# Simulation Plan:

You can click to any of the ones that are completed/ bolded.  
This plan is not fixed, and can be altered according to schedule or other external events.

[**1. Simple Simulation: 1 EV, 1 CS (1 port), Grid, Solar Power.**](#_1._Simple_Simulation:) **(COMPLETED, HAVE MISTAKES WHICH ARE FIXED IN MULTIPLE EVS SIMULATION)**

[**2. Multiple EVs and Charging Stations.**](#_2._Multiple_EVs) **(COMPLETED)**

[**3. Implement any Optimization Algorithm to the Simulation.**](#_3._Implement_any) **(COMPLETED)**

4. Look into existing papers and study how they forecast the load and solar power. Implement that to the simulation.

5. Implement any DRL Algorithm to the Simulation. (This simulator support RL gym environment, so I want to play around with it in this part).

6. Compare the Performances of these 3 Algorithms (Uncontrolled algorithm, Optimization algorithm, DRL algorithm).

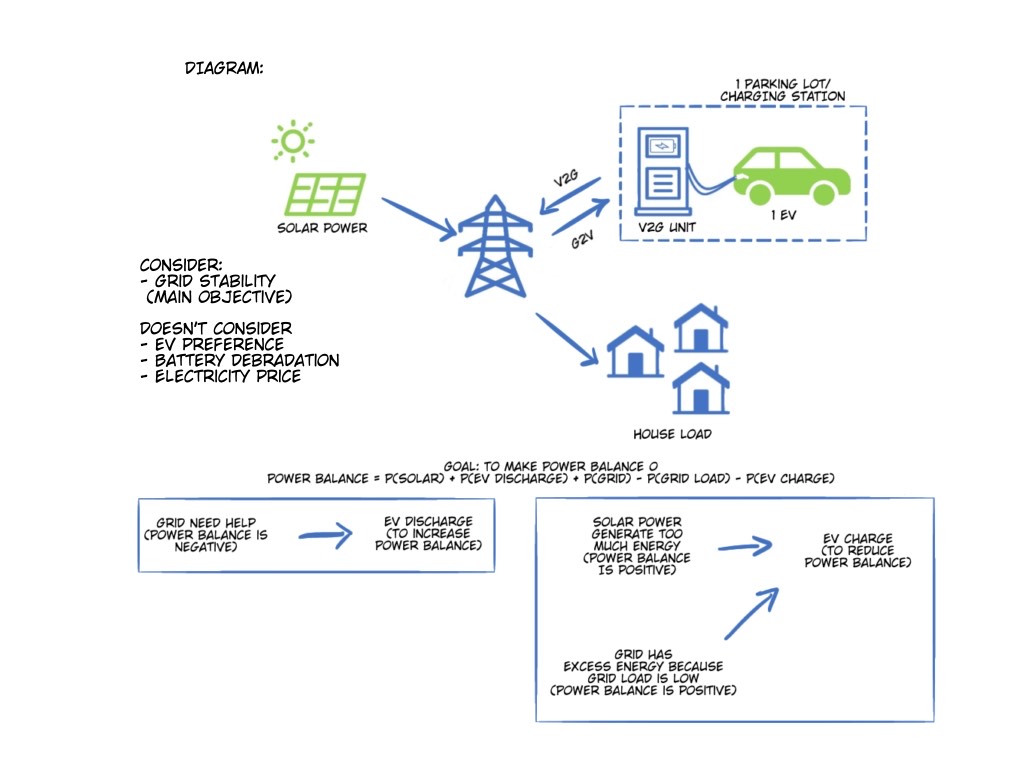
7. Battery degradation. In this part, the goal is to experiment with various battery models that have existed, as well as look into the battery degradation model that has been included in this simulator.

8. Look into carbon aware emission metrics. The goal is to figure out how to keep track the amount of carbon these EVs can save compared to traditional cars as well as compared to uncontrolled charging.

9. Try to replicate an algorithm from a state-of-the-art paper (one of the papers that I have discussed with Dr. Dande).

# **1. Simple Simulation: 1 EV, 1 CS (1 port), Grid, Solar Power. (COMPLETED)**:

In this simulation, there exist **1 EV, 1 charging station with 1 port, grid, load and solar power.**

The grid here does not only generate electricity through solar power, but also generate power through traditional means, which I call traditional power.  
The goal is to achieve power balance: supply == demand  
The diagram below detail the outline of the simulation:  


**My thoughts after finishing the simulation:**

I utilize the **EV2Gym simulator.**

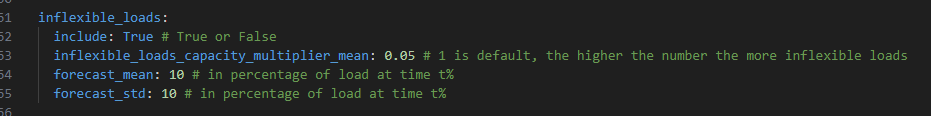
The result is: successful

In this simulation, I mention the use of traditional power, power generated through other means aside from solar power. If there are excess power from these traditional power + solar power, EV will charge. But in actuality, **what we want is to minimize the use of this traditional power, and maximizing the use of the solar power.** So, in the next simulation, for the power balance equation, I won’t be including traditional power generation, as this electricity generation is produced real time to meet the demand. **What we want here is carbon neutrality**, meaning: solar power – load – EV charging + EV discharging = 0.

**Simulation setting:**

- The load, solar power and traditional power generated (power generated not using RE) by the grid **are all provided by the simulator** through the probability distribution that they have created through numerous datasets they have gathered. This also apply to the departure and arrival time of the EV.

- I want to note that it is possible to control these through a config (yaml) file. However, I find them to be hard to control, so there is a huge change that I will modify this or find my own dataset for the grid as well as the load. The config file I am referring to (just a part of it):



What do I mean by hard to control? I use a multiplier of 0.05 above, and it leads to some runs having a “scale negative” bug. My guess is that the minimum value for this setting is supposed to be 1. I put 0.05 here so that I can try to showcase my EV charging. Because if I put it as 1, then the load is too high, forcing the EV to not charge or discharge.

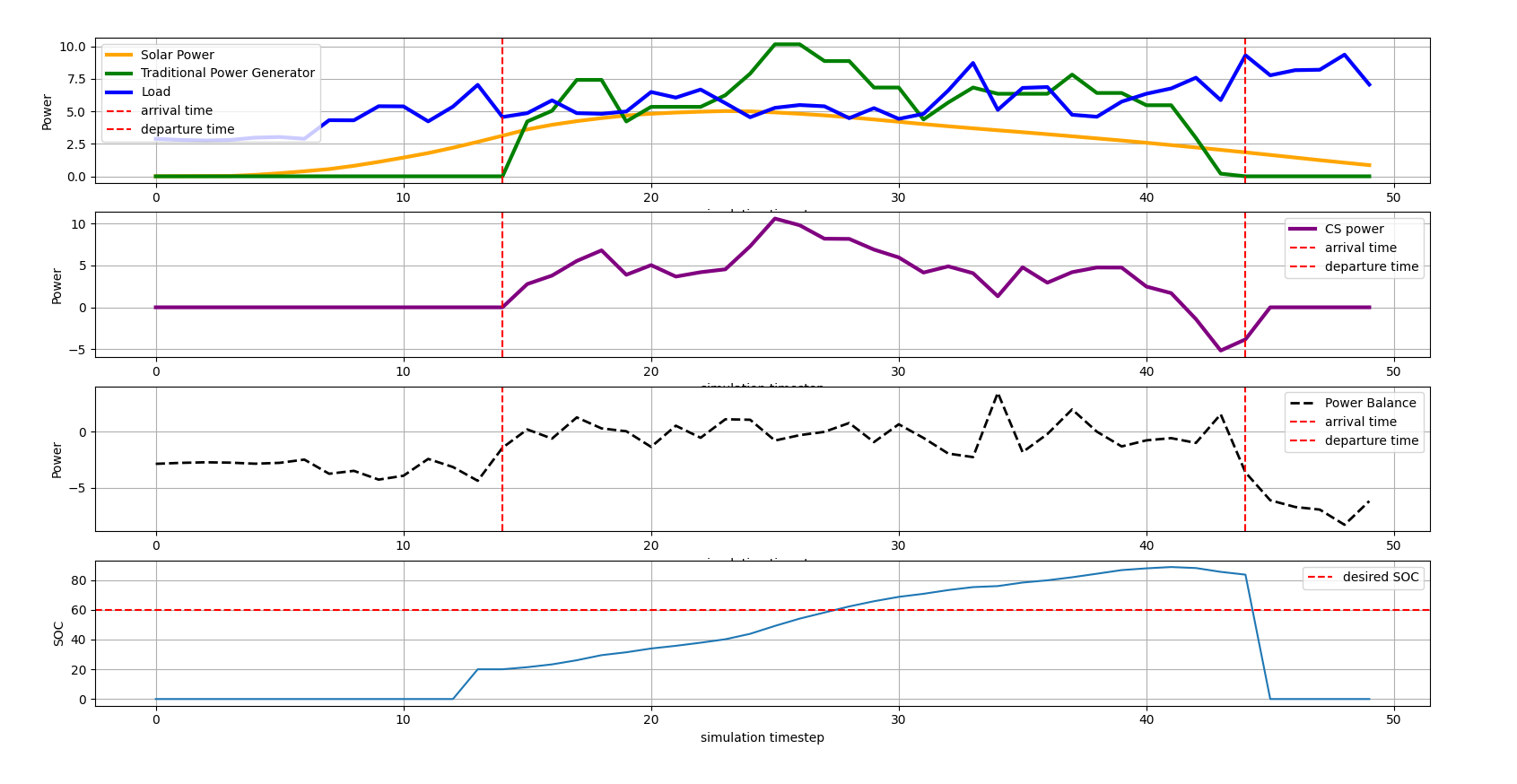
- Furthermore, **the traditional power generated by the grid uses the simulator’s generate\_power\_setpoint function in utils.py**. In actuality, the generate\_power\_setpoint is not used to generate traditional power**, I just use it as a placeholder for now.**

- The **forecasting** of load and solar power for the current timestep is not done through fancy algorithms. It is just using the actual data from the previous timestep.

- In this simulation, **EV’s desired SOC is considered. If the EV’s SOC < desired SOC at the current time step, it will not discharge.**

**-** I do want to point out that there is a bug in their “desired\_capacity” when they spawn the ev. For this simulation, I have fixed it

**Simulation Result:**

  
- The red line is the arrival and departure time of the EV.

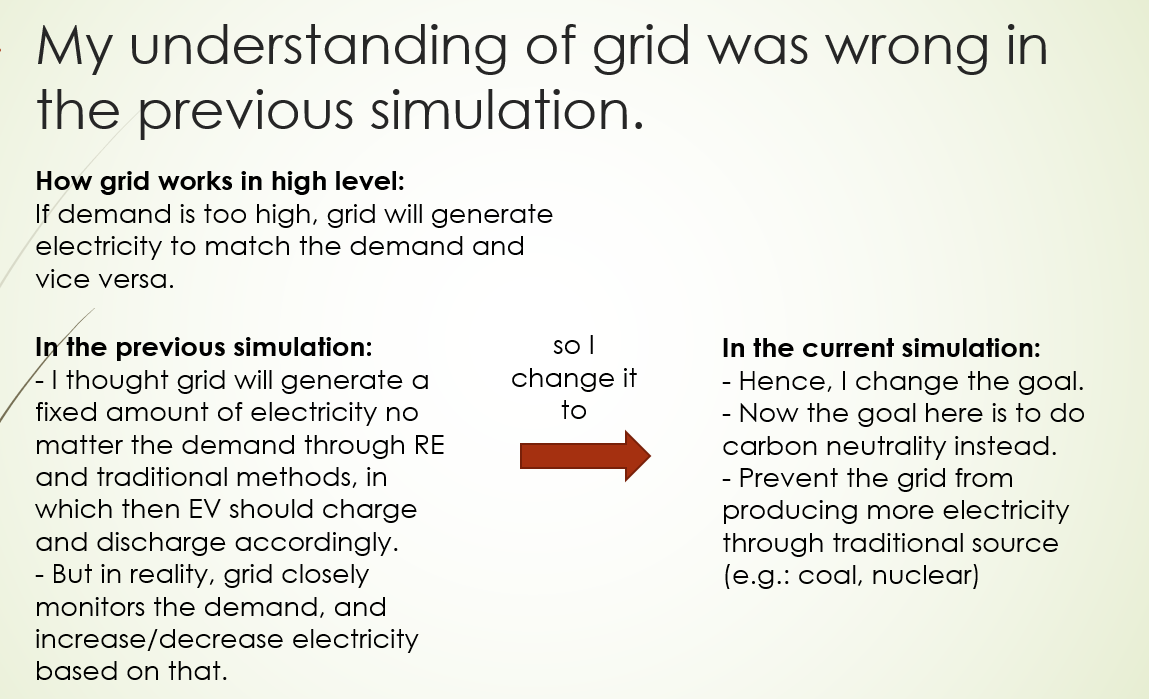
- The second figure (purple line) shows the charging and discharging of EV. If it is below 0, it is discharging.  
- When the EV arrive, the power balance is much closer to 0 compared to the timestep when EV was not there. This shows that EV helps in grid stability.

- V2G is successful as seen in timestep 43. The CS power is negative as EV is discharging to help the grid, because the load is higher than the traditional power and solar power combined.

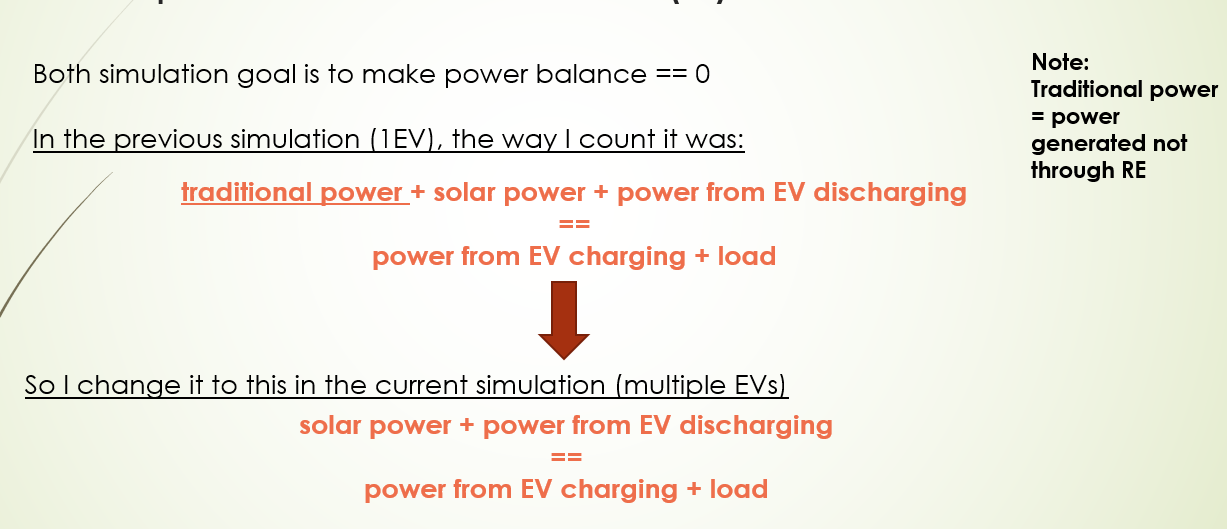
# **2. Multiple EVs and Charging Stations. (COMPLETED)**:

- This simulation involves multiple EVs, multiple CSs, solar power and grid.

- I decided to change the definition of the power balance because:



- Therefore:



- This mean that:

* Power balance = Solar power – Load – sum of EV power
* If the power balance is 0, it means supply == demand without relying on the grid to generate electricity through traditional methods.
* If the power balance is < 0, it means it will need to rely on the grid generating electricity through traditional methods.
* If the power balance is > 0, it means it waste the solar power.

**Simulation settings:**

**-** Multiple EVs. Charging stations: 10

**-** EVs are spawned using the **“workplace” probability distribution** that the simulator provided. (This determine when the EV arrive and depart the CS). Multiplier for this is set to 2.

**-** Simulation runs on the weekdays (5 am, 90 timesteps, 1 timestep is 15 mins)

**-** Solar power and inflexible load still use the distribution the simulator provides as well.

**-** Consider EV’s desired SOC and won’t discharge below that level.

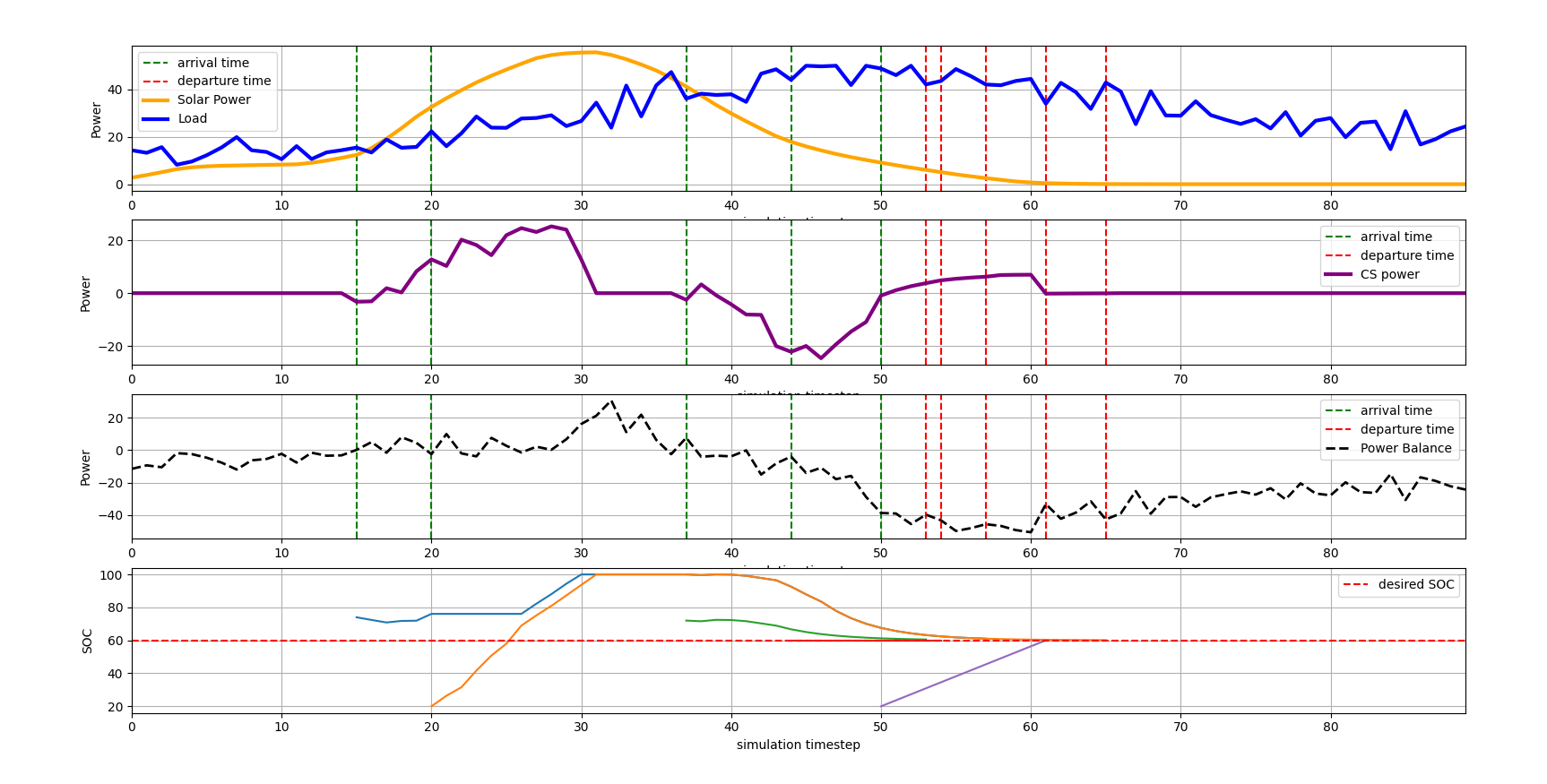
**Simulation algorithm:**

the algorithm of this simulation is simple

* If solar power is greater than the load, charge the EVs with the remaining available power (solar – load).  
  **Prioritize EVs that has its SOC lower than the desired SOC** (Distribute the available power to the prioritized EVs’ first, **equally**)

**The remaining power from charging prioritized EV will be once again distributed equally to the remaining EVs** (EVs that has SOC >= desired SOC)

* If load is greater than solar power, **EVs that have enough SOC will discharge**, **while EVs that has SOC lower than their desired SOC will charge with a power as low as possible (power just enough to reach the desired SOC until they depart)**  
  Of course if EV's SOC is the same as their desired SOC, they will not charge/discharge, basically idle

**Simulation result:**

1st plot: solar power generation and load  
2nd plot: total CS power (negative means discharging > charging, and vice versa)

3rd plot: power balance

4th plot: SOC of each EV

- **Green line** indicates that there is an EV arriving to the CS, and **the red line** indicates that there is an EV departing from the CS.

- We can see that when the solar power is greater than the load, **the EVs are charging** (timestep 20-30)

- On the other hand, when load is greater than the solar power, **EVs are discharging**, and it won’t go below the desired SOC (time step 50 – 60, yellow line)

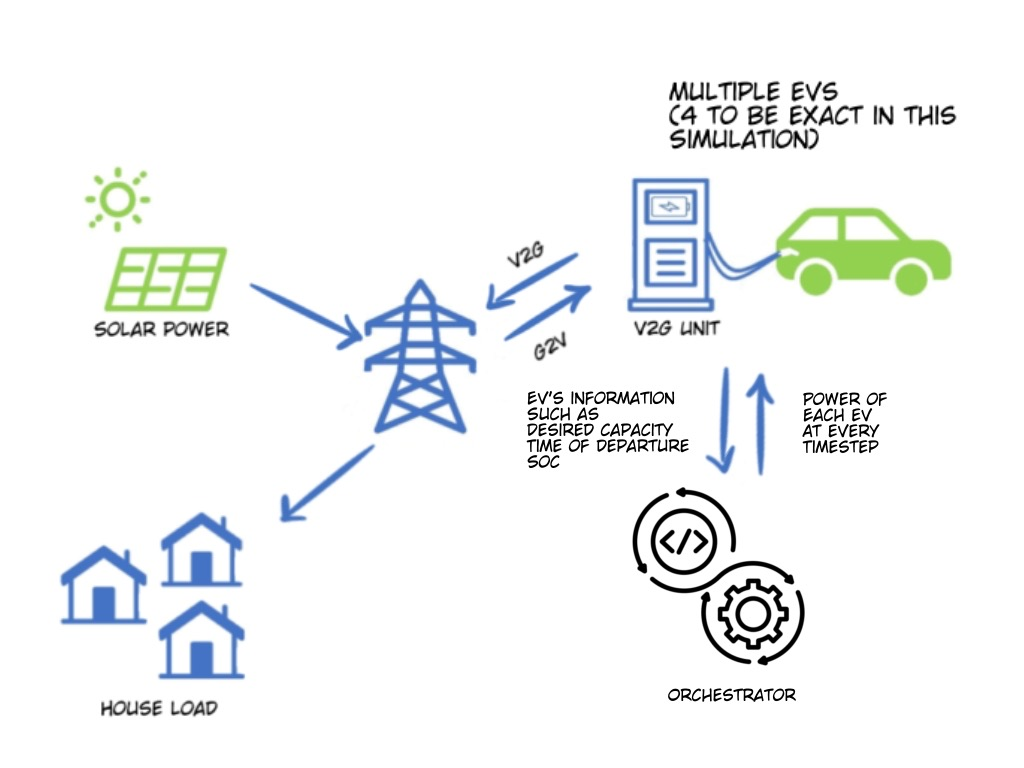
- It is also noted that when **EV’s SOC is below the desired SOC, it won’t discharge**, as indicated in timestep 50 to 60 (purple line, plot no 4).

- This algorithm **guarantees that the desired SOC of the EVs are met**, and as a result, even when load is greater than solar power and power from EV discharging combined, the purple EV will still charge, but with the lowest power possible to meet the desired SOC (indicated by the purple line in plot no 4 in timestep 50 to 60).

- In figure 3 (the dotted black line), indicate the power balance. **When EVs are present such as in timestep 20-35, the power balance is closer to 0**, which mean that it is able to utilize RE well and prevent the grid from using traditional power.

# **3. Implement any Optimization Algorithm to the Simulation (COMPLETED)**:

**Simulation Diagram**



This simulation is similar to the multiple EVs simulation; however, it will be using **MPC (Model Predictive Control)** to solve the problem.

- MPC means determining the best action/control not over one timestep, but over a specified time horizon (known as control horizon).

- **Why MPC?** Because it considers future behaviors as well (considers EV departing time, future load, future solar power)

- This mean that I need to **have a model to forecast the load and solar power** for that specified time horizon.

- For this simulation, I decided to use **SARIMA model** (Seasonal Autoregressive Integrated Moving Average) to forecast the load and solar power. The reason behind why I pick SARIMA is because 1) when there is new data coming in, it doesn’t need to fit the model again 2) I took a time series class this semester, and wanted to use the new knowledge I have.

- The simulation I am doing is taking on a weekday. Hence, I use the previous 5 weekdays to fit the **SARIMA model**.

- Regarding the Optimization problem I have, I will be outlining the objecting function and constraint.

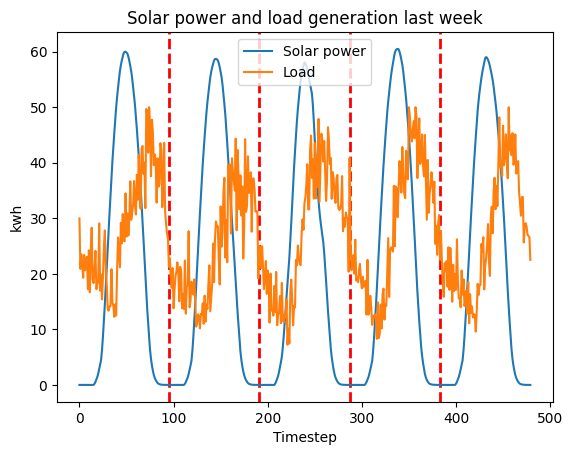
- Also, the goal of this simulation is still power balance. **(For more explanation about power balance, check page 4)**

**Forecasting**

Forecasting model requires past data in order to predict future data.

I want to point out that this simulation takes place on a weekday (July 4, 2022) and on a workplace environment.

Hence, for the data needed for the forecasting model, I take the previous 5 weekdays. It looks like this:



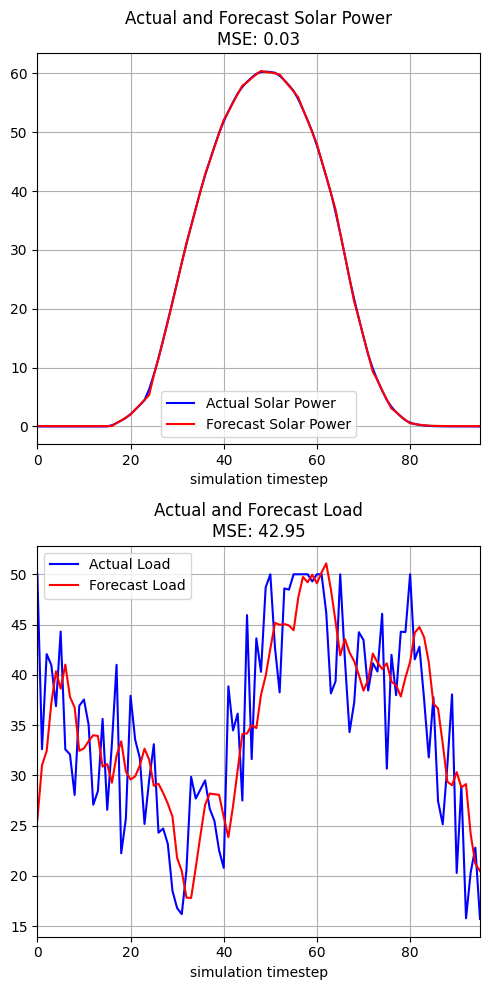
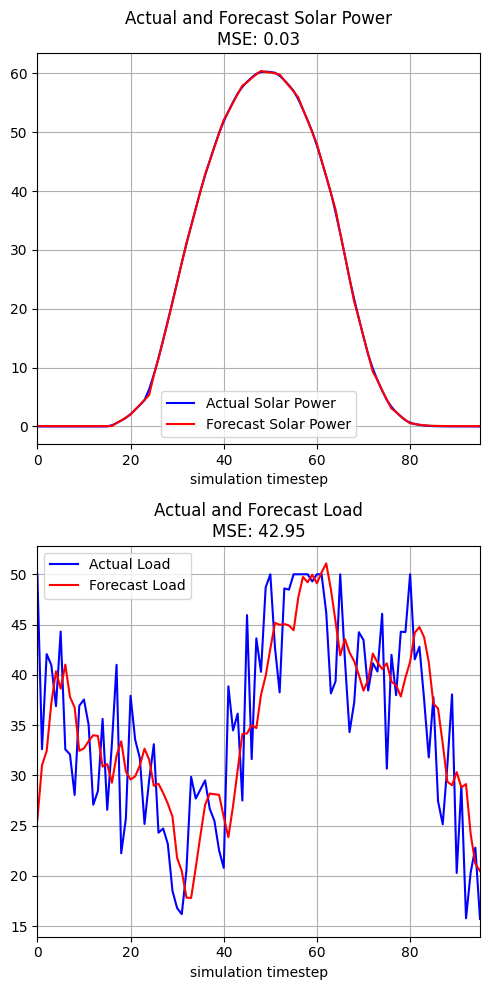
In order to determine the parameters (p,d,q) x (P,D,Q) of the SARIMA model, I analyze the data above (to be more exact, their autocorrelation and partial autocorrelation).

The final exact forecasting models are:

solar power – ARIMA(5,0,0)

load – SARIMA(4,0,0)x(1,0,0)96 -> 96 here means 96 timesteps which is equivalent to 1 day.

**Forecasting Result**

****

\*note: MSE refer to mean squared error.

- In every timestep, the forecasting model will predict the next k horizon. For each timestep, we will take the first prediction and plot it. This is the forecast value.

- From the above graph, the prediction for the solar power is very well done, but the prediction for the load is not the best. I will look into other forecasting models in the future and see if there is any better alternative.

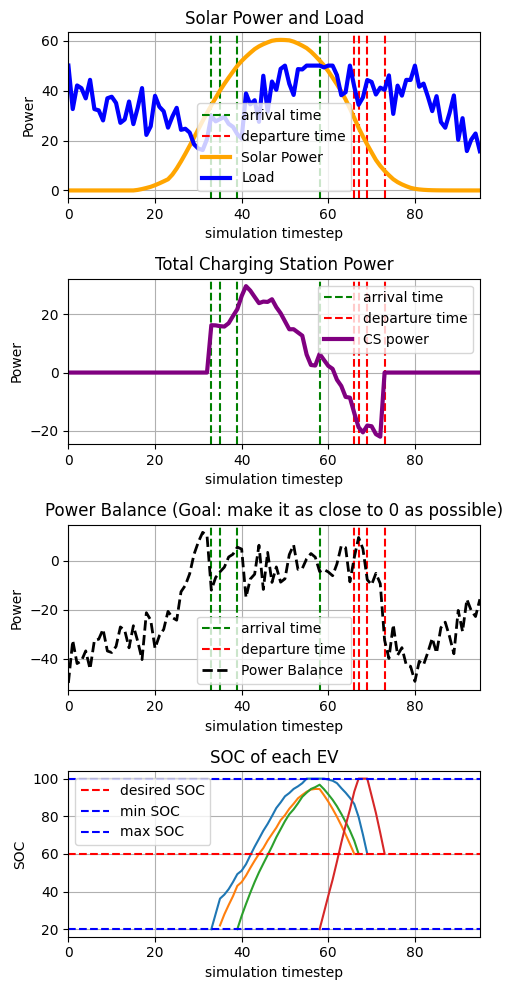
**Objective Function**

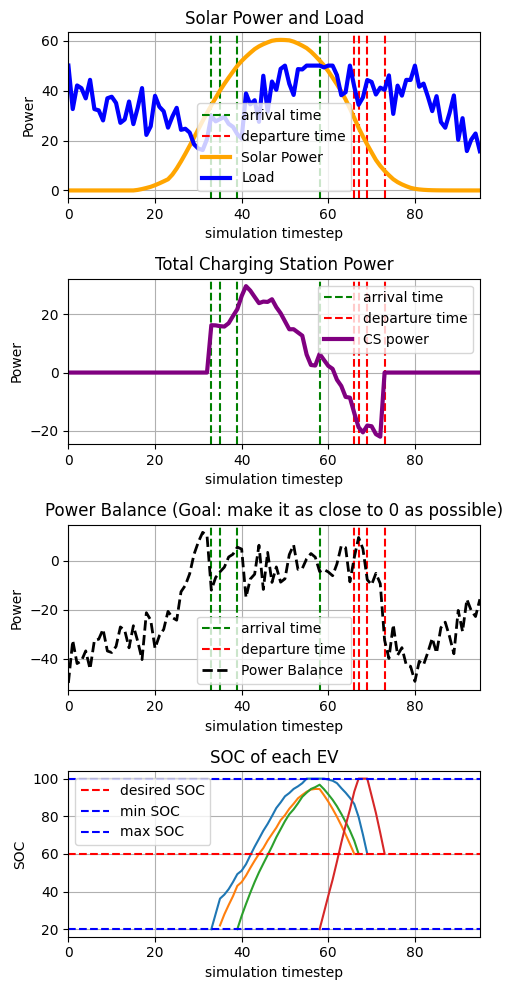
**Explanation:** The objective function is to make power balance as close to 0 as possible.

**Constraints**

**Explanation**: The first constraint ensure that it will not go below the minimum and maximum power of the charging station. The second constraint ensure that it will not go below the minimum and maximum SOC set by the EV user. The third constraint is to try and make the final SOC of the EV user to meet the desired SOC the EV’s user has set up.

For the third constraint, I call it **estimate\_end** because the final SOC is just an estimate. It's hard to know the actual SOC when the EV departs since real battery behavior isn’t linear. For example, batteries can charge faster at certain SOC levels or under specific conditions. I’m using a linear model to keep things simple, but it’s not a perfect match for how things work in real life.

**Result**

****

- When solar power > load, the EVs will try to absorb it for future use (future use means discharge when load > solar power to make power balance == 0)

- When load > solar power, the EVs will discharge in order to make the power balance == 0. This way, we help in saving carbon emission as this will prevent the grid using electricity generated from conventional power plant.

- Looking at the SOC figure, we can see that the red line (I will call it red EV), discharge later compared to other cars. This is because it considers all the cars departure time. It knows that at around timestep 70, there will already be no EV left except for red EV. So, before that come, it charges red EV in order to prepare it for timestep > 70 (because it will be the only EV available to help the grid).