Asynchronous Projective Hedging for Stochastic Programming

David L. Woodruff
Graduate School of Management
UC Davis
Davis CA USA 95616
DLWoodruff@UCDavis.edu

Jonathan Eckstein
Jean-Paul Watson
Introduction

- Background
- Some notation
- Some computational issues
- A few results
Art

Impressionism and Deconstruction
Art

- Character versus Self
- Character versus Nature
- etc.
- ADMM “versus” operator splitting ??
More Art

Algorithms: [Rockafellar and Wets, 1991] [Combettes and Eckstein, 2018] [Eckstein, 2017]
Instances:
  ▶ ALP1P: [Infanger, 1992, Bailey et al., 1999]
  ▶ saphir: [Koberstein et al., 2011, Zverovich et al., 2012]
  ▶ ssn: [Sen et al., 1994]


References II


Pyomo and PySP

- Pyomo.org
- Pyomo provides algebraic modelling and model transformations in Python
- PySP is a stochastic programming extension
- This work is intended for inclusion in PySP
Notation and Problem Setup

- Finite scenario tree $\mathcal{T}$; stages $t \in 1..T$, leaf nodes by $i \in 1..n$.
- Let $\mathcal{T}_t$ denote the set of tree nodes at each stage $t \in 1..T$.
- For each $N \in \mathcal{T}_t$, we let $S(N) \subseteq 1..n$ denote the set of leaf nodes that have $N$ as an ancestor.
- The sets $\{S(N) \mid N \in \mathcal{T}_t\}$ comprise a partition of $1..n$ for each $t \in 1..T$.
- Probability of leaf node (scenario) $i$ is $\pi_i \in (0,1]$. For each node $N$ in the tree, $\pi(N) = \sum_{i \in S(n)} \pi_i$ denotes its probability.
More notation

- Number of decision variables $m_t \geq 1$; $m = \sum_{t=1}^{T} m_t$
- For each $i \in 1..n$, let $\mathcal{X}_i$ denote the space of vectors of possible decisions for scenario $i$, from all stages.
- Element of $\mathcal{X}_i$ is $x_i$, and its subvector corresponding to each stage $t \in 1..T$ is $x_{it} \in \mathbb{R}^{m_t}$.

We give $\mathcal{X}_i$ the inner product

$$\langle x_i, y_i \rangle_{\mathcal{X}_i} = \pi_i (x_i)^\top (y_i) = \pi_i \sum_{t=1}^{T} (x_{it})^\top (y_{it}), \quad (1)$$
Let $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, under the inner product induced by (1),

$$\langle x, y \rangle_{\mathcal{X}} = \sum_{i=1}^{n} \langle x_i, y_i \rangle_{\mathcal{X}} = \sum_{i=1}^{n} \pi_i (x_i)^\top (y_i) = \sum_{i=1}^{n} \pi_i \sum_{t=1}^{T} (x_{it})^\top (y_{it}).$$

(2)

One may envision the elements of $\mathcal{X}$ as expanding decision vectors defined on the scenario tree into a “grid” by making $|S(N)|$ replicas of the decision variables at each node $N$ in the tree. Thus, vectors in $\mathcal{X}$ have $n$ sets of decision variables for each stage $t$, regardless of the number of tree nodes $|T_t|$ at stage $t$. 
\( m' = m - m_T = \sum_{t=1}^{T-1} m_t. \)

For each \( i \in 1..n \), let \( \mathcal{Z}_i \) denote the space of all decision variables for scenario \( i \), except those for stage \( T \).

\[
\langle z_i, u_i \rangle_{\mathcal{Z}_i} = \pi_i(z_i)^\top(u_i) = \pi_i \sum_{t=1}^{T-1} (z_{it})^\top(u_{it}).
\]  

(3)

Let \( \mathcal{Z} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_n \) with the inner product induced by (3),

\[
\langle z, u \rangle_{\mathcal{Z}} = \sum_{i=1}^{n} \langle z_i, u_i \rangle_{\mathcal{Z}_i} = \sum_{i=1}^{n} \pi_i(z_i)^\top(u_i) = \sum_{i=1}^{n} \pi_i \sum_{t=1}^{T-1} (z_{it})^\top(u_{it}).
\]

(4)

Vectors in \( \mathcal{Z} \) have the same “grid” interpretation as vectors in \( \mathcal{X} \), except that the are missing the variables for the final stage.
M maps \( \mathcal{X} \) to \( \mathcal{Z} \)

For each \( i \in 1..n \), let \( M_i \) denote the linear map \( \mathcal{X}_i \to \mathcal{Z}_i \) that simply drops that last-stage variables, \( M_i(x_{i1}, \ldots, x_{iT}) = (x_{i1}, \ldots, x_{i, T-1}) \). We then let \( M : \mathcal{X} \to \mathcal{Z} \) be the linear map given by

\[
M : ((x_{11}, \ldots, x_{1T}), \ldots, (x_{n1}, \ldots, x_{nT})) \mapsto (M_1x_1, \ldots, M_nx_n),
\]

which simply drops all the last-stage variables from its argument.
Nonanticipativity and $\mathcal{N}$

As in [Rockafellar and Wets, 1991], we define a linear subspace $\mathcal{N}$ of $\mathcal{Z}$ by

$$\mathcal{N} = \{ z \in \mathcal{Z} \mid (\forall t \in 1..T) (\forall N \in \mathcal{T}_t) (\forall i, j \in S(N)) : z_{it} = z_{jt} \}.$$  

(6)

Vectors in $z \in \mathcal{N}$ are nonanticipative in the sense that $z_{it} = z_{jt}$ whenever scenarios $i$ and $j$ are indistinguishable at stage $t$. 


The Problem

For each \( i \in 1..n \), let \( h_i : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\} \) be a closed proper convex function and consider the optimization problem

\[
\min_{x \in \mathcal{X}} \sum_{i=1}^{n} \pi_i h_i(x_i) \\
\text{ST} \quad Mx \in \mathcal{N}.
\]  

(7)

This optimization model can subsume any convex stochastic programming problem defined on the scenario tree \( \mathcal{T} \).
One can now derive the progressive hedging algorithm by defining convex functions $F : \mathcal{X} \rightarrow \mathbb{R} \cup \{+\infty\}$ and $G : \mathcal{Z} \rightarrow \mathbb{R} \cup \{+\infty\}$ as follows:

$$F(x) = \sum_{i=1}^{n} \pi_i h_i(x_i) \quad \quad \quad G(z) = \begin{cases} 0, & z \in \mathcal{N} \\ +\infty, & z \notin \mathcal{N} \end{cases} \quad \quad \quad (8)$$

Problem (7) may then be expressed as simply

$$\min_{x \in \mathcal{X}} \{ F(x) + G(Mx) \} \quad \quad \quad (9)$$

which is a standard form for applying the ADMM. Doing so while keeping in mind the inner products defined in (2) and (4), one may derive the progressive hedging method.
\[ \min_{z \in \mathcal{H}_0} \left\{ \sum_{i=1}^{n} f_i(L_i z) \right\}, \]  

(10)

where \( \mathcal{H}_0, \ldots, \mathcal{H}_n \) are real Hilbert spaces with respective inner products \( \langle \cdot, \cdot \rangle_{\mathcal{H}_0}, \ldots, \langle \cdot, \cdot \rangle_{\mathcal{H}_n} \), and, for all \( i \in 1..n \), \( f_i : \mathcal{H}_i \to \mathbb{R} \cup \{ \infty \} \) is a closed proper convex function and \( L_i : \mathcal{H}_0 \to \mathcal{H}_i \) is a bounded linear map.
Elements

We assign the elements of this model as follows:

S1. $H_0 = \mathcal{N}$, the nonanticipativity subspace, under the inner product $\langle \cdot, \cdot \rangle_Z$ from (4).

S2. For all $i \in 1..n$, we let $H_i = \mathcal{Z}_i$, thus using the inner product (3).

S3. For all $i \in 1..n$, we define $L_i$ by $L_i : (z_1, \ldots, z_n) \mapsto z_i$, that is, $L_i$ selects only the portion of its argument corresponding to scenario $i$.

S4. For all $i \in 1..n$, we define $f_i$ by

$$f_i(z_i) = \pi_i \min \left\{ h_i((z_i, x_{iT})) \mid x_{iT} \in \mathbb{R}^{m_T} \right\},$$

that is, given the values of all decision variables $z_i$ for the scenario $i$ except those for the last stage, $f_i$ optimizes over the last-stage variables and returns the resulting objective value scaled by $\pi_i$.

Implicitly minimizing over the last-stage variables within the functions $f_i$ is necessary to avoid generating undesired proximal terms associated with the last-stage variables in the algorithm.
Without completing the setup

1. Start with any $z^0 \in \mathbb{N}$, $w^0 \in \mathbb{N}^\perp$ and $(x_i^{-1}, y_i^{-1} \in \mathbb{R}^{m'})_{i \in 1..n}$
2. for $k = 0, 1, \ldots$ do
   3. Select some $I_k \subseteq 1..n$
   4. for $i \in I_k$ do
      5. Select some integer $d(i, k) \in 0..k$
      6. Select some scalar $\rho_{ik} > 0$
      7. Find $(x^k_i, x^k_{iT}) \in \operatorname{Arg min}_{x_i \in \mathbb{R}^{m'}, x_{iT} \in \mathbb{R}^{mT}} \left\{ h_i(x_i, x_{iT}) + (w^d(i,k)_{iT})^\top (x_i) + \frac{\rho_{ik}}{2} \|x_i - z^d(i,k)_i\|_2^2 \right\}$
      8. $y^k_i = w^d(i,k)_{iT} + \rho_{ik}(x^k_i - z^d(i,k)_i)$
   9. for $i \not\in I_k$ do $(x^k_i, y^k_i) = (x^{k-1}_i, y^{k-1}_i)$
   10. $u^k = \text{proj}_{\mathbb{N}^\perp}(x^k_1, \ldots, x^k_n)$
   11. $v^k = \text{proj}_{\mathbb{N}}(y^k_1, \ldots, y^k_n)$
   12. $\tau_k = \sum_{i=1}^n \pi_i \left( \| u^k_i \|_2^2 + \| v^k_i \|_2^2 \right)$
   13. if $\tau_k > 0$ then
      14. Choose some $\nu_k : 0 < \nu_k < 2$
      15. $\theta_k = \frac{\nu_k}{\tau_k} \max \left\{ 0, \sum_{i=1}^n \pi_i (L_i z^k_i - x^k_i)^\top (w^k_i - y^k_i) \right\}$
   16. else $\theta_k = 0$
   17. $z^{k+1} = z^k + \theta_k v^k$
   18. $w^{k+1} = w^k + \theta_k u^k$
   19. end
20. end

Algorithm 2: Asynchronous projective hedging — Algorithm 1 specialized to the setup S1-S4.
Some Computational Parameters
(with ssn no-bundle values)

- Number of workers (50)
- Threads per worker (2)
- Number of CPUs (96-)
- $\rho$ (1)
- $\gamma$ (1)
- async buffer len ($n/5$)
- termination criteria (iterations)
Some Issues

- Termination criteria
- $x$-hat
- Buffer Length
Conclusions

- A version of Asynchronous Projective Hedging has been implemented in PySP
- It outperforms PH on many instance, but more importantly, it is Asynchronous
- Computational experiments are under way to determine the effects of parameter settings
- Scenario bundling seems to be the most important extension.
N=500, Seed=11134
Buffer=100

(u,v) norm
APL1P