**Mcrogrid simulation and data extraction**

Following on the steps mentioned in chapter 3 above, we created a matlab simulation to extract real simulated data to use in our python environment. We implemented the system for the three locations and recorded a full year’s worth of data for each location. We first simulated a Wind and battery generation connected with our load for the three locations. This shows a 1.5MW generation network with varying generation along a single day “a single figure shows a 75 day period” which shows that for each location there is enough variation in generation for trading to be viable

INSERT 3 FIGURES

The next show a generation network of Solar battery and load at (where is the 200KW load) which shows a the same change in daily generation and also provides a viability to trading.

INSERT FIGURES

The data we got from the simulations were solar and wind generation for a year at one hour intervals, below is a sample of the extracted data for the month on (which is better) at (loc).

INSERT FIGURE FORM DATA

We simulated a fully working microgrid with all distributed generation connected i.e Solar, Wind and the batteries at (loc) which was able to sustain the load for the same 75 day period

INSERT FULLY WORKING MG FIGURE

**RL Simulation and Algorithm Performance**

We used the data we achieved from our microgrid simulation and used it in the designed environment detailed in chapter 3 above. To ran the simulation for a central microgrid at location (location). Before listing the results we need to detail the metric used for the algorithms.

Reinforcement learning works on the notion of a reward for it’s actions; it is the value the policy tries to maximize and it is the metric we used here. Reward design is a major part of RL, designing a good reward that pushes the agent in the direction of the desired action is a design and configuration problem. We tried and optimized rewards to achieve a stability in the grid and to aim to achieve profit from the trading interaction at each point. The reward is a function of total generation, amount of energy stored at battery, total load, network price, unit generation price, trading price, trading amount and the type of action:

If action is to buy:

If action is to sell:

If action is to hold:

This reward pushes the agent to stabilize the load and generation causing no power outages in the network, we set the reward to -10 if after any action the load is unstable, that is load is greater than generation and battery.

Now that we have detailed the reward to use, we ran the algorithms of choice at 50 epochs each, each epochs being 4000 timesteps which is the equivlent of a year in the data obtained from the matlab simulation. When running PPO for the first run, with matching configuration to the matlab simulation at (place) we achieved these results

PPO with wind

This is a per time step average reward of , a clear indicator that at each timestep the grid is achieving profit from trading and never reaching zero reward meaning that the grid is always stable. This good result introduced an opportunity for reducing some of the generation to provide a more challenging environment to the algorithm, so we decided to remove the wind generation form the microgird and trying to stabilize a solar based microgrid.

The solar only generation microgrid was a more challenging configuration for the microgrid but the algorithms were able to achieve the next results. The results of the algorithms in terms of average, maximum, and minimum episode return and average, maximum and minimum timestep return is shown in the table below:

Table of comparison

3 places with 2 algorithms