On investment in power systems

W. van Ackooij[1]

A. Frangioni, R. Lobato, N. Oudjane

[1] OSIRIS Department
EDF R&D
7 Boulevard Gaspard Monge; 9120 Palaiseau ; France

Bergamo, 2024

Outline

Introduction

- The upcoming energy systems showcase a need for flexibility.
- This need stems from increased (intermittent) generation
- Measuring this need requires representing uncertainty and constraints.
- Typically convexity or favourable structure is lost.

Motivation

- For “energy transition" studies there is a need to compute a “good" energy mix.
- This good “mix" serves as the basis for the operational evaluation and possibly policy “illustration";
- There is a question of geographical scale: Europe, NUTS0 - maybe NUTS2 ?

Motivation II

- So “what is the cost optimal mix?"
- This is an optimization problem, but it involves substantial difficulties.
- In particular one needs a “good way" to compute the operational cost at a given investment decision.

Layers

- For a given investment strategy, evaluating the operational cost has typically two layers:
- The first layer is that of “seasonal storage valuation": computing the cost-optimal strategy of long-term storage - classically hydro
- The second layer - underneath seasonal storage - is that of unit-commitment.
- Investment is thus a stacked three layer optimization problem.

Difficulties

- Each layer is already challenging on its own - exacerbated by the geographical scale.
- The seasonal storage layer is typically a multi-stage stochastic program
- Unit-commitment problems can be challenging too, especially with a detailed model.

Schematic problem

- with $\kappa = (\kappa_1, ..., \kappa_n)$ the capacity vector : κ_i being the investment in technology i,
- we face:

$$\min_{\kappa \in \mathcal{K}} F(\kappa) + O(\kappa).$$

- The deterministic operational cost would look like:

$$\begin{aligned} O(\kappa) := \min_x \quad & \sum_{i=1}^{n} \sum_{j=1}^{\kappa_i} c_i(x_{i,j}) \\ \text{s.t.} \quad & x_{i,j} \in X_i \\ & \sum_{i=1}^{n} \sum_{j=1}^{\kappa_i} A_i x_{i,j} \geq d \end{aligned}$$

Primal view

- In the presence of convexity: X_i convex, c_i convex, the inner operational problem is such that the synchronized solution is also optimal:

$$x_{i,j}^{\mathrm{syn}} = \frac{1}{\kappa_i} \sum_{j=1}^{\kappa_i} x_{i,j}^*.$$

- Under these assumptions the operational cost is thus also:

$$\begin{aligned} O(\kappa) := \min_x \quad & \sum_{i=1}^n \kappa_i c_i(x_i) \\ \text{s.t.} \quad & x_i \in X_i \\ & \sum_{i=1}^n \kappa_i A_i x_i \geq d, \end{aligned}$$

computationally much less involved.

- of course convexity is not present: let us look at the dual

Dual view

- If we dualize the power balance equation we get the Lagrangian dual problem:

$$\underline{O}(\kappa) = \max_{\lambda \geq 0} \theta(\kappa, \lambda),$$

with

$$\theta(\kappa, \lambda) = \lambda^\mathsf{T} d + \sum_{i=1}^{n} \kappa_i \left(\min_{x_i \in X_i} c_i(x_i) - \lambda^\mathsf{T} A_i x_i \right)$$

- it is well known that this Lagrangian dual is also the Lagrangian dual of some appropriately convexified primal problem.

Dual view II

- This dual of the convexified primal is:

$$\theta(\kappa, \lambda) = \lambda^{\mathsf{T}} d - \sum_{i=1}^{n} \kappa_i (c_i^X)^* (A_i^{\mathsf{T}} \lambda),$$

with $c_i^X = c_i + \mathbf{1}_{X_i}$ and $(c_i^X)^*$ being Fenchel's conjugate.

- In the Lagrangian dual we recognize once more the favourable multiplicative structure with respect to κ_i.
- It is furthermore known that Lagrangian duals compute effectively.

The surrogate

- We thus suggest to replace the investment problem with the convexified version:
$$\min_{\kappa \in \mathcal{K}} F(\kappa) + \underline{O}(\kappa).$$
- This surrogate has the advantage of being automatically computed by a well-established computational procedure
- The same computational procedure allows for parallelization, hot-starting and many advanced computational “tricks".

Bounding the gap

We can establish:

Theorem (Bounding the approximation gap)

With $O : \mathcal{K} \to \mathbb{R}$ the operational cost map and $\underline{O}$, the "Lagrangian dual" surrogate. Assume moreover that

- *for each $i = 1, \dots, n$, the sets X_i are compact ;*
- *for each $i = 1, \dots, n$, the cost functions c_i are continuous.*
- *the map $c_0(d - \cdot)$ is convex continuously differentiable on (an open set containing) the compact set $\mathrm{Co}(Y)$ (the convex hull of Y) where $Y := \sum_{i=1}^{n} \sum_{j=1}^{\kappa_i} Y_i$ is the Minkowski sum of the sets $Y_i := A_i X_i$. Moreover, c_0 has L_0-Lipschitz gradient w.r.t. the Euclidean norm $\| \cdot \|_2$.*

Then, for any $\kappa \in \mathcal{K}$, the following bound on the duality gap can be exhibited

$$O(\kappa) - \underline{O}(\kappa) \leq \frac{L_0}{2}(T+1) \max_{1 \leq i \leq n} \Delta_i^2, \tag{1}$$

where Δ_i is the diameter of the compact set

$$K_i := \{ w_i = (y_i, z_i) \in \mathbb{R}^T \times \mathbb{R} \mid y_i = A_i x_i \ , \ z_i = c_i(x_i) \ , \ \textit{with } x_i \in X_i \}. \tag{2}$$

Bounding the gap II

With $S(\kappa) := F(\kappa) + O(\kappa)$ and $\underline{S}(\kappa) := F(\kappa) + \underline{O}(\kappa)$

Corollary (Bounding the relative error on system costs)

Suppose moreover that the fixed costs are linear and separable in each technology i.e., $F(\kappa) = \sum_{i=1}^{n} \kappa_i f_i$ for some $f_i \in \mathbb{R}^+$. Then, for any $\kappa \in \mathcal{K}$, the relative error on system costs is bounded as follows

$$0 \leq \frac{S(\kappa) - \underline{S}(\kappa)}{S(\kappa)} \leq \frac{L_0}{2}(T+1)\frac{\max_{1\leq i\leq n} \Delta_i^2}{\sum_{i=1}^{n} \kappa_i \left(f_i + \bar{c}_i(\kappa)\right)}, \tag{3}$$

where $\bar{c}_i(\kappa)$ denotes the average optimal operational cost for technology i, i.e. $\bar{c}_i(\kappa) = \frac{1}{\kappa_i} \sum_{j=1}^{\kappa_i} c_i(x_{i,j}^*(\kappa))$ with $x_{i,j}^*(\kappa)$ being an optimal operational program for the given investment vector $\kappa \in \mathcal{K}$.

Particularizing the result

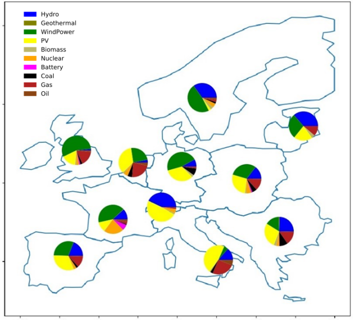
- With $P_i^{\max}$ the maximum power for technology i, we can make the estimate precise:

$$0 \leq \frac{S(\kappa) - \underline{S}(\kappa)}{S(\kappa)} \leq (1 + L^2) L_0 \frac{(T \max_{1 \leq i \leq n} P_i^{\max})^2}{\sum_{i=1}^{n} \kappa_i (f_i + \bar{c}_i(\kappa))} . \quad (4)$$

- So if we invest in many “small" units for large systems, the worst case “gap" - is theoretically small.
- In practice ofcourse, [Frangioni et al.(2011)] show that the gap typically does not exceed 0.5 % anyway.

A case

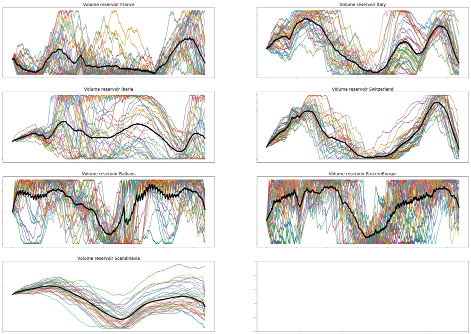
- We pick an 11 zone case from the plan4res H2020 Eu funded project[1]
- The data set is a stochastic mid-term problem with 78 weekly stages, each at hourly granularity
- The problem has 7 reservoirs and more than a 1000 “generators"



[1] https://www.plan4res.eu/ grant : 773897

Results

- The problem is solved through the SMS++ computer code and with the StOpt SDDP solver in roughly 2 hours.
- Volumetric results for the reservoir look as follows:



Structure of time

- We dispose of a larger time horizon $\mathcal{T}$, further split into smaller time periods: subperiods. The latter represents a further subdivision of time.
- Each subperiod represents a stage. Each stage is impacted by uncertainty on load, inflows, renewable generation, (outages)

Structure of storage

- For each stage s and cascaded system c to consider we deal with the following dynamics:
-
$$v_{c,s^+} = v_{c,s} + A^1 f_{:,s} + A^2 \xi_s,$$
where A^1, A^2 are appropriate matrices, $f_{:,s}$ is the vector of flow rates and ξ_s the inflow process.

Structure of program

At a high level the optimization problem to solve appears as:

$$\begin{aligned} \min \quad & \mathbb{E}\left(\sum_s C_s(f_{:,s})\right) \\ \text{s.t.} \quad & f_{c,s} \in \mathfrak{M}_c \ \forall c \ \forall s \\ & f_{c,s} \preceq \sigma(\xi_{[s]}) \ \forall s \\ & v_{c,s^+} = v_{c,s} + A^1 f_{:,s} + A^2 \xi_s, \ \forall c, \end{aligned}$$

- Here C_s is the operational cost attached with flow rate $f_{:,s}$ for all cascading systems.

Transition problem: unit-commitment

- Now the transition problem defines the cost function.
- The transition problem is most naturally modelled as a unit-commitment problem.
- However we would like to be robust to new “sub-models" too.
- Ad-hoc convexification is unlikely to be “reliable" in multiple meanings of the word.

Structure of cost

The cost would typically be of the following form:

$$
\begin{aligned}
C_s(f_{:,s}) := \min \quad & \sum_{i \in I} \hat{C}^i(p_{:,i}), \\
\text{s.t.} \quad & p_{:,i} \in \mathfrak{M}_i, i \in I \\
& (p_{:,c}, f_{c,s}, v_c) \in \mathfrak{M}_c, \ \forall c \\
& \sum_c p_{:,c} + \sum_{i \in I} p_{:,i} = D
\end{aligned}
$$

- Here $\mathfrak{M}_i$ represents the feasible set of generation for various units.
- The cost functions $\hat{C}^i$ can reasonably be assumed convex, as can $\mathfrak{M}_c$, but this is not the case for $\mathfrak{M}_i$.

The Lagrangian

By using the Lagrangian dual for solving the problem defining C_s

- we obtain an efficient solution procedure
- we solve an appropriate convexification, i.e.,

$$\begin{aligned}\bar{C}_s(f_{:,s}) := \min \quad & \sum_{i \in I} \hat{C}^i(p_{:,i}), \\ \text{s.t.} \quad & p_{:,i} \in \mathrm{Co}\, \mathfrak{M}_i, i \in I \\ & (p_{:,c}, f_{c,s}, v_c) \in \mathfrak{M}_c, \ \forall c \\ & \sum_c p_{:,c} + \sum_{i \in I} p_{:,i} = D\end{aligned}$$

- (if the objective function is not linear - we solve a slightly different convexification - harder to make primally explicit)
- we pave the way for being able to use efficient stochastic algorithms for the upper layer

Recursion

The problem admits the following recursion

$$
\begin{aligned}
\bar{\nu}_s(v_{:,s}, \xi_s) = \min\ & \bar{C}_s(f_{:,s}) + \nu_{s^+}(v_{:,s^+}(\xi_{[s]}) \\
\text{s.t.}\, & v_{c,s^+} = v_{c,s} + A^1 f_{:,s} + A^2 \xi_s \ \forall c \\
& f_{c,s} \in \mathfrak{M}_c, \ \forall c.
\end{aligned}
$$

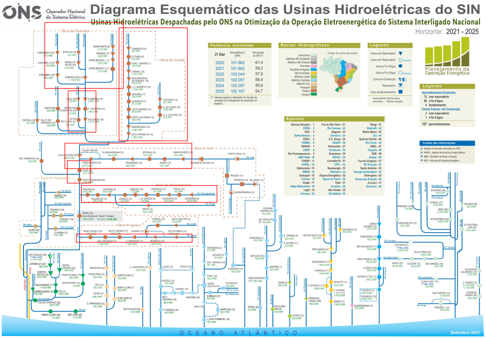
and

$$
\nu_{s^+}(v_{:,s^+}(\xi_{[s]}) := \mathbb{E}\left(\bar{\nu}_{s^+}(v_{:,s^+}, \xi_{s^+}) | \sigma(\xi_{[s]})\right).
$$

- When using $\bar{C}_s$ instead of C_s, the value functions become convex.
- We can thus employ the SDDP algorithm to compute approximations of these value functions.

Seasonal Storage Valuation – some results I

- `SDDPSolver` requires convex problem: continuous relaxation of any formulation or Lagrangian dual
- Brazilian hydro-heavy system: 53 hydro (3 cascade), 98 thermal (coal, gas, nuclear), stochastic inflows (20 scenarios)
- Out-of-sample simulation: 1000 scenarios



	Cont. relax.	Lag. relax.
Cost: Avg. / Std.	4.6023e+9 / 1.3608e+9	4.5860e+9 / 1.3556e+9

- Only 0.4% better, but just changing a few lines in the `Configuration` (Lagrangian about 4 times slower, but can be improved)

Seasonal Storage Valuation – some results II

- Single node (Switzerland)
- 60 stages (1+ year), 37 scenarios, 168 time instants (weekly UC)
- Units: 3 intermittent, 5 thermals, 1 hydro
- Out-of-sample simulation: all 37 scenarios to integer optimality

	Cont. relax.	Lag. relax.
Cost: Avg. / Std.	1.3165e+11 / 2.194e+10	1.2644e+11 / 2.167e+10
Time:	25m	7h30m

- Much longer, but:
  - simulation cost ≈ 30m per scenario, largely dominant
  - save 4% just changing a few lines in the configuration
  - LR time can be improved (`ParallelBundleSolver` not used)

Seasonal Storage Valuation – some results III

- A different single node (France)
- 60 stages (1+ year), 37 scenarios, 168 time instants (weekly UC)
- 83 thermals, 3 intermittent, 2 batteries, 1 hydro
- Out-of-sample simulation: all 37 scenarios to integer optimality

	Cont. relax.	Lag. relax.
Cost: Avg. / Std.	3.951e+11 / 1.608e+11	3.459e+11 / 8.903e+10
Time:	5h43m	7h54m

- Time not so bad (and 3h20m on average simulation per scenario) using `ParallelBundleSolver` with 5 threads per scenario
- That's 14% just changing a few lines in the configuration
- Starts happening regularly enough (and lower variance) to be believable

Energy System Investment Problem – some results I

- Simplified version: solve SDDP only once, run optimization with fixed value-of-water function + simulation (`SDDPGreedySolver`)
- EdF EU scenario: 11 nodes (France, Germany, Italy, Switzerland, Eastern Europe, Benelux, Iberia, Britain, Balkans, Baltics, Scandinavia), 20 lines
- Units: 1183 battery, 7 hydro, 518 thermal, 40 intermittent
- 78 weeks hourly (168h), 37 scenarios (demand, inflow, RES generation)
- Investments: 3 thermal units + 2 transmission lines.
- Average cost: original (operational) 6.510e+12
  optimized (investment + operational) 5.643e+12
- This is ≈ 1 Trillion Euro, 15%
- Running time: ??? hours for value-of-water functions (EdF provided) + 10 hours (4 scenarios in parallel + `ParallelBundleSolver` with 6 threads) for the investment problem

Energy System Investment Problem – some results II

- Simplified version (fixed value-of-water with continuous relaxation)
- Same 11 nodes, 19 lines
- Less units: 7 hydros, 44 thermals, 24 batteries, and 42 intermittent
- More investments: 82 units + 19 transmission lines.
- 78 weeks hourly (168h), 37 scenarios (demand, inflow, RES generation)
- Average cost: original (operational) 3.312e+12
  optimized (investment + operational) 1.397e+12
- This is ≈ 2 Trillion Euro, 137%
- Running time: 48 hours for value-of-water functions (2 nodes = 96 cores) + 5h 20m to solve the investment problem (1 nodes = 48 core)

Energy System Investment Problem

- The true version: value-of-water recomputed anew for each investment
- Still simplified: only one scenario (long way to go, but `TwoStageStochasticBlock` and `BendersDecompositionSolver` currently under active development, we'll get there eventually)
- EU scenario: 14 nodes (France, Germany, Italy, Switzerland, Eastern EU, Benelux, Iberia, Britain, Balkans, Baltics, Denmark, Finland, Sweden, Norway), 28 lines, 62 thermals, 54 intermittent, 8 hydros, 39 batteries
- 78 weeks hourly (168h), 37 scenarios (demand, inflow, RES generation)
- Investments: 99 units of all kinds + all transmission lines
- Two dedicated top-level servers with (each) 2 AMD Epyc 9654 (2.4Ghz, 96 cores, 192 threads, 384MB cache) with 1.5TB RAM (DDR5-4800)
- Requires extensive support for checkpointing and restarts (but less than on CINECA machines that had 24h time limit)

Energy System Investment Problem: first steps

- Huge problem, so three steps approach
  - solve the Seasonal Storage Valuation with initial system (no investment)
  - solve Energy System Investment Problem with fixed value-of-water function out of SDDP (simulation-based optimization)
  - improve investment by dynamically recomputing value-of-water at every iteration

- Original system cost: (operational) 3.467e+12
  Optimized cost: operational 4.505e+11 + investment 2.284e+11 = total 6.789e+11

- Half an order of magnitude saving (suspect most value of lost load), 511% better investing on just 4 lines and 10 hydrogen power plants

- Running time: 15h18m for future cost function of the original system, 5h18m simulation-based investment problem (74 threads max)

The Little-Big Kahuna results

- Starting from previous solution, optimize with variable value-of-water
  - iteration 0: op. 4.505e+11 + inv. 2.284e+11 = total 6.789e+11 (1.8h) (very sparse investment decision)
  - iteration 1: op. 6.670+10 + inv. 5.635e+12 = total 5.702e+12 (22h) (almost completely dense investment decision)
  - iteration 2: op. 1.505e+12 + inv. 2.221e+11 = total 1.727e+12 (21h) (less dense investment decision)
  - iteration 3: op. 2.286e+11 + inv. 7.263e+11 = total 9.549e+11 (20h) (less dense investment decision)
- Already a factor of 2 better than original system (no investment)
- Using LPs in SDDP (many numerical issues), Lagrangian will be better and will be able to use way more threads (`ParallelBundleSolver`)
- Will improve over the fixed value-of-water, just not there as yet
- But we are getting there, thanks to `SMS++`

The Little-Big Kahuna results

- Starting from previous solution, optimize with variable value-of-water
  - iteration 0: op. 4.505e+11 + inv. 2.284e+11 = total 6.789e+11 (1.8h) (very sparse investment decision)
  - iteration 1: op. 6.670+10 + inv. 5.635e+12 = total 5.702e+12 (22h) (almost completely dense investment decision)
  - iteration 2: op. 1.505e+12 + inv. 2.221e+11 = total 1.727e+12 (21h) (less dense investment decision)
  - iteration 3: op. 2.286e+11 + inv. 7.263e+11 = total 9.549e+11 (20h) (less dense investment decision)
- Already a factor of 2 better than original system (no investment)
- Using LPs in SDDP (many numerical issues), Lagrangian will be better and will be able to use way more threads (`ParallelBundleSolver`)
- Will improve over the fixed value-of-water, just not there as yet
- But we are getting there, thanks to `SMS++`

The Little-Big Kahuna results

- Starting from previous solution, optimize with variable value-of-water
  - iteration 0: op. 4.505e+11 + inv. 2.284e+11 = total 6.789e+11 (1.8h) (very sparse investment decision)
  - iteration 1: op. 6.670+10 + inv. 5.635e+12 = total 5.702e+12 (22h) (almost completely dense investment decision)
  - iteration 2: op. 1.505e+12 + inv. 2.221e+11 = total 1.727e+12 (21h) (less dense investment decision)
  - iteration 3: op. 2.286e+11 + inv. 7.263e+11 = total 9.549e+11 (20h) (less dense investment decision)
- Already a factor of 2 better than original system (no investment)
- Using LPs in SDDP (many numerical issues), Lagrangian will be better and will be able to use way more threads (`ParallelBundleSolver`)
- Will improve over the fixed value-of-water, just not there as yet
- But we are getting there, thanks to `SMS++`

The Little-Big Kahuna results

- Starting from previous solution, optimize with variable value-of-water
  - iteration 0: op. 4.505e+11 + inv. 2.284e+11 = total 6.789e+11 (1.8h) (very sparse investment decision)
  - iteration 1: op. 6.670+10 + inv. 5.635e+12 = total 5.702e+12 (22h) (almost completely dense investment decision)
  - iteration 2: op. 1.505e+12 + inv. 2.221e+11 = total 1.727e+12 (21h) (less dense investment decision)
  - iteration 3: op. 2.286e+11 + inv. 7.263e+11 = total 9.549e+11 (20h) (less dense investment decision)
- Already a factor of 2 better than original system (no investment)
- Using LPs in SDDP (many numerical issues), Lagrangian will be better and will be able to use way more threads (`ParallelBundleSolver`)
- Will improve over the fixed value-of-water, just not there as yet
- But we are getting there, thanks to `SMS++`

Summary

- In this talk we discussed the question of investment in power systems.
- We have shown how we can leverage the convexifying effect of the Lagrangian to balance accuracy and computability: both in investment and SDDP.

Some references I

- W. van Ackooij and N. Oudjane. On supply and network investment in power systems.
  4OR, pages 1–17, 2024.
  doi: 10.1007/s10288-024-00566-8
- W. van Ackooij, I. Danti Lopez, A. Frangioni, F. Lacalandra, and M. Tahanan.
  Large-scale unit commitment under uncertainty: an updated literature survey.
  Annals of Operations Research, 271(1):11–85, 2018.
  doi: 10.1007/s10479-018-3003-z

Bibliography I

[[Frangioni et al.(2011)]]A. Frangioni, C. Gentile, and F. Lacalandra.
Sequential Lagrangian-MILP Approaches for Unit Commitment Problems.
International Journal of Electrical Power and Energy Systems, 33:585–593, 2011.

[[van Ackooij and Oudjane(2024)]]W. van Ackooij and N. Oudjane.
On supply and network investment in power systems.
4OR, pages 1–17, 2024.
doi: 10.1007/s10288-024-00566-8.

[[van Ackooij et al.(2018)]]W. van Ackooij, I. Danti Lopez, A. Frangioni, F. Lacalandra, and M. Tahanan.
Large-scale unit commitment under uncertainty: an updated literature survey.
Annals of Operations Research, 271(1):11–85, 2018.
doi: 10.1007/s10479-018-3003-z.