Time decomposition methods for optimal management of energy storage under stochasticity

T. Rigaut PhD defense
Under the supervision of F. Bourquin and J.-Ph. Chancelier

16 May, 2019

efficacity
IFSTTAR
École des Ponts
ParisTech

Local renewable energies are spreading

To lower CO_2 emissions from our electricity generation

We tend to consume energy where it is produced

But they require a storage that has to be managed

When intermittent renewables generation does not match demand
we rely on fossil fuels

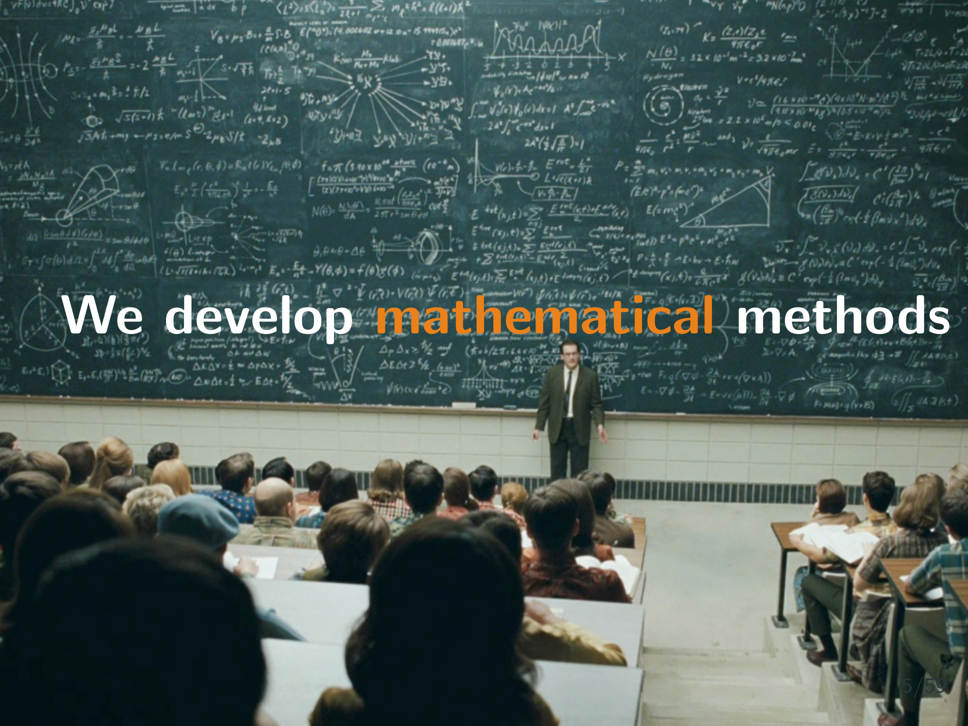
Storage cleans our electricity generation
as long as we optimize its management to make it sustainable

Real problems adressed by the optimization team at Efficacity

Our team solves energy management problems
for the energy transition of cities with our industrial partners

RATP case study

VINCI Energies case study

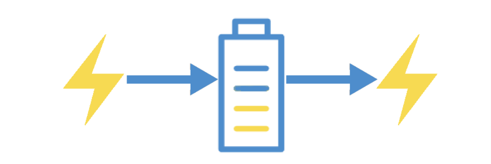


We develop mathematical methods

handling uncertain outcomes

to optimize storage management on multiple time scales

to store/consume clean energy at the right minutes of the day



and to ensure a sustainable battery life lasting many years

Table of contents of the thesis: 5 preprints, 2 articles

Part I: Contributions to time decomposition in multistage stochastic optimization

1. **Time blocks decomposition of multistage stochastic optimization problems**,
   P. Carpentier, J-Ph. Chancelier, M. De Lara, T. Rigaut
2. **A template to design online policies for multistage stochastic optimization problems**, P. Carpentier, J-Ph. Chancelier, M. De Lara, F. Pacaud, T. Rigaut

Part II: Stochastic optimization of storage energy management in microgrids

3. **Energy and air quality management in a subway station using stochastic dynamic optimization**, IEEE Transactions on Power Systems,
   P. Carpentier, J-Ph. Chancelier, M. De Lara, T. Rigaut, J. Waeytens
4. **Power management in a DC micro grid integrating renewables and storage**,
   Control Engineering Practices, G. Damm, E. De Santis, M.D. Di Benedetto, A. Iovine, T. Rigaut
5. **Algorithms for two-time scales stochastic optimization with applications to long term management of energy storage**,
   P. Carpentier, J-Ph. Chancelier, M. De Lara, T. Rigaut

Part III: Softwares and experimentations

6. **DynOpt: a generic library for stochastic dynamic optimization**, T. Rigaut
7. **Energy aware temperature control of a house using stochastic dual dynamic programming: a first test bed implementation**, F. Bourquin, T. Rigaut, J. Waeytens

Outline of the PhD defense

A: Stochastic optimization of energy and air quality in a subway station 10’

We optimize battery and ventilation control

to minimize daily electricity bill of a subway station

- Chapters 2 and 3

B: Algorithms for two-time scales stochastic optimization 25’

We optimize on two time scales

to minimize electricity bill of a solar home

and maximize long term sustainability of batteries

- Chapters 1 and 5

C: Software and experimentations 5’

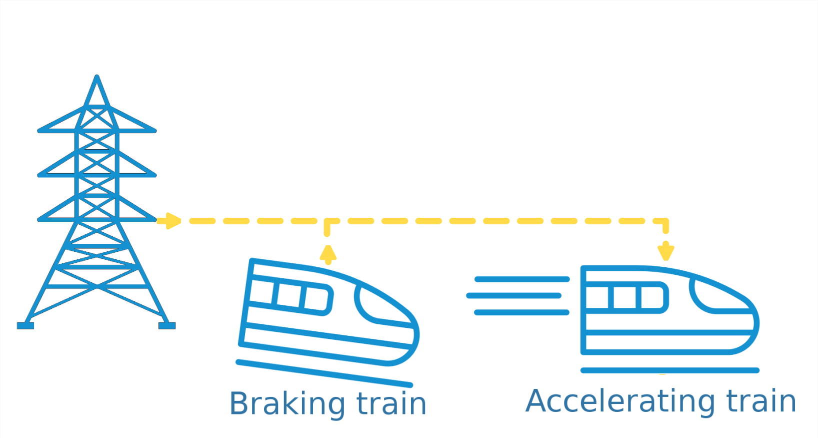
We deploy our algorithms in the real world

- Chapters 6 and 7

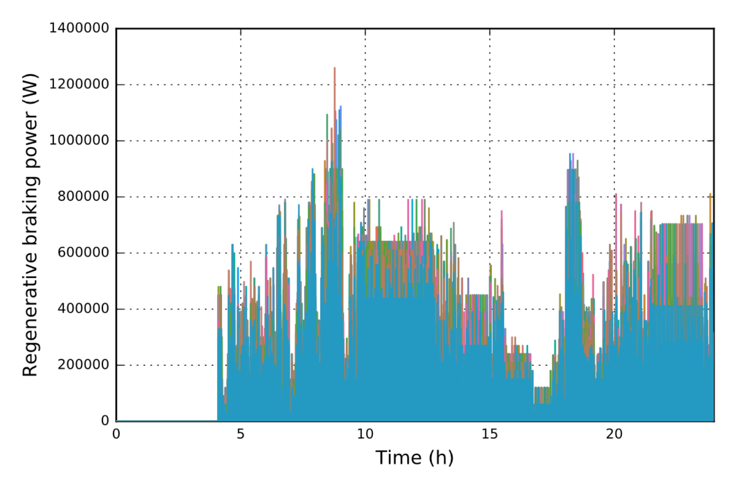
A: Stochastic optimization of energy and air quality in a subway station

We design energy management strategies for a subway station

Subway stations have unexploited energy ressources

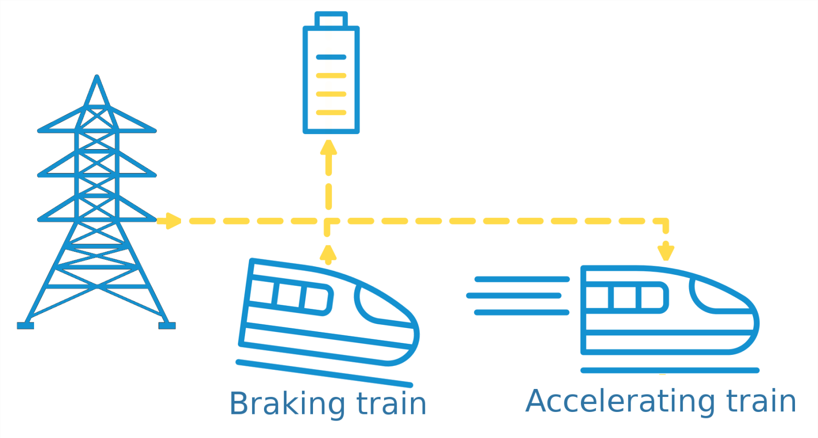


Subway stations have unexploited erratic energy ressources



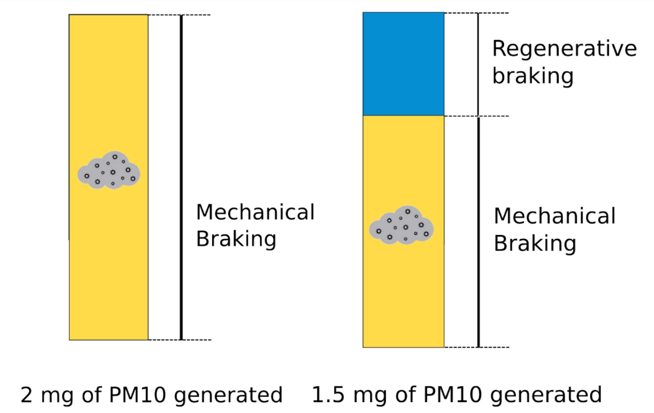
Braking energy scenarios

That can be recovered through storage



Subways mechanical braking generates particulate matters

Ventilation represents 25% of electricity consumption



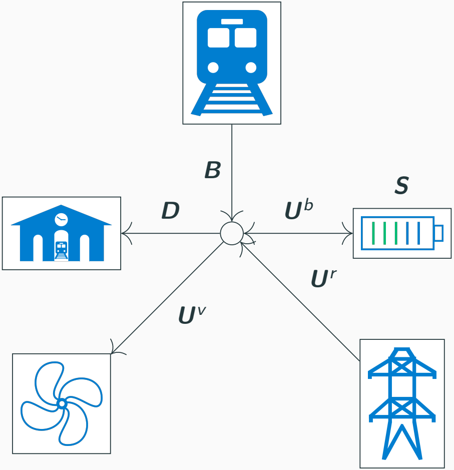
Win Win: recovering braking energy improves air quality

So let us optimize battery and ventilation in a subway station!

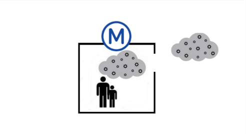
A: Stochastic optimization of energy and air quality in a subway station

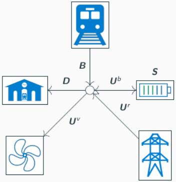
Statement of the problem

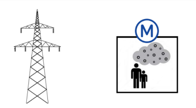
A subway station with battery and ventilation



Physical model and optimization objective







- Battery state of charge dynamics

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t + \underbrace{\rho_c (\boldsymbol{U}_t^b)^+}_{\text{charge}} - \underbrace{\rho_d^{-1} (\boldsymbol{U}_t^b)^-}_{\text{discharge}}$$

- PM10 concentration dynamics

$$\boldsymbol{C}_{t+1} = \boldsymbol{C}_t \underbrace{-\Delta\delta \boldsymbol{C}_t + \Delta\alpha \boldsymbol{N}_{t+1}^2}_{\text{deposition and generation}} + \underbrace{\left(\frac{\rho_v}{v} \boldsymbol{U}_t^v + \Delta\beta \boldsymbol{N}_{t+1}\right)\left(\boldsymbol{C}_{t+1}^o - \boldsymbol{C}_t\right)}_{\text{inside/outside exchange}}$$

- Balance equation

$$\underbrace{\boldsymbol{U}_t^r + \boldsymbol{B}_t}_{\text{import+braking}} = \underbrace{\boldsymbol{D}_t + \boldsymbol{U}_t^v + \boldsymbol{U}_t^b}_{\text{demand+ventilation+battery}}$$

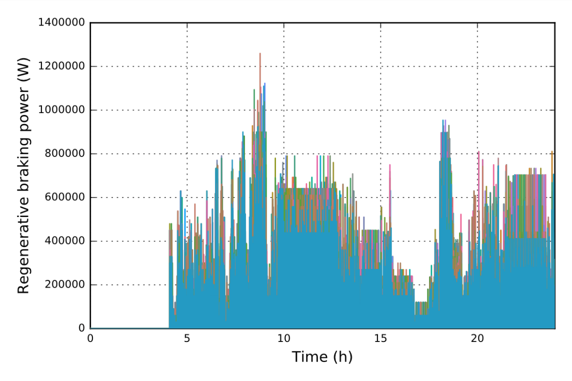
- Cost: energy consumption and PM10 concentration

$$\mathbb{E}\left[\sum_{t=0}^{T-1} \underbrace{p_{t+1} \left(\boldsymbol{U}_{t+1}^r\right)^+}_{\text{energy expense}} + \underbrace{\lambda \boldsymbol{C}_{t+1}}_{\text{air quality cost}}\right]$$

We focus on braking energy uncertainty

We design algorithms to handle this uncertainty based on multiple scenarios of

$$(\boldsymbol{W}_0, \dots, \boldsymbol{W}_T) \text{ with } \boldsymbol{W}_t = \boldsymbol{B}_t$$



Braking energy scenarios (RATP)

Stochastic optimization problem statement

We gather all previous equations to state a standard **stochastic optimal control problem**

- States: $\boldsymbol{X}_t = (\boldsymbol{S}_t, \boldsymbol{C}_t)$
- Controls: $\boldsymbol{U}_t = (\boldsymbol{U}_t^b, \boldsymbol{U}_t^v)$
- Uncertainty: $\boldsymbol{W}_t = \boldsymbol{B}_t$
- Costs: $L_t(\boldsymbol{X}_t, \boldsymbol{U}_t, \boldsymbol{W}_{t+1})$
- Dynamic: $f_t(\boldsymbol{X}_t, \boldsymbol{U}_t, \boldsymbol{W}_{t+1})$
- Constraints: Γ_t

$$
\begin{aligned}
\min_{\boldsymbol{X},\boldsymbol{U}} \; & \mathbb{E}\Big[\sum_{t=0}^{T-1} L_t(\boldsymbol{X}_t, \boldsymbol{U}_t, \boldsymbol{W}_{t+1})\Big] \\
\text{s.t } & \boldsymbol{X}_{t+1} = f_t(\boldsymbol{X}_t, \boldsymbol{U}_t, \boldsymbol{W}_{t+1}) \\
& (\boldsymbol{X}_t, \boldsymbol{U}_t, \boldsymbol{W}_{t+1}) \in \Gamma_t, \; \mathbb{P}\text{-a.s} \\
& \sigma(\boldsymbol{U}_t) \subset \sigma(\boldsymbol{W}_0, \dots, \boldsymbol{W}_t)
\end{aligned}
$$

The non anticipativity constraint states that we take our decisions based on past uncertainties observations

A: Stochastic optimization of energy and air quality in a subway station

Comparison of 4 control strategies

We are looking for energy management strategies

We compare 4 methods to produce a strategy π_t taking into account current state $x_t \in \mathbb{X}_t$ and last noise $w_t \in \mathbb{W}_t$ and producing a control $u_t \in \mathbb{U}_t$

$$\pi_t : \mathbb{X}_t \times \mathbb{W}_t \to \mathbb{U}_t,$$
$$u_t = \pi_t(x_t, w_t)$$

1. Open Loop Feedback Control (**OLFC**)
2. Certainty Equivalent Control (**CEC**)
3. Stochastic Dynamic Programming with online law (**SDPO**)
4. Stochastic Dynamic Programming with augmented state (**SDPA**)

These methods are detailed in Chapter 2:

A template to design online policies for multistage stochastic optimization problems

OLFC solves a deterministic problem online

Online at time t in state x_t and knowing last noise w_t we draw S scenarios $\{\tilde{w}^s_{t+1}, \dots, \tilde{w}^s_T\}_{1 \leq s \leq S}$ with probabilities $\{p_s\}_{1 \leq s \leq S}$ and solve

$$
\begin{aligned}
\pi_t(x_t, w_t) \in \mathop{\arg\min}_{u_t \in \mathbb{U}_t} \min_{(u_{t+1}, \dots, u_{T-1})} \sum_{s=1}^{S} p_s \sum_{t=t'}^{T-1} L_t(x^s_{t'}, u_{t'}, \tilde{w}^s_{t'+1}) \\
\text{s.t } x^s_{t'+1} = f_{t'}(x^s_{t'}, u_{t'}, \tilde{w}^s_{t'+1}) \\
x^s_t = x_t
\end{aligned}
$$

- OLFC is a kind of stochastic Model Predictive Control (MPC)
- CEC (traditionally called MPC) is OLFC with one scenario

SDPO computes value functions offline to use them online

- **Offline** we use a family of offline discrete laws $\{\mu_1^{of}, \dots, \mu_T^{of}\}$
and compute value functions solving Bellman equation

$$
\begin{aligned}
&V_T = 0\\
&V_t(x_t) = \min_{u_t \in \mathbb{U}_t} \int_{\mathbb{W}_{t+1}} \Big[L_t(x_t, u_t, w_{t+1}) + V_{t+1}(x_{t+1}) \Big] \mu_{t+1}^{of}(dw_{t+1})\\
&\qquad\qquad \text{s.t } x_{t+1} = f_t(x_t, u_t, w_{t+1})
\end{aligned}
$$

- **Online** at time t in state x_t and knowing last noise w_t,
we use an online discrete conditional law $\mu_{t+1}^{on}(w_t, \cdot)$ and solve

$$
\begin{aligned}
&\pi_t(x_t, w_t) \in \arg\min_{u_t \in \mathbb{U}_t} \int_{\mathbb{W}_{t+1}} \Big[L_t(x_t, u_t, w_{t+1}) + V_{t+1}(x_{t+1}) \Big] \mu_{t+1}^{on}(w_t, dw_{t+1})\\
&\qquad\qquad \text{s.t } x_{t+1} = f_t(x_t, u_t, w_{t+1})
\end{aligned}
$$

SDPA computes state augmented value functions

- **Offline** noises are modeled as $\log \boldsymbol{W}_{t+1} = a \log \boldsymbol{W}_t + \boldsymbol{Z}_{t+1}$

and we use discrete laws $\{\rho_1, \dots, \rho_T\}$ for $(\boldsymbol{Z}_1, \dots, \boldsymbol{Z}_T)$

Then we compute value functions for each x and w

$$V_t(x, w) = \min_{u \in \mathbb{U}_t} \int_{\mathbb{Z}_{t+1}} \Big[L_t(x, u, w_{t+1}) + V_{t+1}\Big(x_{t+1}, w_{t+1}\Big) \Big] \rho_{t+1}(dz_{t+1})$$

$$\text{s.t } x_{t+1} = f_t(x, u, w_{t+1})$$

$$w_{t+1} = \exp(a \log w + z_{t+1})$$

- **Online** at time t in state x_t and knowing last noise w_t, we solve

$$\pi_t(x_t, w_t) \in \arg\min_{u \in \mathbb{U}_t} \int_{\mathbb{Z}_{t+1}} \Big[L_t(x_t, u, w_{t+1}) + V_{t+1}\Big(x_{t+1}, w_{t+1}\Big) \Big] \rho_{t+1}(dz_{t+1})$$

$$\text{s.t } x_{t+1} = f_t(x_t, u, w_{t+1})$$

$$w_{t+1} = \exp(a \log w_t + z_{t+1})$$

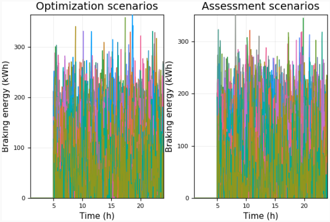
A: Stochastic optimization of energy and air quality in a subway station

Assessment by simulation: numerical results

Assessment of the strategies

We perform Monte Carlo simulations on common assessment scenarios

Optimization scenarios were used to design our algorithms



We compare how the strategies perform in terms of

1. Computation times
2. Energy expenses
3. Air quality

SDP computes online decisions faster than OLFC and CEC

We obtain the following computation times

	OLFC	CEC	SDPO	SDPA
Offline time			$0h06$	$3h47$
Mean online time	54 ms	5.7 ms	0.04 ms	0.30 ms

SDP methods require offline time but are faster online

We decrease expenses by 46 %: SDPA beats other strategies

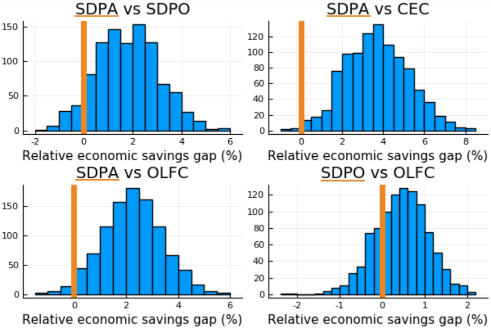
- Without battery and constant ventilation at $60m^3/s$

Energy expenses: 161€

- SDPA is the best on average

	OLFC	CEC	SDPO	SDPA
Saved money (€)	-72.4 ± 0.29	-71.4 ± 0.27	-73.0 ± 0.28	-74.1 ± 0.30

- SDPA is the best in almost every scenario



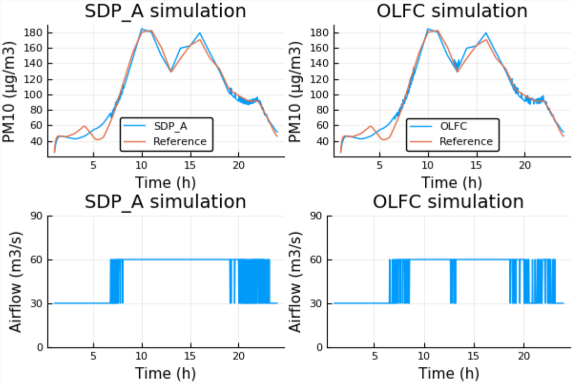
We lower ventilation by 30 % without deteriorating air quality

- Without battery and constant ventilation at $60m^3/s$

Mean PM10: 108 $\mu g/m^3$

- The algorithms do not deteriorate air quality

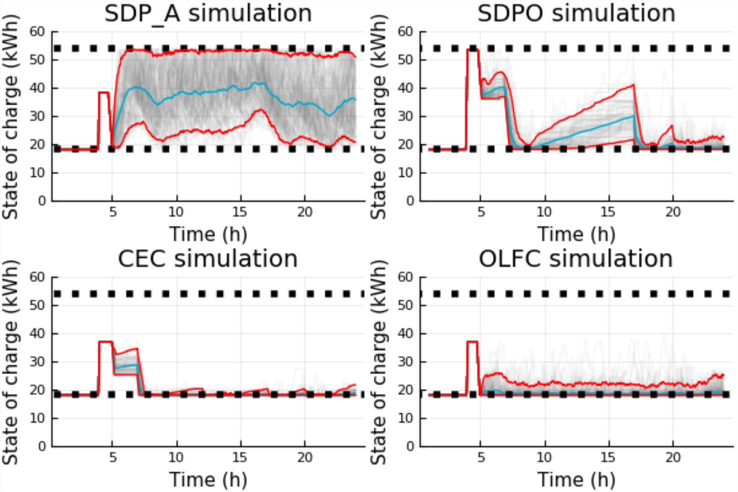
	OLFC	CEC	SDPO	SDPA
Mean PM10 ($\mu g/m^3$)	106 $\pm$0.01	107 $\pm$0.01	107 $\pm$0.01	106 $\pm$0.01



We have compared stochastic optimization algorithms

1. **We have shown that SDP outperforms MPC to handle highly uncertain energy sources**
2. **We have shown that we can decrease energy expenses by 46%**
3. **We have shown that we can lower ventilation by 30% without deteriorating air quality**

The algorithms solicitate the battery differently



State of charge of the battery on multiple simulations (gray), mean (blue), 5% confidence interval (red)

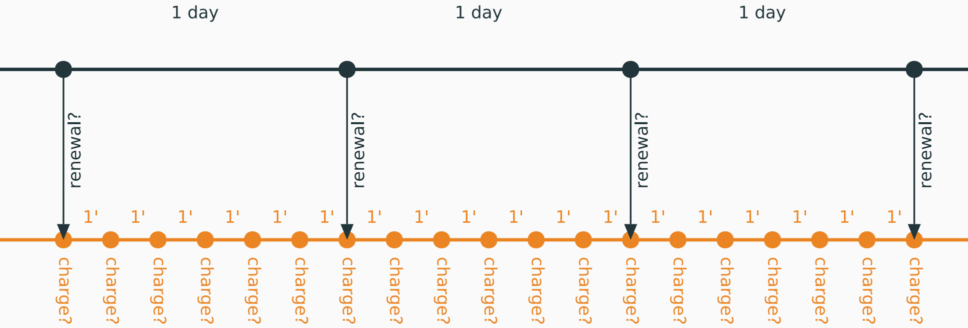
Does it pay to install a battery in a subway station?

We have to take into account investments and battery aging!

B: Algorithms for two-time scales stochastic optimization

We tackle battery control problems on two time scales

We will decompose the scales



Two time scales stochastic optimal control problem

$$
\begin{aligned}
\min_{\boldsymbol{X}_{0:D+1}, \boldsymbol{U}_{0:D}} \quad & \mathbb{E}\Big[\sum_{d=0}^{D} L_d(\boldsymbol{X}_d, \boldsymbol{U}_d, \boldsymbol{W}_d) + K(\boldsymbol{X}_{D+1})\Big] \\
\text{s.t } & \boldsymbol{X}_{d+1} = f_d(\boldsymbol{X}_d, \boldsymbol{U}_d, \boldsymbol{W}_d) \\
& \boldsymbol{U}_d = (\boldsymbol{U}_{d,0}, \dots, \boldsymbol{U}_{d,m}, \dots, \boldsymbol{U}_{d,M}) \\
& \boldsymbol{W}_d = (\boldsymbol{W}_{d,0}, \dots, \boldsymbol{W}_{d,m}, \dots, \boldsymbol{W}_{d,M}) \\
& \sigma(\boldsymbol{U}_{d,m}) \subset \sigma\big(\boldsymbol{W}_{d',m'}; (d', m') \leq (d, m)\big)
\end{aligned}
$$

We have a non standard problem

- with daily time steps
- but a non anticipativity constraint every minute

Next to come: outline of part B

I. Illustration with an energy storage management application

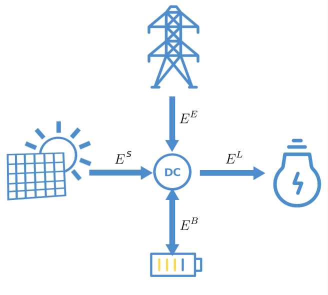
II. Two algorithms for two-time scales stochastic optimization

III. Numerical results for a house with solar panels and batteries

B: Algorithms for two-time scales stochastic optimization

Application to energy storage management

Physical model: a home with load, solar panels and storage



- Two time scales uncertainties
  - $\boldsymbol{E}^L_{d,m}$: Uncertain demand
  - $\boldsymbol{E}^S_{d,m}$: Uncertain solar electricity
  - $\boldsymbol{P}^b_d$: Uncertain storage cost
- Two time scales controls
  - $\boldsymbol{E}^E_{d,m}$: National grid import
  - $\boldsymbol{E}^B_{d,m}$: Storage charge/discharge
  - $\boldsymbol{R}_d$: Storage renewal
- Two time scales states
  - $\boldsymbol{B}_{d,m}$: Storage state of charge
  - $\boldsymbol{H}_{d,m}$: Storage health
  - $\boldsymbol{C}_d$: Storage capacity
- Balance equation:

  $\boldsymbol{E}^E_{d,m} + \boldsymbol{E}^S_{d,m} = \boldsymbol{E}^B_{d,m} + \boldsymbol{E}^L_{d,m}$
- Battery dynamic:

  $\boldsymbol{B}_{d,m+1} = \boldsymbol{B}_{d,m} - \frac{1}{\rho_d}\boldsymbol{E}^{B-}_{d,m} + \frac{1}{\rho_d}\rho_c\boldsymbol{E}^{B+}_{d,m}$

New dynamics: aging and renewal model

- At the end of every day d, we can buy a new battery at cost $\boldsymbol{P}_d^b \times \boldsymbol{R}_d$

$$\text{Storage capacity: } \boldsymbol{C}_{d+1} = \begin{cases} \boldsymbol{R}_d \,, & \text{if } \boldsymbol{R}_d > 0 \\ \boldsymbol{C}_d \,, & \text{otherwise} \end{cases}$$

example: a Tesla Powerwall 2 with 14 kWh costs $430 \times 14 = 6020$ €

- A new battery can make a maximum number of cycles $N_c(\boldsymbol{R}_d)$:

$$\text{Storage health: } \boldsymbol{H}_{d+1,0} = \begin{cases} 2 \times N_c(\boldsymbol{R}_d) \times \boldsymbol{R}_d \,, & \text{if } \boldsymbol{R}_d > 0 \\ \boldsymbol{H}_{d,M} \,, & \text{otherwise} \end{cases}$$

$\boldsymbol{H}_{d,m}$ is the amount of exchangeable energy day d, minute m

$$\boldsymbol{H}_{d,m+1} = \boldsymbol{H}_{d,m} - \frac{1}{\rho_d} \boldsymbol{E}_{d,m}^{B-} - \rho_c \boldsymbol{E}_{d,m}^{B+}$$

example: a Tesla Powerwall 2 can make 3200 cycles or exchange 90 MWh

- A new battery is empty

$$\text{Storage state of charge: } \boldsymbol{B}_{d+1,0} = \begin{cases} \underline{B} \times \boldsymbol{R}_d \,, & \text{if } \boldsymbol{R}_d > 0 \\ \boldsymbol{B}_{d,M} \,, & \text{otherwise} \end{cases}$$

We build a non standard stochastic optimal control problem

- Objective to be minimized

$$\mathbb{E}\Big[\sum_{d=0}^{D}\Big(\underbrace{\boldsymbol{P}^b_d \times \boldsymbol{R}_d}_{\text{renewal}} + \sum_{m=0}^{M-1}\underbrace{p^e_{d,m}}_{\textit{price}} \times \big(\underbrace{\boldsymbol{E}^B_{d,m} + \boldsymbol{E}^L_{d,m+1} - \boldsymbol{E}^S_{d,m+1}}_{\text{national grid energy consumption}}\big)\Big)\Big]$$

- Controls

$$\boldsymbol{U}_d = (\boldsymbol{E}^B_{d,0} \dots, \boldsymbol{E}^B_{d,m}, \dots, \boldsymbol{E}^B_{d,M-1}, \boldsymbol{R}_d)$$

- Uncertainties

$$\boldsymbol{W}_d = \left(\begin{pmatrix}\boldsymbol{E}^S_{d,1}\\ \boldsymbol{E}^L_{d,1}\end{pmatrix}, \dots, \begin{pmatrix}\boldsymbol{E}^S_{d,m}\\ \boldsymbol{E}^L_{d,m}\end{pmatrix}, \dots, \begin{pmatrix}\boldsymbol{E}^S_{d,M-1}\\ \boldsymbol{E}^L_{d,M-1}\end{pmatrix}, \begin{pmatrix}\boldsymbol{E}^S_{d,M}\\ \boldsymbol{E}^L_{d,M}\\ \boldsymbol{P}^b_d\end{pmatrix}\right)$$

- States and dynamics

$$\boldsymbol{X}_d = \begin{pmatrix}\boldsymbol{C}_d\\ \boldsymbol{B}_{d,0}\\ \boldsymbol{H}_{d,0}\end{pmatrix} \text{ and } \boldsymbol{X}_{d+1} = f_d\big(\boldsymbol{X}_d, \boldsymbol{U}_d, \boldsymbol{W}_d\big)$$

Two time scales stochastic optimal control problem

$$
\begin{aligned}
\mathcal{P}: \quad \min_{\boldsymbol{X}_{0:D+1}, \boldsymbol{U}_{0:D}} \quad & \mathbb{E}\Big[\sum_{d=0}^{D} L_d(\boldsymbol{X}_d, \boldsymbol{U}_d, \boldsymbol{W}_d) + K(\boldsymbol{X}_{D+1})\Big] , \\
\text{s.t } & \boldsymbol{X}_{d+1} = f_d(\boldsymbol{X}_d, \boldsymbol{U}_d, \boldsymbol{W}_d) , \\
& \boldsymbol{U}_d = (\boldsymbol{U}_{d,0}, \dots, \boldsymbol{U}_{d,m}, \dots, \boldsymbol{U}_{d,M}) \\
& \boldsymbol{W}_d = (\boldsymbol{W}_{d,0}, \dots, \boldsymbol{W}_{d,m}, \dots, \boldsymbol{W}_{d,M}) \\
& \sigma(\boldsymbol{U}_{d,m}) \subset \sigma\big(\boldsymbol{W}_{d',m'}; (d', m') \leq (d, m)\big)
\end{aligned}
$$

Two time scales because of the non anticipativity constraint

Information grows every minute!

- Intraday time stages: $M = 24 * 60 = 1440$ minutes
- Daily time stages: $D = 365 * 20 = 7300$ days
- $D \times M = 10,512,000$ stages!

B: Algorithms for two-time scales stochastic optimization

Time decomposition by daily dynamic programming

Daily management when "end of the day" cost is known

On day d assume that we have a final cost $V_{d+1} : \mathbb{X}_{d+1} \to [0, +\infty]$

giving a price to a battery in state $\boldsymbol{X}_{d+1} \in \mathbb{X}_{d+1}$

Solving the intraday problem with a final cost

$$
\begin{aligned}
\min_{\boldsymbol{X}_{d+1}, \boldsymbol{U}_d} \quad & \mathbb{E}\left[L_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d) + V_{d+1}(\boldsymbol{X}_{d+1})\right] \\
\text{s.t } \boldsymbol{X}_{d+1} &= f_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d) \\
\boldsymbol{U}_d &= (\boldsymbol{U}_{d,0}, \dots, \boldsymbol{U}_{d,m}, \dots, \boldsymbol{U}_{d,M}) \\
\sigma(\boldsymbol{U}_{d,m}) &\subset \sigma(\boldsymbol{W}_{d,0:m})
\end{aligned}
$$

Gives a minute scale policy for day d that takes into account the future through V_{d+1}, the daily value of energy storage

We write a Bellman equation with daily time blocks

Daily Independence Assumption

$\{\boldsymbol{W}_d\}_{d=0,\dots,D}$ is a sequence of independent random variables

We set $V_{D+1} = K$ and then by backward induction:

$$V_d(x) = \min_{\boldsymbol{X}_{d+1}, \boldsymbol{U}_d} \mathbb{E}\left[L_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d) + V_{d+1}(\boldsymbol{X}_{d+1})\right]$$

$$\text{s.t } \boldsymbol{X}_{d+1} = f_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d)$$

$$\sigma(\boldsymbol{U}_{d,m}) \subset \sigma(\boldsymbol{W}_{d,0:m})$$

where $\boldsymbol{W}_{d,0:m} = (\boldsymbol{W}_{d,0}, \dots, \boldsymbol{W}_{d,m})$ = non independent random variables

Proposition (see Chapter 1 of the thesis)

Under Daily Independence Assumption V_0 is the value of problem $\mathcal{P}$

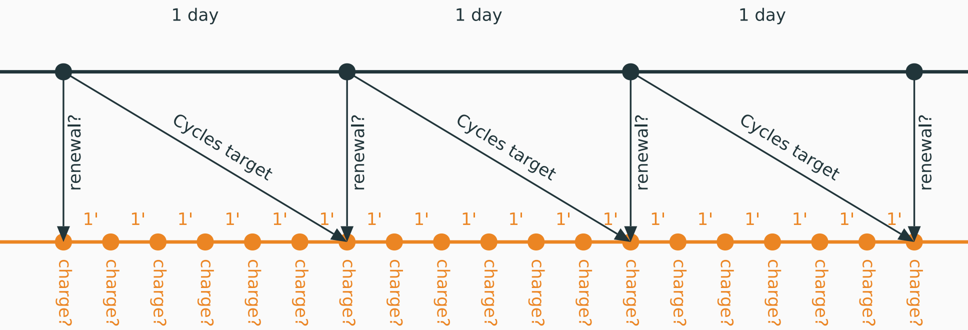
We present two efficient time decomposition algorithms to compute upper and lower bounds of the daily value functions

1. **Targets decomposition gives an upper bound**
2. **Weights decomposition gives a lower bound**

B: Algorithms for two-time scales stochastic optimization

Targets decomposition algorithm

Decomposing by sending targets



Stochastic targets decomposition

We introduce the stochastic target intraday problem

$$\phi_{(d,=)}\big(x_d, \boldsymbol{X}_{d+1}\big) = \min_{\boldsymbol{U}_d} \quad \mathbb{E}\Big[L_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d)\Big]$$

$$\text{s.t } f_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d) = \boldsymbol{X}_{d+1}$$

$$\sigma(\boldsymbol{U}_{d,m}) \subset \sigma(\boldsymbol{W}_{d,0:m})$$

Proposition

Under Daily Independence Assumption, V_d satisfies

$$V_d(x) = \min_{\boldsymbol{X} \in L^0(\Omega, \mathcal{F}, \mathbb{P}; \mathbb{X}_{d+1})} \Big(\phi_{(d,=)}\big(x, \boldsymbol{X}\big) + \mathbb{E}\big[V_{d+1}(\boldsymbol{X})\big]\Big)$$

$$\text{s.t } \sigma(\boldsymbol{X}) \subset \sigma(\boldsymbol{W}_d)$$

Relaxed stochastic targets decomposition

We introduce a relaxed target intraday problem

$$\phi_{(d,\geq)}\big(x_d, \boldsymbol{X}_{d+1}\big) = \min_{\boldsymbol{U}_d} \quad \mathbb{E}\Big[L_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d)\Big]$$

$$\text{s.t } f_d(x, \boldsymbol{U}_d, \boldsymbol{W}_d) \geq \boldsymbol{X}_{d+1}$$

$$\sigma(\boldsymbol{U}_{d,m}) \subset \sigma(\boldsymbol{W}_{d,0:m})$$

A relaxed daily value function

$$V_{(d,\geq)}(x) = \min_{\boldsymbol{X} \in L^0(\Omega,\mathcal{F},\mathbb{P};\mathbb{X}_{d+1})} \Big(\phi_{(d,\geq)}\big(x, \boldsymbol{X}\big) + \mathbb{E}\big[V_{(d+1,\geq)}(\boldsymbol{X})\big]\Big)$$

$$\text{s.t } \sigma(\boldsymbol{X}) \subset \sigma(\boldsymbol{W}_d)$$

Because of relaxation $V_{(d,\geq)} \leq V_d$ but $V_{(d,\geq)}$ is hard to compute due to the stochastic targets

Relaxed deterministic targets decomposition

Now we can define value functions with deterministic targets:

$$V_{(d,\geq,\mathbb{X}_{d+1})}(x) = \min_{X\in\mathbb{X}_{d+1}} \Big(\phi_{(d,\geq)}\big(x, X\big) + V_{(d+1,\geq,\mathbb{X}_{d+1})}(X)\Big)$$

Monotonicity Assumption

The daily value functions V_d are non-increasing

Theorem

Under Monotonicity Assumption

- $V_{(d,\geq)} = V_d$
- $V_{(d,\geq,\mathbb{X}_{d+1})} \geq V_{(d,\geq)} = V_d$

There are efficient ways to compute the upper bounds $V_{(d,\geq,\mathbb{X}_{d+1})}$

Numerical efficiency of deterministic targets decomposition

$$V_{(d,\geq,\mathbb{X}_{d+1})}(x) = \overbrace{\min_{X \in \mathbb{X}_{d+1}} \Big(\underbrace{\phi_{(d,\geq)}(x, X)}_{\text{Hard to compute}} + V_{(d+1,\geq,\mathbb{X}_{d+1})}(X) \Big)}^{\text{Easy to compute by dynamic programming}}$$

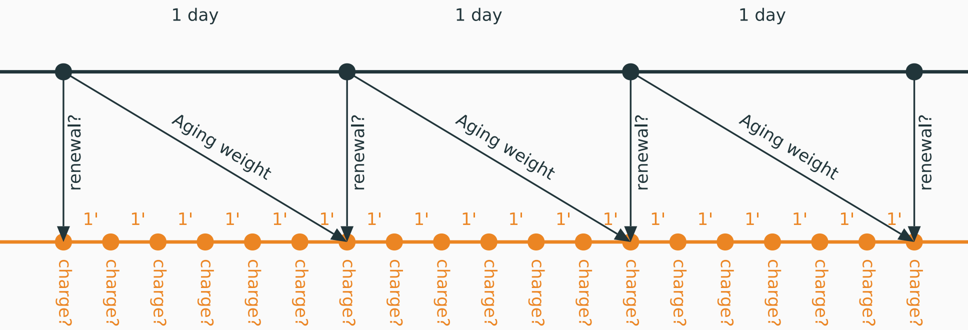
It is challenging to compute $\phi_{(d,\geq)}(x, X)$ for each couple (x, X) and each day d but

- We can exploit periodicity of the problem, e.g $\phi_{(d,\geq)} = \phi_{(0,\geq)}$
- In some cases $\phi_{(d,\geq)}(x, X) = \phi_{(d,\geq)}(x - X, 0)$
- We can parallelize $\phi_{(d,\geq)}$ computation on day d
- We can use any suitable method to solve the multistage intraday problems $\phi_{(d,\geq)}$ (SDP, scenario tree based SP...)

B: Algorithms for two-time scales stochastic optimization

Weights decomposition algorithm

Decomposing by sending weights



Stochastic weights decomposition

We introduce the dualized intraday problems

$$\psi_{(d,\star)}(x_d, \boldsymbol{\lambda}_{d+1}) = \min_{\boldsymbol{U}_d} \mathbb{E}\Big[L_d(x_d, \boldsymbol{U}_d, \boldsymbol{W}_d) + \langle \boldsymbol{\lambda}_{d+1}, f_d(x_d, \boldsymbol{U}_d, \boldsymbol{W}_d)\rangle\Big]$$

$$\text{s.t } \sigma(\boldsymbol{U}_{d,m}) \subset \sigma(\boldsymbol{W}_{d,0:m})$$

Note that $\psi_{(d,\star)}$ might be simpler than $\phi_{(d,\geq)}$ (state reduction)

Stochastic weights daily value function

$$V_{(d,\star)}(x_d) = \sup_{\boldsymbol{\lambda}_{d+1} \in L^q(\Omega,\mathcal{F},\mathbb{P};\Lambda_{d+1})} \psi_{(d,\star)}(x_d, \boldsymbol{\lambda}_{d+1}) - \Big(\mathbb{E}V_{(d+1,\star)}\Big)^{\star}(\boldsymbol{\lambda}_{d+1})$$

$$\text{s.t } \sigma(\boldsymbol{\lambda}_{d+1}) \subset \sigma(\boldsymbol{X}_{d+1})$$

where $\Big(\mathbb{E}V\Big)^{\star}(\boldsymbol{\lambda}_{d+1}) = \sup_{\boldsymbol{X} \in L^p(\Omega,\mathcal{F},\mathbb{P};\mathbb{X}_{d+1})} \langle \boldsymbol{\lambda}_{d+1}, \boldsymbol{X}\rangle - \mathbb{E}\big[V(\boldsymbol{X})\big]$
is the Fenchel transform of $\mathbb{E}V$

Deterministic weights decomposition

We define value functions with deterministic weights

$$V_{(d,\star,\mathbb{E})}(x_d) = \sup_{\lambda_{d+1} \in \Lambda_{d+1}} \psi_{(d,\star)}(x_d, \lambda_{d+1}) - V^*_{(d+1,\star,\mathbb{E})}(\lambda_{d+1})$$

Theorem

By weak duality and restriction, we get $V_{(d,\star,\mathbb{E})} \leq V_{(d,\star)} \leq V_d$

If $ri\Big(dom(\psi_{(d,\star)}(x_d, \cdot)) - dom(\mathbb{E} V_{d+1}(\cdot))\Big) \neq \emptyset$ and $\mathcal{P}$ is convex then we have $V_{(d,\star,\mathbb{E})} \leq V_{(d,\star)} = V_d$

There are efficient ways to compute the lower bounds $V_{(d,\star,\mathbb{E})}$

Numerical efficiency of deterministic weights decomposition

$$\overbrace{V_{(d,\star,\mathbb{E})}(x_d) = \sup_{\lambda_{d+1}\in\Lambda_{d+1}} \underbrace{\psi_{(d,\star)}(x_d,\lambda_{d+1})}_{\text{Hard to compute}} - V^*_{(d+1,\star,\mathbb{E})}(\lambda_{d+1})}^{\text{Easy to compute by dynamic programming}}$$

It is challenging to compute $\psi_{(d,\star)}(x,\lambda)$ for each couple (x,λ) and each day d but

- Under Monotonicity Assumption,
  we can restrict to positive weights $\lambda \geq 0$
- We can exploit periodicity of the problem $\psi_{(d,\star)} = \psi_{(0,\star)}$
- We can parallelize $\psi_{(d,\star)}$ computation on day d

We will use the daily value functions upper and lower bounds

Back to daily intraday problems with final costs

We obtained two bounds $V_{(d,\star,\mathbb{E})} \leq V_d \leq V_{(d,\geq,\mathbb{X}_{d+1})}$

Now we can solve all intraday problems with a final cost

$$\min_{\boldsymbol{X}_{d+1},\boldsymbol{U}_d} \mathbb{E}\left[L_d(x,\boldsymbol{U}_d,\boldsymbol{W}_d) + \widetilde{V}_{d+1}(\boldsymbol{X}_{d+1})\right]$$

$$\text{s.t } \boldsymbol{X}_{d+1} = f_d(x,\boldsymbol{U}_d,\boldsymbol{W}_d)$$

$$\sigma(\boldsymbol{U}_{d,m}) \subset \sigma(\boldsymbol{W}_{d,0:m})$$

with $\widetilde{V}_{d+1} = V_{(d,\geq,\mathbb{X}_{d+1})}$ or $\widetilde{V}_{d+1} = V_{(d,\star,\mathbb{E})}$

We obtain one targets and one weights minute scale policies

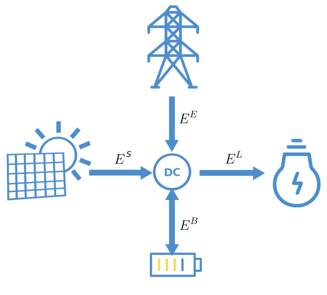
B: Algorithms for two-time scales stochastic optimization

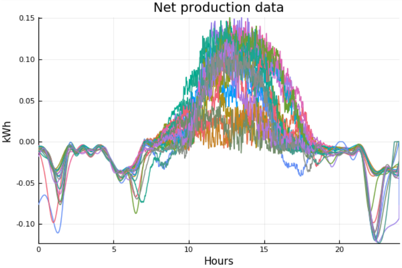
Numerical results

We present numerical results associated to two real use cases

Common data: load/production from a house with solar panels

1. Managing a given battery charge and health on 5 days
   to compare our algorithms to references on a “small” instance
2. Managing batteries purchases, charge and health on 7300 days
   to show that targets decomposition scales





Application 1: managing charge and aging of a battery

We control a battery

- capacity $c_0 = 13$ kWh
- $h_{0,0} = 100$ kWh of exchangeable energy (4 cycles remaining)
- over $D = 5$ days or $D \times M = 7200$ minutes
- with 1 day periodicity

We compare 4 algorithms

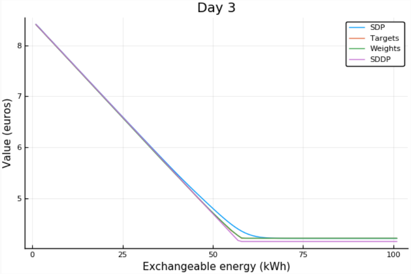
1. Stochastic dynamic programming
2. Stochastic dual dynamic programming
3. Targets decomposition (+ SDDP for intraday problems)
4. Weights decomposition (+ SDP for intraday problems)

Decomposition algorithms provide tighter bounds

We know that

- $V_d^{sddp} \leq V_d \leq V_d^{sdp}$
- $V_{(d,\star,\mathbb{E})} \leq V_d \leq V_{(d,\geq,\mathbb{X}_{d+1})}$

We observe that $V_d^{sddp} \leq V_{(d,\star,\mathbb{E})} \leq V_{(d,\geq,\mathbb{X}_{d+1})} \leq V_d^{sdp}$



We beat SDP and SDDP (that cannot fully handle 7200 stages)

Computation times and convergence

	SDP	Weights	SDDP	Targets
Total time (with parallelization)	22.5 min	5.0 min	3.6 min	0.41 min
Gap ($200 \times \frac{mc-v}{mc+v}$)	0.91 %	0.32 %	0.90 %	0.28 %

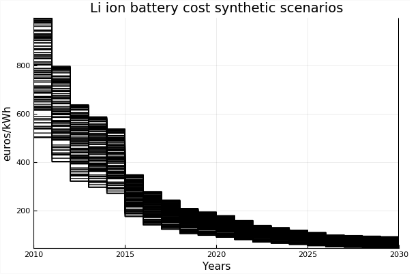
The Gap is between Monte Carlo simulation (upper bound)

and value functions at time 0

- Decomposition algorithms display smaller gaps
- Targets decompositon + SDDP is faster than SDDP
- Weights decomposition + SDP is faster than SDP

Application 2: managing batteries purchases, charge and aging

- 20 years, $10,512,000$ minutes, 1 day periodicity
- Battery capacity between 0 and 20 kWh
- Synthetic scenarios for batteries prices



SDP and SDDP fail to solve such a problem over $10,512,000$ stages!

Target decomposed SDDP solves $10,512,000$ stages problems

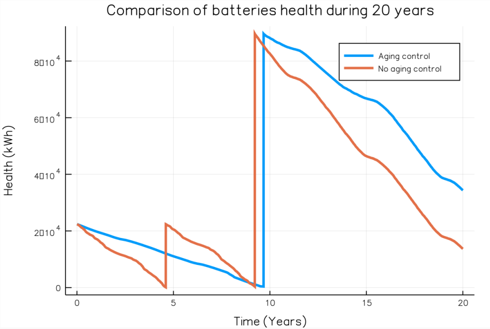
$$\overbrace{V_{(d,\geq,\mathbb{X}_{d+1})}(x) = \min_{X\in\mathbb{X}_{d+1}}\Big(\underbrace{\phi_{(d,\geq)}(x,X)}_{\text{Computing } \phi_{(d,\geq)}(\cdot,\cdot) \text{ with SDDP takes 60 min}} + V_{(d+1,\geq,\mathbb{X}_{d+1})}(X)\Big)}^{\text{Computing daily value functions by dynamic programming takes 45 min}}$$

Complexity: 45 min + $D \times$ 60 min

- Periodicity: 45 min + $N \times$ 60 min with $N << D$
- Parallelization: 45 min + 60 min

Does it pay to control aging?

We draw one battery prices scenario and one solar/demand scenario over $10,512,000$ minutes and simulate the policy of targets algorithm



We make a simulation of $10,512,000$ decisions in 45 minutes

We compare to a policy that does not control aging

- Without aging control: 3 battery purchases
- With aging control: 2 battery purchases

It pays to control aging with targets decomposition!

Conclusion

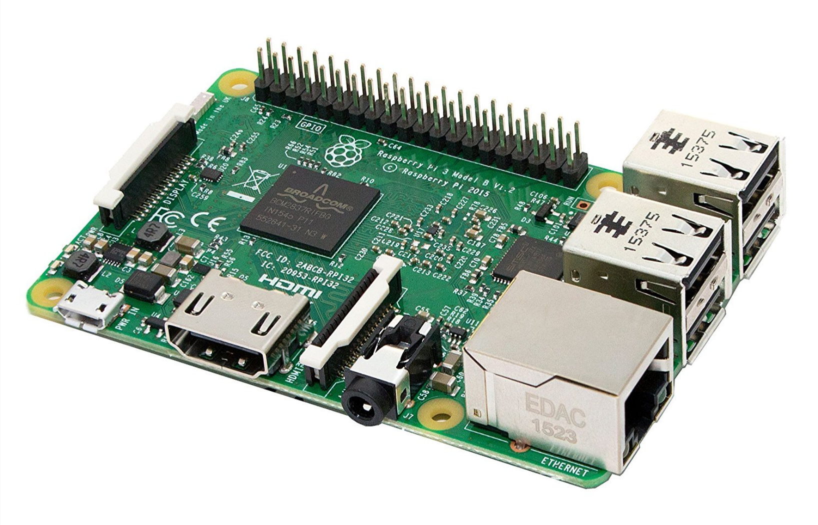
1. **We have solved problems with millions of time steps using targets decomposed SDDP**
2. **We have designed control strategies for sizing/charging/aging/investment of batteries**
3. **We have used our algorithms to improve results obtained with algorithms sensitive to the number of time steps (SDP, SDDP)**

What about implementability?

Our algorithms should be usable in real world applications!

C: Software and experimentations

We design embeddable strategies for real energy systems



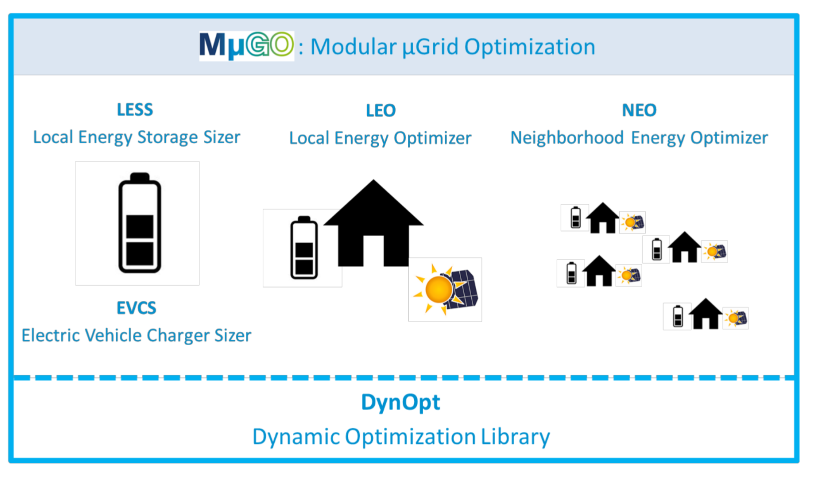
DynOpt.jl: a Julia package for stochastic optimization

```julia
using DynOpt

const Δt = 0.01   # discretization step of SDP

const n_stages = 6              # number of stages of the SP problem
const c = [sin(3*t)-1 for t in 1:n_stages-1]

const umax = [0.5]         # bounds on the control
const umin = [0.]

const xmax = [1.]
const xmin = [0.]

const ξmax = 0.3              # bounds on the noise
const ξmin = 0.
const n_ξ = 10                # discretization of the noise

dynamic!(new_x, t, x, u, xi) = new_x[1] = x[1] + u[1] - xi[1]
cost_t(t, x, u, w) =  c[t] * u[1]
constraints(args...) = true
final_cost(x) = 0

sp = DynOpt.DynamicOptimizationModel(n_stages, cost_t, dynamic!, constraints,
                                     final_cost, xmin, xmax, umin, umax,
                                     [HereAndNow])

proba = fill(1/n_ξ, n_ξ)
ξ_support = collect(range(ξmin, ξmax, length=n_ξ))
ξ_law = DiscreteMarginalLaw(ξ_support, proba)
W = WhiteNoise(vec(fill(ξ_law, n_stages-1)))

X = DiscreteSpace(xmin, xmax, [Δt])
U = DiscreteSpace(umin, umax, [Δt])

@time V = DynOpt.solvedp(sp, X, U, W)
```

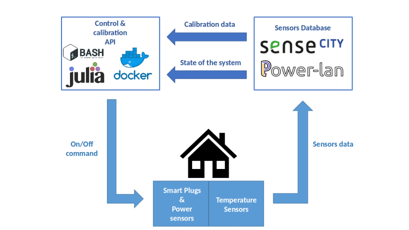
Features:

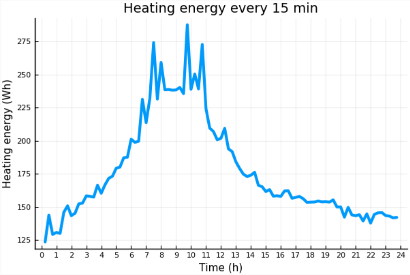
- Julia API to formulate stochastic optimization problems
- Julia API to build and simulate policies
- Resolution algorithms:
  - OLFC - MPC
  - SDP
  - Value iterations
  - SDDP
  - MIDAS (experimental)
- Easy deployment in any Linux machine using Docker

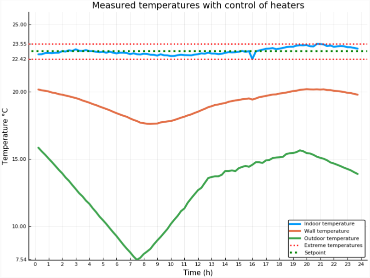
MμGO: energy management packaging of DynOpt



We used DynOpt to control the temperature in a house







Contributions and perspectives

- We have applied a fair method to compare stochastic optimal control methods on a subway station use case displaying promising energy efficiency results
  There remains to apply on a real demonstrator
- We have designed two time decomposition algorithms to tackle large multistage problems
  - Both can efficiently decompose problems for algorithms sensitive to the number of time stages (SDP, SDDP)
    There remains to experiment with other algorithms (MIDAS, Progressive Hedging, SDDIP...)
  - Targets decomposition is computationally efficient to solve very large problems (more than $10,000,000$ time stages)
    There remains to apply weights decomposition
- We have developed a generic library to solve stochastic optimization problems and have used it to manage the temperature in a house
  There remains to implement our library for larger microgrids with Efficacity and our partners

Thank you for your attention