Regularized optimization techniques for multistage stochastic programming

Felipe Beltrán[1], Welington de Oliveira[2], Guilherme Fredo[1], Erlon Finardi[1]

[1] UFSC/LabPlan – Universidade Federal de Santa Catarina

[2] MINES ParisTech, PSL-Research University
CMA - Centre de Mathématiques Appliquées

SESO 2018, May 22, 2018

Multistage stochastic linear programs - MSLPs

Consider the multistage linear stochastic program:

$$\min_{\substack{A_1 x_1 = b_1 \\ x_1 \geq 0}} c_1^\top x_1 + \mathbb{E}\left[\min_{\substack{B_2 x_1 + A_2 x_2 = b_2 \\ x_2 \geq 0}} c_2^\top x_2 + \mathbb{E}\left[\cdots + \mathbb{E}[\min_{\substack{B_T x_{T-1} + A_T x_T = b_T \\ x_T \geq 0}} c_T^\top x_T]\right]\right]$$

- Some elements of the data $\xi_t = (c_t, B_t, A_t, b_t)$ depend on uncertainties ($\xi_1 = (c_1, A_1, b_1)$ is supposed to be known)
- We view the sequence $\xi = (\xi_1, \ldots, \xi_T)$ as a stochastic process
- We say that the stochastic process is stagewise independent if ξ_{t+1} is independent of $\xi_{[t]} := (\xi_1, \ldots, \xi_t)$, $t = 1, \ldots, T$

In multistage stochastic linear programs (MSLP) the uncertain data is revealed gradually over time $t = 1, \ldots, T$

It is convenient to write the dynamic equations of the above MSLP

Multistage stochastic linear programming

$$\min_{\substack{A_1 x_1 = b_1 \\ x_1 \geq 0}} c_1^\top x_1 + \mathbb{E}\left[\min_{\substack{B_2 x_1 + A_2 x_2 = b_2 \\ x_2 \geq 0}} c_2^\top x_2 + \mathbb{E}\left[\cdots + \mathbb{E}[\min_{\substack{B_T x_{T-1} + A_T x_T = b_T \\ x_T \geq 0}} c_T^\top x_T]\right]\right]$$

Dynamic equations: $\xi_t = (c_t, B_t, A_t, b_t)$

- Stage $t = T$

$$Q_T(x_{T-1}, \xi_t) := \min_{\substack{B_T x_{T-1} + A_T x_T = b_T \\ x_T \geq 0}} c_T^\top x_T$$

- At stages $t = 2, \ldots, T-1$

$$Q_t(x_{t-1}, \xi_t) := \min_{\substack{B_t x_{t-1} + A_t x_t = b_t \\ x_t \geq 0}} c_t^\top x_t + \mathcal{Q}_{t+1}(x_t)$$

- Stage $t = 1$

$$\min_{\substack{A_1 x_1 = b_1 \\ x_1 \geq 0}} c_1^\top x_1 + \mathcal{Q}_2(x_1)$$

- **Recourse function**

$$\mathcal{Q}_{t+1}(x_t) := \mathbb{E}\left[Q_{t+1}(x_t, \xi_{t+1})\right]$$

Multistage stochastic linear programming

$$\min_{\substack{A_1 x_1 = b_1 \\ x_1 \geq 0}} c_1^\top x_1 + \mathbb{E}\left[\min_{\substack{B_2 x_1 + A_2 x_2 = b_2 \\ x_2 \geq 0}} c_2^\top x_2 + \mathbb{E}\left[\cdots + \mathbb{E}[\min_{\substack{B_T x_{T-1} + A_T x_T = b_T \\ x_T \geq 0}} c_T^\top x_T]\right]\right]$$

Dynamic equations: cutting-plane approximations

- Stage $t = T$

$$Q_T(x_{T-1}, \xi_t) := \min_{\substack{B_T x_{T-1} + A_T x_T = b_T \\ x_T \geq 0}} c_T^\top x_T$$

- At stages $t = 2, \ldots, T-1$

$$\underline{Q}_t(x_{t-1}, \xi_t) := \min_{\substack{B_t x_{t-1} + A_t x_t = b_t \\ x_t \geq 0}} c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t)$$

- Stage $t = 1$

$$\min_{\substack{A_1 x_1 = b_1 \\ x_1 \geq 0}} c_1^\top x_1 + \check{\mathcal{Q}}_2(x_1)$$

- **$\check{\mathcal{Q}}_t$ is a cutting-plane approximation of the recourse function**

$$\check{\mathcal{Q}}_{t+1}(x_t) := \max_{j=1,\ldots,k} \{\beta_{t+1}^{j\,\top} x_t + \alpha_{t+1}^j\} \leq \mathcal{Q}_{t+1}(x_t)$$

SDDP algorithm

Stochastic Dual Dynamic Programming, Pereira and Pinto, 1991

Let $\{\xi^1, \dots, \xi^N\}$ be the set of scenarios representing the considered tree

Forward pass

At iteration k, the forward step of the SDDP algorithm consists in choosing $M < N$ scenarios $\mathcal{J}^k := \{\tilde{\xi}^1, \dots, \tilde{\xi}^M\}$ and computing x_t^k as a solution of

$$\begin{aligned} \min_{x_t \geq 0} \quad & {c_t}^\top x_t + \check{\mathcal{Q}}_{t+1}^k(x_t) \\ \text{s.t.} \quad & A_t x_t = b_t - B_t x_{t-1}^k \end{aligned}$$

for all[1] $t = 2, \dots, T$ and all $\tilde{\xi} \in \mathcal{J}^k$

- Estimate an upper bound for the MSLP

Backward pass

By considering the new trial points x_t^k, in this step the algorithm comes backward computing new cuts to improve the cutting-plane models to $\check{\mathcal{Q}}_t^{k+1}$, $t = T, T-1, \dots, 2$

- A valid lower bound is available

$$\begin{aligned} \underline{z}^k = \min_{x_1 \geq 0} \quad & {c_1}^\top x_1 + \check{\mathcal{Q}}_2^{k+1}(x_2) \\ \text{s.t.} \quad & A_1 x_1 = b_1 \end{aligned}$$

[1] We define $\check{\mathcal{Q}}_{T+1} \equiv 0$.

SDDP: ITS ENGINE

The workhorse in this kind of decomposition is the *Kelley's cutting-plane method*

$$
\begin{aligned}
x_t^k \in \arg\min_{x_t \geq 0} \quad & {c_t}^\top x_t + \check{\mathcal{Q}}_{t+1}^k(x_t) \\
\text{s.t.} \quad & A_t x_t = b_t - B_t x_{t-1}^k
\end{aligned}
$$

- Unstable, slow convergence

In two-stage stochastic programming (and in deterministic convex optimization) regularized methods provide faster convergence than the Kelley's cutting-plane method (L-Shaped method)

It is then a natural idea to try to accelerate the SDDP algorithm by employing some kind of regularization

Multistage regularized decomposition

The *Regularized Decomposition* for MSLPs ([2]) replaces[3] the cutting-plane master problem at each visited node (j, t) with

$$x_t^k \in \arg\min_{x_t \geq 0} \quad {c_t}^\top x_t + \check{\mathcal{Q}}_{t+1}^k(x_t) + \frac{1}{2\tau_k} \|G(x_t - \hat{x}_t)\|^2$$
$$\text{s.t.} \quad A_t x_t = b_t - B_t x_{t-1}^k$$

- where G is a square matrix, typically $G = I$, $G = [0\ I]$, $G = [I\ 0]$
- It is necessary that $\tau_k \to +\infty$ for convergence
- The authors define $\hat{x}_t = x_t^{k-1}$
- Notice that $\hat{x}_t = x_t^{k-1}$ may not be feasible at iteration k and visited node (j, t)!

---

[2] T. Asanov, Q. Powell. Regularized Decomposition of High-Dimensional Multistage Stochastic Programs with Markov Uncertainty, 2018

[3] A related idea is present in (Sen and Zhou 2014)

Multistage level decomposition

The *Level Decomposition* for MSLPs ([4]) replaces the cutting-plane master problem at each visited node (j,t) with

$$\begin{aligned} x_t^k \in \arg\min_{x_t \geq 0} \quad & \frac{1}{2}\|x_t\|^2 \\ \text{s.t.} \quad & A_t x_t = b_t - B_t x_{t-1}^k \\ & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t) \leq f_{\texttt{lev}}^t \end{aligned}$$

- Ideally, $f_{\texttt{lev}}^t$ is the optimal value of

$$\begin{aligned} \min_{x_t \geq 0} \quad & c_t{}^\top x_t + \mathcal{Q}_{t+1}(x_t) \\ \text{s.t.} \quad & A_t x_t = b_t - B_t x_{t-1}^k \end{aligned}$$

  In practice, $f_{\texttt{lev}}^t$ is defined via heuristics

- When $f_{\texttt{lev}}^t$ is too small the level QP is infeasible: it is necessary to resort back to the standard SDDP in this case

[4] W. van Ackooij, W. de Oliveira and Y. Song. On regularization with normal solutions in decomposition methods for multistage stochastic programming, 2017

Multistage level decomposition

There exists a constant $\bar{\tau} > 0$ such that for all $\tau \geq \bar{\tau}$ the x_t-solution of the QP[5]

$$\begin{cases} \min\limits_{r,x_t} & r + \frac{1}{2\tau}\|x_t\|^2 \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t) \leq r \\ & x_t \geq 0,\ f_{\texttt{lev}}^t \leq r \end{cases}$$

either solves

$$\min_{x_t \geq 0} \frac{1}{2}\|x_t\|^2 \quad \text{s.t. } A_t x_t = b_t - B_t x_{t-1}^k, \quad c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t) \leq f_{\texttt{lev}}^t$$

(if it is feasible)

Level Iterate

or computes the normal solution of

$$\min_{x_t \geq 0} c_t^\top x_t + \check{\mathcal{Q}}_{t+1}(x_t) \quad \text{s.t. } A_t x_t = b_t - B_t x_{t-1}^k$$

Normal Iterate

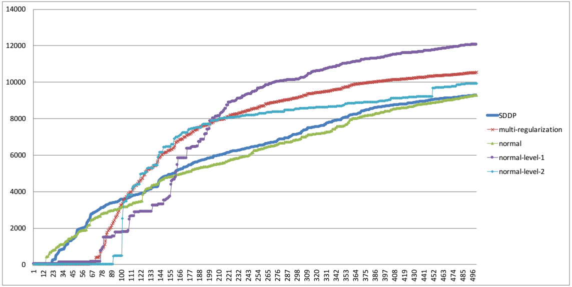
Normal solution is the solution of minimal norm

Specialized QP solver: K.C. Kiwiel. Finding normal solutions in piecewise linear programming. Applied Mathematics and Optimization, 32(3):235–254, 1995.

[5] Finite perturbation of convex programs, Ferris and Mangasarian 1991

- The objective of the model is to minimize the (expected) total cost over T stages, including the power generation cost and the penalty of insufficient power to satisfy the demand, under the uncertainty of the amount of rainfall in the future
- Power can be generated by 30 hydro power plants (16 with reservoir) and 38 thermal power plants
- The inflow of water into each reservoir is random, and a finite set of scenarios for each time stage (monthly by default) in the planning horizon is available from prediction

- **Improvement of LB w.r.t. SDDP**: $T = 61$, $\#\ \text{Nodes}_t = 50$



- The objective of the model is to minimize the (expected) total cost over T stages, including the power generation cost and the penalty of insufficient power to satisfy the demand, under the uncertainty of the amount of rainfall in the future
- Power can be generated by 30 hydro power plants (16 with reservoir) and 38 thermal power plants
- The inflow of water into each reservoir is random, and a finite set of scenarios for each time stage (monthly by default) in the planning horizon is available from prediction
- Fixed time limit of one hour (3600 seconds) for every solver and employ 10 sample paths in each forward step
- An exception is that for the multistage regularized decomposition, which uses only one sample path per iteration as suggested by Asamov and Powell, 2015 (we also observed in our numerical experiments that this variant yielded better results)

• Improvement of LB w.r.t. SDDP

T	# Nodes$_t$	Reg. Decomp.		Level Decomp.	
		LB	Iter	LB	Iter
25	20	0.2%	1222	3.3%	130
	50	2.8%	1125	5.9%	122
	80	2.9%	1067	8.0%	115
61	20	-10.8%	817	-2.2%	87
	50	-8.4%	744	2.5%	79
	80	-4.3%	692	8.8%	74
97	20	-12.2%	685	-6.9%	73
	50	-0.3%	617	3.3%	67
	80	-1.1%	567	2.0%	61

CMA Centre de Mathématiques Appliquées

- The objective of the model is to minimize the (expected) total cost over T stages, including the power generation cost and the penalty of insufficient power to satisfy the demand, under the uncertainty of the amount of rainfall in the future
- Power can be generated by 30 hydro power plants (16 with reservoir) and 38 thermal power plants
- The inflow of water into each reservoir is random, and a finite set of scenarios for each time stage (monthly by default) in the planning horizon is available from prediction
- Fixed time limit of one hour (3600 seconds) for every solver and employ 10 sample paths in each forward step
- An exception is that for the multistage regularized decomposition, which uses only one sample path per iteration as suggested by Asamov and Powell, 2015 (we also observed in our numerical experiments that this variant yielded better results)

• Improvement of LB w.r.t. SDDP

T	# Nodes$_t$	Reg. Decomp.		Level Decomp.	
		LB	Iter	LB	Iter
25	20	0.2%	1222	3.3%	130
	50	2.8%	1125	5.9%	122
	80	2.9%	1067	8.0%	115
61	20	-10.8%	817	-2.2%	87
	50	-8.4%	744	2.5%	79
	80	-4.3%	692	8.8%	74
97	20	-12.2%	685	-6.9%	73
	50	-0.3%	617	3.3%	67
	80	-1.1%	567	2.0%	61

- The Forward step is more time consuming
- QPs are harder than LPs
- What if we try a simpler regularization scheme?

Centre de Mathématiques Appliquées

Central cutting-plane algorithm: Chebyshev center

Consider the deterministic convex optimization problem

$$\min_{x\in X} f(x), \quad f:\Re^n \to \Re$$

and let

$$\check{f}^k(x) := \max_{j\le k}\{\beta^{j\,\top} x + \alpha^j\}$$

be a cutting-plane for f

To overcome the instability inherent in the cutting-plane method, the work ([6]) proposes to define iterates in X as the Chebyshev center of the polyhedron:

$$S_k := \left\{ (x,r)\in\Re^{n+1} \;\middle|\; \begin{array}{lll} r & \le \bar{z}^k & \\ \beta^{j\,\top} x + \alpha^j & \le r\,, & \forall j \le k \end{array} \right\}$$

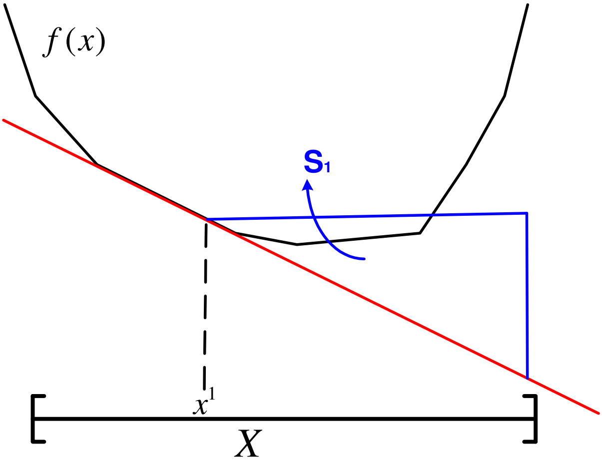
where $\bar{z}^k$ is an upper bound on the optimal value of the above problem

This amounts to solving the LP

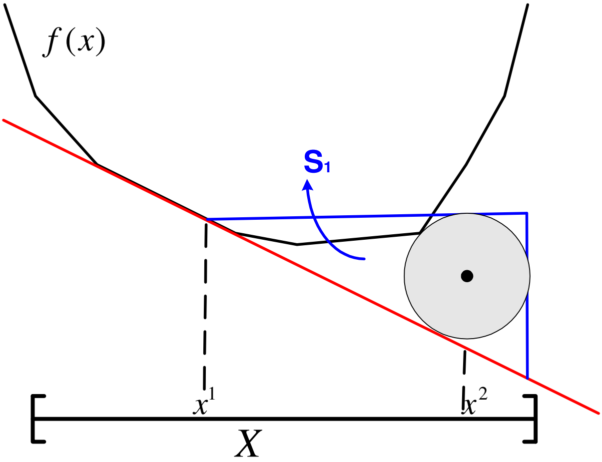
$$(x^{k+1},\tilde{r},\tilde{\sigma}) \in \arg \left\{ \begin{array}{lll} \max_{x,r,\sigma} & \sigma & \\ s.t. & r+\sigma & \le \bar{z}^k \\ & \beta^{j\,\top} x + \alpha^j + \sigma\sqrt{1+\|\beta^j\|^2} & \le r,\; j=1,\ldots,k \\ & x\in X,\; r,\sigma\in\Re, & \end{array}\right.$$

[6] J. Elzinga and T. G. Moore. A central cutting plane algorithm for the convex programming problem, Math. Program., 1975.

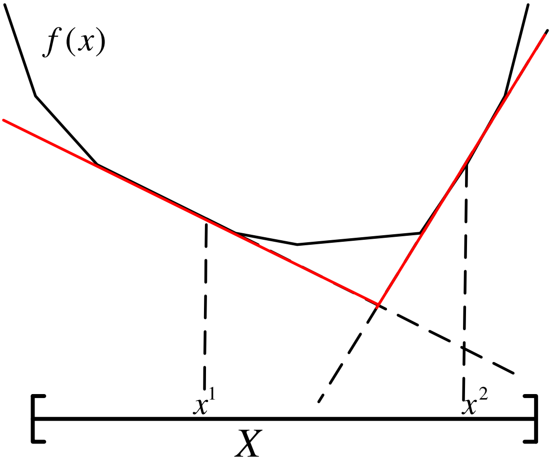
Central Path Cutting-plane Method



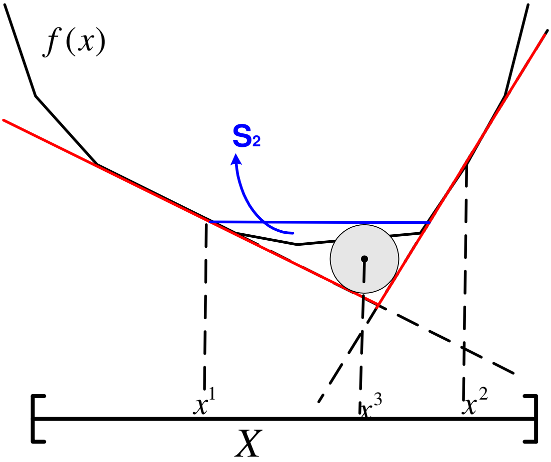
Central Path Cutting-plane Method



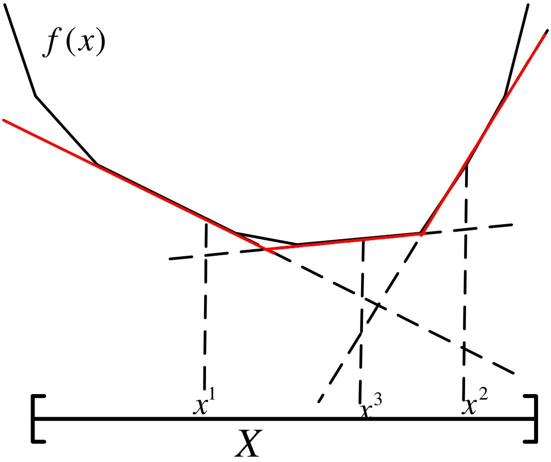
Central Path Cutting-plane Method



Central Path Cutting-plane Method

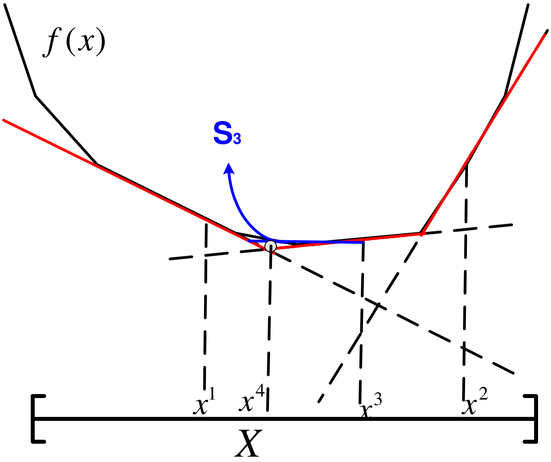


Central Path Cutting-plane Method



CMA Centre de Mathématiques Appliquées

Central Path Cutting-plane Method



Central cutting-plane algorithm for MSLPs

Our idea is to employ this technique to SDDP

To this end, we replace the SDDP subproblems in the forward step

$$\begin{cases} \min\limits_{x_t \geq 0} & c_t^\top x_t + \check{\mathcal{Q}}_{t+1}^k(x_t) \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \end{cases} \equiv \begin{cases} \min\limits_{x_t, r_t} & r_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & (c_t + \beta_{t+1}^j)^\top x_t + \alpha_{t+1}^j \leq r_t \quad \forall j \leq k \\ & x_t \geq 0 \end{cases}$$

with

$$\begin{cases} \max\limits_{x_t, r_t, \sigma_t} & \sigma_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & (c_t + \beta_{t+1}^j)^\top x_t + \alpha_{t+1}^j + \sigma_t \sqrt{1 + \|(c_t + \beta_{t+1}^j)\|} \leq r_t \quad \forall j \leq k \\ & \sigma_t + r_t \leq \bar{z}(\xi_{[t]})^k \\ & x_t \geq 0,\, r_t,\, \sigma_t \in \Re \end{cases}$$

- This is a LP with only two additional variables!
- The upper bound $\bar{z}(\xi_{[t]})^k$ is not reliable. moreover, it depends on each node of the underlying scenario tree and decisions made in previous stages
- This hinders a direct application of the *central path CP* algorithm to multistage stochastic programming

CMA Centre de Mathématiques Appliquées

Central cutting-plane algorithm for MSLPs

$$\begin{cases} \max\limits_{x_t, r_t, \sigma_t} & \sigma_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & (c_t + \beta_{t+1}^j)^\top x_t + \alpha_{t+1}^j + \sigma_t \sqrt{1 + \|(c_t + \beta_{t+1}^j)\|} \leq r_t \quad \forall j \leq k \\ & \sigma_t + r_t \leq \bar{z}(\xi_{[t]})^k \\ & x_t \geq 0, \; r_t, \sigma_t \in \Re \end{cases}$$

If we fix the radius σ_t then the (difficult-to-estimate) upper bound can be dismissed. We can thus reformulate the subproblem as

The new proposal (only in the forward step)

$$\begin{cases} \min\limits_{x_t, r_t} & c_t^\top x + r_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & \beta_{t+1}^{j\,\top} x_t + \alpha_{t+1}^j + \bar{\sigma}_t \sqrt{1 + \|(c_t + \beta_{t+1}^j)\|} \leq r_t \quad \forall j \leq k \\ & x_t \geq 0, \; r_t \in \Re \end{cases}$$

In this case, $\bar{\sigma}_t$ is a parameter

The classical SDDP

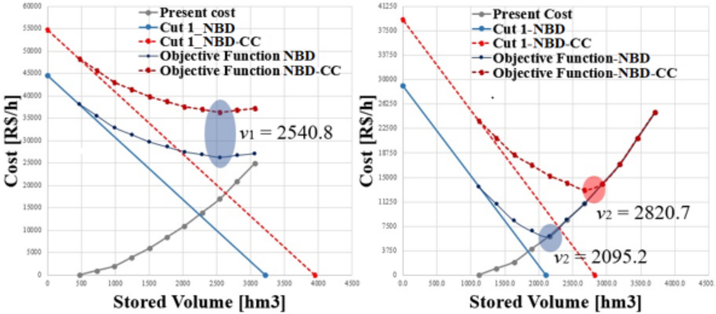
$$\begin{cases} \min\limits_{x_t, r_t} & c_t^\top x + r_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & \beta_{t+1}^{j\,\top} x_t + \alpha_{t+1}^j \leq r_t \quad \forall j \leq k \\ & x_t \geq 0, \; r_t \in \Re \end{cases}$$

CMA Centre de Mathématiques Appliquées

Geometric interpretation: example

Consider a reduced power system with, 1 hydro, 5 five thermal plants, $T = 4$ time steps and 125 scenarios (# Nodes$_t = 5$, $t = 2, 3, 4$). The classical SDDP computes the exact solution and cost $R\$34,051.40$ in 9 iterations, while the SDDP with Chebyshev centers requires only 6 iterations

$$
\begin{cases}
\min \; 10 f_{1t} + 20 f_{2t} + 25 f_{2t} + 30 f_{4t} + 40 f_{5t} + r_{t+1} \\
\text{s.t.} \; f_{1t} + f_{2t} + f_{2t} + f_{4t} + f_{5t} + q_t = 1000 \\
\qquad v_t + K_0 \cdot (q_t + s_t) = v_{t-1} + K_0 \cdot \xi_t^n \\
\qquad \beta_{t+1}^{jT} v_t + \alpha_{t+1}^j + \varpi_t \left\| (1, c_t + \beta_{t+1}^j) \right\|_* \le r_{t+1}, \; j \in J_{t+1} \\
\qquad p_{it} \le 200, \; q_t \le 1000, \; s_t \ge 0, v_t \le 4000, r_{t+1} \in \Re.
\end{cases}
$$



(a) Stage 1. (b) Stage 2.

SDDP algorithm with Chebyshev centers

Let $\mathcal{S}_N := \{\xi^1, \ldots, \xi^N\}$ be the set of scenarios representing the considered tree

Forward pass

At iteration k, choose $M < N$ scenarios $\mathcal{J}^k := \{\tilde{\xi}^1, \ldots, \tilde{\xi}^M\}$ and solve

$$x_t^k \in \begin{cases} \min\limits_{x_t, r_t} & c_t^\top x + r_t \\ \text{s.t.} & A_t x_t = b_t - B_t x_{t-1}^k \\ & \beta_{t+1}^{j\,\top} x_t + \alpha_{t+1}^j + \bar{\sigma}_t^k \sqrt{1 + \|(c_t + \beta_{t+1}^j)\|} \le r_t \qquad \forall j \le k \\ & x_t \ge 0,\ r_t \in \Re \end{cases}$$

for all[7] $t = 2, \ldots, T$ and all $\tilde{\xi} \in \mathcal{J}^k$

Backward pass

As in the SDDP algorithm: compute new cuts and update the cutting-plane model $\check{\mathcal{Q}}_{t+1}$ but considering only points x_t^k related to the sample set $\mathcal{J}^k$

If $\sigma_t^k = 0$ for all stages and iterates, then the above algorithm is nothing but a SDDP algorithm

Convergence analysis. It follows from the SDDP analysis: just make sure that $\lim_{k\to\infty} \sigma_t^k = 0$ for all t

[7] We define $\check{\mathcal{Q}}_{T+1} \equiv 0$.

Numerical assessment

- We consider the Brazilian multistage hydro-thermal power generation planning problem with individualized decisions per plant over a five-year planning horizon ($T = 60$) with monthly decisions
- The objective of the model is to minimize the total cost over the horizon, including power generation cost and penalty of insufficient power to satisfy the demand, under the uncertainty of the amount of rainfall in the future
- Power can be generated by 294 power plants (153 hydro and 141 thermal plants
- Every stage t and every node of the scenario tree is composed of 2 886 variables and 1 459 constraints
- The inflow uncertainties are handled via a PAR model with a scenario tree with 20 realizations per stage ($N = 20^{59}$ scenarios for each one of the 153 hydro plants)
- The comparative analysis is carried out among five solvers for three different seeds, which generate distinct scenario trees
- The forward step considers 216 scenarios per iteration with resampling
- We considered risk-neutral and risk-averse cases
- A parallel processing algorithm strategy is used within servers that have a configuration Xeon CPU with 2.60GHz, using 8 threads and 128 GB RAM. All LPs are solved using Gurobi called from environment C++.

CMA Centre de Mathématiques Appliquées

Numerical assessment

In our experiments, we consider the following variants of SDDP for comparison:

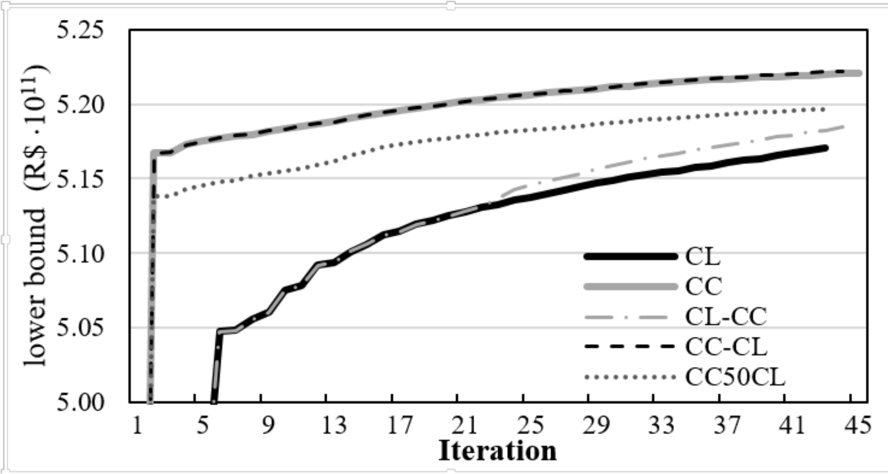
1. The classical SDDP algorithm *CL*
2. The new proposal: SDDP algorithm with Chebyshev centers *CC*
3. CC-CL the 24 first hours with CC and the last 24 hours with the classical SDDP
4. CL-CC the inverse of the previous strategy
5. CC50CL Chebyshev centers are computed for 50% of scenarios in the forward step, and for the other half we solve the SDDP subproblems

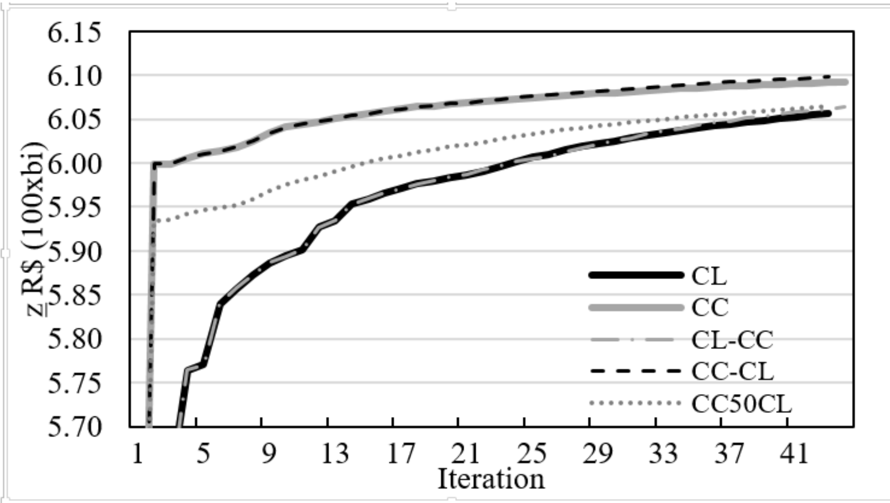
With the purpose of obtaining reasonable lower bound stabilizations all our solvers were stopped with 48 hours of CPU time

The considered heuristic for updating the parameter $\overline{\sigma}_t$ is:

$$\overline{\sigma}_t = 5 \cdot 10^5 \cdot \frac{\underline{z}^k - \underline{z}^{k-1}}{\sum_{\text{if } nk<10,\ k=0 \text{ else } k=nk-10}^{nk} (\underline{z}^k - \underline{z}^{k-1})} \cdot \left(1 - e^{-10(\underline{z}^k - \underline{z}^{k-1})}\right).$$

The constant $5 \cdot 10^5$ was tuned for a better scalability of our problem. Such a constant is weighted by a proportion of the lower bound improvement w.r.t. the last ten iterations. The main idea is to make $\overline{\sigma}_t$ a function of the lower bound progress. For instance, such parameter increases when the lower bound presents a high increase rate since, at this point, the convergence is not achieved, and new regions of the cost-to-go-function can be explored. Otherwise, a stabilization of the $\underline{z}$ indicates that new trial points are not improving the model and $\overline{\sigma}_t$ must decrease to attend the algorithm convergence. Finally, the last term ensures that $\overline{\sigma}_t = 0$ when the lower bound rate increase is null. Several simple updating rules can be formulated by incorporating the dynamic of the problem


5.25
5.20
5.15
5.10
5.05
5.00
lower bound (R$ ·10^11)
1
5
9
13
17
21
25
29
33
37
41
45
Iteration
CL
CC
CL-CC
CC-CL
CC50CL



CMA
Centre de Mathématiques Appliquées

TABLE II – Comparative analysis.

Solver	Risk-level	Lower bound increase (%) scenario tree			Time reduction to reach the CL lower bound (%) scenario tree		
	λ / ρ (%)	1	2	3	1	2	3
CC	0 / 0	0.98	0.86	0.73	99.34	98.32	96.10
CL-CC		0.28	0.27	0.13	37.39	37.58	36.85
CC-CL		1.00	0.90	0.75	99.34	98.15	96.09
CC50CL		0.51	0.38	0.24	89.10	88.93	60.17
CC	0.1 /10%	0.60	0.51	0.46	91.94	78.19	78.93
CL-CC		0.12	0.05	0.04	20.29	20.31	11.39
CC-CL		0.69	0.62	0.55	90.55	85.77	78.85
CC50CL		0.13	0.15	0.10	25.60	26.55	18.03

To comprehend the results, consider the lower bound value increase of 0.86 highlighted in Table II. This means that, for the risk-neutral case, the CC provides a $\underline{z}$ value 0.86% greater than the CL one (R$ $5.1964 \cdot 10^{11}$ in Table I) in 48 hours of processing. Considering the CL increasing rate of $\underline{z}$ for the last 10 iterations, the solver CL would require approximately 24 extra iterations (95 hours) to reach the value given by CC. For the risk-averse case, the CL would reach an increase of 0.51% with 15 more iterations (50 hours).

TABLE IV - Comparative simulation results.

Solver	Risk-level λ / ρ (%)	Expected cost difference (%) scenario tree 1	2	3
CC	0 / 0	0.24	0.02	0.02
CL-CC		0.52	1.29	1.61
CC-CL		-0.06	-0.21	-0.21
CC50CL		-0.06	-0.12	-0.01
CC	0.1 /10%	0.001	0.002	0.001
CL-CC		0.004	0.005	0.003
CC-CL		-0.13	-0.16	-0.18
CC50CL		-0.01	0.11	0.11

Notice that CC-CL is the only solver that obtains significant cost reductions in all scenario trees and risk-measure models. It is then natural to inquire why CC solvers, which achieves similarly $\underline{z}$ values in relation to CC-CL, are more expensive in out-of-sample simulation than CL ones. The key aspect is that the CC forward step provides trial points in a small region of the problem (near to the solution); accordingly, the cutting-plane model is improved exclusively in such region, obtaining non-sufficiently robust policies for other feasible regions of the state variables. On the other hand, the second half of the CC-CL optimization process permits to construct cuts in a broader state variables domain.

Concluding remarks

1. Regularize the forward step of SDDP by increasing the cuts' intercept
2. The strategy's inspiration is the Central Path Cutting-plane model, that uses Chebyshev centers
3. Instead of using estimated upper bounds we fix the ball radius
4. The proposed approach requires
   - solving LPs along the forward step
   - properly tuning the radius parameters
5. No need for stability centers
6. The proposed technique computes better lower bounds and (nearly-optimal) feasible policies in less than 90% of the CPU time required by the classical SDDP.

Thank you!

Contact

✉ welington.oliveira@mines-paristech.fr

www.oliveira.mat.br