CT5141 2019/2020 Assignment 1

James McDermott

Topic Solving a Unit Commitment (electricity generator scheduling) problem using linear programming.

Deadline 5pm Friday 30 October.

Submission When finished, make a single zip file containing the problem data and EITHER a very short `.pdf` report and one `.py` file, OR a single `.ipynb` file containing both the report and code. These files should be named `UC_*`, with your ID number(s), and the correct filename suffix, e.g. `UC_12345678.py` or `UC_12345678_87654321.ipynb`. Your name and student ID should be noted at the top of the `.pdf`, `.py`, and `.ipynb` as appropriate. Submit by going to Blackboard - CT5141 - Assessment.

Groups You may choose to work alone, or in a group of two students. However, you may not work with a student you have already worked with in any other MScAI module group assignment.

Requirements In a small country there are 10 generators of four types: Hydroelectric, Solid fuel, Gas, Solar. Each generator has a lower and upper bound on its production per hour (in MW/h). Between these bounds, production is not limited to discrete levels. Each generator also has a cost for producing each MW, and produces a certain amount of CO2 per MW. All of this information is given in `generator_info.csv`.

The maximum supply from any Solar generator depends on the time of day. Relative to the generator's maximum, it can achieve 50% from 6am to 10am, 100% from 11am to 3pm, 50% from 4pm to 6pm, and 0% otherwise. This is given in `solar_curve.csv`.

The Solid fuel generators cannot change their amount of production from one hour to the next.

The demand from electricity customers varies by hour of the day. This is given in `demand.csv`. If we over-supply relative to demand, it can damage the electricity infrastructure. If we under-supply, then some customers can't boil their kettles.

The goal of the problem is therefore to meet demand at each hour at the minimum cost, by choosing how much energy each generator should produce per hour.

Solve this problem using linear programming. Formulate the problem and include the formulation in a report. Then program it and solve it. Report the solution and the value of the objective. Visualise and interpret the best solution you find, and consider using post-solution analysis to add insight. The report should be very short, e.g. 500 words, with some equations and graphics as needed.

For the code, use Python and any linear programming library you prefer (e.g. OR-tools).

Grading This is worth 15% of the module. Grading is weighted as follows:

- 30% Formulation
- 30% Code
- 40% Solution, interpretation, visualisation, insight (including good writing).

Possible penalties: incorrect submission format and other deviations from the spec.

This assignment has several parts and some are intended to be a challenge. There is no need to worry if you cannot complete all parts. You can receive significant partial credit for partial solutions. You can also receive partial credit for the Solution part even if the Code didn't work, e.g. if you can demonstrate understanding of the problem. I reserve the right to give extra weighting to parts that have worked well in that case.

Support For queries, please post questions on the Discussion Board, but do not post details of your work in progress – instead send email directly to me in those cases. You can also avail of office hours.

Policy Students are reminded of the University's policy on plagiarism. Students may discuss the assignment with other students/groups but must not look at other students' work, or allow others to look at theirs. Any online sources used must be cited with URL and date of access in a comment. Materials from CT5141 need not be cited. By submitting, you declare that you have abided by these conditions. Suspected infringements will be investigated, including by interview, and may be referred to NUI Galway authorities.