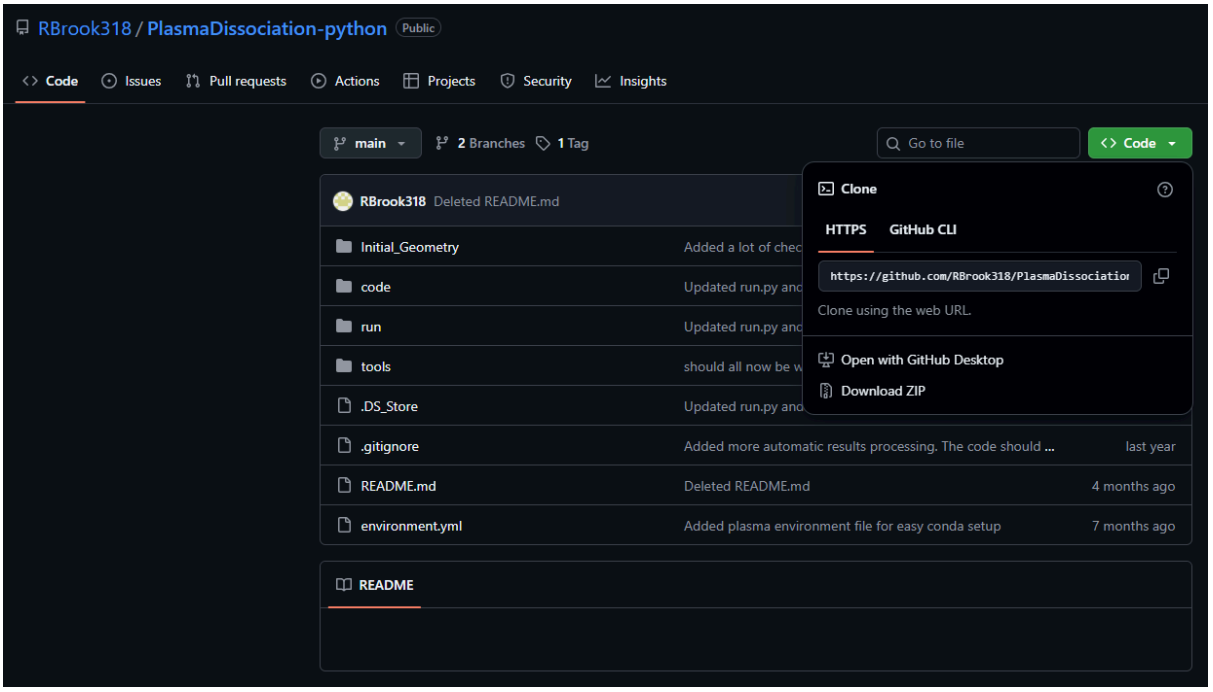
There should be 4 main steps to setting up the code.

1. Copying from Github
2. Setting up the conda environment
3. Obtaining the QChem tarball
4. Running the code itself

IMPORTANT LINKS:
DROPBOX:
https://www.dropbox.com/scl/fo/zw54csla9fhi13zq5ybsx/AFg6yqDp4l29KNjEhMMrgRU?rlkey=u8wo5ibxxs55qz1xhc3a503zh&st=b227u2hd&dl=0
GITHUB:
https://github.com/RBrook318/PlasmaDissociation-python

Copying from GitHub

The code itself can be found in the github repository (https://github.com/RBrook318/PlasmaDissociation-python)



This HTTPS clone link can then be copied to the clipboard (with the green code button and the copy button). Once you have logged into aire you can then use the command

```shell
git clone https://github.com/RBrook318/PlasmaDissociation-python.git
```

Which should copy the entire repository into your home directory of Aire.

Setting up the conda environment

The conda environment should be able to be built easy from the environment.yml file that is in the shared plasma folder. Just copying that into your home directory in Aire and typing these 4 commands should be sufficient. This only needs to be done once. After this, the environment can simply be activated.

```Shell
module load miniforge
conda activate base
conda env create -f environment.yml
conda activate plasma_env
```

Running the code itself

Once everything has been properly installed, running the code should be as simple as running two commands.

```Shell
cd PlasmaDissociation-python/run
python run.py
```

It is worth noting that the conda environment will have to be activated every time you relog into the HPC before running the code again.

```Python
module load miniforge
conda activate plasma_env
```

The only thing that should be done before submitting a job is checking inputs.json will give you the trajectory you would like. For instance a good test for the code is to run something small and very cheap like methanol or even methane.

```JSON
{
    "HPC":{
        "Runfolder":"Methanol-test",
        "repeats":1,
        "cores":8,
        "initre":0,
        "hours": 48
    },

    "Molecule_data":{
        "Geom_flg": "PubChem",
        "Molecule":"67-56-1",
        "States":1,
        "Multiplicity":3,
        "Start_State": 1,
        "Temp":5000,
        "Cutoff": 100,
        "Geom_file_start":1
    },

    "Elec_structure":{
        "method": "QChem",
        "Spin_flip":1,
        "Basis":"6-31+G*",
        "GPU": 0
    },

    "Propagation":{
        "Branches":0,
        "Timestep":10,
        "Tot_timesteps":2100,
        "Checks": 1,
        "Remove_atoms": 1
    }

}
```

The only things that are really changed between runs are the variables:
-Runfolder: Name of the runfolder created in your scratch directory
- Number of repeats

-Geom_flg: which can be ‘Full’ to run inputs of molecules in the style of Dima’s code, ‘Initial’ where it shall read in coordinates from the folder of initial_geometries or ‘PubChem’ where it will search PubChem for the code given.

Temperature: Normally set to 5000

Processing of results

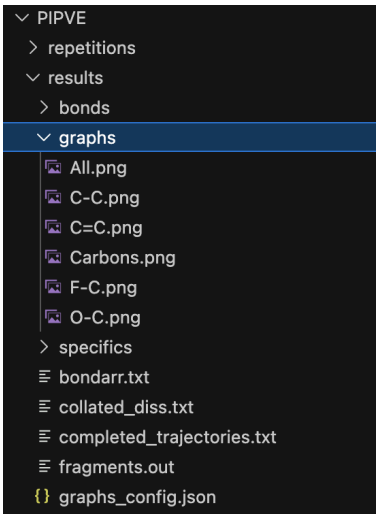
Once a job has been submitted the queue can be checked with the command

```shell
squeue --me
```

Which will print an output akin to

```shell
     JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
       3250536_1     nodes Plasma_t   cm18rb  R       2:11      1 node033
```

Where there will be a job number x_y for every y trajectory you are running. If the ST does not say R, then the job is currently not running, a status of PD means the job is currently pending and is sat in the queue. Once sufficient time has passed for a couple runs to complete it is worth going to your scratch directory where you can find the run folder. Inside this runfolder



Inside repetitions will be folders rep-1,rep-2 etc. which will contain checks and the output of individual trajectories. The results folder will be updated every time a trajectory finishes. The

important files to check on while there are still trajectories running is the completed_trajectory.txt which will have the number of finished runs. The graphs will also be updated so checking those can give an idea of which bonds are breaking quickest.
The bondarr.txt is also a really important file as it contains which atoms in the geometry are bonded to which others and via what bond type. The specifics file contains information about which bonds break first, second, and what bond breaks most frequently after another.