Reconnection-Intro

July 21, 2022

1 Motivation

In this module we will discuss current computational methods to modeling magnetic reconnection. This is motivated by the fact that many students studying this field are poorly taught the computational skills needed to succeed in this field. From more broad skills like using Git version control and plotting in Python, to more narrow knowledge of magnetohydrodynamic equations, we will aim to properly introduce these vital skills to students, taking care to cater to various levels of prior knowledge. (**figures demonstrating high performance computing**)

2 Background

2.1 Understanding Magnetic Reconnection

Magnetic Reconnection is a process when the magnetic topology is rearranged in the presence of a plasma. This causes a burst of energy in the form of heat, kinetic energy for electrons and ions, large scale plasma flow. When this occurs, the magnetic field lines snap and connect with the other field lines (*see the gif below*). Magnetic field linesare frozen into the plasma. Since plasma is a highly conductive material and field lines have inherent tension, (reconnection is intutitive due to tension, nontrivial since ideal plasmas cant reconnect)

Source: By ChamouJacoN - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=7496329

2.2 How does magnetic reconnection occur

Magnetic reconnection occurs with a pair of oppositely directed magnetic field lines. It occurs when the system experiences high heat levels of heat. A common example of this are coronal mass ejections and when the solar wind impacts Earth's magnetosphere. ## How do we model magnetic reconnection? (start with ideal ohms law, frozen in theorem, nonideal plasma cause reconnection We start with the generalized Ohm's Law

$$E + u \times B = \eta J + \frac{J \times B}{n|e|} - \frac{\nabla \cdot P_e}{n|e|} \times [\frac{\partial J}{\partial t} + \nabla \cdot (uJ + Ju - \frac{JJ}{n|e|})]$$

, where the left hand side comes from the ideal Ohm's Law, the first term on the right hand side is known as the resistive dissipation term. The $J \times B$ term is called the hall term, and P_e is known as the pressure tensor (**EXPAND**). The last portion of the equation is derived from electron inertia. In this computational demo we will start with the 5 moment simulation, which symplifies the pressure tensor down to a scalar. This is Eulers equations where the force is lorentz force (write eulers equations with general force, F=E+(VxB) which is 0 in an ideal plasma. In

generalized ohms law, there are other terms, called nonideal terms, that cause reconnection) Then we will expand the tensor out and compute the 10 moment simulation. Normally, only including the resistive dissipation term causes reconnection to occur far too slowly to only be attributed to the physics. Instead at that rate, the numerics of the simulation cause reconnection. This can be explored in the Apendix. ### 5 moment models As mentioned before, 5 moment models simplify the pressure tensor down to a scalar. The 5 moment model is in fact a generalization of the Hall-MHD model (taking Ohm's law until the hall term) which can be found from the 5 moment model by approximating the speed of light to be infinite. By simplifying the pressure tensor, the 5 moment model is significantly easier to run, but can be an inaccurate model outside of the reconnection zone.

3 Setup Procedure - Windows

3.1 Install WSL 2

- follow directions on https://gkeyll.readthedocs.io/en/latest/install.html#note-on-the-windows-linux-subsystem-wsl (Ubuntu installation preferred, the following instructions may not work on other distributions)
- If installing CUDA, use https://docs.nvidia.com/cuda/wsl-user-guide/index.html#cuda-support-for-wsl2
- Install common libraries with `sudo apt install build-essential`
- Run `sudo apt update`, `sudo apt upgrade` and `sudo apt autoremove` ## Install Conda
- (Recommended) If you want to run Jupyter Lab in WSL 2 on your local machine's browser, install a conda distribution if you do not have one
  - Either install Miniconda (lightweight, easier installation) or Anaconda (far more packages, useful if you also need more features for other applications)
    - Miniconda: first download the installer with the command `wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh`. Then run `sudo chmod +x Miniconda3-latest-Linux-x86_64.sh`. After the executable is finished, run `./Miniconda3-latest-Linux-x86_64.sh` and follow the prompts. When you have installed Miniconda correctly, jump to Install Jupyter Lab below.
    - Anaconda: follow https://docs.anaconda.com/anaconda/install/linux/ up to step 9 and jump to Install Jupyter Lab below.
- If you want to install the standalone application, follow directions on https://github.com/jupyterlab/jupyterlab-desktop#download, then jump to "Add WSL 2 terminal in JL" ## Install Jupyter Lab
- First, either open a new terminal or run `source ~/.bashrc`, then activate conda with `conda activate`
- Then update conda with,

```
conda update conda
conda update --all
```

- (Recommended) Create a new conda virtual environment with `conda create --name [NAME]` with a name of your choice. If you choose to do this, you must also run `conda activate [NAME]` with the same name as before if you want to use anything you will install in the next steps. If you create the virtual environment correctly, you should see **(NAME)** text before the terminal prompt. (**ADD IMAGE**)

- Then install Jupyter Lab and dependencies with

```
conda install matplotlib jupyter jupyterlab pip git
conda install -c conda-forge jupyter_contrib_nbextensions
conda update conda
conda update --all
```

- Then launch Jupyter Lab by entering `jupyter lab --no-browser` in the terminal, and copying one of the URLs listed at the end of the output into a browser on your *local* machine. (**ADD IMAGE OF JL**)

4 Setup Procedure - Macos

4.1 Install GkeyllZero

4.1.1 Add WSL 2 terminal in JL

Run `jupyter-lab --generate-config` Open generated file and add `c.NotebookApp.terminado_settings = { 'shell_command': ['C:\\Users\\d3mon\\AppData\\Local\\Microsoft\\WindowsApps\\ubuntu.exe'] }` to the end (change path) and restart JL

4.2 Install JL in WSL2

https://towardsdatascience.com/configuring-jupyter-notebook-in-windows-subsystem-linux-wsl2-c757893e9d69

4.3 Using Git Version Control

4.4 Cloning and setting up GkeyllZero

- Follow instructions on https://github.com/ammarhakim/gkylzero
- Make sure that you can run configure with CUDA if installed
- Test installation with `lua_gem_g0.lua` file

[ ]: