FAST: Finally an SDDP Toolbox

The toolbox we've all been waiting for

Léopold Cambier

ICME, Stanford University
CME500 Seminar

October 10, 2016

Acknowledgements

- Work performed mainly at Université catholique de Louvain (Belgium), one year ago
- Joint work with Damien Scieur (Ph.D. student in convex optimization at ENS Paris/INRIA)
- Original SDDP algorithm is due to Pereira and Pinto (Multi-stage stochastic optimization applied to energy planning, Mathematical Programming, 52 (1991), 359-375).

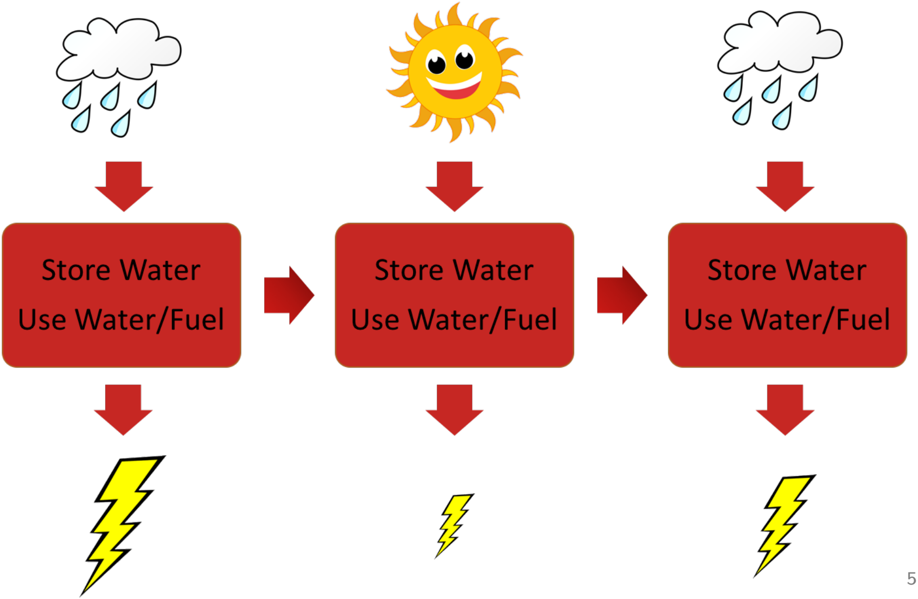
1. Motivations

2. The SDDP Algorithm

3. FAST

Motivations

A First Example : Hydro-Thermal Scheduling



Hydro-Thermal Scheduling :

- H time-steps ;

Hydro-Thermal Scheduling :

- H time-steps ;
- Meet known demand d_t ;

Hydro-Thermal Scheduling :

- H time-steps ;
- Meet known demand d_t ;
- using fuel p_t at price C...

Hydro-Thermal Scheduling :

- H time-steps ;
- Meet known demand d_t ;
- using fuel p_t at price C...
- or use y_t / store x_t water for free up to some limit W...

Hydro-Thermal Scheduling :

- H time-steps ;
- Meet known demand d_t ;
- using fuel p_t at price C...
- or use y_t / store x_t water for free up to some limit W...
- subject to random rain $r(\xi_t)$.

Hydro-Thermal Scheduling :

- H time-steps ;
- Meet known demand d_t ;
- using fuel p_t at price C...
- or use y_t / store x_t water for free up to some limit W...
- subject to random rain $r(\xi_t)$.

The goal is to minimize the *expected total cost*.

At each time t,

$$V^t(x_{t-1}) = \mathbb{E}_{\xi_t}\left(\min_{x_t,y_t,p_t} Cp_t(\xi_t) + V^{t+1}(x_t(\xi_t))\right)$$

s.t.

$$x_t(\xi_t) \le W$$

$$x_t(\xi_t) = x_{t-1} + r(\xi_t) - y_t(\xi_t)$$

$$p_t(\xi_t) + y_t(\xi_t) \ge d$$

$$x_t(\xi_t), y_t(\xi_t), p_t(\xi_t) \ge 0$$

At each time t,

$$V^t(x_{t-1}) = \mathbb{E}_{\xi_t}\left(\min_{x_t,y_t,p_t} Cp_t(\xi_t) + V^{t+1}(x_t(\xi_t))\right)$$

s.t.

$$x_t(\xi_t) \leq W$$

$$x_t(\xi_t) = x_{t-1} + r(\xi_t) - y_t(\xi_t)$$

$$p_t(\xi_t) + y_t(\xi_t) \geq d$$

$$x_t(\xi_t), y_t(\xi_t), p_t(\xi_t) \geq 0$$

Goal :

$$\min_{x_1,y_1,p_1} Cp_1 + V^2(x_1(\xi_1))$$

(Usually no randomness at time $t = 1$.)

LP formulation : we need 3 variables *per scenarios*. Say

- Rain takes k random values ;
- There are H time steps.

LP formulation : we need 3 variables *per scenarios*. Say

- Rain takes k random values ;
- There are H time steps.

$\Rightarrow k^H$ scenarios

LP formulation : we need 3 variables *per scenarios*. Say

- Rain takes k random values ;
- There are H time steps.

$\Rightarrow k^H$ scenarios
$\Rightarrow$ intractable using stanford LP solver

LP formulation : we need 3 variables *per scenarios*. Say

- Rain takes k random values ;
- There are H time steps.

$\Rightarrow$ k^H scenarios
$\Rightarrow$ intractable using stanford LP solver
$\Rightarrow$ need for an *iterative algorithm*.

The SDDP Algorithm

Fact 1 : V^t is convex

In general, we have, $\forall t = 1, \ldots, H$,

$$V^t(x_{t-1}) = \mathbb{E}_{\xi_t}\left(\min_{x_t(\xi_t)} c_H^\top x_t(\xi_t) + V^{t+1}(x_t(\xi_t))\right)$$

s.t., $\forall \xi_t$,

$$W_t x_t(\xi_t) = h(\xi_t) - T_t(\xi_t) x_{t-1}$$
$$x_t(\xi_t) \geq 0$$

Fact 1 : V^t is convex

Consider V^H :

$$V^H(x_{H-1}) = \mathbb{E}_{\xi_H}\left(\min_{x_H(\xi_H)} c_H^\top x_H(\xi_H)\right)$$

s.t., $\forall \xi_H$,

$$W_H x_H(\xi_H) = h(\xi_H) - T_H(\xi_H)x_{H-1}$$

$$x_H(\xi_H) \geq 0$$

Fact 1 : V^t is convex

Consider V^H :

$$V^H(x_{H-1}) = \mathbb{E}_{\xi_H}\left(\min_{x_H(\xi_H)} c_H^\top x_H(\xi_H)\right)$$

s.t., $\forall \xi_H$,

$$W_H x_H(\xi_H) = h(\xi_H) - T_H(\xi_H) x_{H-1}$$

$$x_H(\xi_H) \geq 0$$

$$\Updownarrow$$

Fact 1 : V^t is convex

Consider V^H :

$$V^H(x_{H-1}) = \mathbb{E}_{\xi_H}\left(\min_{x_H(\xi_H)} c_H^\top x_H(\xi_H)\right)$$

s.t., $\forall \xi_H$,

$$W_H x_H(\xi_H) = h(\xi_H) - T_H(\xi_H) x_{H-1}$$
$$x_H(\xi_H) \geq 0$$

$$\Updownarrow$$

$$f(b) = \sum_{k=1}^{K} g^{(k)}(b)$$

$$g^{(k)}(b) = \min_x c^\top x \text{ s.t. } Wx \geq h^{(k)} - T^{(k)} b, x \geq 0$$

Fact 1 : V^t is convex

$$f(\tilde{b}) = f(\theta b_1 + (1-\theta) b_2) \overset{?}{\leq} \theta f(b_1) + (1-\theta) f(b_2)$$

Fact 1 : V^t is convex

$$f(\tilde{b}) = f(\theta b_1 + (1-\theta) b_2) \overset{?}{\leq} \theta f(b_1) + (1-\theta) f(b_2)$$

- Let $\tilde{x}^{(k)} = \theta x_1^{(k)} + (1-\theta) x_2^{(k)}$;

Fact 1 : V^t is convex

$$f(\tilde{b}) = f(\theta b_1 + (1-\theta) b_2) \overset{?}{\leq} \theta f(b_1) + (1-\theta) f(b_2)$$

- Let $\tilde{x}^{(k)} = \theta x_1^{(k)} + (1-\theta) x_2^{(k)}$;
- We see that

$$\begin{aligned} W\tilde{x}^{(k)} &= \theta W x_1^{(k)} + (1-\theta) W x_2^{(k)} \\ &\geq \theta (h^{(k)} - T^{(k)} b_1) + (1-\theta)(h^{(k)} - T^{(k)} b_2) \\ &= h^{(k)} - T^{(k)} \tilde{b} \end{aligned}$$

and clearly, $\tilde{x}^{(k)} \geq 0$.

Fact 1 : V^t is convex

$$f(\tilde{b}) = f(\theta b_1 + (1-\theta) b_2) \overset{?}{\leq} \theta f(b_1) + (1-\theta) f(b_2)$$

- Let $\tilde{x}^{(k)} = \theta x_1^{(k)} + (1-\theta) x_2^{(k)}$;
- We see that

$$\begin{aligned} W\tilde{x}^{(k)} &= \theta W x_1^{(k)} + (1-\theta) W x_2^{(k)} \\ &\geq \theta(h^{(k)} - T^{(k)} b_1) + (1-\theta)(h^{(k)} - T^{(k)} b_2) \\ &= h^{(k)} - T^{(k)} \tilde{b} \end{aligned}$$

  and clearly, $\tilde{x}^{(k)} \geq 0$.
- $\Rightarrow$ so $\tilde{x}^{(k)}$ feasible with cost

$$\sum_{k=1}^{K} c^\top (\theta x_1^{(k)} + (1-\theta) x_2^{(k)}) = \theta f(b_1) + (1-\theta) f(b_2)$$

Fact 1 : V^t is convex

$$f(\tilde{b}) = f(\theta b_1 + (1-\theta) b_2) \overset{?}{\leq} \theta f(b_1) + (1-\theta) f(b_2)$$

- Let $\tilde{x}^{(k)} = \theta x_1^{(k)} + (1-\theta) x_2^{(k)}$;
- We see that

$$\begin{aligned} W\tilde{x}^{(k)} &= \theta W x_1^{(k)} + (1-\theta) W x_2^{(k)} \\ &\geq \theta(h^{(k)} - T^{(k)} b_1) + (1-\theta)(h^{(k)} - T^{(k)} b_2) \\ &= h^{(k)} - T^{(k)} \tilde{b} \end{aligned}$$

  and clearly, $\tilde{x}^{(k)} \geq 0$.
- $\Rightarrow$ so $\tilde{x}^{(k)}$ feasible with cost

$$\sum_{k=1}^{K} c^\top (\theta x_1^{(k)} + (1-\theta) x_2^{(k)}) = \theta f(b_1) + (1-\theta) f(b_2)$$

- So by minimizing, we can do better :

$$f(\tilde{b}) \leq \theta f(b_1) + (1-\theta) f(b_2)$$

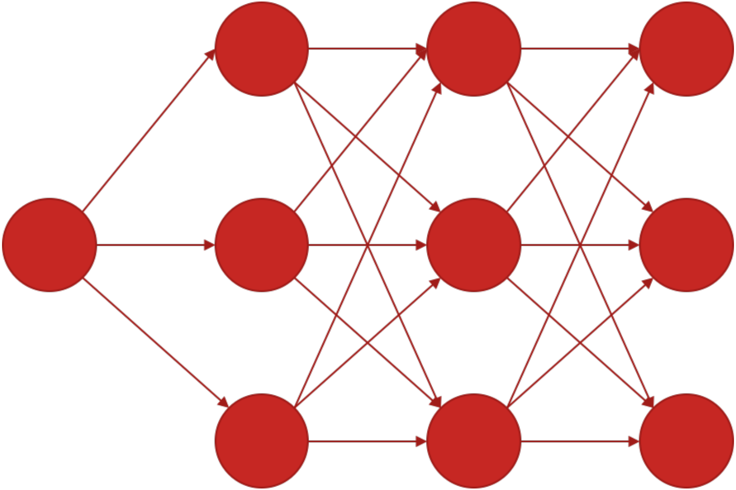
Fact 1 : V^t is convex

The same reasoning extends to all stage by induction and shows that

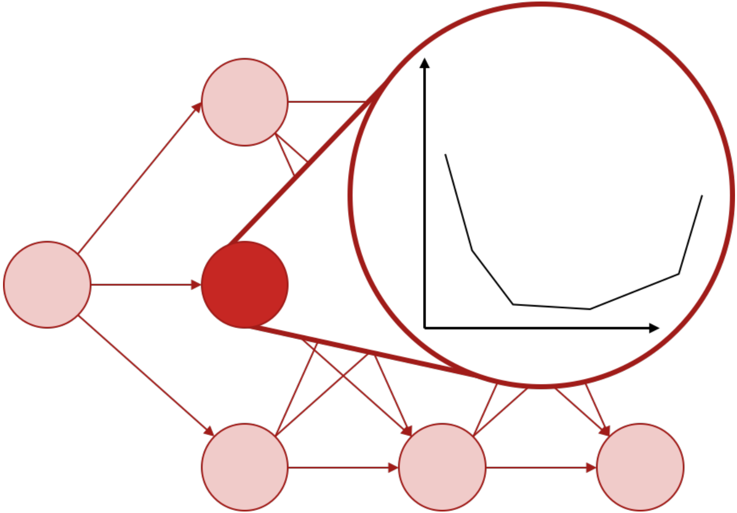
$$\forall t = 1, \ldots, H : V^t(x) \text{ is convex.}$$

Lattice

Use a Lattice to encode randomness

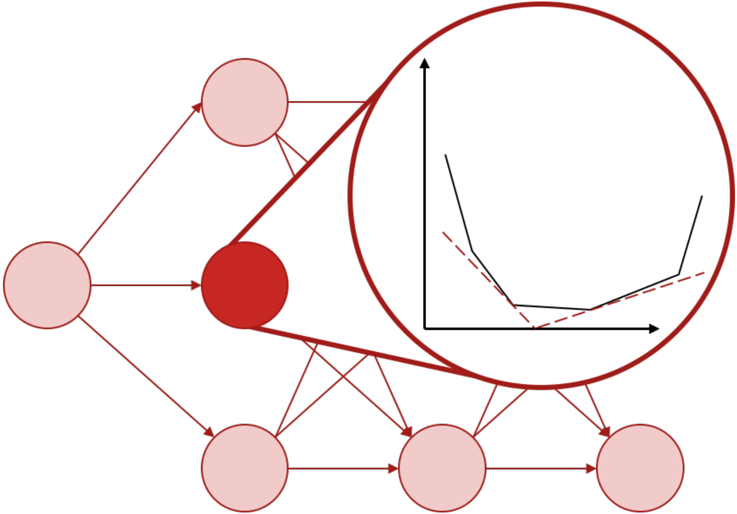


Lattice + V^t



Main Idea

We approximate V^t using *supporting hyperplanes*



Fact 2 : Supporting Hyperplanes

Using duality,

$$f(b) = \min c^\top x$$
$$Wx = h - Tb$$
$$x \geq 0$$

Fact 2 : Supporting Hyperplanes

Using duality,

$$
\begin{aligned}
f(b) = \min c^\top x \\
Wx = h - Tb \\
x \geq 0
\end{aligned}
$$

$$\Updownarrow$$

Fact 2 : Supporting Hyperplanes

Using duality,

$$f(b) = \min c^\top x$$
$$Wx = h - Tb$$
$$x \geq 0$$

$$\Updownarrow$$

$$f(b) = \max \pi^\top h - \pi^\top Tb$$
$$W^\top \pi \leq c$$

Fact 2 : Supporting Hyperplanes

1. Given b, compute $x^*(b)$ and $\pi^*(b)$:

$$f(b) = c^\top x^*(b) = (h - Tb)^\top \pi^*(b)$$

Fact 2 : Supporting Hyperplanes

1. Given b, compute $x^*(b)$ and $\pi^*(b)$:

$$f(b) = c^\top x^*(b) = (h - Tb)^\top \pi^*(b)$$

2. For any other $\tilde{b}$,

Fact 2 : Supporting Hyperplanes

1. Given b, compute $x^*(b)$ and $\pi^*(b)$:

$$f(b) = c^\top x^*(b) = (h - Tb)^\top \pi^*(b)$$

2. For any other $\tilde{b}$,

$$f(\tilde{b}) = \begin{cases} \max (h - T\tilde{b})^\top \pi \\ W^\top \pi \leq c \end{cases}$$

Fact 2 : Supporting Hyperplanes

1. Given b, compute $x^*(b)$ and $\pi^*(b)$:

$$f(b) = c^\top x^*(b) = (h - Tb)^\top \pi^*(b)$$

2. For any other $\tilde{b}$,

$$f(\tilde{b}) = \begin{cases} \max (h - T\tilde{b})^\top \pi \\ W^\top \pi \leq c \end{cases}$$

$$= \begin{cases} \max (h - Tb)^\top \pi + \pi^\top T(b - \tilde{b}) \\ W^\top \pi \leq c \end{cases}$$

Fact 2 : Supporting Hyperplanes

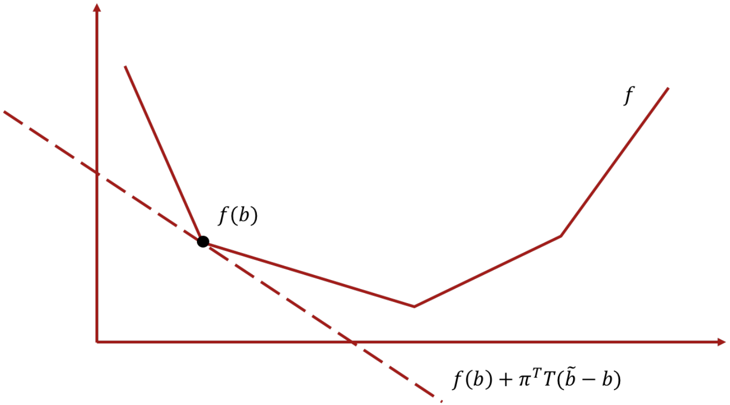
1. Given b, compute $x^*(b)$ and $\pi^*(b)$:

$$f(b) = c^\top x^*(b) = (h - Tb)^\top \pi^*(b)$$

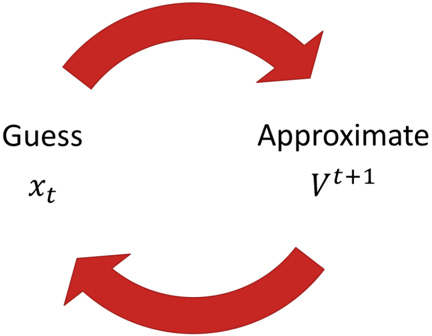
2. For any other $\tilde{b}$,

$$\begin{aligned} f(\tilde{b}) &= \begin{cases} \max (h - T\tilde{b})^\top \pi \\ W^\top \pi \leq c \end{cases} \\ &= \begin{cases} \max (h - Tb)^\top \pi + \pi^\top T(b - \tilde{b}) \\ W^\top \pi \leq c \end{cases} \\ &\geq f(b) + \pi^{*\top}(b) T(\tilde{b} - b) \end{aligned}$$

Fact 2 : Supporting Hyperplanes



SDDP Algorithm



SDDP Algorithm

- While UpperBound >> LowerBound
  1. Forward pass (Approximate objective + Guess x_t)
     1. Ramdomly pick M paths in the lattice ;
     2. Solve for x_t for each pass and approximate mean cost C ;
     3. Use cost at stage 1 to compute a lower-bound on the cost L.
  2. Backward pass (Build supporting hyperplanes)
     For $t = H, \ldots, 2$:
     1. Using the forward solutions x_{t-1} solve each node for x_t and π_t ;
     2. Use π_t to build supporting hyperplane for V^t.

FAST

Why FAST

- SDDP = Great algorithm, but hard to program properly ;
- Hard to change the model easily since relies on dual multipliers ;
- Usual modeling languages (AMPL, ...) not well-suited to programming ;
- No existing open-source toolbox.

FAST

- FAST (Finally An SDDP Toolbox)
- Open-Source
- In "plain" Matlab
- Useful for rapid prototyping ; handles both
  - Modeling (linear)
  - Solver (SDDP + commercial or open-source solver for subproblems)

Using FAST

Hydro-Scheduling

Back to hydro-scheduling...
At each time t,

$$V^t(x_{t-1}) = \mathbb{E}\left(\min_{x_t, y_t, p_t} Cp_t + V^{t+1}(x_t)\right)$$

s.t.

$$x_t \leq W$$

$$x_t = x_{t-1} + r(\xi_t) - y_t$$

$$p_t + y_t \geq d$$

$$x_t, y_t, p_t \geq 0$$

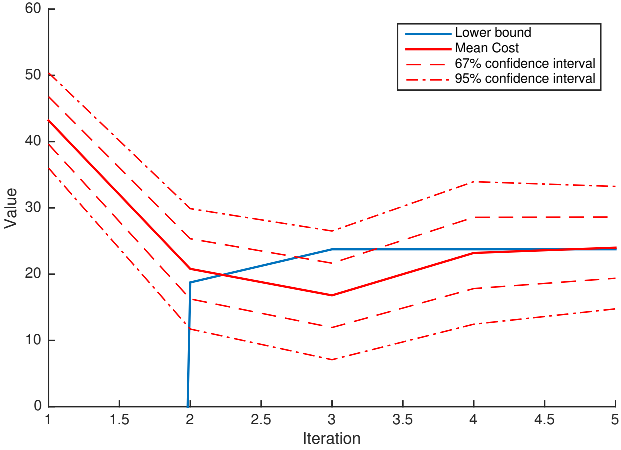
NLDS

```matlab
function [cntr, obj] = nlds(scenario, x, y, p)
t = scenario.getTime();
i = scenario.getIndex();
% Objective
fuel_cost = 5 * p(t);
% Constraints
reservoir_max_level = x(t) <= 8;
meet_demand = p(t) + y(t) >= 5;
positivity = [x(t) >= 0, y(t) >= 0, p(t) >= 0];
if t == 1
    reservoir_level = x(1) + y(1) == rainfall(t,i)
else
    reservoir_level = x(t) - x(t-1) + y(t) == rainfall(t,i);
end
obj = fuel_cost;
cntr = [reservoir_max_level, ...
        meet_demand, ...
        positivity, ...
        reservoir_level];
end
```

Lattice + SDDP

```
% Creating a simple 5 stages lattice with 2 nodes at each stage
lattice = Lattice.latticeEasy(5, 2);
% Run SDDP
params = sddpSettings('algo.McCount',25, ...
                      'stop.iterationMax',10,...
                      'stop.pereiraCoef',2,...
                      'solver','gurobi');
x = sddpVar(5) ; % The reservoir level at time t
y = sddpVar(5) ; % For how much we use the water at time t
p = sddpVar(5) ; % For how much we use the fuel generator at time t
lattice = compileLattice(lattice,@(scenario)nlds(scenario,x,y,p),params);
output = sddp(lattice,params);
% Visualise output
plotOutput(output);
```

Output


Lower bound
Mean Cost
67% confidence interval
95% confidence interval
Value
Iteration
60
50
40
30
20
10
0
1
1.5
2
2.5
3
3.5
4
4.5
5

Other Features

- Use any solver you want (Linprog, Gurobi, Cplex, Mosek)
- Deterministic exact version for debugging
- Extract cuts
- Extraxt optimal solution
- Various options

Links

- Github : https://github.com/leopoldcambier/FAST
- Website (tutorial, examples) : http://www.baemerick.be/fast/

Questions ?