A RANDOMIZED PROGRESSIVE HEDGING METHOD FOR MULTI-STAGE STOCHASTIC PROGRAMMING

ϖ

Abstract. RPH

1. Elements of intro, thrown randomly

Progressive Hedging introduced by Wets = basic algorithm textbook Shapiro&cie + numerous applications (cite some of them). Based on a different and randomized take on the link between Douglas-Rachford splitting and Progressive Hedging formalized notably in the handbook (Ruszczyński, 2003, Chap. 9).

PH = decomposition by scenarios = iterative method with subproblems corresponding to a given scenario of uncertainty.

Opens the way to distributed version of the algorithm + asynchrone

Extension for C-var.

2. Multistage Stochastic Programs

2.1. **Stochastic programs.** In mathematical optimization, stochastic programming is the field tackling optimization problems involving uncertainty. The goal of these problems is to find a feasible solution that is *optimal* in some sense relatively to the uncertainty of the problem. Is is generally assumed that the underlying probability distributions are known (or at least can be estimated) which leads to the generic formulation

$$\min_{x\in\mathbb{R}^n} \mathcal{R}(f(x(\xi),\xi)) \tag{2.1}$$

where $\mathcal{R}$ denotes the risk measure considered in the problem. For an extensive review of stochastic programming, see e.g. Shapiro et al. (2009).

2.2. **Multistage model.** Multistage stochastic problems have important applications in energy optimization, finance, etc. In this section, we lay down the multistage model considered in this paper as well as our notation; for an extensive presentation of multistage stochastic programming, see e.g. (Ruszczyński and Shapiro, 2003, Sec. 3).

In the multistage setting, the uncertainty of the problem is revealed sequentially, delimitating T *stages*. More precisely, the random variable ξ is split into $T-1$ chunks, $\xi = (\xi_1, .., \xi_{T-1})$, and the problem at hand is to decide at each stage $t = 1, .., T$ what is the optimal action, $\mathrm{x}_t(\xi_{[1,t-1]})$, given the *past* observations $\xi_{[1,t-1]} := (\xi_1, .., \xi_{t-1})$. The global variable of this problem thus writes

$$\mathrm{x}(\xi) = (\mathrm{x}_1, \mathrm{x}_2(\xi_1), .., \mathrm{x}_T(\xi_{[1,t-1]}) \in \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_T} = \mathbb{R}^n$$

where $(n_1, .., n_T)$ are the size of the decision variable at each stage and $n = \sum_{t=1}^{T} n_t$ is the total size of the problem.

In this paper, we will focus on the case where the random variable ξ can only take a finite number S of values called *scenarios* and denoted by $\xi^1, .., \xi^S$. Each scenario $s \in \{1, .., S\}$ occurs with probability $p^s = \mathbb{P}[\xi = \xi^S]$ and is revealed in T stages $\xi^S = (\xi_1^S, .., \xi_T^S)$, it is thus natural to represent the scenarios as the outcome of a probability tree, see Fig. 1.

Figure 1: Scenarios as the outcomes of a probability tree.

For each scenario $s \in \{1, .., S\}$, the target of multistage stochastic programming is to provide a decision $\mathrm{x}^s = (\mathrm{x}_1, \mathrm{x}_2(\xi_1^s), .., \mathrm{x}_T(\xi^s_{[1,t-1]}))$, and thus the full problem variable writes

$$\mathrm{x} = \begin{pmatrix} \mathrm{x}_1 & \mathrm{x}_2\left(\xi_1^1\right) & \dots & \mathrm{x}_{T-1}\left(\xi^1_{[1,..,T-2]}\right) & \mathrm{x}_T\left(\xi^1_{[1,..,T-1]}\right) \\ \mathrm{x}_1 & \mathrm{x}_2\left(\xi_1^2\right) & \dots & \mathrm{x}_{T-1}\left(\xi^2_{[1,..,T-2]}\right) & \mathrm{x}_T\left(\xi^2_{[1,..,T-1]}\right) \\ \vdots & & & & \vdots \\ \vdots & & & & \vdots \\ \mathrm{x}_1 & x_2\left(\xi_1^S\right) & \dots & \mathrm{x}_{T-1}\left(\xi^S_{[1,..,T-2]}\right) & \mathrm{x}_T\left(\xi^S_{[1,..,T-1]}\right) \end{pmatrix} = \begin{pmatrix} \mathrm{x}^1 \\ \mathrm{x}^2 \\ \vdots \\ \vdots \\ \mathrm{x}^S \end{pmatrix} \tag{2.2}$$

and thus $\mathrm{x} \in \mathbb{R}^{S \times n}$.

From Eq. (2.2), we see that by construction of the randomness, the decision at stage 1, x_1, must be the same for all the scenarios. Indeed, as no random variables have been observed, the user does not have any information about the scenarios. In the same vein, given the specificity of these random variables, an important feature of finite multistage problems is that if two scenarios s_1 and s_2 coincide up to stage $t-1$ (i.e. $\xi^{s_1}_{[1,t-1]} = \xi^{s_2}_{[1,t-1]}$), then the obtained decision variables must be equal up to stage t (i.e. $(\mathrm{x}_1, \mathrm{x}_2(\xi_1^{s_1}), .., \mathrm{x}_t(\xi^{s_1}_{[1,t-1]})) = (\mathrm{x}_1, \mathrm{x}_2(\xi_1^{s_2}).., \mathrm{x}_t(\xi^{s_2}_{[1,t-1]}))$).

These constraints are called *non-anticipativity* and will denoted in the following by

$$\mathcal{W} = \left\{ \mathrm{x} \in \mathbb{R}^{S \times n} : \forall s_1, s_2 \in \{1, .., S\} \;\middle|\; \begin{array}{l} \mathrm{x}_1^{s_1} = \mathrm{x}_1^{s_2} \\ \text{ and} \\ \forall t \in \{2, ..T\}\, \mathrm{x}_t^{s_1} = \mathrm{x}_t^{s_2} \text{ if } \xi^{s_1}_{[1,t-1]} = \xi^{s_2}_{[1,t-1]} \end{array} \right\}. \tag{2.3}$$

It is easy to see that these non-anticipativity constraints lead to a variable x with a block structure as depicted in Fig. 2.

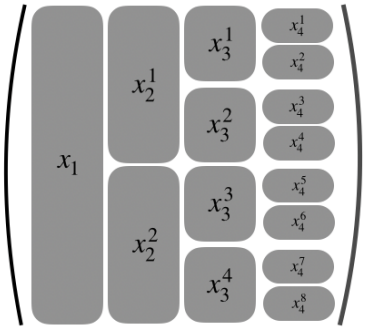

Figure 2: Block structure of a nonanticipative strategy in a 4-stage stochastic problem

2.3. **Study of the Risk neutral-case.** For each scenario $s \in \{1, .., S\}$, let us denote by $f^s(\mathrm{x}^s) = f(\mathrm{x}\,\xi^s, \xi^s)$ the cost of the decision sequence ξ^s taken for scenarios s when it occurs. In the risk-neutral case, $\mathcal{R}$ is simply the expectation and thus Problem (2.1) rewrites as

$$\min_{\mathrm{x} \in \mathcal{W}} \sum_{s=1}^{S} p_s f^s(\mathrm{x}^s) \tag{2.4}$$

which will be out problem of interest in the remaining of the paper.

3. Efficient Randomized versions of Progressive Hedging

In this section, we present the celebrated Progressive Hedging algorithm to solve Problem (2.4) and discuss its practical implementation. Then, our main contribution is to propose two randomized versions of Progressive Hedging: one leading to a single-thread algorithm with cheap iterations, and one leading to an asynchronous multi-thread method.

3.1. **Algorithm.** The difficulty of Problem (2.4) comes from the fact that there can be a large number of scenarios since, given the tree structure of the scenarios, it can be exponential in the number of stages and that these scenarios are all linked by the non-anticipativity constraints. For that reason, algorithms decoupling the objective (separable among the scenarios) and the constraints (linking them) were developed. Among them, *Progressive Hedging* is arguably one of the most popular methods. Much like the Alternating Direction Method of Multipliers (ADMM, see notably Lions and Mercier (1979)), the Progressive Hedging (PH) algorithm relies on performing Douglas-Rachford splitting on the subgradient of the dual problem[1]. Out of clarity, we recall below the progressive hedging algorithm as Alg. 1, its main proof elements following (Ruszczyński, 2003, Sec. 9) are recalled in Appendix A.

Algorithm 1 Progressive Hedging

Initialize: $\mathrm{x}^0 \in \mathcal{W}, \mathrm{u}^0 \in \mathcal{W}^\perp, \mu > 0$

For $k = 0, 1, ..$ **do:**

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

Theorem 3.1. *Consider a multistage stochastic problem* (2.4) *with positive probabilities* ($p_s > 0$ *for all* s)*, convex objectives* f^s*, and constraints* $\mathcal{W}$ *as defined in* (2.3) *such that* $\operatorname{ri}\operatorname{dom} f \cap \mathcal{W} \neq \emptyset$ *holds. Then, the sequence* (x^k) *generated by the Progressive Hedging algorithm is feasible* ($\mathrm{x}^k \in \mathcal{W}$ *for all* k) *and converges to a solution of Problem* (2.4).

Implementation. The Progressive Hedging (Alg. 1) consists of three steps:

i) a proximal operation over all scenarios. When the scenario costs (f^s) are (constrained) linear or quadratic functions[2], this amounts to computing the solution of a (constrained) quadratic program per scenario. This operation can be parallelized over all scenarios; however, since the number of scenarios might be large (exponential in the number of stages), the parallelization might be limited by the number of available threads.

ii) a projection over the non-anticipativity constraints $\mathcal{W}$. This step involves all the variables computed at the previous step which is constraining for the parallelization. Nevertheless, this projection is relatively cheap to compute.

iii) a simple variable update.

3.2. **Randomized version.** Thanks to parallel established between Progressive Hedging and Douglas-Rachford/ADMM, we can employ randomized coordinate descent methods (see e.g. Iutzeler et al. (2013)) on the underlying operator subsuming progressive hedging in order to produce a randomized version of the algorithm. These methods consist in updating only a subset of the coordinates at each iteration (for instances, the ones corresponding to *one* scenario) and leaving the other unchanged. As this has to be performed at the operator level, one has to be careful with this operation as parts of the operator link the variables with each other. These derivations as well as the proof of Th. 3.2 are provided in Appendix B.

Theorem 3.2. *Consider a multistage stochastic problem* (2.4) *with positive probabilities* ($p_s > 0$ *for all* s)*, convex objectives* f^s*, and constraints* $\mathcal{W}$ *as defined in* (2.3) *such that* $\operatorname{ri}\operatorname{dom} f \cap \mathcal{W} \neq \emptyset$ *holds. Then, the sequence* (x^k) *generated by the Randomized Progressive Hedging algorithm is feasible* ($\mathrm{x}^k \in \mathcal{W}$ *a.s. for all* k) *and converges almost surely to a solution of Problem* (2.4).

Implementation. The Randomized Progressive Hedging (Alg. 2) consists of four steps:

i) a sampling of one[3] scenario s^k among all with the probabilities (q_s).

[1] See Eckstein and Bertsekas (1992) for a formal link between Douglas-Rachford and ADMM and (Ruszczyński, 2003, Sec. 9) for a formal link between Douglas-Rachford and PH.

[2] The constraints are incorporated in the function for simplicity as minimizing $\tilde{f}^s$ over some set $\mathcal{C}^s$ is the same as minimizing $f^s = \tilde{f}^s + \iota_{\mathcal{C}^s}$ over the full space (with $\iota_{\mathcal{C}^s}(x) = 0$ if $x \in \mathcal{C}^s$ and $+\infty$ elsewhere).

[3] The algorithm presented here can be easily extended to the sampling of multiple scenarios per iteration.

Algorithm 2 Randomized Progressive Hedging

Initialize: $z^0 \in \mathbb{R}^{S\times n}, \mu > 0$

For $k = 0, 1, ..$ **do:**

Draw a scenario $s^k \in \{1, .., S\}$ with probability $\mathbb{P}[s^k = s] = q_s$

$x_t^{k+1,s^k} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma z_t^{k,\sigma}$ for all $t = 1, .., T$ — projection only on constraints involving s^k

$y^{k+1,s^k} = \text{argmin}_{u\in\mathbb{R}^n} \left\{ f^{s^k}(u) + \frac{1}{2\mu} \left\| u - 2x^{k+1,s^k} + z^{k,s^k} \right\|^2 \right\}$ — optimization sub-problem only concerning scenario s^k

$z^{k+1,s^k} = z^{k,s^k} + y^{k+1,s^k} - x^{k+1,s^k}$

$z^{k+1,s} = z^{k,s}$ for all $s \neq s^k$

Specify output ?

ii) a projection over the non-anticipativity constraints $\mathcal{W}$ *associated with* s^k. As before, this projection is relatively cheap to compute. Notice that, due to the application of coordinate descent on the underlying operators (see Appendix B), this projection step is performed before the computations at each iteration[4].

iii) a proximal operation over the selected scenario s^k. As this step has to be performed after the previous ones, this algorithm is naturally adapted to single-thread implementations and its incremental nature makes it computationally more efficient than the vanilla Progressive Hedging, to almost no additional implementation complicatedness.

why so?

iv) a simple variable update.

3.3. **Asynchronous Randomized version.** One can furthermore generate asynchronous iterations using slightly different coordinate descent methods *à la* ARock Peng et al. (2016). In that setting, each of the computing agents asynchronously reads[5] the master variable (this *read version* and the variables computed from it are denoted by a hat), computes an update associated with *one randomly drawn scenario*, then incorporates it to the master variable.

As there may be several agents[6], multiple updates may occurred between the time of reading and updating, i.e. $\hat{z}^k = z^{k-d^k}$ where d^k is called the *delay* suffered by the agent updating at time k. In the following, we will assume that this delay is uniformly bounded: $d^k \leq \tau < \infty$ for all k.

With the delay bound τ and a stepsize $\eta^k \in \left[\eta_{\min}, \frac{cSq_{\min}}{2\tau\sqrt{q_{\min}}+1}\right]$ for $q_{\min} = \min_s q_s$ and any $\eta_{\min} > 0$ and $0 < c < 1$, the asynchronous algorithm obtained by transposing the ARock scheme to the operators at hand is displayed as Algorithm 3. The derivations details and proofs are reported in Appendix C.

Algorithm 3 Asynchronous Randomized Progressive Hedging

Initialize: $z^0 \in \mathbb{R}^{S\times n}, \mu > 0$

Every agent asynchronously do:

Draw a scenario $s^k \in \{1, .., S\}$ with probability $\mathbb{P}[s^k = s] = q_s$ — the computation is done with potentially outdated variable $\hat{z}^k$

$\hat{x}_t^{k,s^k} = \frac{1}{\sum_{\sigma\in\mathcal{B}^{s^k}} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma \hat{z}_t^{k,\sigma}$ for all $t = 1, .., T$ — projection only on constraints involving s^k

$\hat{y}^{k+1,s^k} = \text{argmin}_{u\in\mathbb{R}^n} \left\{ f^{s^k}(u) + \frac{1}{2\mu} \left\| u - 2\hat{x}^{k,s^k} + \hat{z}^{k,s^k} \right\|^2 \right\}$ — optimization sub-problem only concerning scenario s^k

$z^{k+1,s^k} = z^{k,s^k} + \frac{2\eta^k}{Sp_{s^k}} \left(\hat{y}^{k+1,s^k} - \hat{x}^{k,s^k} \right)$

$z^{k+1,s} = z^{k,s}$ for all $s \neq s^k$

$k \leftarrow k + 1$

[4] This ordering is actually paramount as the projection links several variables which means that even computing part of the projection would imply computing several scenarios updates beforehand.

[5] The algorithm presented here assumes *consistent*, i.e. complete reads of the full variable, extensions to *inconsistent* reads is discussed in (Peng et al., 2016, Sec. 1.2)

[6] In the case of one agent, it is natural for it to update from the latest available variable, i.e. $\hat{z}^k = z^k$, and the algorithm boils down to Randomized Progressive Hedging with $\eta^k = Sp_{s^k}/2$.

Theorem 3.3. *Consider a multistage stochastic problem (2.4) with positive probabilities ($p_s > 0$ for all s), convex objectives f^s, and constraints $\mathcal{W}$ as defined in (2.3) such that* $\operatorname{ri}\operatorname{dom} f \cap \mathcal{W} \neq \emptyset$ *holds. Then, the sequence (x^k) generated by the Asynchronous Randomized Progressive Hedging algorithm is feasible ($\mathrm{x}^k \in \mathcal{W}$ a.s. for all k) and converges almost surely to a random variable supported by the solution set of Problem (2.4).* Different formulation => different optimal so

Implementation. The Asynchronous Randomized Progressive Hedging (Alg. 3) consists of the same steps per iteration as the Randomized Progressive Hedging (Alg. 2) but these steps are performed *asynchronously by several agents in parallel.* This makes this algorithm amenable to an asynchronous parallel master-slave implementation. For convenience, we reformulate Alg. 3 to make clear how it can be implemented in a master-slave setting in Alg. 4.

Algorithm 4 Master-Slave implementation of the Asynchronous Randomized Progressive Hedging

Master

Initialize: $\mathrm{z}^0 \in \mathbb{R}^{S\times n}, \mu > 0, k = 0, \mathrm{x}[j], \mathrm{v}[j] \in \mathbb{R}^n$ and $s[j] = j$ for every agent j

Send the scenario/point pair $(s[j], \mathrm{v}[j])$ to every agent j

maintain hats ?

As soon as an agent finishes its computation:

Receive $\mathrm{y}^{s[i]}$ from an agent, say i

$\mathrm{z}^{k+1,s[i]} = \mathrm{z}^{k,s[i]} + \frac{2\eta^k}{S p_{s[i]}} \left(\mathrm{y}^{s[i]} - \mathrm{x}[i]\right)$

$\mathrm{z}^{k+1,s} = \mathrm{z}^{k,s}$ for all $s \neq s[i]$

Draw a new scenario for i : $s[i] \in \{1, .., S\}$ with probability $\mathbb{P}[s[i] = s] = q_s$ clarity ?

$\mathrm{x}_t[i] = \frac{1}{\sum_{\sigma\in\mathcal{B}^{s[i]}} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^{s[i]}} p_\sigma \, \mathrm{z}_t^{k+1,\sigma}$ for all $t = 1, .., T$

$\mathrm{v}[i] = 2\,\mathrm{x}[i] - \mathrm{z}^{k+1,s[i]}$

Send the scenario/point pair $(s[i], \hat{\mathrm{v}}[i])$ to agent i

$k \leftarrow k + 1$

Agent i

Initialize: $\mu > 0$

As soon as a scenario/point pair is received:

Receive scenario/point pair $(s[i], \mathrm{v}[i])$

$\mathrm{y}^{s[i]} = \operatorname{argmin}_{u\in\mathbb{R}^n} \left\{ f^{s[i]}(u) + \frac{1}{2\mu} \|u - \mathrm{v}[i]\|^2 \right\}$

Send $\mathrm{y}^{s[i]}$ to the Master

4. Numerical Experiments

References

Eckstein, Jonathan, Dimitri P Bertsekas. 1992. On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming* **55**(1-3) 293–318.

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

Lions, Pierre-Louis, Bertrand Mercier. 1979. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis* **16**(6) 964–979.

Peng, Zhimin, Yangyang Xu, Ming Yan, Wotao Yin. 2016. Arock: an algorithmic framework for asynchronous parallel coordinate updates. *SIAM Journal on Scientific Computing* **38**(5) A2851–A2879.

Ruszczyński, Andrzej, Alexander Shapiro. 2003. Stochastic programming models. *Handbooks in operations research and management science* **10** 1–64.

Ruszczyński, Andrzej. 2003. Decomposition methods. *Stochastic Programming*, *Handbooks in Operations Research and Management Science*, vol. 10. Elsevier, 141 – 211.

Shapiro, Alexander, Darinka Dentcheva, Andrzej Ruszczyński. 2009. *Lectures on stochastic programming: modeling and theory*. SIAM.

Appendix A. Convergence of the Progressive Hedging and Relations with the Douglas-Rachford method

First of all, noting $f(x) := \sum_{s=1}^{S} p_s f^s(x)$ and $\iota_{\mathcal{W}}(x) = 0$ if $x \in \mathcal{W}$ and $+\infty$ otherwise, solving (2.4) amounts to finding $x^\star$ such that

$$0 \in \partial f(x^\star) + \partial \iota_{\mathcal{W}}(x^\star) \tag{A.1}$$

provided that the constraint qualification condition ri $\mathrm{dom} f \cap \mathcal{W} \neq \emptyset$ holds.

In order to emphasize that i) the two parts of this equation have to be treated independently; and ii) the scenarios have different probabilities, let us define two $\mathbb{R}^{S\times n} \to \mathbb{R}^{S\times n}$ operators[7]:

$$A(x) := P^{-1}\partial f(x) \;\text{ and }\; B(x) := P^{-1}\partial \iota_{\mathcal{W}}(x) \tag{A.2}$$

where $P = \mathrm{diag}(p_1, .., p_S)$.

Then, under the same constraint qualification conditions, solving (2.4) amounts to finding a zero of $A + B$ i.e. $x^\star$ such that $0 \in A(x^\star) + B(x^\star)$. The useful properties of A and B are recalled in the following lemmas.

Lemma A.1. *Consider a multistage stochastic problem (2.4) with positive probabilities ($p_s > 0$ for all s), convex objectives f^s, and constraints $\mathcal{W}$ as defined in (2.3). Then, the operators A and B defined in (A.2) are maximal monotone.*

Lemma A.2. *Let M be maximal monotone. Then, for any $\mu > 0$:*

i) $J_{\mu M} := (I + \mu M)^{-1}$ *is well-defined and firmly non-expansive;*
ii) $O_{\mu M} = 2J_{\mu M} - I$ *is non-expansive;*
iii) Any $z \in \mathbb{R}^{S\times n}$ can be uniquely represented as $z = x + \mu u$ with $u \in M(x)$, thus $J_{\mu M}(z) = x$ and $O_{\mu M}(z) = O_{\mu M}(x + \mu u) = x - \mu u$.

Let us take a careful look at $O_{\mu A}$ and $O_{\mu A}$.

Lemma A.3. *Let $z \in \mathbb{R}^{S\times n}$,*

i) $O_{\mu A}(z) = x - \mu u$ *with*

$$x^s = \operatorname*{argmin}_{u\in\mathbb{R}^n} \left\{ f^s(u) + \frac{1}{2\mu}\|u - z^s\|^2 \right\} \;\text{ for all } s = 1, .., S \tag{A.3}$$

and $u = (z - x)/\mu$;
ii) $O_{\mu B}(z) = x - \mu u$ *with*

$$x^s_t = \frac{1}{\sum_{\sigma\in\mathcal{B}^s_t} p_\sigma} \sum_{\sigma\in\mathcal{B}^s_t} p_\sigma\, z^\sigma_t \;\text{ for all } s = 1, .., S \text{ and } t = 1, .., T \tag{A.4}$$

and $u = (z - x)/\mu$;
iii) The point x obtained by (A.4) is the orthogonal projection *of z to $\mathcal{W}$ (in the space endowed by the norm weighted by the probabilities (p_s)). Thus, z writes uniquely as $z = x + \mu u$ with $x \in \mathcal{W}$ obtained by (A.4) and $u \in \mathcal{W}^\perp$.*

The above lemma shows that these operator are very different in nature:

- $O_{\mu A}$ *is separable by scenario* but involves *solving a subproblem*;
- $O_{\mu B}$ *links the scenarios* but only amounts to *computing a weighted average.*

The Douglas-Rachford method finds a zero of $A + B$ by iterating

$$z^{k+1} = \frac{1}{2} O_{\mu A}(O_{\mu B}(z^k)) + \frac{1}{2} z^k \tag{A.5}$$

and converge as soon as $\mu > 0$ and A, B are maximal monotone as stated in the following theorem.

Theorem A.4. *Let $\mu > 0$ and A, B be maximal monotone operators, then the sequence (z^k) generated by (A.5) converges to a point $z^\star$ such that $x^\star := (O_{\mu B}(z^\star) + z^\star)/2$ is a zero of $A + B$, i.e. $0 \in A(x^\star) + B(x^\star)$.*

= \mu B(z) ?

[7] Note that in the whole paper the ambient space is the space of $S \times n$ real matrices with the Frobenius norm and scalar product.

Laying down the iterations obtained by Douglas-Rachford method (A.5) with the expressions obtained in Lemma A.3, we get

$$\left\{\begin{array}{ll} \mathrm{x}_t^{k,s} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \mathrm{z}_t^{k,\sigma} \ \text{ for all } s=1,..,S \text{ and } t=1,..,T & \mathrm{x}^k\in\mathcal{W} \\ \mathrm{w}^k = \mathrm{O}_{\mu\mathrm{B}}(\mathrm{z}^k) = 2\,\mathrm{x}^k - \mathrm{z}^k = \mathrm{x}^k - \mu\,\mathrm{u}^k & \text{with } \mathrm{u}^k=(\mathrm{z}^k-\mathrm{x}^k)/\mu\in\mathcal{W}^\perp \\ & \text{thus } \mathrm{u}^k=\mathrm{u}^{k-1}+\frac{1}{\mu}(\mathrm{y}^k-\mathrm{x}^k) \\ \mathrm{y}^{k+1,s} = \mathrm{argmin}_{u\in\mathbb{R}^n}\left\{f^s(u)+\frac{1}{2\mu}\left\|u-\mathrm{w}^{k,s}\right\|^2\right\} \ \text{ for all } s=1,..,S & \\ \mathrm{z}^{k+1} = \frac{1}{2}(2\,\mathrm{y}^{k+1}-\mathrm{w}^k)+\frac{1}{2}\,\mathrm{z}^k = \mathrm{z}^k+\mathrm{y}^{k+1}-\mathrm{x}^{k+1} = \mathrm{y}^{k+1}+\mu\,\mathrm{u}^k & \end{array}\right. \tag{A.6}$$

Reorganizing the equations and eliminating intermediate variables (notably using the fact that, provided that the algorithm is initialized with $\mathrm{x}^0\in\mathcal{W}$ and $\mathrm{u}\in\mathcal{W}^\perp$, (x^k) and (u^k) are in $\mathcal{W}$ and $\mathcal{W}^\perp$ respectively), we get the *Progressive Hedging algorithm* as per (Ruszczyński, 2003, Fig. 10):

$$\left\{\begin{array}{ll} \mathrm{y}^{k+1,s} = \mathrm{argmin}_{u\in\mathbb{R}^n}\left\{f^s(u)+\frac{1}{2\mu}\left\|u-\mathrm{x}^{k,s}+\mu\,\mathrm{u}^{k,s}\right\|^2\right\} \ \text{ for all } s=1,..,S & \\ \mathrm{x}_t^{k+1,s} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \mathrm{y}_t^{k+1,\sigma} \ \text{ for all } s=1,..,S \text{ and } t=1,..,T & \mathrm{x}^k\in\mathcal{W} \text{ converges to a solution of (2.4)} \\ \mathrm{u}^{k+1} = \mathrm{u}^k+\frac{1}{\mu}(\mathrm{y}^{k+1}-\mathrm{x}^{k+1}) & \end{array}\right. \tag{A.7}$$

Appendix B. Derivation and Proof of Randomized Progressive Hedging

Getting back to the Douglas-Rachford method,

$$\mathrm{z}^{k+1} = \frac{1}{2}\mathrm{O}_{\mu\mathrm{A}}(\mathrm{O}_{\mu\mathrm{B}}(\mathrm{z}^k)) + \frac{1}{2}\,\mathrm{z}^k,$$

an interesting randomized counterpart consists in updating the part corresponding with scenario s with probability q_s, the other staying unchanged. Mathematically, this writes

$$\text{Draw a scenario } s^k\in\{1,..,S\} \text{ with probability } \mathbb{P}[s^k=s]=q_s \tag{B.1}$$

$$\left|\begin{array}{l} \mathrm{z}^{k+1,s^k} = \frac{1}{2}\left[\mathrm{O}_{\mu\mathrm{A}}(\mathrm{O}_{\mu\mathrm{B}}(\mathrm{z}^k))\right]^{s^k} + \frac{1}{2}\,\mathrm{z}^{k,s^k} \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \ \text{ for all } s\neq s^k \end{array}\right. \tag{B.2}$$

Theorem B.1. *Let $\mu>0$, A,B be maximal monotone operators, and $q_s>0$ for all $s=1,..,S$. Then, the sequence (z^k) generated by (B.1) converges almost surely to a point $\mathrm{z}^\star$ such that $x^\star := (\mathrm{O}_{\mu\mathrm{B}}(\mathrm{z}^\star)+\mathrm{z}^\star)/2$ is a zero of $\mathrm{A}+\mathrm{B}$, i.e. $0\in\mathrm{A}(\mathrm{x}^\star)+\mathrm{B}(\mathrm{x}^\star)$.*

needs proof ?

Laying down the iterations obtained by that randomized Douglas-Rachford method (B.1) with the expressions obtained in Lemma A.3, we get

$$\left\{\begin{array}{ll} \text{Draw a scenario } s^k\in\{1,..,S\} \text{ with probability } \mathbb{P}[s^k=s]=q_s & \\ \mathrm{x}_t^{k,s} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \mathrm{z}_t^{k,\sigma} \ \text{ for all } s=1,..,S \text{ and } t=1,..,T & \mathrm{x}^k\in\mathcal{W} \\ \mathrm{w}^k = \mathrm{O}_{\mu\mathrm{B}}(\mathrm{z}^k) = 2\,\mathrm{x}^k - \mathrm{z}^k = \mathrm{x}^k - \mu\,\mathrm{u}^k & \text{with } \mathrm{u}^k=(\mathrm{z}^k-\mathrm{x}^k)/\mu\in\mathcal{W}^\perp \\ \mathrm{y}^{k+1,s} = \mathrm{argmin}_{u\in\mathbb{R}^n}\left\{f^s(u)+\frac{1}{2\mu}\left\|u-\mathrm{w}^{k,s}\right\|^2\right\} \ \text{ for all } s=1,..,S & \\ \left|\begin{array}{l} \mathrm{z}^{k+1,s^k} = \frac{1}{2}(2\,\mathrm{y}^{k+1,s^k}-\mathrm{w}^{k,s^k})+\frac{1}{2}\,\mathrm{z}^{k,s^k} = \mathrm{z}^{k,s^k}+\mathrm{y}^{k+1,s^k}-\mathrm{x}^{k+1,s^k} = \mathrm{y}^{k+1,s^k}+\mu\,\mathrm{u}^{k,s^k} \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \ \text{ for all } s\neq s^k \end{array}\right. & \end{array}\right. \tag{B.3}$$

The next step is to carefully prune unneccessary computations. First, only y^{k+1,s^k} needs to be computed, so the other $\mathrm{y}^{k+1,s}$ ($s\neq s^k$) can be safely dropped. The same holds for x^{k,s^k}, w^{k,s^k}, and u^{k,s^k}. However, even though only x^{k,s^k} need to be computed, it depends on all the other scenarios through the projection operator $(2\mathrm{O}_{\mu\mathrm{B}}-I)$, so the iterates *have to be computed sucessively* without asynchronous parallelization, and with only a partial update of u^k, it does not belong to $\mathcal{W}$ anymore and thus cannot be dropped out of the projection. It is then simpler to keep directly

the global variable z^k updated.

$$\left\{\begin{array}{ll}
\text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s & \\
\mathrm{x}_t^{k,s^k} = \frac{1}{\sum_{\sigma \in \mathcal{B}_t^{s^k}} p_\sigma} \sum_{\sigma \in \mathcal{B}_t^{s^k}} p_\sigma \, \mathrm{z}_t^{k,\sigma} \text{ for all } t = 1,..,T & \text{projection only on constraints involving } s^k \\
\mathrm{w}^{k,s^k} = 2\,\mathrm{x}^{k,s^k} - \mathrm{z}^{k,s^k} & \\
\mathrm{y}^{k+1,s^k} = \operatorname{argmin}_{u \in \mathbb{R}^n} \left\{ f^{s^k}(u) + \frac{1}{2\mu} \left\| u - \mathrm{w}^{k,s^k} \right\|^2 \right\} & \text{optimization sub-problem only concerning scenario } s^k \\
\left| \begin{array}{l} \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} + \mathrm{y}^{k+1,s^k} - \mathrm{x}^{k+1,s^k} \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \text{ for all } s \neq s^k \end{array} \right. &
\end{array}\right. \tag{B.4}$$

Reorganizing the equations and eliminating intermediate variables (note also that the order of the equations cannot be changed contrary to the vanilla progressive hedging), we get the *Randomized Progressive Hedging algorithm*:

$$\left\{\begin{array}{ll}
\text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s & \\
\mathrm{x}_t^{k+1,s^k} = \frac{1}{\sum_{\sigma \in \mathcal{B}_t^{s^k}} p_\sigma} \sum_{\sigma \in \mathcal{B}_t^{s^k}} p_\sigma \, \mathrm{z}_t^{k,\sigma} \text{ for all } t = 1,..,T & \text{projection only on constraints involving } s^k \\
\mathrm{y}^{k+1,s^k} = \operatorname{argmin}_{u \in \mathbb{R}^n} \left\{ f^{s^k}(u) + \frac{1}{2\mu} \left\| u - 2\,\mathrm{x}^{k+1,s^k} + \mathrm{z}^{k,s^k} \right\|^2 \right\} & \text{optimization sub-problem only concerning scenario } s^k \\
\left| \begin{array}{l} \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} + \mathrm{y}^{k+1,s^k} - \mathrm{x}^{k+1,s^k} \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \text{ for all } s \neq s^k \end{array} \right. &
\end{array}\right. \tag{B.5}$$

Appendix C. Derivation and Proof of Asynchronous Randomized Progressive Hedging

As we are willing to find fixed points of the non-expansive operator $\mathsf{T} = \mathrm{O}_{\mu\mathsf{A}} \circ \mathrm{O}_{\mu\mathsf{B}}$, or equivalently zeros of $\mathsf{S} = I - \mathrm{O}_{\mu\mathsf{A}} \circ \mathrm{O}_{\mu\mathsf{B}}$, the ARock asynchronous parallel methods boils down to

Every agent asynchronously do (C.1)

$$\left\{\begin{array}{l}
\text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s \\
\left| \begin{array}{l} \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} - \frac{\eta^k}{S p_{s^k}} \left(\hat{\mathrm{z}}^{k,s^k} - \left[\mathrm{O}_{\mu\mathsf{A}}(\mathrm{O}_{\mu\mathsf{B}}(\hat{\mathrm{z}}^k)) \right]^{s^k} \right) \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \text{ for all } s \neq s^k \end{array} \right. \\
\text{where } \hat{\mathrm{z}}^k \text{ is the value of } \mathrm{z}^k \text{ used for its computation by the updating agent at time } k
\end{array}\right. \tag{C.2}$$

One can notice that the main difference between this version with the randomized one is the introduction of $\hat{\mathrm{z}}^k$:

- if there is only one agent, it just computes its new point with the latest value so $\hat{\mathrm{z}}^k = \mathrm{z}^k$ and one can notice that taking $\eta^k = S p_{s^k}/2$, we recover exactly the randomized Douglas-Rachford method (B.1);
- where there are more agents, $\hat{\mathrm{z}}^k$ is usually an older version of the main variable (as other workers may have updated the main variable during the computation of the updating worker) so that $\hat{\mathrm{z}}^k = \mathrm{z}^{k-d^k}$ where d^k is the *delay* suffered by the updating agent at time k.

Theorem C.1. *Let $\mu > 0$, A, B be maximal monotone operators, and $q_s > 0$ for all $s = 1,..,S$. Consider the sequence (z^k) generated by (C.1) with*

i) the maximal delay (i.e. the maximal number of updates between two updates of one agent) bounded by τ;
ii) $\eta^k \in \left[\eta_{\min}, \frac{cSq_{\min}}{2\tau\sqrt{q_{\min}}+1} \right]$ for $q_{\min} = \min_s q_s$ and any $\eta_{\min} > 0$ and $0 < c < 1$.

Then, (z^k) converges almost surely to a random variable $\mathrm{z}^\star$ such that $\mathrm{x}^\star := (\mathrm{O}_{\mu\mathsf{B}}(\mathrm{z}^\star) + \mathrm{z}^\star)/2$ is a zero of $\mathsf{A} + \mathsf{B}$, i.e. $0 \in \mathsf{A}(\mathrm{x}^\star) + \mathsf{B}(\mathrm{x}^\star)$ almost surely.

Proof. See (Peng et al., 2016, Th. 3.7). ∎

Using again Lemma A.3, we get

Every agent asynchronously do (C.3)

$$\left\{\begin{array}{l} \text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s \\ \hat{\mathrm{x}}_t^{k,s} = \frac{1}{\sum_{\sigma\in\mathcal{B}_t^s} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^s} p_\sigma\, \hat{\mathrm{z}}_t^{k,\sigma} \;\text{ for all } s = 1,..,S \text{ and } t = 1,..,T \\ \hat{\mathrm{w}}^k = \mathrm{O}_{\mu\mathrm{B}}(\hat{\mathrm{z}}^k) = 2\,\hat{\mathrm{x}}^k - \hat{\mathrm{z}}^k \\ \hat{\mathrm{y}}^{k+1,s} = \mathrm{argmin}_{u\in\mathbb{R}^n} \left\{ f^s(u) + \frac{1}{2\mu}\left\|u - \hat{\mathrm{w}}^{k,s}\right\|^2 \right\} \;\text{ for all } s = 1,..,S \\ \left[\mathrm{O}_{\mu\mathrm{A}}(\mathrm{O}_{\mu\mathrm{B}}(\hat{\mathrm{z}}^k))\right]^{s^k} = 2\,\hat{\mathrm{y}}^{k+1,s^k} - \hat{\mathrm{w}}^{k,s^k} \\ \left|\begin{array}{l} \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} - \frac{\eta^k}{S p_{s^k}} \left(\hat{\mathrm{z}}^{k,s^k} - \left[\mathrm{O}_{\mu\mathrm{A}}(\mathrm{O}_{\mu\mathrm{B}}(\hat{\mathrm{z}}^k))\right]^{s^k} \right) \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \;\text{ for all } s \neq s^k \end{array}\right. \end{array}\right. \tag{C.4}$$

The next step is again to carefully prune unnecessary computations.

Every agent asynchronously do (C.5)

$$\left\{\begin{array}{ll} \text{Draw a scenario } s^k \in \{1,..,S\} \text{ with probability } \mathbb{P}[s^k = s] = q_s & \\ \hat{\mathrm{x}}_t^{k,s^k} = \frac{1}{\sum_{\sigma\in\mathcal{B}^{s^k}} p_\sigma} \sum_{\sigma\in\mathcal{B}_t^{s^k}} p_\sigma\, \hat{\mathrm{z}}_t^{k,\sigma} \;\text{ for all } t = 1,..,T & \text{projection only on constraints involving } s^k \\ \hat{\mathrm{y}}^{k+1,s^k} = \mathrm{argmin}_{u\in\mathbb{R}^n} \left\{ f^{s^k}(u) + \frac{1}{2\mu}\left\|u - 2\,\hat{\mathrm{x}}^{k,s^k} + \hat{\mathrm{z}}^{k,s^k}\right\|^2 \right\} & \text{optimization sub-problem only concerning scenario } s^k \\ \left|\begin{array}{l} \mathrm{z}^{k+1,s^k} = \mathrm{z}^{k,s^k} + \frac{2\eta^k}{S p_{s^k}} \left(\hat{\mathrm{y}}^{k+1,s^k} - \hat{\mathrm{x}}^{k,s^k} \right) \\ \mathrm{z}^{k+1,s} = \mathrm{z}^{k,s} \;\text{ for all } s \neq s^k \end{array}\right. & \end{array}\right. \tag{C.6}$$