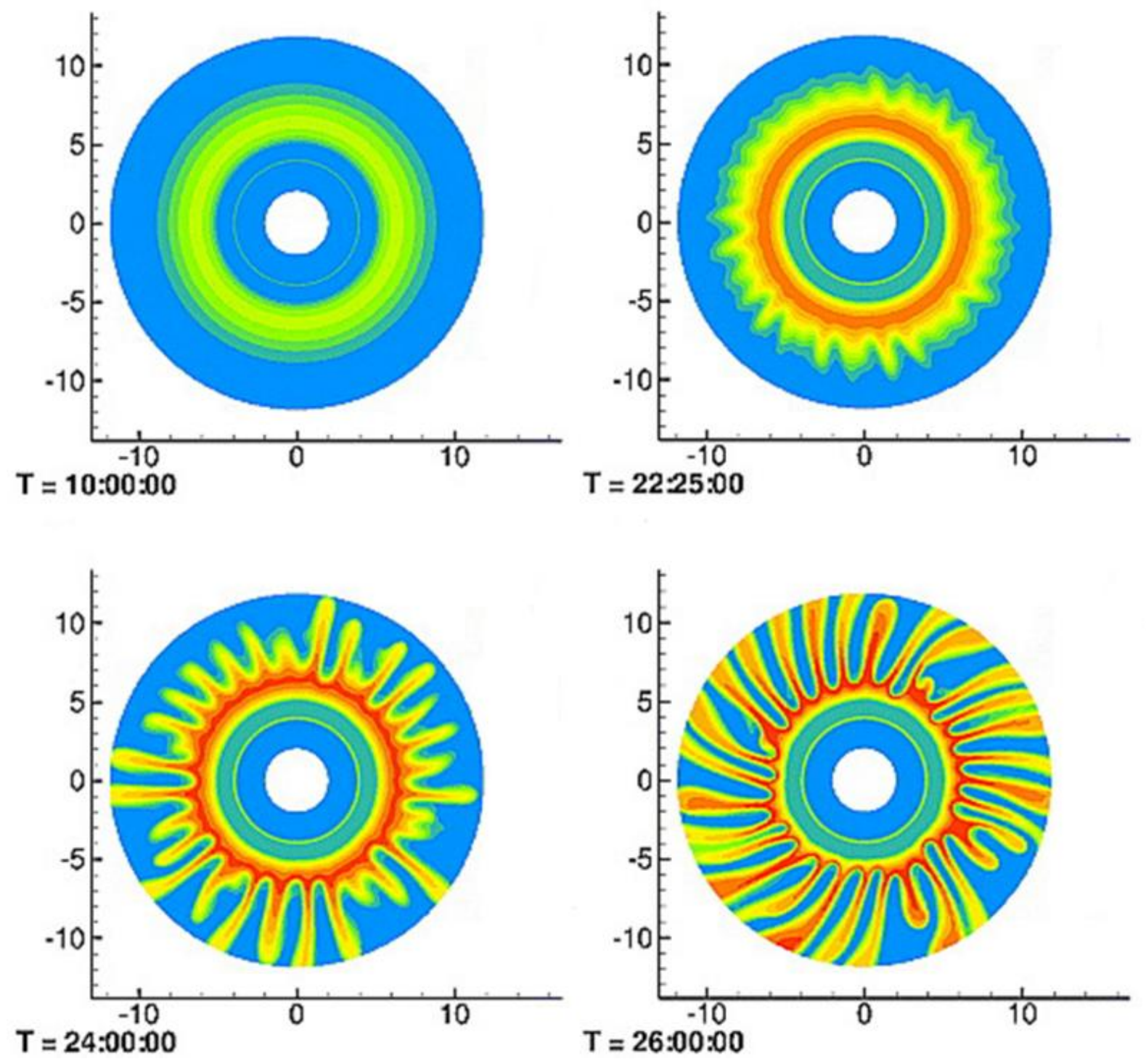
JERICHO: a Kinetic-Ion, Fluid-Electron Hybrid Plasma Model for the Outer Planets

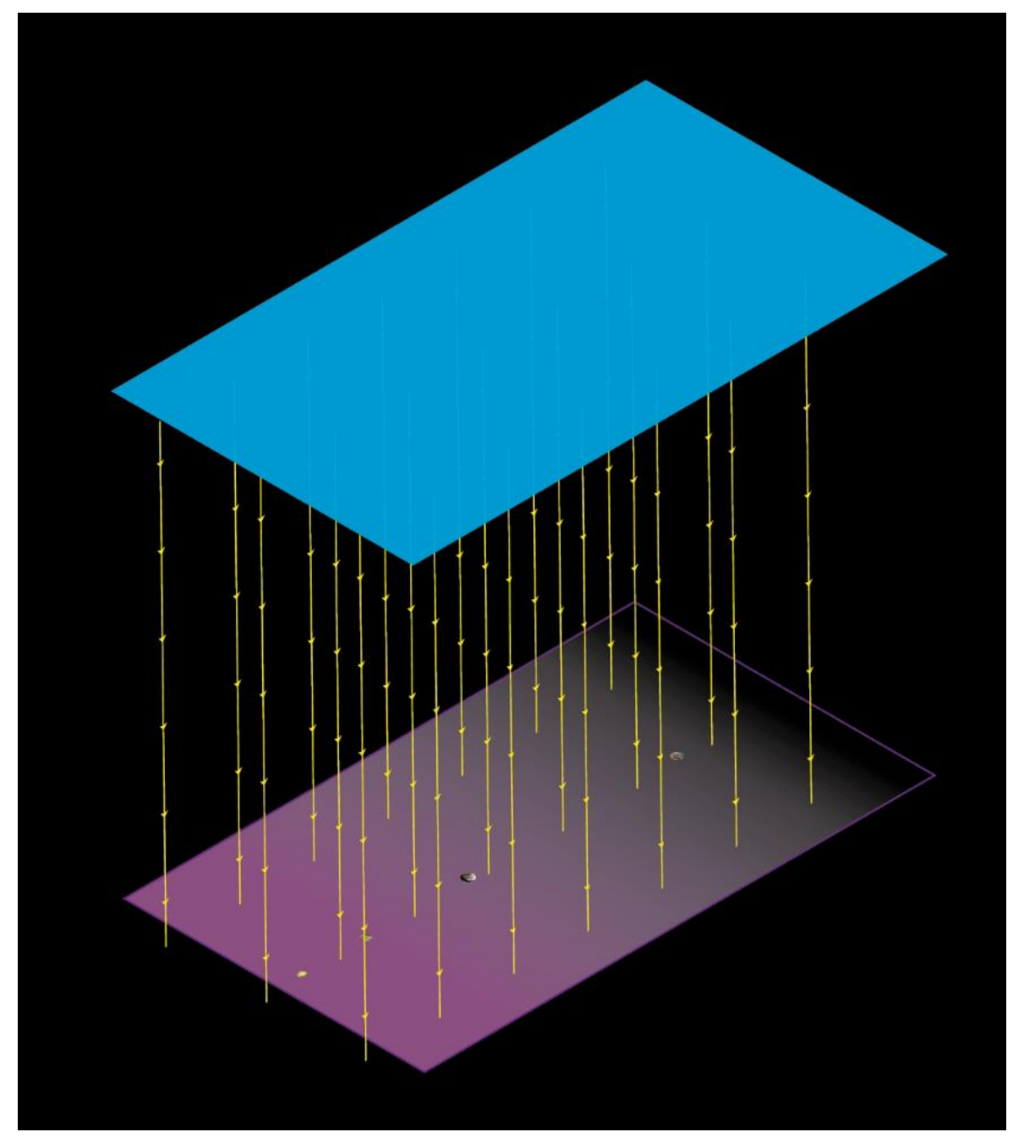
Josh Wiggs & Chris Arridge
j.wiggs@lancaster.ac.uk
Department of Physics, Lancaster University,
Lancaster, LA1 4YB, UK

1. Why Model Magnetospheres?



Liu+ 2010

We are interested in the simulation of plasma convection from Jupiter's plasma torus radially outwards. This convecting plasma is theorised to undergo the centrifugal interchange instability, analogous to the Rayleigh-Taylor instability with centrifugal force taking the place of gravity. Interchange motions occur between magnetic flux tubes and are responsible for the bulk transport of plasma from Io into the inner & middle magnetosphere (*Southwood & Kivelson, 1987 & 1989*). It is therefore necessary to examine the plasma at the ion-inertial scale in order capture the motion of particles between flux tubes whilst maintaining the computational capacity to resolve length scales on the order of the planetary radii. Current state-of-the-art models predicts transport will break-out into interchange fingers (*Yang+ 1994, Liu+ 2010*), but this has recently been disputed (*Vasyliunas, 2019*).



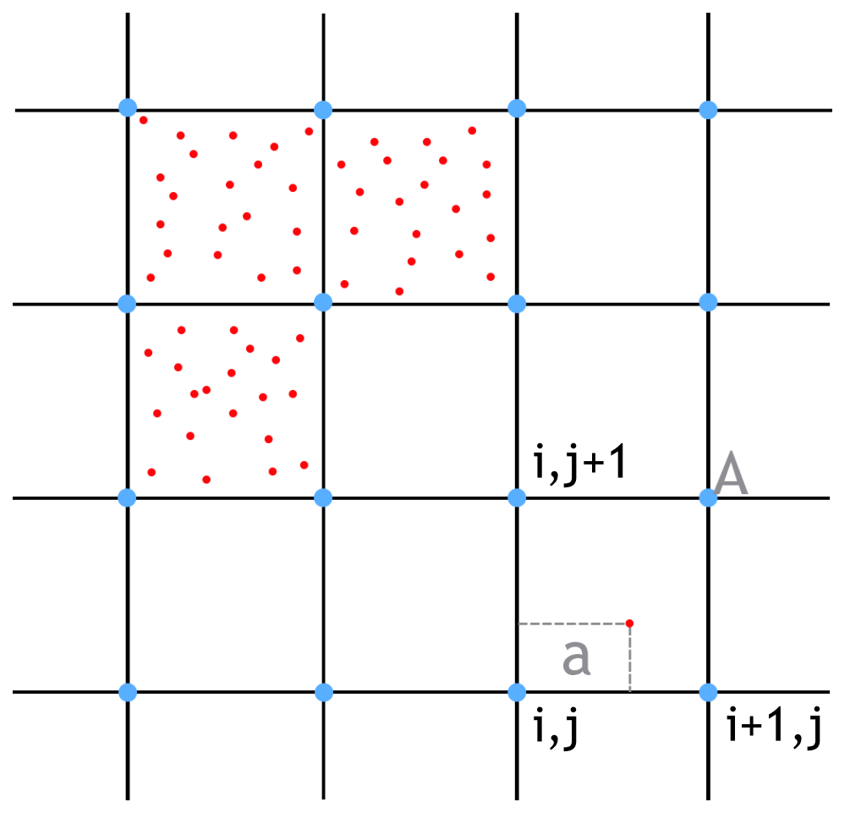
JERICHO (Jovian magnEtospheRIC ion kinetic fluid electron Hybrid plasma mOdel) is a 2.5D kinetic ion, fluid electron hybrid plasma model (*Wiggs & Arridge, in prep*) with codebases in both Python and c++. It has been developed with the aim of analysing radial outflows and magnetic flux transport from Io's torus into the middle magnetosphere over time scales comparable to the Jovian day. JERICHO currently reproduces a 2D magnetosphere which will be coupled to the Ionosphere with orthogonal field lines, this will allow insights into interchange ion motions.

Bagdonat , T. 2004, PhD Thesis, University of Braunschweig
Liu, X., Hill, T. W., Wolf, R. A., Sazykin, S., Spiro, R. W., & Wu, H. 2010, J. Geophys. Res., 115, A12254
Southwood, D. J., & Kivelson, M. G. 1987, J. Geophys. Res., 92, 109
Southwood, D. J., & Kivelson, M. G. 1989, J. Geophys. Res., 94, 299
Vasyliunas, V. M. 2019. *Poster presented at AGU Fall Meeting 2019., SM33E-3256*
Wiggs, J. A., & Arridge. C. S. *in prep*
Winske , D., Yin, L., Omidi , N., et al. 2003. In Space Plasma Simulation, ed. J. Büchner , C. T. Dum, & M. Scholer (Springer, Berlin), 136
Yang, Y. S., Wolf, R. A., Spiro, R. W., Hill, T. W., & Dessler, A. J. 1994, J. Geophys. Res., 99, 8755

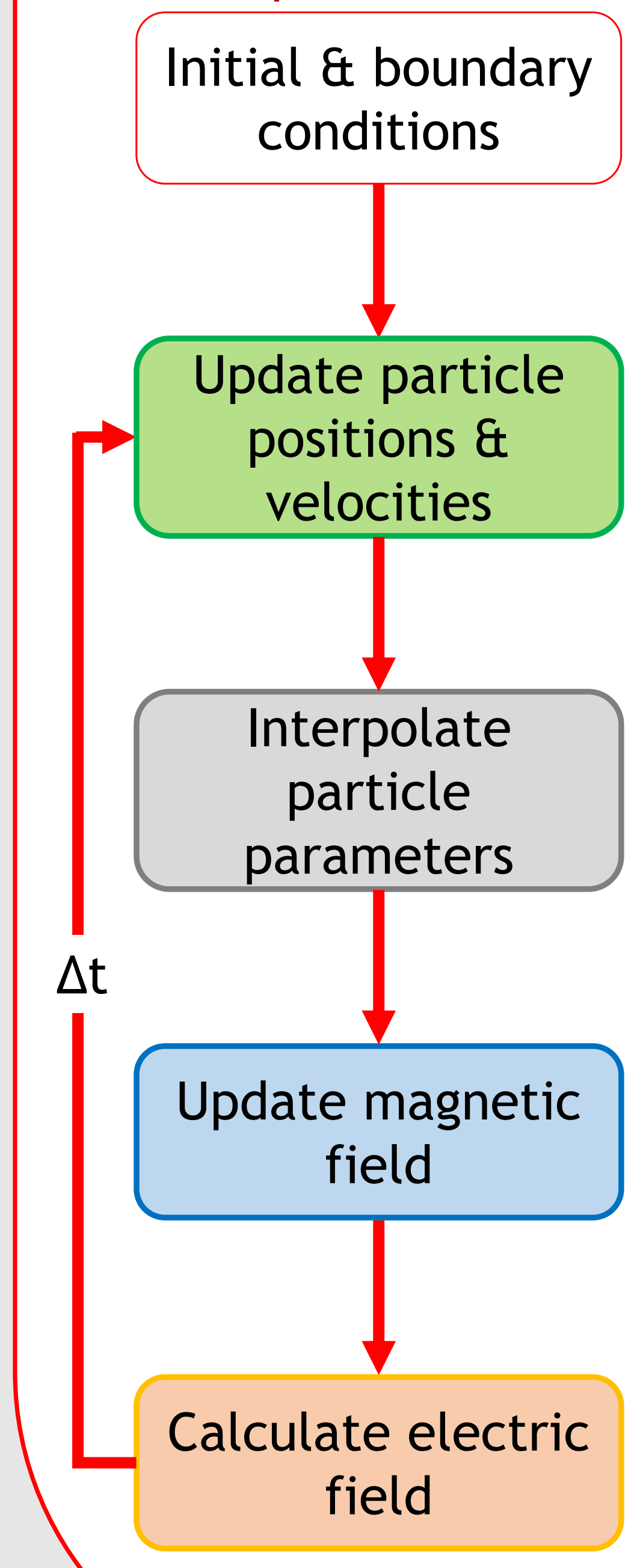
Space & Planetary Physics | Lancaster University

THE ROYAL SOCIETY

2. How Does JERICHO work?



We have developed a 2.5D hybrid kinetic-ion, fluid-electron model, JERICHO. The ions are modelled using a Particle-In-Cell (PIC) description and the electrons are a neutralising magnetohydrodynamic (MHD) fluid (*Winske+ 2003, Bagdonat 2004*). A Cartesian grid is overlaid across the simulation region on the vertex's of which the electromagnetic (EM) fields are calculated. The model is advanced through time numerically, with the magnetic field being obtained with a modified MacCormack Predictor-Corrector scheme in order to minimise numerical instabilities allowing larger time steps.



Ions are pushed numerically considering the Electromagnetic, Coriolis and Centrifugal forces

$$\frac{d\underline{v}}{dt} = \frac{q}{m}(\underline{E} + \underline{v} \times \underline{B}) - 2\underline{\Omega} \times \underline{v} - \underline{\Omega} \times (\underline{\Omega} \times \underline{r})$$
$$\frac{d\underline{r}}{dt} = \underline{v}$$

$$\rho_c = \sum_p W\left(\underline{x}_p, q_p\right)$$

Parameters are obtained from ion distributions using first-order interpolation onto the EM field grid vertices

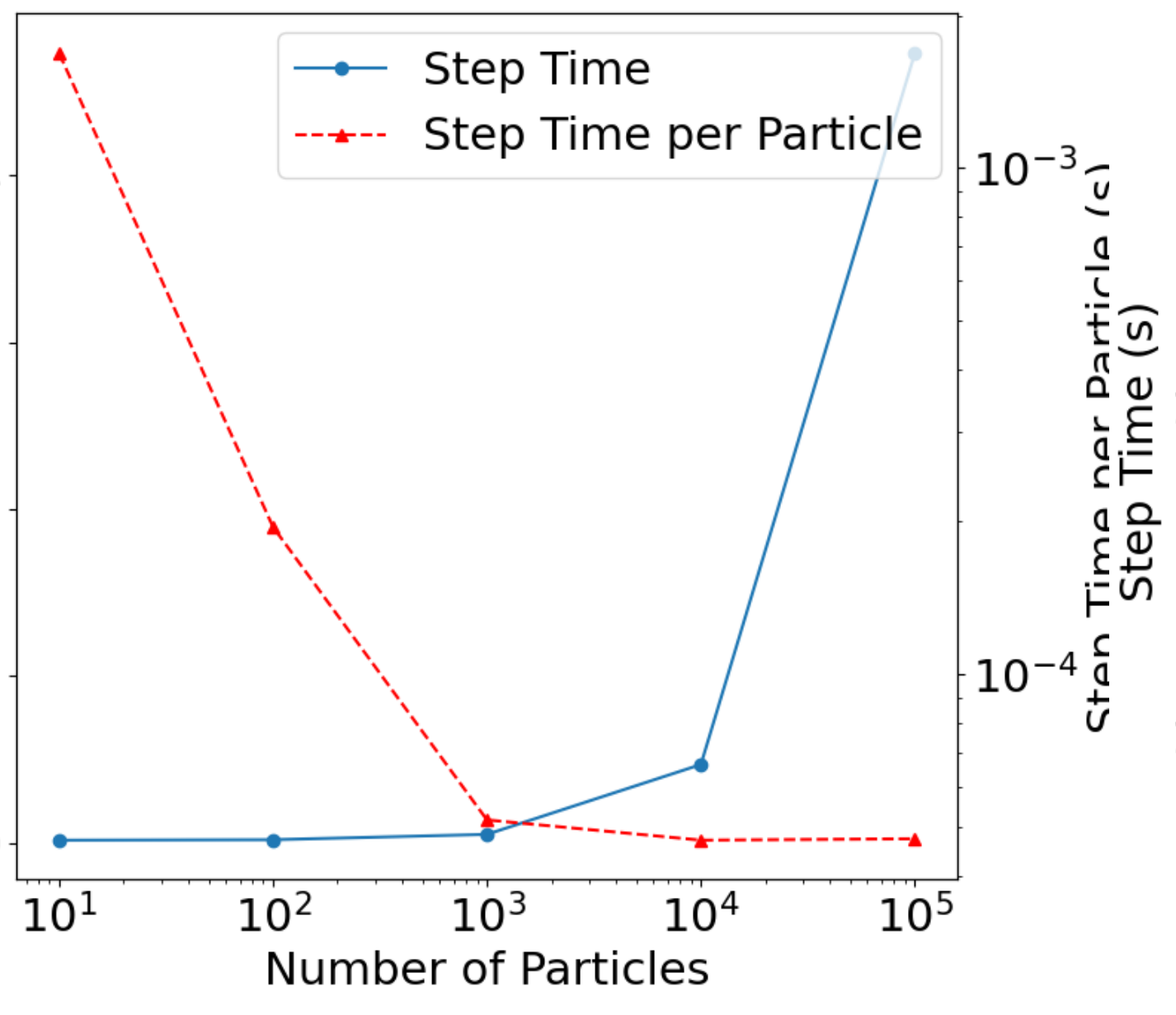
$$\frac{\partial \underline{B}}{\partial t} = -\left(\nabla \times \underline{E}\right)$$

The magnetic field is updated using Faraday's Law and the electric field using the MHD momentum equation for massless electrons

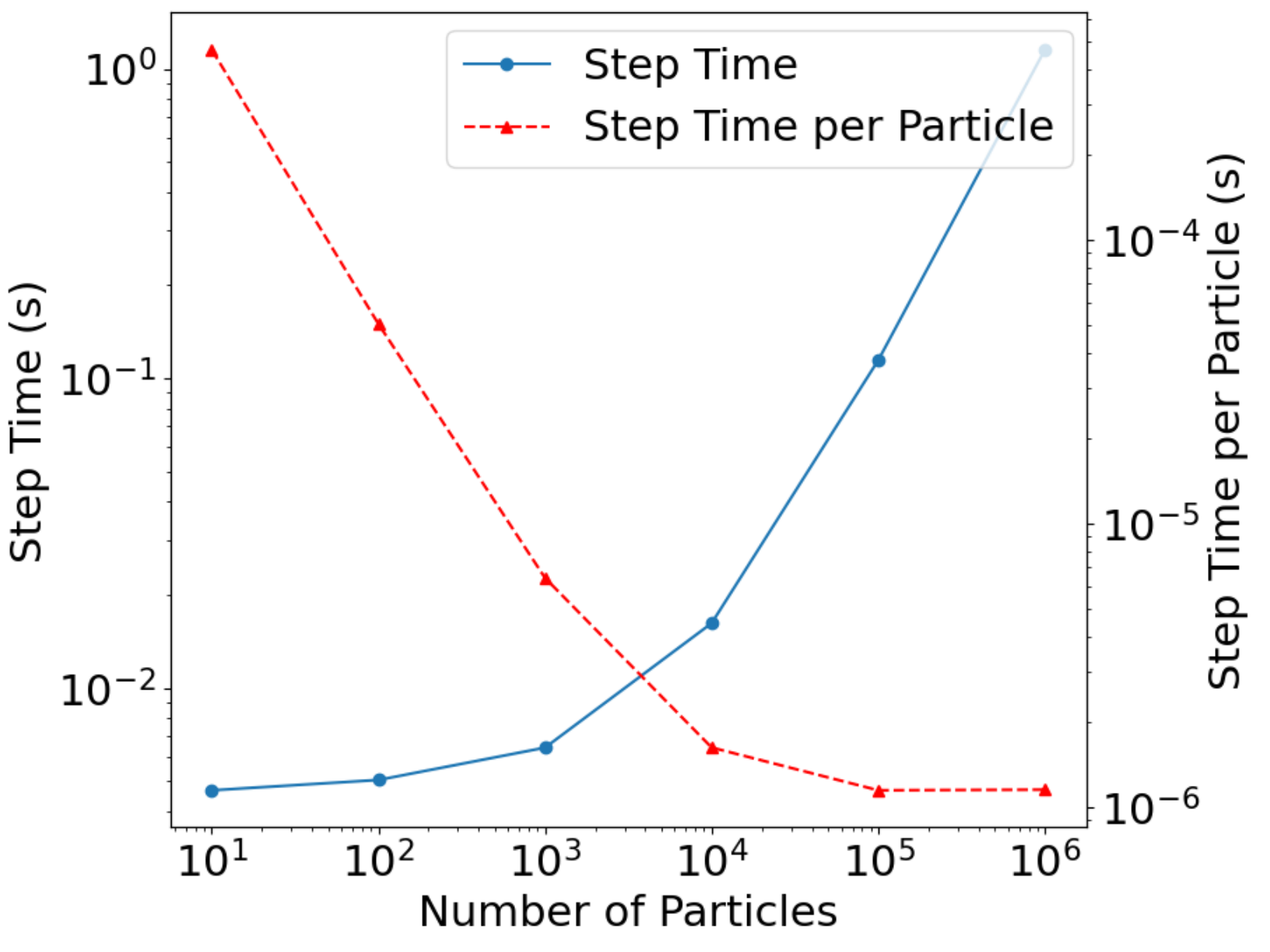
$$\underline{E} = -\frac{1}{\rho_c}\nabla \cdot \underline{\underline{p_e}} - \left[\underline{U}_i \times \underline{B} - \frac{1}{\mu_0 \rho_c}(\nabla \times \underline{B}) \times \underline{B}\right]$$

3. From Python to c++

A series of performance tests on the Python and c++ versions of JERICHO were carried out on a 10x10m surface with a 51x51 grid. It can be seen that as the number of particles increases then the step time per particle decreases linearly up to a critical value. This value is where the particle operations dominate the run time. The critical value is reached at 47μs in the Python version and 1.1μs in the c++ version.

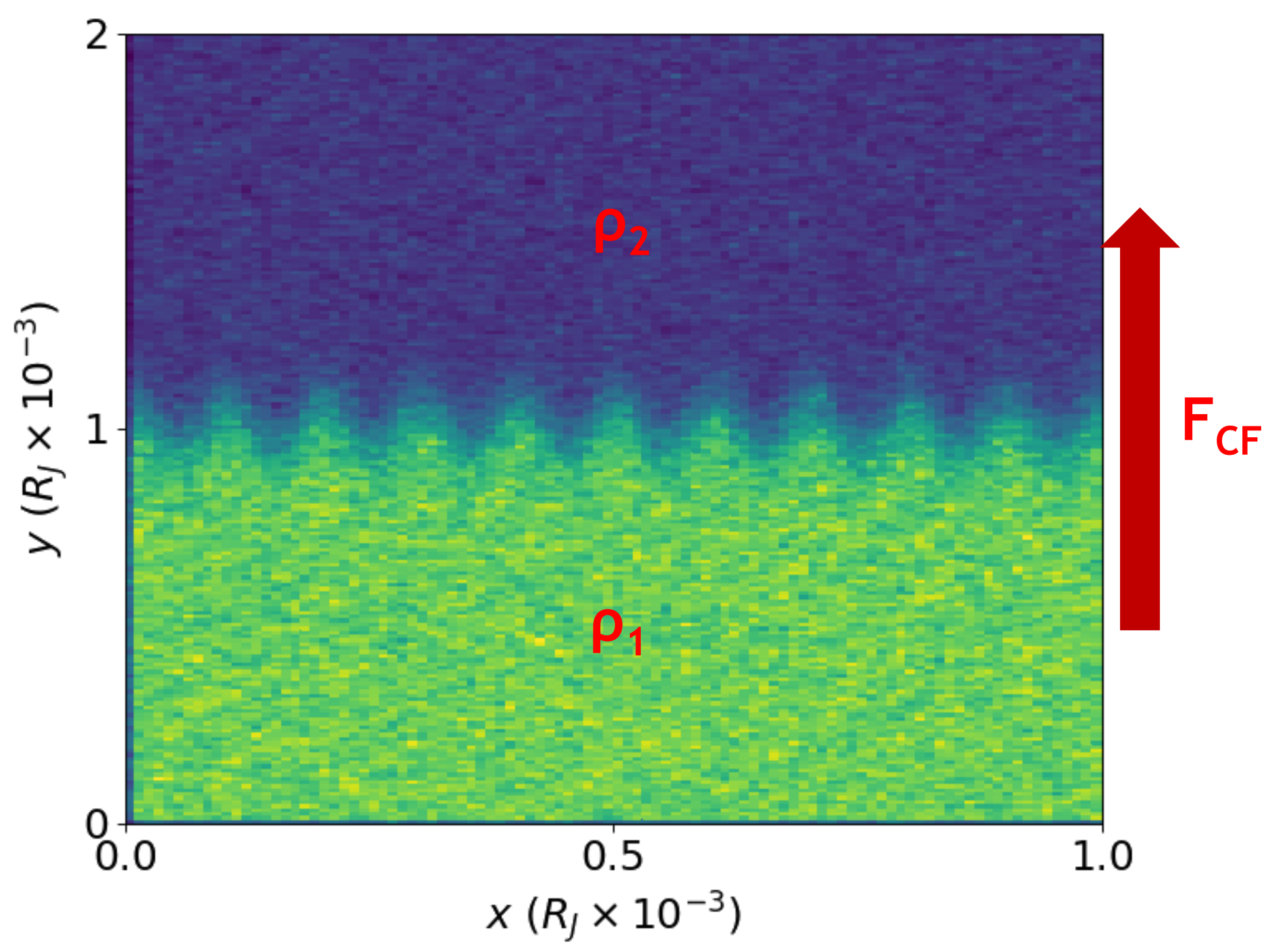


Python

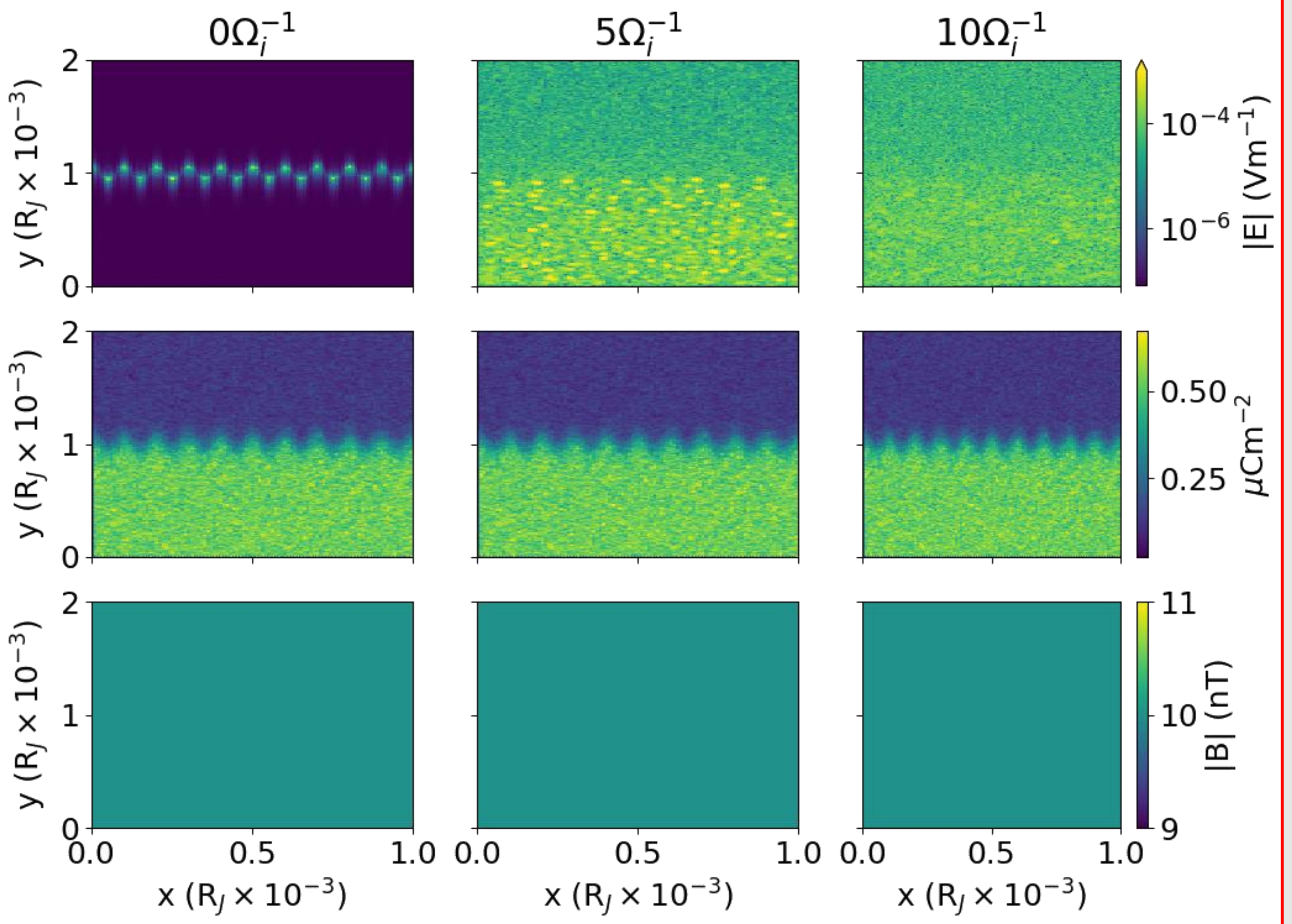


C++

4. Centrifugal-Interchange



In order to investigate the centrifugal-interchange (or radial-interchange) instability a 101x201 grid is constructed over a $1 \times 10^{-3} \times 2 \times 10^{-3}$ R_J domain with its origin placed at 10 R_J in such a way as to ensure that centrifugal force is pointed solely in the y-direction. The boundaries at the left and right of the domain are periodic (both for particles and EM fields) and the top and bottom boundaries are hard (reflecting particles and using a background value for EM operations). The domain has 2 surfaces placed on top of one another within it, each $1 \times 10^{-3} \times 1 \times 10^{-3}$ R_J, the densities of these differ with the bottom being heavier than the top ($\rho_1 > \rho_2$). The surfaces are filled with low temperature O^+ ions with 50 macroparticles per cell utilised. The interface between these is then perturbed with domain parameters utilised such as the usual 'mushroom-head' spike and bubble instability topology should be formed.



Currently, we do not see the expected evolution of the centrifugal-interchange instability in our simulated domain over the expected temporal scales. Rather, the perturbed interface has a tendency to relax to a more diffusion like density gradient rather than producing the spike and bubble pattern on temporal scales longer than presented above.

5. Future Work

Lookout for our AGU poster for further results!

- Continue investing parameters needed to produce centrifugal-interchange instability in simulated Jovian magnetospheric domain
- Couple magnetospheric plane of JERICHO to planetary ionosphere
- Parallelise c++ code base for JERICHO using MPI to decrease overall runtime