Solving stochastic programming problems with randomized scenario sampling

Gilles Bareilles[1], Dmitry Grishchenko[1], Franck Iutzeler[1], Yassine Laguel[1], Jérôme Malick[2]

[1] Univ. Grenoble Alpes, Grenoble, France
{prenom.nom}@univ-grenoble-alpes.fr
[2] CNRS, LJK, Grenoble, France
{prenom.nom}@univ-grenoble-alpes.fr

Mots-clés : *Multistage Stochastic Programming, Progressive Hedging, Julia.*

Multistage Stochastic Programming [3, Chap. 9] consists in minimizing the expected cost of some decision over a set of coupled scenarios. Progressive Hedging is a popular strategy for solving such problems, based on scenario decomposition. In this talk, we present a randomized version of this algorithm able to compute an update as soon as a scenario subproblem is solved. This is of crucial importance when run on parallel computing architectures. We prove that the randomized version has the same converge properties as the standard one and we release an easy-to-use Julia toolbox.

1 Large-scale (but structured) multistage problems

We briefly recall the general multistage stochastic problems and the existing scenario-based decomposition algorithm. We follow closely the notation of the textbook [3, Chap. 9].

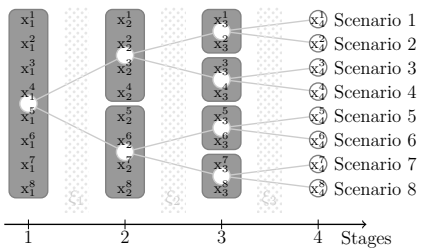
For each scenario $s \in \{1, .., S\}$, we associate a random variable ξ^s encompassing its stochastic nature and its probability p_s. We denote by $f^s(\mathrm{x}^s) = f(\mathrm{x}^s, \xi^s)$ the cost of the decision sequence $\mathrm{x}^s \in \mathbb{R}^n$ taken for scenario s when it occurs. Both variables are split in bits x_t^s, $t \in \{1, \dots, T\}$, which model the sequential aspect of the problem at hand. A simple formulation of a Multistage Stochastic Program is

$$\min_{\mathrm{x} \in \mathcal{W}} f(\mathrm{x}) = \sum_{s=1}^{S} p_s f^s(\mathrm{x}^s).$$

where the difficulty lies in the coupling of the scenarios by *non-anticipativity* constraints :

$$\mathcal{W} = \left\{ \mathrm{x} \in \mathbb{R}^{S \times n} : \forall s_1, s_2 \in \{1, ..., S\} \;\middle|\; \begin{array}{l} \mathrm{x}_1^{s_1} = \mathrm{x}_1^{s_2} \\ \text{ and} \\ \forall t \in \{2, ..., T\}\, \mathrm{x}_t^{s_1} = \mathrm{x}_t^{s_2} \;\text{ if } \xi_{[1,t-1]}^{s_1} = \xi_{[1,t-1]}^{s_2} \end{array} \right\} \quad (1)$$

This indeed leads to a block-coupling of total decision variable $\mathrm{x} \in \mathbb{R}^{S \times n}$ of the form :



The *Progressive hedging* algorithm solves this problem by solving each scenario independently and then projecting the obtained solution on the non-anticipativity constraints :

$$\begin{cases} \mathrm{y}^{k+1,s} = \mathrm{argmin}_{y\in\mathbb{R}^n} \left\{ f^s(y) + \frac{1}{2\mu} \left\| y - \mathrm{x}^{k,s} + \mu\, \mathrm{u}^{k,s} \right\|^2 \right\} \text{ for all } s = 1,..,S \\ \mathrm{x}_t^{k+1,s} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \mathrm{y}_t^{k+1,\sigma} \text{ for all } s = 1,..,S \text{ and } t = 1,..,T \\ \mathrm{u}^{k+1} = \mathrm{u}^k + \frac{1}{\mu}(\mathrm{y}^{k+1} - \mathrm{x}^{k+1}) \end{cases}$$

where we use the notion of *bundle* $\mathcal{B}_t^s$ of the scenarios that are indistinguishable from scenario s at time t (see Eq. (1)).

2 Decomposition by randomized sampling

Instead of involving all the scenarios at each iteration, our randomized version of progressive hedging samples one scenario at each iteration :

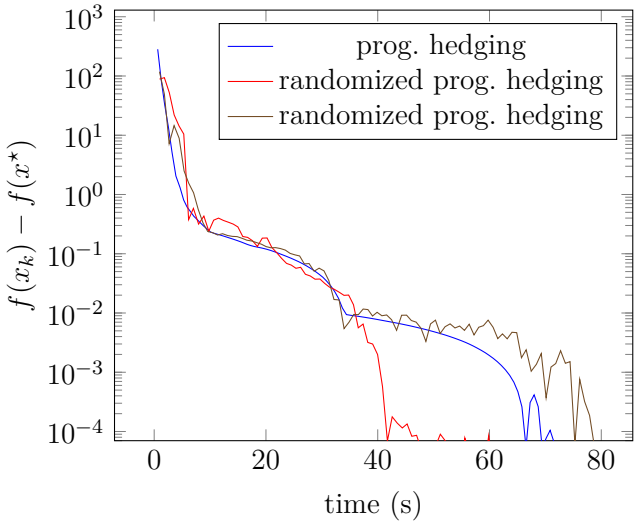
$$\begin{cases} \text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s \\ \mathrm{x}_t^{k+1,s^k} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma\, \mathrm{z}_t^{k,\sigma} \text{ for all } t = 1,..,T & \text{projection only on constraints involving } s^k \\ \mathrm{y}^{k+1,s^k} = \mathrm{argmin}_{y\in\mathbb{R}^n} \left\{ f^{s^k}(y) + \frac{1}{2\mu} \left\| y - 2\,\mathrm{x}^{k+1,s^k} + \mathrm{z}^{k,s^k} \right\|^2 \right\} & \text{optimization sub-problem only concerning scenario } s^k \\ \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} + \mathrm{y}^{k+1,s^k} - \mathrm{x}^{k+1,s^k} \text{ and } \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \text{ for all } s \neq s^k \end{cases}$$

Return : $\tilde{\mathrm{x}}^{k+1} = \dfrac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \displaystyle\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \mathrm{y}_t^{k+1,\sigma}$ for all $s = 1,..,S$ and $t = 1,..,T$

Our approach is based on the randomized operator theory (see e.g. [1]). Mathematically, we show, in the convex case, that the sequence (x^k) generated by this method is feasible ($\mathrm{x}^k \in \mathcal{W}$ a.s. for all k) and converges almost surely to a solution of the problem.

In practive, randomly sampling one scenario per iteration enables us to dramatically reduce the cost of an iteration (both for the scenario solving part and the projection part) which results in better performances.

We implemented our method in an easy-to-use Julia toolbox (expected release : Dec. 2020) along with several examples. We plot the performance of our method compared on a simple hydro-thermal scheduling problem following from [2] and the FAST toolbox [1].



Références

[1] Franck Iutzeler, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. Asynchronous distributed optimization using a randomized alternating direction method of multipliers. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 3671–3676. IEEE, 2013.

[2] Mario VF Pereira and Leontina MVG Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical programming*, 52(1-3) :359–375, 1991.

[3] Andrzej Ruszczyński. Decomposition methods. In *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*, pages 141 – 211. Elsevier, 2003.

1. `https://stanford.edu/ lcambier/cgi-bin/fast/index.php`