In this assignment, you will explore the application of reinforcement learning techniques for the control of a battery energy storage system in the real-time electricity market. Similar to other assignments, the tasks involved require a degree of mathematical modeling, implementation in your preferred programming language (preferably Python), the generation and discussion of results, as well as the presentation of your work in a concise report.

The aim of Assignment 3 is to evaluate

- Your comprehension of the fundamentals of reinforcement learning,
- Your capability to implement basic reinforcement learning techniques, specifically value iteration and policy iteration,
- Your critical analysis of the outcomes obtained.

The expected outcome of Assignment 3 includes:

- a report of maximum 10 pages (excluding appendices),
- code delivered as supplementary material.

The evaluation of Assignment 3 will count for 33.33% of the final grade. **Individual contributions to the assignment must be clearly stated in the report.** If not, equal contribution will be assumed. Good luck!

Description of the assignment:

In this assignment, you will explore the utilization of the learning techniques presented for the real-time control of a battery. The primary goal is to maximize the profit for the owner of the battery while considering electricity prices and the state of charge (SOC) of the battery. For simplicity, let us assume instead of day-ahead and intra-day markets, we have a continuous market which is cleared on hourly basis in a continuous manner, and you are able to bid in real-time, i.e. the electricity price of the current hour is known but not of future hours.

For the hourly electricity prices, you can utilize the price dataset attached to the assignment and pick your desired zone. Note that similar to other assignments, you can partition the data into a training dataset for the learning process and a testing dataset for evaluation purposes.

The battery possesses the ability to charge, discharge, or remain inactive during each time period. Let us consider a battery with a total capacity of 500 MWh, featuring charging and discharging capabilities of 100 MW per hour. Consequently, when the battery is at a state of charge (SOC) of 0, it can be fully charged within a span of 5 hours. It is essential to note that at an SOC of 0, the battery cannot discharge, and at an SOC of 500 MWh, it cannot be charged.

Step 1: Problem Formulation as a Markov Decision Process (MDP)

Define the problem within the framework of a Markov decision process (MDP), specifying key components such as the state space, action space, and reward function. To maintain simplicity, consider employing a discrete action space.

Step 2: Discretization of State Space and Transition Probability Estimation

Assume a discrete state space for this step, even if some state variables were initially continuous. Apply discretization, and estimate state transition probabilities.

Step 3: Implementation of value and policy iteration algorithms

Having acquired a model for the MDP, use the value iteration and policy iteration algorithms to solve the MDP with the obtained transition probabilities. Analyze and compare the optimal policy and its corresponding value obtained from the algorithms.

Step 4: Optional - Performance Evaluation and Comparison

Optionally, propose a method for evaluating the performance of the learning algorithms and conduct a comparative analysis of their effectiveness and efficiency. What metrics or criteria can be used to highlight the performance of the algorithms?

Evaluation Metrics:
- Total profit obtained during testing.
- Convergence rate (number of iterations for optimal policy/value).
- Computational time.

Comparative Analysis:
- Use different lengths of dataset (2023, 2022-23, 21-23)
- ~~Analyze sensitivity to different price volatility levels or SOC initial conditions.~~
- ~~Compare with simple heuristic~~
- Compute the rate of change in value iteration
- Compare the cumulative profit from policies found during iterations
- Expand price states and compare

Optional Extensions:
- Include inefficiencies (e.g., round-trip efficiency of the battery).
- ~~Add randomness to test robustness of the algorithms.~~
- Incentive for specific action in reward