setting, while the plot on the right shows convergence error against computation time. Examining both plots is important, because a setting that converges more quickly in terms of simulation time steps may perform worse in execution time, as using more iterations requires more computer resources.

Indeed, the results show that although the greater accuracy gained by a higher number of electric potential iterations leads to faster convergence per time step, there is a critical point at which the additional computation time required outweighs the diminishing returns in accuracy. Based on these results, the optimal setting for electric potential iterations is between 300 and 750 iterations, and values substantially beyond that range lead to a significant slowdown in computation.

Figure 10 below shows the result of varying the number of pressure solver iterations. The left-hand plot shows that, unlike for electric potential, performing more pressure iterations brings no significant benefit to the rate of convergence. This is supported by the right-hand plot, which shows that higher number of iterations consistently lead to slower convergence in terms of computation time. The lowest iteration count shown on these plots is five, but we have obtained optimal outcomes with as few as one. We conclude that higher numbers of pressure solver iterations consistently result in slower computation.

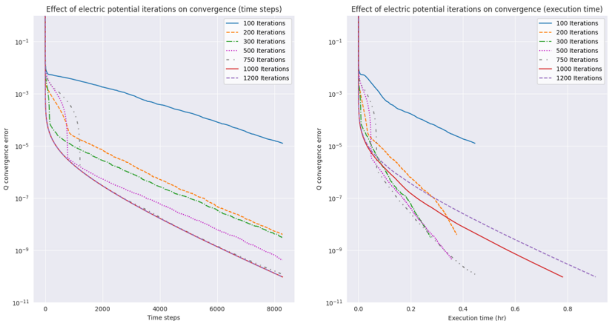


Figure 9: Simulation convergence error for a selection of electric field iteration settings plotted against time steps (left) and execution time (right). More iterations leads to faster convergence per simulation time step, but execution time is minimized with 300–750 iterations.

This finding is particularly significant to our work, as our number of pressure iterations was initially set to 1000 by default; as shown by the plots in Figure 10, changing it to between one and five can reduce computation time by nearly two thirds.

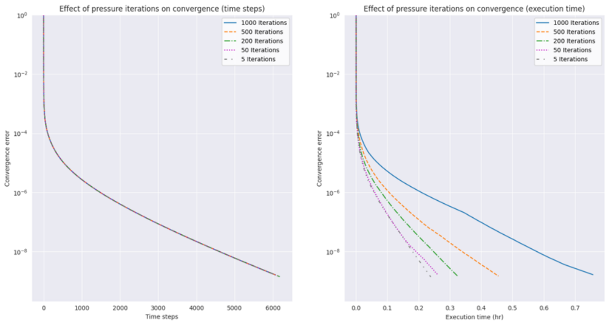


Figure 10: Simulation convergence error for a selection of pressure iteration settings plotted against time steps (left) and execution time (right). Performing more iterations requires more computation time and has no significant effect on the time step convergence rate. Minimizing the number of pressure iterations therefore leads to optimal performance.

6 Conclusion

Verification and validation efforts of the new MHD code are ongoing, but have shown promising preliminary results, as presented in this paper and in those by Isabella Marshall and Vincent Galvan. Optimization of solver iteration counts has proven to be an effective avenue for increasing computational efficiency. Future work on this project will examine Hunt's case with conducting walls as described above, as well as other more complex test cases discussed in Smolentsev et. al. 2015.

7 Acknowledgements

This work was made possible by funding from the Department of Energy for the Summer Engineering Internship program. This work is supported by the US DOE Contract No. DE-AC02-09CH11466.

Special thanks to Dr. Andrei Khodak of the Princeton Plasma Physics Laboratory (PPPL), Daniel Suárez Cambra of the Polytechnic University of Catalonia, and fellow PPPL interns Isabella Marshall and Vincent Galvan.

References

[1] Mas de les Valls Ortiz, E. (2011). Development of a simulation tool for MHD flows under nuclear fusion conditions (Doctoral dissertation). UPC, Department of Physics and Nuclear Engineering. http://hdl.handle.net/2117/95157

[2] Smolentsev, S., Badia, S., Bhattacharyay, R., Bühler, L., Chen, L., Huang, Q., Jin, H.-G., Krasnov, D., Lee, D.-W., de les Valls, E. M., Mistrangelo, C., Munipalli, R., Ni, M.-J., Pashkevich, D., Patel, A., Pulugundla, G., Satyamurthy, P., Snegirev, A., Sviridov, V., ... Zikanov, O. (2015). An approach to verification and validation of MHD codes for fusion applications. Fusion Engineering and Design, 100, 65–72. https://doi.org/10.1016/j.fusengdes.2014.04.049

[3] Shercliff, J. (1953). Steady motion of conducting fluids in pipes under transverse magnetic fields. Mathematical Proceedings of the Cambridge Philosophical Society, 49(1), 136-144. https://doi.org/10.1017/S0305004100028139

[4] Hunt, J. (1965). Magnetohydrodynamic flow in rectangular ducts. Journal of Fluid Mechanics, 21(4), 577-590. https://doi.org/10.1017/S0022112065000344

[5] Perić, M. (2016). Computation of incompressible flows: SIMPLE and related algorithms. CoMeT Continuum Mechanics Technologies GmbH. http://www.prague-sum.com/download/2016/Peric_lectures_SIMPLE.pdf

[6] Celik, I.B. (2008). Procedure for estimation and reporting of uncertainty due to discretization in CFD applications. Journal of Fluids Engineering, 130(7). https://doi.org/10.1115/1.2960953