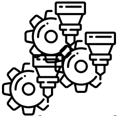
Coupled industrial production and energy supply planning
ROADEF

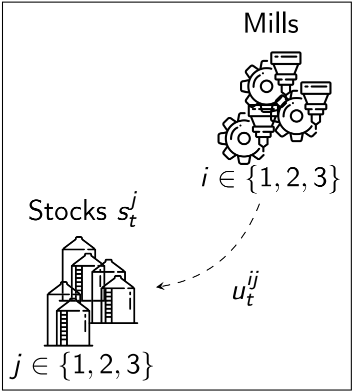
Zoé Fornier
November 2021

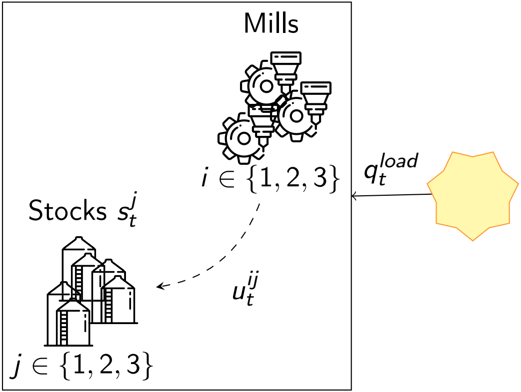
Contents

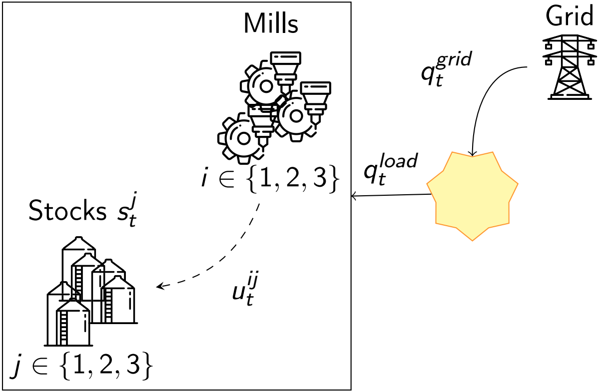
Mills

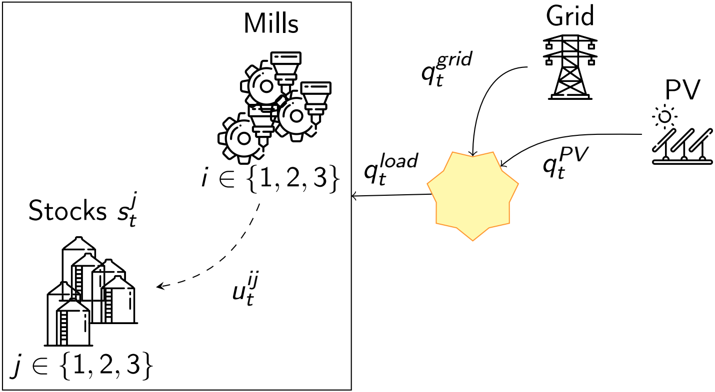


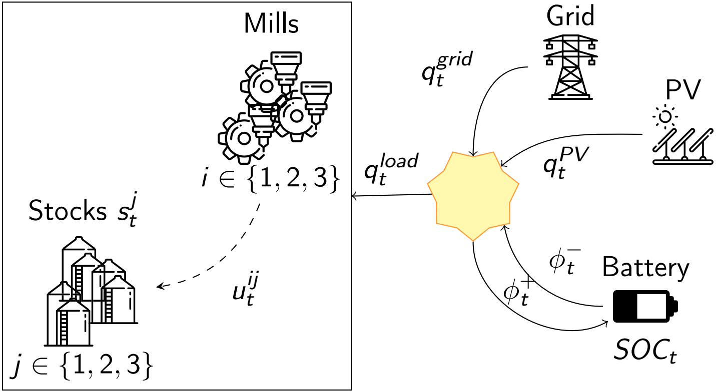
$i \in \{1, 2, 3\}$

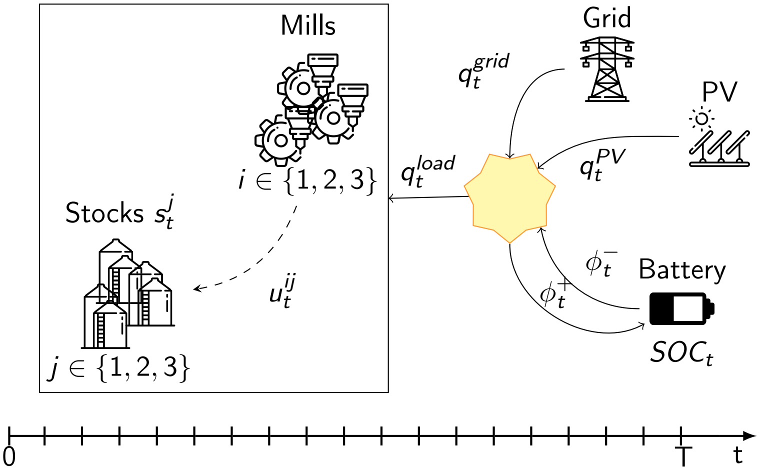

Mills
$i \in \{1, 2, 3\}$
Stocks s_t^j
u_t^{ij}
$j \in \{1, 2, 3\}$

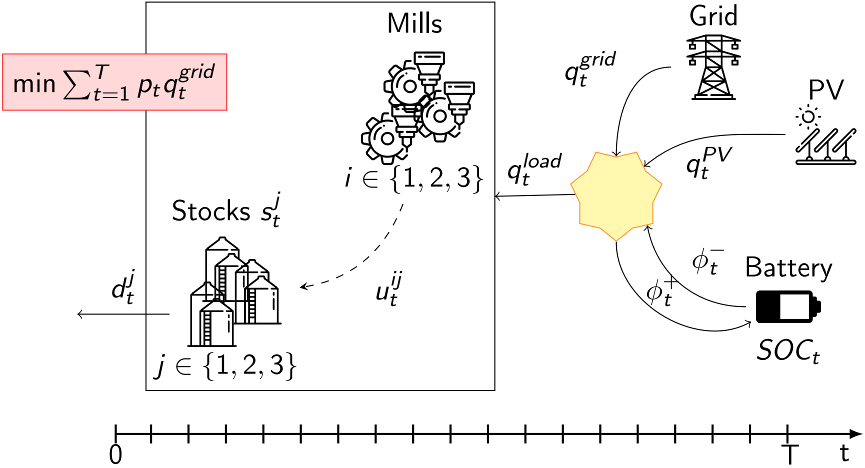

Mills
$i \in \{1, 2, 3\}$
Stocks s_t^j
$j \in \{1, 2, 3\}$
u_t^{ij}
q_t^{load}


Mills
Grid
q_t^{grid}
$i \in \{1, 2, 3\}$
q_t^{load}
Stocks s_t^j
u_t^{ij}
$j \in \{1, 2, 3\}$


Mills
$i \in \{1,2,3\}$
Stocks s_t^j
$j \in \{1,2,3\}$
u_t^{ij}
q_t^{load}
q_t^{grid}
Grid
PV
q_t^{PV}


Mills
$i \in \{1, 2, 3\}$
Stocks s_t^j
$j \in \{1, 2, 3\}$
u_t^{ij}
q_t^{load}
q_t^{grid}
Grid
PV
q_t^{PV}
ϕ_t^-
ϕ_t^+
Battery
SOC_t


Mills
$i \in \{1, 2, 3\}$
Stocks s_t^j
$j \in \{1, 2, 3\}$
u_t^{ij}
q_t^{load}
Grid
q_t^{grid}
PV
q_t^{PV}
ϕ_t^-
ϕ_t^+
Battery
SOC_t
0
T
t


$\min \sum_{t=1}^{T} p_t q_t^{grid}$
Mills
$i \in \{1, 2, 3\}$
Stocks s_t^j
u_t^{ij}
d_t^j
$j \in \{1, 2, 3\}$
Grid
q_t^{grid}
PV
q_t^{PV}
q_t^{load}
ϕ_t^-
ϕ_t^+
Battery
SOC_t
0
T
t

Contents

Deterministic formulation

$$\min \quad \sum_{t=1}^{T} L_t(x_{t-1}, u_t, q_t^{PV})$$

- State variables: $x_t := (SOC_t, s_t^1, s_t^2, s_t^3)$
- Controls: $u_t := (\underbrace{q_t^{grid}, \phi_t^+, \phi_t^-, (u_t^{ij})_{i,j\in[3]}}_{\in\mathbb{R}^+}, \underbrace{(b_t^{ij})_{i,j\in[3]}}_{\in\{0,1\}})$
- Instantaneous cost : $L_t(x_{t-1}, u_t, q_t^{PV}) := p_t q_t^{grid}$

Deterministic formulation

$$\min \quad \sum_{t=1}^{T} L_t(x_{t-1}, u_t, q_t^{PV})$$

$$\text{s.c} \quad x_t = D_t(x_{t-1}, u_t), \qquad x_0 \text{ fixed} \qquad \forall t \geq 1$$

- Dynamic equations:

$$D_t(x_{t-1}, u_t) = \begin{cases} s_t^j = s_{t-1}^j - d_t^j + \sum_i u_t^{ij} & \forall j \\ SOC_t = SOC_{t-1} - \frac{1}{\rho^-}\phi_t^- + \rho^+\phi_t^+ \end{cases}$$

- Initial conditions : $s_0^j = 0 \ \forall j, \ SOC_0 = SOC_{min}$

Deterministic formulation

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} L_t(x_{t-1}, u_t, q_t^{PV}) \\ \text{s.c} \quad & x_t = D_t(x_{t-1}, u_t), \qquad x_0 \text{ fixed} \qquad \forall t \geq 1 \\ & u_t \in \mathbb{U}_t(x_t, q_t^{PV}) \end{aligned}$$

- Feasible domain of controls:

$$\mathbb{U}_t(x_t, q_t^{PV}) = \begin{cases} b_t^{ij} \in \{0, 1\} & \forall i, j \\ u_{min}^{ij} b_t^{ij} \leq u_t^{ij} \leq u_{max}^{ij} b_t^{ij} & \forall i, j \\ q_t^{grid}, \phi_t^+, \phi_t^- \geq 0 & \\ \phi_t^+ \leq \phi_{max}^+ \quad \phi_t^- \leq \phi_{max}^- & \\ \dots & \end{cases}$$

Deterministic formulation

$$\min \quad \sum_{t=1}^{T} L_t(x_{t-1}, u_t, q_t^{PV})$$

$$\text{s.c} \quad x_t = D_t(x_{t-1}, u_t), \qquad x_0 \text{ fixed} \qquad \forall t \geq 1$$

$$u_t \in \mathbb{U}_t(x_t, q_t^{PV})$$

- Controls constraints:

$$\mathbb{U}_t(x_t, q_t^{PV}) = \begin{cases} \dots \\ \sum_j b_t^{ij} \leq 1 & \text{1 product per mill} \\ \max_i b_t^{i1} + \max_i b_t^{i3} \leq 1 & \text{Incompatibility} \\ q_t^{load} \leq q_t^{grid} + q_t^{PV} + \phi_t^- - \phi_t^+ & \text{Load balance} \end{cases}$$

Deterministic formulation

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} L_t(x_{t-1}, u_t, q_t^{PV}) \\ \text{s.c} \quad & x_t = D_t(x_{t-1}, u_t), \qquad x_0 \text{ fixed} \qquad \forall t \geq 1 \\ & u_t \in \mathbb{U}_t(x_t, q_t^{PV}), \qquad x_t \in \mathbb{X}_t \qquad \forall t \geq 1 \end{aligned}$$

- State variables' feasible domain:

$$\mathbb{X}_t = \begin{cases} 0 \leq s_t^j \leq s_{max}^j & \forall j \\ SOC_{min} \leq SOC_t \leq SOC_{max} \end{cases}$$

Contents

NP-completeness

- L production lines and C cements; no incompatibility

Proposition 1

Given a set of parameters, and K $\in \mathbb{R}$: is there a solution whose cost is inferior to K ?
The corresponding decision problem is NP-complete.

A polynomial case

- L production lines and C cements

Proposition 2

Given a set of parameters: is there a feasible solution ?
If C is fixed, and all u^{ij}_{max} are independent from i, the corresponding decision problem is Polynomial.

NP-completeness

- L production lines and C cements; no incompatibility

Proposition 1

Given a set of parameters, and K $\in \mathbb{R}$: is there a solution whose cost is inferior to K ?
The corresponding decision problem is **NP-complete.**

A polynomial case

- L production lines and C cements

Proposition 2

Given a set of parameters: is there a feasible solution ?
If C is fixed, and all u_{max}^{ij} are independent from i, the corresponding decision problem is **Polynomial.**

Contents

Stochastic formulation : Hazard Decision

$$\min \quad \mathbb{E}\left[\sum_{t=1}^{T} L_t(\boldsymbol{x_{t-1}}, \boldsymbol{u_t}, \boldsymbol{q_t^{PV}})\right]$$

$$\text{s.c} \quad \boldsymbol{x_t} = D_t(\boldsymbol{x_{t-1}}, \boldsymbol{u_t}), \qquad x_0 \text{ fixed} \qquad \forall t \geq 1$$

$$\boldsymbol{u_t} \in \mathbb{U}_t(\boldsymbol{x_t}, \boldsymbol{q_t^{PV}}), \qquad \boldsymbol{x_t} \in \mathbb{X}_t \qquad \forall t \geq 1$$

$$\underbrace{\sigma(u_t) \subset \sigma(\boldsymbol{q_1^{PV}}, ..., \boldsymbol{q_t^{PV}})}_{\text{non-anticipativity constraints}}$$

- $(\boldsymbol{q_t^{PV}})_{t\in[T]}$ are random variables: assumed **independent**
- We minimize the expected cost
- We don't know what happens in the future (after t)

Contents

Algorithm 1: Model predictive control

Input: x_0, $\hat{q}^{PV}$ solar prediction for the whole horizon

for t : 1,...,T **do**

Observe q_t^{PV} realization of solar energy at the current time, and using the prediction for the future, solve the following **deterministic** subproblem:

$$(u_{t'}^{\sharp})_{t' \geq t} = \underset{u_t, (u_{t'})_{t'>t}}{\arg\min} \; L_t(x_{t-1}, u_t, q_t^{PV}) + \sum_{t'=t+1}^{T} L_t(x_{t'-1}, u_{t'}, \hat{q}_{t'}^{PV})$$

$$x_{t'} = D_t(x_{t'-1}, u_{t'}) \in \mathbb{X}_{t'}$$

$$u_{t'} \in \mathbb{U}_{t'}(x_{t'}, \hat{q}_{t'}^{PV})$$

$$x_t = \quad D_t(x_{t-1}, u_t^{\sharp})$$

This method will be our reference model.

Contents

- Define the Bellman's functions $V_t(x)$, a.k.a *cost-to-go* function, as the expected optimal cost on $[t, \dots, T]$ starting from x.
- $V_t(x)_{t \in [T]}$ can be computed recursively backward.
- $V_t(x)_{t \in [T]}$ cannot usually be exactly computed (e.g. when there is a continuous number of possible state x), therefore we must consider interpolations and approximations methods:
  - **discretization** of the space set $\boldsymbol{x_t^D} \in \mathbb{X}_{\boldsymbol{N}}$
  - **convex interpolation** of $(x_{t-1}^D, v_t^{x_{t-1}^D})$ to approximate V_t

Algorithm 2: Stochastic dynamic programming

```
// Initialization
```
$V_{T+1} := 0$

```
// Backward phase
```
for *t : T, T-1,...,1* **do**

  for $x^D_{t-1} \in \mathbb{X}_N$ **do**

    for ω *realization of* $\boldsymbol{q_t^{PV}}$ **do**

$$v_t^{x^D_{\boldsymbol{t-1}}} \mathrel{+}= p_\omega \min_{\boldsymbol{u_t} \in \mathbb{U}_t(\boldsymbol{y}, \boldsymbol{q_t^{PV}})} \left\{ L_t(\boldsymbol{x^D_{t-1}}, \boldsymbol{u_t}, \omega) + \tilde{V}_{t+1}(\boldsymbol{y}) \right\}$$

$$\boldsymbol{y} = F_t(\boldsymbol{x^D_{t-1}}, \boldsymbol{u_t})$$

  Define $\tilde{V}_{t+1}$ convex interpolation of $(x^D_t, v^{x^D_t}_{t+1})_{x^D_t \in \mathbb{X}_N}$

Curse of dimensionality: $\mathcal{O}(T.N^4.|\Omega|_t)$ MILP to solve.

- **Stochastic Dual Dynamic Programming** (SDDP) is a classic algorithm which solves continuous multistage linear stochastic problems.

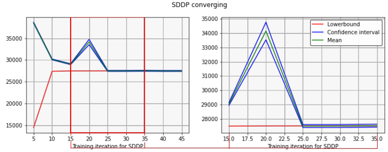
- The algorithm consists in a fixed number of successive steps involving:
  - a forward phase: we **randomly draw** $(\xi_t^k)_t$, a scenario, on which we compute the optimal **trajectory** $(x_t^k)_{t\geq 0}$ for the strategy given by the current cost-to-go approximation.
  - a backward phase: we **refine our** V_t^{k-1} **approximation** by constructing **new cuts** (similar to Benders) from x_{t-1}^k.

- **Stochastic Dual Dynamic Programming** (SDDP) is a classic algorithm which solves continuous multistage linear stochastic problems.

- The algorithm consists in a fixed number of successive steps involving:
  - a forward phase: we **randomly draw** $(\xi_t^k)_t$, a scenario, on which we compute the optimal **trajectory** $(x_t^k)_{t\geq 0}$ for the strategy given by the current cost-to-go approximation.
  - a backward phase: we **refine our** V_t^{k-1} **approximation** by constructing **new cuts** (similar to Benders) from x_{t-1}^k.

- **Stochastic Dual Dynamic Programming** (SDDP) is a classic algorithm which solves continuous multistage linear stochastic problems.

- The algorithm consists in a fixed number of successive steps involving:
  - a forward phase: we **randomly draw** $(\xi_t^k)_t$, a scenario, on which we compute the optimal **trajectory** $(x_t^k)_{t\geq 0}$ for the strategy given by the current cost-to-go approximation.
  - a backward phase: we **refine our V_t^{k-1} approximation** by constructing **new cuts** (similar to Benders) from x_{t-1}^k.



Heuristic based on SDDP

- We consider offline time (construction of a decision strategy) and online time (simulation).
- Compared to DP and MPC, SDDP is very fast.
- However, SDDP solves a convex continuous relaxation of the problem.
- We aim at finding a hybrid method which takes advantage of SDDP's fast computational time, and returns a feasible solution (with binary constraint).

Contents

Cuts Heuristic

Main Idea : estimate V_t using SDDP cuts.

- SDDP **estimates V_t^r, the continuous relaxation of V_t**
- Given a solar energy scenario $(\omega_1, \dots, \omega_T)$, we compute :

$$u_t^\sharp := \underset{u_t \in \mathbb{U}_t(y,\omega_t)}{\arg\min} \; L_t(x_{t-1}^\sharp, u_t, \omega_t) + \boldsymbol{V_{t+1}^r(y)}$$

$$y = F_t(x_{t-1}^\sharp, u_t)$$

$$x_t^\sharp := F_t(x_{t-1}^\sharp, u_t^\sharp)$$

Pros: the offline time amounts to SDDP computation time, the online time is similar to dynamic programming online time.

Cons: the coarse approximation used may be inadequate to get feasible solutions.

Main Idea : estimate V_t using SDDP cuts.

- SDDP **estimates V_t^r, the continuous relaxation of V_t**
- Given a solar energy scenario $(\omega_1, \dots, \omega_T)$, we compute :

$$u_t^\sharp := \underset{u_t \in \mathbb{U}_t(y, \omega_t)}{\arg\min} \; L_t(x_{t-1}^\sharp, u_t, \omega_t) + \boldsymbol{V_{t+1}^r(y)}$$

$$y = F_t(x_{t-1}^\sharp, u_t)$$

$$x_t^\sharp := F_t(x_{t-1}^\sharp, u_t^\sharp)$$

Pros: the offline time amounts to SDDP computation time, the online time is similar to dynamic programming online time.

Cons: the coarse approximation used may be inadequate to get feasible solutions.

Main Idea : estimate V_t using SDDP cuts.

- SDDP **estimates V_t^r, the continuous relaxation of V_t**
- Given a solar energy scenario $(\omega_1, \dots, \omega_T)$, we compute :

$$u_t^\sharp := \underset{u_t \in \mathbb{U}_t(y, \omega_t)}{\arg\min} \; L_t(x_{t-1}^\sharp, u_t, \omega_t) + \boldsymbol{V_{t+1}^r(y)}$$

$$y = F_t(x_{t-1}^\sharp, u_t)$$

$$x_t^\sharp := F_t(x_{t-1}^\sharp, u_t^\sharp)$$

Pros: the offline time amounts to SDDP computation time, the online time is similar to dynamic programming online time.

Cons: the coarse approximation used may be inadequate to get feasible solutions.

Contents

Main Idea: improving DP using 2-stage problems.

- DP reduces a T-stage problem to T consecutive 1-stage problems.
- SDDP solves a convex continuous relaxation.
- Instead, let's consider $T-1$ consecutive **2-stage problems**:
  - **non-anticipativity** constraint, we need to consider all realizations at $t+1$ and the corresponding decision variables ($\times|\Omega|_t$)
  - we minimize the sum of
    - instantaneous cost at t,
    - expected instantaneous cost at $t+1$,
    - and cost-to-go from $t+2$.

Pros : better handling of the binary variables.

Cons : the problem size grows significantly.

Main Idea: improving DP using 2-stage problems.

- DP reduces a T-stage problem to T consecutive 1-stage problems.
- SDDP solves a convex continuous relaxation.
- Instead, let's consider $T-1$ consecutive **2-stage problems**:
  - **non-anticipativity** constraint, we need to consider all realizations at $t+1$ and the corresponding decision variables ($\times|\Omega|_t$)
  - we minimize the sum of
    - instantaneous cost at t,
    - expected instantaneous cost at $t+1$,
    - and cost-to-go from $t+2$.

Pros : better handling of the binary variables.

Cons : the problem size grows significantly.

Main Idea: improving DP using 2-stage problems.

- DP reduces a T-stage problem to T consecutive 1-stage problems.
- SDDP solves a convex continuous relaxation.
- Instead, let's consider $T-1$ consecutive **2-stage problems**:
  - **non-anticipativity** constraint, we need to consider all realizations at $t+1$ and the corresponding decision variables ($\times|\Omega|_t$)
  - we minimize the sum of
    - instantaneous cost at t,
    - expected instantaneous cost at $t+1$,
    - and cost-to-go from $t+2$.

Pros : better handling of the binary variables.

Cons : the problem size grows significantly.

Main Idea: improving DP using 2-stage problems.

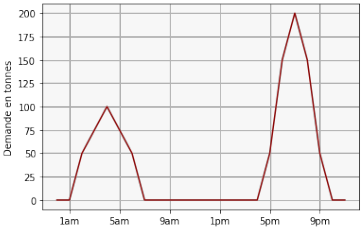
- DP reduces a T-stage problem to T consecutive 1-stage problems.
- SDDP solves a convex continuous relaxation.
- Instead, let's consider $T-1$ consecutive 2-stage problems:
  - **non-anticipativity** constraint, we need to consider all realizations at $t+1$ and the corresponding decision variables $(\times|\Omega|_t)$
  - we minimize the sum of
    - instantaneous cost at t,
    - expected instantaneous cost at $t+1$,
    - and cost-to-go from $t+2$.

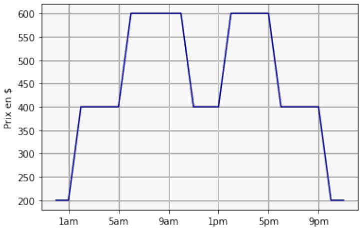
Pros : better handling of the binary variables.

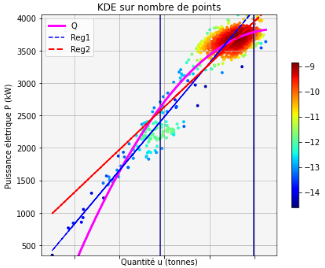
Cons : the problem size grows significantly.

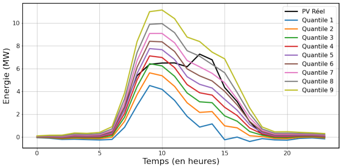
Contents

Fitting parameters to data

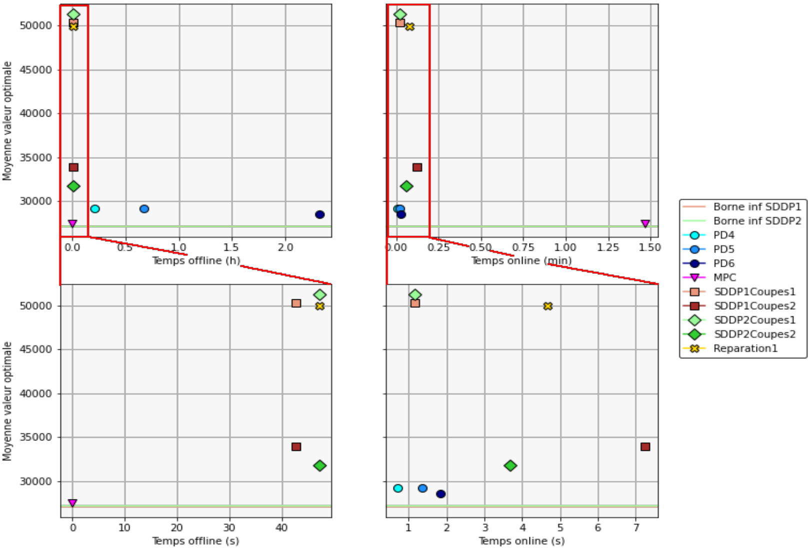








Contents


Moyenne valeur optimale
50000
45000
40000
35000
30000
0.0
0.5
1.0
1.5
2.0
Temps offline (h)
0.00
0.25
0.50
0.75
1.00
1.25
1.50
Temps online (min)
0
10
20
30
40
Temps offline (s)
1
2
3
4
5
6
7
Temps online (s)
Borne inf SDDP1
Borne inf SDDP2
PD4
PD5
PD6
MPC
SDDP1Coupes1
SDDP1Coupes2
SDDP2Coupes1
SDDP2Coupes2
Reparation1

In a nutshell:

- we studied the complexity of the **deterministic** problem;
- we test two classical methods on the stochastic problem: DP (with convex interpolation) and MPC;
- finally, we developed various **heuristics** taking advantage of SDDP for speed and multi-step Bellman equation for better handling of binary variables.

Futur works:

- incorporate day-ahead problem;
- test more advanced algorithm like SDDiP;
- develop a new heuristic to deal with the few binary variables.

In a nutshell:

- we studied the complexity of the **deterministic** problem;
- we test two classical methods on the stochastic problem: **DP** (with convex interpolation) and **MPC**;
- finally, we developed various **heuristics** taking advantage of **SDDP** for speed and multi-step Bellman equation for better handling of binary variables.

Futur works:

- incorporate **day-ahead** problem;
- test more advanced algorithm like **SDDiP**;
- develop a new heuristic to deal with the few binary variables.