Electrical Power Generation

Sophie Demassey et Welington de Oliveira
CMA, Mines ParisTech
sophie.demassey,welington.oliveira@mines-paristech.fr

10 mars 2021

In this electrical power generation problem, also known as the Unit Commitment Problem, subsets of power generation units must be selected and scheduled over a given (typically 24-hour) time horizon in order to satisfy anticipated power demand at minimum cost, while meeting technical requirements on the operation of the units.

This problem is based on example 16 from the fifth edition of *Model Building in Mathematical Programming* by H. Paul Williams on pages 271-272 and 326-327.

A first implementation as a Python code using the Gurobi MILP solver is available in Google Colab, a free online Jupyter Notebook environment that allows you to write and execute Python code through your browser.

You will have to duplicate, modify, and complete **the code** and **the documentation** according to the directions below. The final `.ipynb` files must be sent by email and will be evaluated. The main criteria for the evaluation will be, in this order : 1. the technical correctness of the models, 2. the clarity of the documentation and explanations (either in French or in English), 3. the number of questions processed. In particular, the questions marked *(*)* should be answered after all the others.

Questions below are organized in two parts : modelling with discrete variables (Section 2) and modelling uncertainties (Section 3). Answers must be sent in separated files, to sophie.demassey@mines-paristech.fr for the first part, and to welington.oliveira@mines-paristech.fr for the second part.

1 Basic problem description

The time horizon is divided into consecutive discrete time periods $p \in \{0, \dots, P\}$ of $\Delta_p \in \mathbb{N}$ hours each. The total power demand expected on periode p is denoted by $D_p \in \mathbb{R}_+$ (in MW).

Power generation units are grouped by type. There are $N_t \in \mathbb{N}$ installed units for each type $t \in T$.

Units can be set to on or off at each time period. If a unit of type $t \in T$ is on, then its power output is constant during the period and has a value between $\underline{L}_t \in \mathbb{R}_+$ and $\overline{L}_t \in \mathbb{R}_+$ (in MW). The production costs (in euros/hour) are : $C^b_t \in \mathbb{R}_+$ to operate the unit at its minimum power output, and $C^r_t \in \mathbb{R}_+$ (in euros) for each extra MWh.

The model decides the commitment and the power output of the units for each time period in order to satisfy the total demand at minimum cost.

2 Discrete Optimization

2.1 Modelling (un)differentiated units

1. Model the basic problem as a Mixed Binary Linear Program (with only binary or continuous variables) considering differentiated units.
2. Assume that, for each time period, the committed units of a same type operate with the exact same power output. Model the basic problem with this additional assumption as a Mixed Integer Linear Program without binary variables.
3. Is one model a relaxation of the other one and why? Show that the two models are actually identical.

2.2 Additional models

Include the following details into the two mathematical models above (when possible) :

1. *reserve.* The model also captures a reserve requirement, where the committed units must be capable of increasing their output, while still respecting their maximum output, in order to cope with the situation where actual demand exceeds predicted demand by a given factor $F \in [0, 1]$.
2. *startup costs.* A startup cost $C_t^s \in \mathbb{R}_+$ (in euros) is associated with transitioning a unit of type $t \in T$ from off to on. We are given the number $A_t \in \mathbb{N}$ of units of type $t \in T$ that are already on at time 0.
3. *hydro power generation.* Hydroelectric plants $h \in H$ are available with fixed power output L_h (when on), hourly operation cost $C_h^b \in \mathbb{R}_+$, and startup cost $C_h^s \in \mathbb{R}_+$. The plants are gravity fed from a common reservoir. The depletion of the water in the reservoir happens at rate $R_h \in \mathbb{R}_+$ (in meter/hour) for unit $h \in H$. The reservoir can be replenished by pumping water into in. Pumping water into the reservoir consumes electricity at a rate of $E \in \mathbb{R}_+$ MWh of electricity per meter of height. The height of the reservoir at the end of the time horizon must be equal to the height at the beginning. All the hydroelectric units participate to the reserve.
4. *ramping (*).* When a unit of type $t \in T$ remains on between two consecutive periods, then the power output is given a maximum rate of change (in MW) : $L_t^+ \in \mathbb{R}_+$ for power increase and $L_t^- \in \mathbb{R}_+$ for power decrease. When the unit is turned on, the power output is also limited by fraction $L_t^S \in \mathbb{R}_+$. Conversely, the unit cannot be turned off while the power output exceeds $L_t^E \in \mathbb{R}_+$. L_{0it} denotes the load for the i-th unit of type $t \in T$ right before period 0.
5. *up/down time (*).* Minimum down time $\Delta_t^- \in \mathbb{R}_+$ and up time $\Delta_t^+ \in \mathbb{R}_+$ (in hours) must be observed to prevent turning units of type $t \in T$ on or off too quickly. $\Delta_{0it}^- \in \mathbb{R}_+$ (resp. Δ_{0it}^+) denotes the time the i-th unit of type $t \in T$ has been off (resp. on) before period 0.

2.3 Implementation

Retrieve the Jupyter Notebook example at
https://colab.research.google.com/drive/19WNrTomQnD12aScfmJRxQZGdxwsL3ehc
then save a copy in your Google Drive or a Github directory.

1. Implement and document the two basic models in the Jupyter Notebook (in separate files).
2. Implement and document the additional models of Section 2.2, considering the following additional data values for each unit type $t \in T = \{0, 1, 2\}$:

data		*value/type*		
	$t=$	0	1	2
min down time (h)	Δ_t^-	9	9	9
min up time (h)	Δ_t^+	9	9	9
max ramp-down (MW)	L_t^-	800	1200	1700
max ramp-up (MW)	L_t^+	800	1200	1700
max switch-on (MW)	L_t^S	1000	1500	2000
max switch-off (MW)	L_t^E	1000	1500	2000

   The production plan is repeated every day in a cyclic way, i.e. periods $p = -1$ and $p = P - 1$ coincide.
3. Solve the models. Is a reserve constraint active (i.e. satisfied at equality by the optimal solution) ? If not, what is the maximum reserve factor value satisfied by the current optimal solution ? Otherwise, what is the cost of a 5% increase of the factor value ?
4. Get optimal dual values of the (root) LP relaxations : with gurobipy, use `Model.relax()` to get the LP relaxation, solve it and retrieve the dual values of the demand satisfaction constraints as the attribute `Constr.pi`.
5. *(*)* Relax the reserve constraints and dualize the demand satisfaction constraints, then solve the lagrangian subproblem with the optimal dual values as the multipliers. What is a possible descent direction (in the space of the multipliers) ? Runs 1 (or more) iterations of the subgradient algorithm.

3 Stochastic Optimization Model : handling the random power demand

In this section we consider a stochastic variant of the problem. We assume that the load demand is uncertain from the second period on, but follows a known probability distribution. The goal is to find an optimal set of power stations to turn on in order to satisfy the random power demand with a probability of at least $p_{level} \in [0, 1]$:

— *Random Demand* : Total power generated across all generator types must meet the random demand plus pumping for each time period with probability p_{level} :

$$\mathbb{P}\left[\sum_{t\in \text{Types}} \text{output}_{t,p} + \sum_{h\in \text{HydroUnits}} \text{hydro}_{load\ h} * \text{hydro}_{h,p} \geq \ \textbf{demand}_p + \text{pumping}_p \quad \forall p \in \text{Periods}\right] \geq p_{level}$$

The uncertainties will be represented by $Nscen$ scenarios of demand, drawn from a Gaussian probability distribution with average given by the vector `demand`, and covariance matrix :

$$Cov = \begin{bmatrix} 265000 & 475000 & 650000 & 700000 \\ 475000 & 1300000 & 750000 & 1650000 \\ 650000 & 750000 & 2500000 & 2000000 \\ 700000 & 1650000 & 2000000 & 4700000 \end{bmatrix}$$

The Jupiter Notebook with the deterministic implementation is available at
https://colab.research.google.com/drive/1E9mRlIvrKLMP9HxwlAPu_QOx7BeVayXy?usp=sharing

— Task 1 Reformulate the above probability constraint by using $Nscen$ scenarios $\textbf{demand}_p^i \in R^5$ of demand, having probability Pr_i, $i = 1, \ldots, N$.

— Task 2 Write a code to generate $Nscen \geq 1$ correlated scenarios following the stated probability distribution. Set a seed for the random number generator : use np.random.seed(0).

— Task 3 Using your code for Task 2, generate a sample of $Nscen = 300$ correlated scenarios. Assign the probability $Pr_i = 1/Nscen$ for every scenario.
a) Plot this sample.
b) What is the percentage of scenarios satisfied by the determinist solution ?

— Task 4 Using the above sample, solve the resulting chance-constrained problem (Task 1) with probability level $p_{level} = 0.9$. Compare the following outputs with those from the deterministic model :
a) Costs
b) Thermal production
c) hydro production
d) Pumping
What happens when reliability is increased to $p_{level} = 0.95$

— Task 5 (Bonus) This task is optional.
Generate a sample with $Nscen = 1000$ scenarios. Pre-process this sample to reduce its size. Repeat Task 4 with this pre-processed sample.