Optimal energy management of an urban district

The unbearable lightness of SDDP

F. Pacaud
Advisors: P. Carpentier, J.-P. Chancelier, M. De Lara

May 31, 2017

A paradigm shift in energy transition

The ambition of Efficacity is to improve urban energy efficiency.


Une loi encourage l'autoconsommation d'électricité

Jean-Claude Bourbon, le 17/02/2017 à 10h14
Mis à jour le 17/02/2017 à 10h31

Les professionnels n'ont pas attendu la fixation du cadre réglementaire pour lancer des offres.

De nombreuses jeunes sociétés investissent le créneau.

Un projet de loi vise à développer l'autoconsommation d'électricité. / diyanadimitrova/Fotolia

Le texte était réclamé depuis longtemps par les professionnels des én renouvelables, en particulier dans le photovoltaïque. Le Parlement a

Self-consumption

Domestic storage

Energy management system

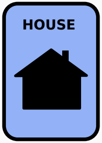
Our team focus on the control of *energy management system*.

What do we do

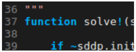
How to control storage inside urban microgrid ?

We follow a common procedure in operation research:

1. We consider a real world problem
   How to control a bunch of storage ?

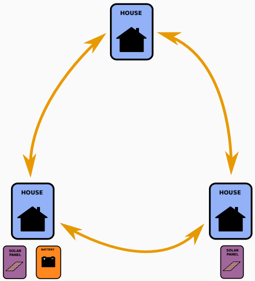


2. We model it as an optimization problem
   As demands are not predictable, we formulate a stochastic optimization problem



3. We develop algorithms to solve this particular optimization problem
   Dynamic Programming based methods, Model Predictive Control, ...

Analyzing the real world problem



We consider a system where different units (houses) are connected together via a local network (microgrid).

The houses have different storage available:

- batteries,
- electrical hot water tank

and are equipped with solar panels.

We control the stocks every 15mn and we want to

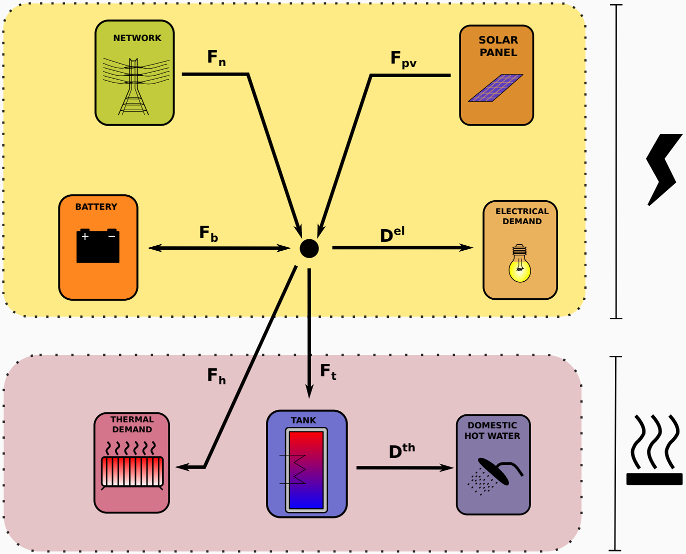
- minimize electrical's bill
- maintain a comfortable temperature inside the house

Outline

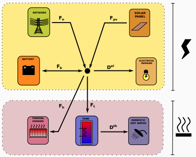
Outline

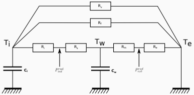
Outline

For each house, we consider the following devices



We introduce states, controls and noises

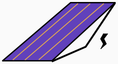


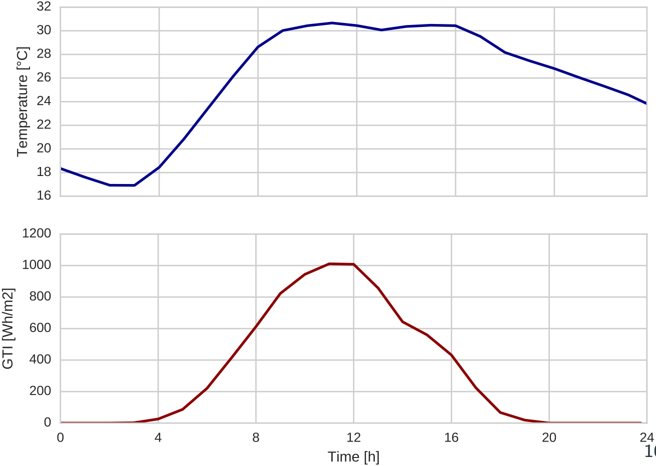


- **Stock variables** $\mathbf{X}_t = \left(\mathbf{B}_t, \mathbf{H}_t, \theta^i_t, \theta^w_t\right)$
  - $\mathbf{B}_t$, battery level (kWh)
  - $\mathbf{H}_t$, hot water storage (kWh)
  - θ^i_t, inner temperature ($^\circ C$)
  - θ^w_t, wall's temperature ($^\circ C$)
- **Control variables** $\mathbf{U}_t = \left(\mathbf{F}^+_{\mathbf{B},t}, \mathbf{F}^-_{\mathbf{B},t}, \mathbf{F}_{T,t}, \mathbf{F}_{\mathbf{H},t}\right)$
  - $\mathbf{F}_{\mathbf{B},t}$, energy exchange with the battery (kW)
  - $\mathbf{F}_{T,t}$, energy used to heat the hot water tank (kW)
  - $\mathbf{F}_{\mathbf{H},t}$, thermal heating (kW)
- **Uncertainties** $\mathbf{W}_t = \left(\mathbf{D}^E_t, \mathbf{D}^{DHW}_t\right)$
  - $\mathbf{D}^E_t$, electrical demand (kW)
  - $\mathbf{D}^{DHW}_t$, domestic hot water demand (kW)

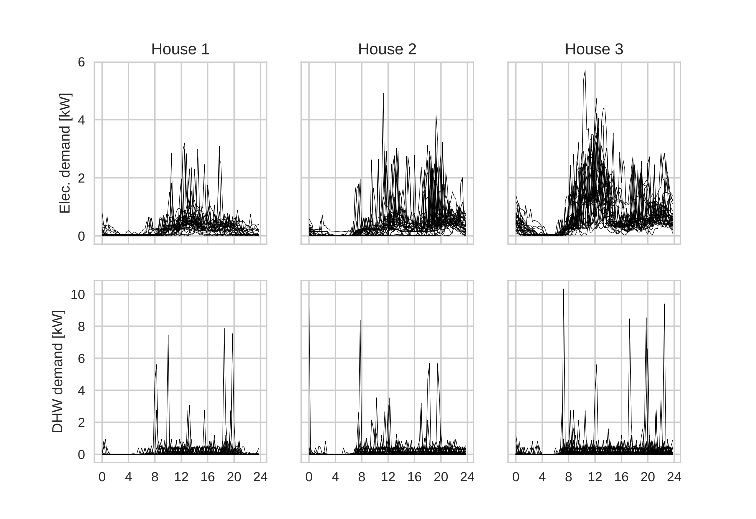
We work with real data

We consider one day during summer 2015 (data from Meteo France):





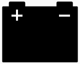
We generate scenarios of demands during this day



These scenarios are generated with StRoBE, a generator open-sourced by KU-Leuven

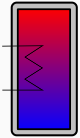
Discrete time state equations

We have the four state equations (all linear):



$$\mathbf{B}_{t+1} = \alpha_{\mathbf{B}} \mathbf{B}_t + \Delta T \left(\rho_c \mathbf{F}^{+}_{\mathbf{B},t} - \frac{1}{\rho_d} \mathbf{F}^{-}_{\mathbf{B},t} \right)$$

$$\mathbf{H}_{t+1} = \alpha_{\mathbf{H}} \mathbf{H}_t + \Delta T \left[\mathbf{F}_{T,t} - \mathbf{D}^{DHW}_t \right]$$



$$\boldsymbol{\theta}^w_{t+1} = \boldsymbol{\theta}^w_t + \frac{\Delta T}{c_m} \left[\frac{\boldsymbol{\theta}^i_t - \boldsymbol{\theta}^w_t}{R_i + R_s} + \frac{\boldsymbol{\theta}^e_t - \boldsymbol{\theta}^w_t}{R_m + R_e} + \gamma \mathbf{F}_{\mathbf{H},t} + \frac{R_i}{R_i + R_s} P^{int}_t + \frac{R_e}{R_e + R_m} P^{ext}_t \right]$$

$$\boldsymbol{\theta}^i_{t+1} = \boldsymbol{\theta}^i_t + \frac{\Delta T}{c_i} \left[\frac{\boldsymbol{\theta}^w_t - \boldsymbol{\theta}^i_t}{R_i + R_s} + \frac{\boldsymbol{\theta}^e_t - \boldsymbol{\theta}^i_t}{R_v} + \frac{\boldsymbol{\theta}^e_t - \boldsymbol{\theta}^i_t}{R_f} + (1 - \gamma) \mathbf{F}_{\mathbf{H},t} + \frac{R_s}{R_i + R_s} P^{int}_t \right]$$

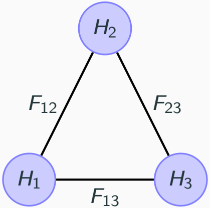
which will be denoted:

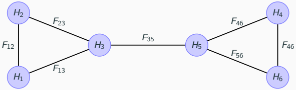
$$\boxed{\mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1})}$$

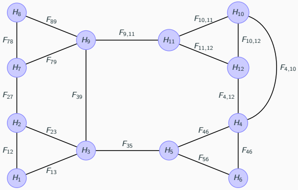
Outline

Viewing the network as a directed graph

We consider three different configurations





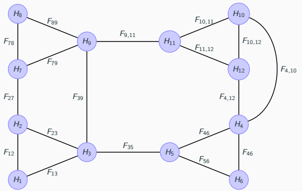


*H*1	House 1	PV + Battery
*H*2	House 2	PV
*H*3	House 3	.

*H*1	House 1	PV + Battery
*H*2	House 2	PV
*H*3	House 3	.
*H*4	House 4	PV + Battery
*H*5	House 5	PV
*H*6	House 6	.

*H*1	House 1	PV + Battery
*H*2	House 2	PV
*H*3	House 3	.
*H*4	House 4	PV + Battery
*H*5	House 5	PV
*H*6	House 6	.
*H*7	House 7	PV + Battery
*H*8	House 8	PV
*H*9	House 9	.
*H*10	House 10	PV + Battery
*H*11	House 11	PV
*H*12	House 12	.

Modeling exchange through the graph



We denote by $\mathbf{Q}$ the flows through the arcs, and $\mathbf{\Delta}$ the balance at each node.

The flows must satisfy the Kirchhoff's law:

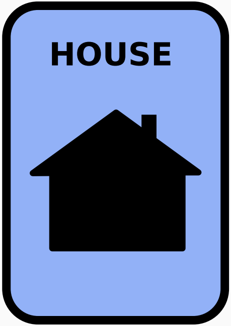
$$A\mathbf{Q} = \mathbf{\Delta}$$

where A is the node-incidence matrix.

We suppose furthermore that losses occurs through the arcs ($\eta = 0.96$).

Outline

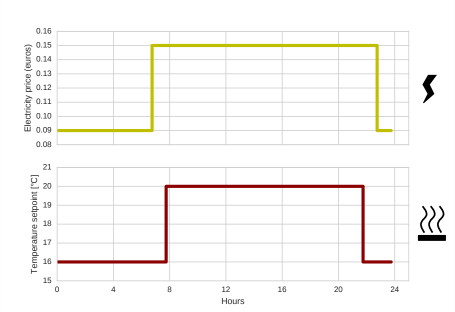
Two commandments to rule them all



Thou shall:

- Satisfy thermal comfort
- Optimize operational costs

Prices and temperature setpoints vary along time



- $T_f = 24\text{h}$, $\Delta T = 15\text{mn}$
- Electricity peak and off-peak hours
  $\pi_t^E = 0.09$ or 0.15 euros/kWh
- Temperature set-point
  $\bar{\theta}_t^i = 16^\circ C$ or $20^\circ C$

The costs we have to pay

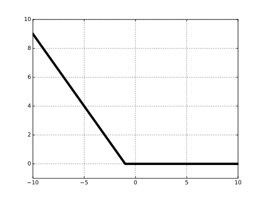
- Cost to import electricity from the network

$$\underbrace{-b_t^E \max\{0, -\mathbf{F}_{NE,t+1}\}}_{\text{selling}} + \underbrace{\pi_t^E \max\{0, \mathbf{F}_{NE,t+1}\}}_{\text{buying}}$$

where we define the recourse variable (electricity balance):

$$\underbrace{\mathbf{F}_{NE,t+1}}_{Network} = \underbrace{\mathbf{D}_{t+1}^E}_{Demand} + \underbrace{\mathbf{F}_{\mathbf{B},t}}_{Battery} + \underbrace{\mathbf{F}_{\mathbf{H},t}}_{Heating} + \underbrace{\mathbf{F}_{T,t}}_{Tank} - \underbrace{\mathbf{F}_{pv,t}}_{Solar\ \ panel} + \underbrace{\mathbf{\Delta}_t}_{Exchange}$$

- Virtual Cost of thermal discomfort: $\kappa_{th}(\underbrace{\boldsymbol{\theta}_t^i - \bar{\boldsymbol{\theta}}_t^i}_{\text{deviation from setpoint}})$



κ_{th}

Piecewise linear cost

Penalize temperature if below given setpoint

Instantaneous and final costs for a single house

- The instantaneous convex costs are

$$L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{\Delta}_t, \mathbf{W}_{t+1}) = \underbrace{-b_t^E \max\{0, -\mathbf{F}_{NE,t+1}\}}_{buying} + \underbrace{\pi_t^E \max\{0, \mathbf{F}_{NE,t+1}\}}_{selling} + \underbrace{\kappa_{th}(\boldsymbol{\theta}_t^i - \bar{\boldsymbol{\theta}}_t^i)}_{discomfort}$$

- We add a final linear cost

$$K(\mathbf{X}_{T_f}) = -\pi^{\mathbf{H}} \mathbf{H}_{T_f} - \pi^{\mathbf{B}} \mathbf{B}_{T_f}$$

to avoid empty stocks at the final horizon T_f

That gives the following stochastic optimization problem for the global problem

$$\min_{X^h, U^h} \quad \sum_h J(X^h, U^h)$$
$$s.t \quad AQ = \Delta$$

where for each house h:

$$J(X,U) = \mathbb{E}\left[\sum_{t=0}^{T_f - 1} \underbrace{L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{\Delta}_t, \mathbf{W}_{t+1})}_{instantaneous\ \ cost} + \underbrace{K(\mathbf{X}_{T_f})}_{final\ \ cost}\right]$$

$$\begin{aligned} s.t \quad \mathbf{X}_{t+1} &= f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) && \text{Dynamic} \\ X^\flat \leq \mathbf{X}_t &\leq X^\sharp \\ U^\flat \leq \mathbf{U}_t &\leq U^\sharp \\ X_0 &= X_{ini} \\ \sigma(\mathbf{U}_t) &\subset \sigma(\mathbf{W}_1, \ldots, \mathbf{W}_t) && \text{Non-anticipativity} \end{aligned}$$

Outline

Outline

How to solve this stochastic optimal control problem?

We have 96 timesteps (4 x 24) and for each problem

	3 houses	6 houses	12 houses
Stocks	10	20	40
Controls	14	30	68
Uncertainties	8	8	8

How to solve this stochastic optimal control problem?

We have 96 timesteps (4 x 24) and for each problem

	3 houses	6 houses	12 houses
Stocks	10	20	40
Controls	14	30	68
Uncertainties	8	8	8

The state dimension is high (≥ 10), the problem is not tractable by a straightforward use of *dynamic programming* because of the curse of dimensionality! :-(

How to solve this stochastic optimal control problem?

We have 96 timesteps (4 x 24) and for each problem

	3 houses	6 houses	12 houses
Stocks	10	20	40
Controls	14	30	68
Uncertainties	8	8	8

The state dimension is high (≥ 10), the problem is not tractable by a straightforward use of *dynamic programming* because of the curse of dimensionality! :-(

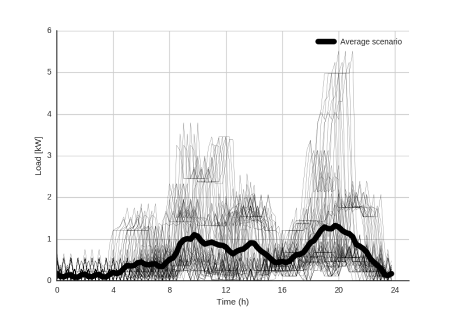
We will compare two methods that overcome this curse:

1. **Model Predictive Control** (MPC)
2. **Stochastic Dual Dynamic Programming** (SDDP)

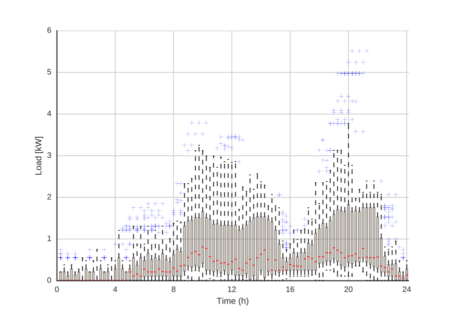
MPC vs SDDP: uncertainties modelling

The two algorithms use optimization scenarios to model the uncertainties:

MPC



SDDP



MPC vs SDDP: online resolution

At the beginning of time period $[\tau, \tau+1]$, do

MPC

- Consider a **rolling horizon** $[\tau, \tau + H[$
- Consider a **deterministic scenario** of demands (forecast) $(\overline{W}_{\tau+1}, \dots, \overline{W}_{\tau+H})$
- Solve the **deterministic optimization** problem

$$\min_{X,U} \left[\sum_{t=\tau}^{\tau+H} L_t(X_t, U_t, \overline{W}_{t+1}) + K(X_{\tau+H}) \right]$$

$$\begin{aligned} s.t. \qquad X_\cdot &= (X_\tau, \dots, X_{\tau+H}) \\ U_\cdot &= (U_\tau, \dots, U_{\tau+H-1}) \\ X_{t+1} &= f(X_t, U_t, \overline{W}_{t+1}) \\ X^\flat &\leq X_t \leq X^\sharp \\ U^\flat &\leq U_t \leq U^\sharp \end{aligned}$$

- Get optimal solution $(U^\#_\tau, \dots, U^\#_{\tau+H})$ over horizon $H = 24h$
- Send first control $U^\#_\tau$ to assessor

SDDP

- We consider the approximated value functions $\left(\widetilde{V}_t\right)_0^{T_f}$

$$\underbrace{\widetilde{V}_t}_{\text{Piecewise affine functions}} \leq V_t$$

- Solve the **stochastic optimization problem**

$$\min_{u_\tau} \mathbb{E}_{W_{\tau+1}} \Big[L_\tau(X_\tau, u_\tau, W_{\tau+1}) + \widetilde{V}_{\tau+1}\Big(f_\tau(X_\tau, u_\tau, W_{\tau+1})\Big)\Big]$$

$$\iff \min_{u_\tau} \sum_i \pi_i \Big[L_\tau(X_\tau, u_\tau, W^i_{\tau+1}) + \widetilde{V}_{\tau+1}\Big(f_\tau(X_\tau, u_\tau, W^i_{\tau+1})\Big)\Big]$$

$\Rightarrow$ this problem resumes to solve a LP at each timestep

- Get optimal solution $U^\#_\tau$
- Send $U^\#_\tau$ to assessor

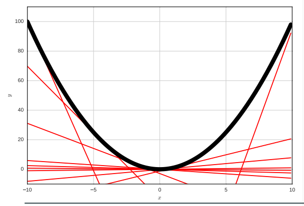
A brief recall on Dynamic Programming

Dynamic Programming

μ_t is the probability law of W_t and is being used to estimate expectation and compute **offline** value functions with the backward equation:

$$V_T(x) = K(x)$$
$$V_t(x_t) = \min_{U_t} \mathbb{E}_{\mu_t}\Big[\underbrace{L_t(x_t, U_t, W_{t+1})}_{\text{current cost}} + \underbrace{V_{t+1}\Big(f(x_t, U_t, W_{t+1})\Big)}_{\text{future costs}}\Big]$$

Stochastic Dual Dynamic Programming [1]



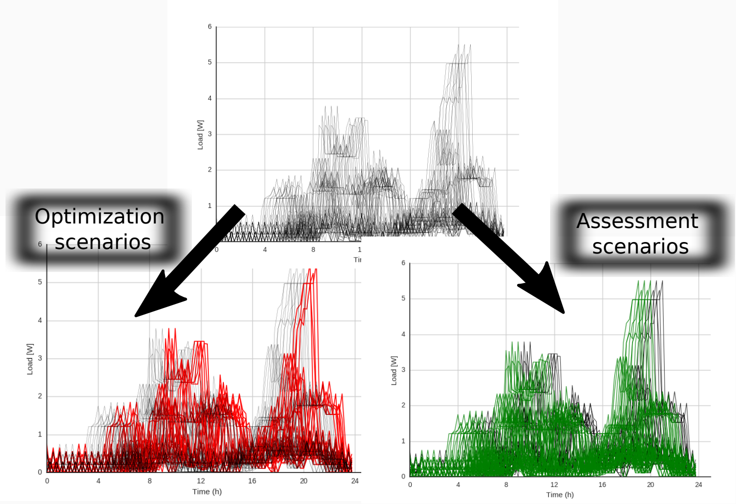
- Convex value functions V_t are approximated as a supremum of a finite set of affine functions
- Affine functions (=cuts) are computed during forward/backward passes, till convergence
- SDDP makes an extensive use of LP solver

$$\widetilde{V}_t(x) = \max_{1 \le k \le K} \{\lambda_t^k x + \beta_t^k\} \le V_t(x)$$

[1] Here, we use a variant of SDDP to compute cuts in Decision-Hazard

Outline

Out-of-sample comparison



Outline

Outline

Our stack is deeply rooted in Julia language

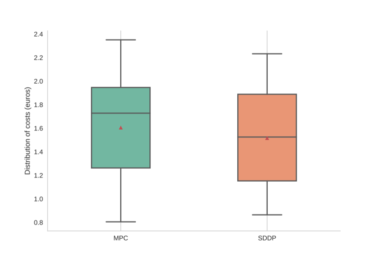
- Modeling Language: `JuMP`
- Open-source SDDP Solver: `StochDynamicProgramming.jl`
- LP Solver: `Gurobi 7.0`

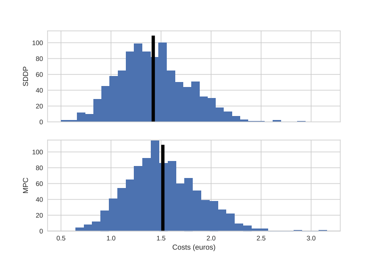
`https://github.com/JuliaOpt/StochDynamicProgramming.jl`

Outline

Comparison of MPC and SDDP

We compare MPC and SDDP over 1000 assessment scenarios



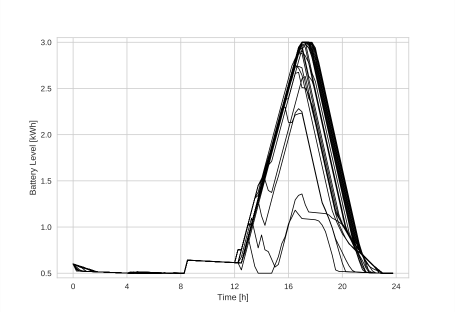


	MPC	SDDP	Diff
3 houses			
Costs	1.52	1.42	-6.6 %
t_c	0.8	2.8	x3.5
6 houses			
Costs	3.04	2.85	-6.3 %
t_c	1.7	4.6	x2.7
12 houses			
Costs	6.08	5.74	-5.6 %
t_c	3.5	8.6	x2.5

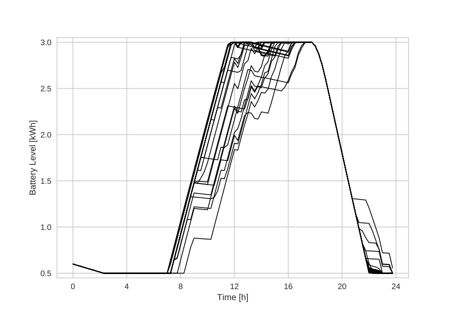
t_c: average time to compute the control online (in ms)

MPC and SDDP use differently the battery

MPC



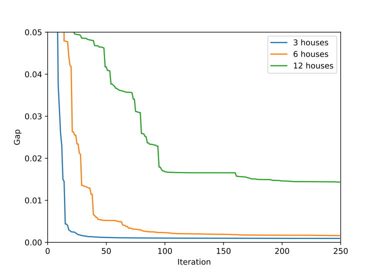
SDDP

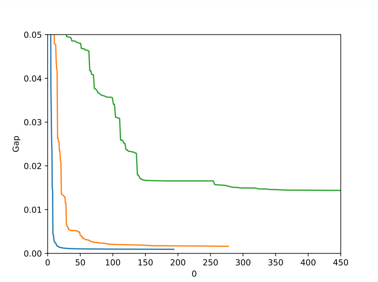


Trajectories of battery for the "3 houses" problem.

Discussing the convergence of SDDP w.r.t. the dimension

We compute the upper-bound afterward, with a great number of scenarios (10000) We define the gap as : $gap = (ub - lb)/ub$.





We compare the time taken to achieve a particular gap:

gap	3 houses	6 houses	12 houses
2 %	7.0	21.0	137.8
1 %	8.0	28.8	.
0.5 %	8.0	47.2	.
0.1 %	65.1	.	.

Outline

Conclusion

- SDDP scales up to 40 dimensions!
- We have to use a variant of SDDP to compute cuts in Decision-Hazard, because classical SDDP gives poor results
- SDDP beats MPC, however the difference narrows along the number of dimensions (because of the convergence of SDDP)
- Both MPC and SDDP are penalized if dimension became too high

Perspectives

Mix SDDP with spatial decomposition like *Dual Approximate Dynamic Programming* (DADP) to control bigger urban neighbourhood