**Python Code Documentation**

* Visual Studio CODE

Need to create and activate the virtual environment in comment line environment (cmd)

Then activate the virtual environment

(.venv) C:\2021\Sloan Project\ABM\_Model\SloanABM\.venv\Scripts>

Then type “ activate”

Then you can install the package by typing: pip install package\_name

* Main program for calibration: MainScript\_price\_predict\_Validation\_0505\_random\_search.py

Monthly demand and price data at load zones are retrieved every year to generate supply curves for installation decision-making.

* Renewable agents perceived different (and lower) costs and Renewable Energy Credit (REC, a proxy for the financial incentive) for renewables and non-renewable agents (NG) also perceived different (lower) costs but received no RECs.
* Agents are assumed risk-averse. That is, when agents observed an opportunity for investment (difference between their predicted capacity based on the demand forecast and supply curves and the current generation capacity in the market), they would invest only a fraction of the capacity needed (hesitation).
* The calibration processes:
  + Select initial values for Optimal REC, Cost Perception (mu, std of the distribution), and Risk Attitude (mu, std of the distribution).
  + Varies the REC and select the best REC value based on the KGE values (comparing historical data and simulation results)
  + Varies cost perception distribution parameters (mu, std): choose the best values based on KGE values of the simulation
  + Varies risk attitude distribution parameters (mu std): choose the best values based on KGE values of the simulation.
  + The results are stored in \Sloan Project\ABM\_Model\SloanABM\00 Results\Calibration\, and the final calibration results are in the Calibration\_May-04\Best\_case\
* Main program of the future simulation (baseline: MainScript\_Future\_Scenario\_0517\_no\_change\_baseline.py)
* Costs of the generation technology are assume the same at the costs in 2020
* Optimal REC, Cost Perception (mu, std of the distribution), and Risk Attitude (mu, std of the distribution) are inputs from the calibration
* Demand data is retrieved from ERCORT’s forecast
* Market\_Class.py:
* Market(obj): this defines a market object (class) – many of the functions can be integrated into market object. However, in its current form, it only serves as a data reader
* Read\_Future\_Data(obj): read future data
* generater(): this is a generator class.
* Functions: below are the functions for evaluating an investment
  + Pay\_back\_period(…): calculate the payback period of an investment. I did not use this function since it is not included in current ABM
  + NPV(…): this converts the investment into net present value (NPV)
  + Evaluation(…): this is the main function that evaluates the NPV and IRR of investment in a generation technology.
* ABM\_function.py: The functions needed to simulate the ABM are defined here, excepted for the evaluation. Originally, I planned to define classes in the Market\_class.py and functions in this ABM\_function.py. However, as the project is still under development, I haven’t had the time to review and consolidate the codes.
* Some of the functions are legacy functions that are not called in the calibration and future simulation.
* Supply(…): This function utilizes supply curve to determine future market price.
* ZoneInvest(…): determine the generation technology with the highest IRR
* Agent\_Investment(…): determine the amounts of investments in the load zones. If more than one Load zone (LZ) has IRR higher than the threshold, agents will invest in those LZs proportional to the IRR associated with them (for example, two LZs. have IRRs 0.08 and 0.08, and the required capacity is 100 MW, the agent will invest 50MW in each tech.)
* Aggregation\_fuel(…): aggregate the results by fuel type.
* Price\_predict.py: Predicting future price
* hist\_price: read historical demand and price data and generate a linear regression model
* future\_price: read future demand and price data and generate a linear regression model
* Generate\_future\_data.py: This code generates future demand, cists, and price data

**Preliminary Results**

Calibration: \Sloan Project\ABM\_Model\SloanABM\00 Results\Calibration\Calibration\_May-04\Best\_case\