Reinforcement learning project: wind farm optimization

Nicolas Gast

October 3, 2022

Reminder: plagiarism is illegal.

1 Introduction

For this project, you work for a wind-farm company and you need to design the control algorithm of a wind turbine. The wind turbine is located in a environnement in which the wind is hard to predict. Your goal is to decide in real time of the angle of the turbine in order to maximize the energy production while minimizing rotation costs. There are two parts in the project:

- In the first part, we consider a simple stochastic model for which we know all parameters. You need to solve this model by using the theory of Markov decision processes.
- In the second part, you will use a simulator provided by the company *Eol Robotics* `https://github.com/paulaubin/wind_turbine_env`, and you need to implement RL-based algorithms to solve the optimization problem.

The objective of this project is to make you familiar with the tools that we learn in the course (MDPs and RL algorithms), by applying them in a realistic example.

2 Scientific work

In most of the project, you control a single wind turbine for which you can control the angle of the turbine. We assume that time is discrete at at each time instant, there are three control decisions that you can apply: turn one degree left, turn one degree right and stay still. Turning left or right costs you some energy but might help you produce more energy if you can get better aligned with the wind. In the first part, we model the wind by a simple stochastic process and you will need to use MDP theory to solve the problem optimally. In the second part, we only give access to a simulator and you will need to implement RL based algorithms.

2.1 First part: MDP and optimal control

We first assume that the wind orientation moves by plus or minus one degree at each time step. If θ is the difference of angle between the wind orientation and the turbine orientation, and V the wind speed, then the energy produced during one time step is $f(V)\cos(\theta)$, where $f(V) = V^3$ if $V \leq V_{\max}$ and $f(V) = 0$ if $V > V_{\max}$. For simplicity, we assume that $V_{\max} = 1$.

You have to solve three questions:

1. Assume that V is constant in time and equal to V_{max}, and that moving left or right costs a unit of energy. Model the problem as a MDP, and explain how to compute the optimal solution for a given discount factor γ.

2. Implement an algorithm that finds the optimal solution and illustrate the optimal solution. You are free to choose your values of a and γ, but try to find ones that provide interesting results (Hint: Choose a close to 0 and γ close to 1).
3. (*) Assume now that the wind at time $t+1$ evolves as $V(t+1) = \max(0, \min(1.1, V(t) + b\Delta(t)))$, where $\Delta(t)$ is a value uniformly distributed between -1 and 1 and $b > 0$ is a parameter. Redo questions 1 and 2 above with the new model. Comment on the change of model that is needed.
4. (*) In addition to the above, assume that the wind has some momentum: with 90% changes, the change in the orientation of the wind turbine will be the same as the one last time. Redo question 1 and 2 above and comment.

2.2 Second part: RL algorithms

In this second part, you are asked to use the code https://github.com/paulaubin/wind_turbine_env to simulate the behavior of the wind turbine and wind.

1. The code proposes a random heuristic. Evaluate its performance numerically and compare it with a heuristic that consists in tracking the wind's orientation as perfectly as possible.
2. Implement an agent that uses a tabular Q-learning algorithm and study its performance and its learning speed (Hint: choose wisely your state representation, and explain your choices).
3. (*) Implement a deep-Q-learning algorithm to solve the same task (Hint: again, choose wisely your state representation and your neural architecture).
4. (*) Implement a policy gradient algorithm (Hint: explain your design of a policy, your parametrization and how it learns).

3 Deadlines and practical remarks

3.1 Groups and deadlines

A group is composed of **four to six** students. Each group will handle two reports, to be handle on the *moodle* page of the course. https://cours.univ-grenoble-alpes.fr/course/view.php?id=16628. Each report should consist of **one jupyter notebook** plus any number of additional files that may contain code or data.

- The notebook should be executable as is (*i.e.*, if it needs to download or compile files, this should be included in the notebook).
- The notebook should serve as a report (= should contain all comments / explanation / figures / mathematical formulas / ...) needed to understand what you did.

There are two deadlines ("end of day, anywhere on earth"):

- The first part of the project is to be finished by handled on **October 14, 2022**.
- The full project is to be finished by **November, 18, 2022**.

Penalty for delay: d days of delay = $d^4/4$ day of penalty.

3.2 Grading

- 30% = presentation quality (= organization, quality of figures, comments of results...)
- 70% = scientific content (= algorithms that work, with good performances)

The asterisks (*) are supposed to indicate more difficult questions. If the presentation is reasonable, answering all the questions without a (*) guarantee you a grade of at least 12/20.