Extending SDDP-style Algorithms for Multistage Stochastic Programming

Dave Morton
Industrial Engineering & Management Sciences
Northwestern University

Joint work with:
Oscar Dowson, Daniel Duque, and Bernardo Pagnoncelli
Collaborators
Hydroelectric Power

Itaipu (14 GW)
Yuba, Bear and South Feather Hydrological Basin
SDDP
Stochastic Dual Dynamic Programming
SLP-\(T\)

\[
    z^* = \min_{x_1 \geq 0} c_1 x_1 + \mathbb{E}_{\xi_2|\xi_1} V_2(x_1, \xi_2)
\]

s.t. \(A_1 x_1 = B_1 x_0 + b_1\)

where for \(t = 2, \ldots, T\),

\[
    V_t(x_{t-1}, \xi_t) = \min_{x_t \geq 0} c_t x_t + \mathbb{E}_{\xi_{t+1}|\xi_1, \ldots, \xi_t} V_{t+1}(x_t, \xi_{t+1})
\]

s.t. \(A_t x_t = B_t x_{t-1} + b_t\)

and where \(V_{T+1} \equiv 0\)

\(V_t(\cdot, \xi_t)\) is piecewise linear and convex
SLP-T Assumptions for SDDP

• Relatively complete recourse, finite optimal solution

• \( \xi_t = (A_t, B_t, b_t, c_t) \) is inter-stage independent

• Or, \( (A_t, B_t, c_t) \) is inter-stage independent and \( b_t \) satisfies, e.g.,
  \[ -b_t = \Psi(b_{t-1}) + \varepsilon_t \text{ with } \varepsilon_t \text{ inter-stage independent}; \text{ or,} \]
  \[ -b_t = \Psi(b_{t-1}) \cdot \varepsilon_t \text{ with } \varepsilon_t \text{ inter-stage independent} \]

• Sample space: \( \Omega_t = \Sigma_2 \times \Sigma_3 \times \cdots \times \Sigma_t \) with \( |\Sigma_t| \) modest

• \( T \) may be large
What Does “Solution” Mean?

A solution is a *policy*
SDDP

(a) Forward Pass

(b) Backward Pass
SDDP Master Programs

\[
\begin{align*}
\min_{x_t, \theta_t} & \quad c_t x_t + \theta_t \\
\text{s.t.} & \quad A_t x_t = B_t x_{t-1} + b_t \\
& \quad -G^k_t x_t + \theta_t \geq g^k_t, k = 1, 2, \ldots, K \\
& \quad x_t \geq 0
\end{align*}
\]
Partially Observable Multistage Stochastic Programming

Or, an alternative to DRO when you don’t really know the distribution

An apology: Not talking about Wasserstein-based DRO for SLP-$T$ via an SDDP Algorithm (with Daniel Duque)
Policy Graphs (Dowson)

A policy graph for SLP-3 with inter-stage independence:

```
1  -->  2  -->  3
```

Unfolds to a scenario tree:

```
1  -->  2H  -->  3HL
     \   /     /   \
      \ /  3HH  /  3HL
       \       /     /
        \     2L   3LH
         \   /     /   \
          \ /  3LL  /  3HH
           \       /     /
            \     1   2H
```


Policy Graphs

A Markov-switching model:

Random transitions:
Inventory Example

Demand model $A$: $\mathbb{P}(\omega = 1) = 0.2 \quad \mathbb{P}(\omega = 2) = 0.8$

Demand model $B$: $\mathbb{P}(\omega = 1) = 0.8 \quad \mathbb{P}(\omega = 2) = 0.2$

\[ D_i : D_i(x) = \min_{u,x' \geq 0} u + \mathbb{E}_\omega[H_i(x',\omega)] \]
\[ \text{s.t. } x' = x + u \]

\[ H_i : H_i(x,\omega) = \min_{u,x' \geq 0} 2u + x' + \rho D_i(x) \]
\[ \text{s.t. } x' = x + u - \omega \]
Policy Graphs

Each node $i$:

$$
\Omega_i \xrightarrow{\omega} x \xrightarrow{\pi_i(x, \omega)} u \xrightarrow{T_i(x, u, \omega)} x' \xrightarrow{C_i(x, u, \omega)}
$$

A policy graph:

- $\mathcal{G} = (R, N, \mathcal{E}, \Phi)$
- $\omega_j \in \Omega_j$: node-wise independent noise
- feasible controls: $u \in U_i(x, \omega)$
- transition function: $x' = T_i(x, u, \omega)$
- one-step cost function: $C_i(x, u, \omega)$
Policy Graphs

\[
\min_{\pi} \mathbb{E}_{i \in R^+; \omega \in \Omega_i}[V_i(x_R, \omega)]
\]

where

\[
V_i(x, \omega) = \min_{u, \bar{x}, x'} C_i(\bar{x}, u, \omega) + \mathbb{E}_{j \in i^+; \varphi \in \Omega_j}[V_j(x', \varphi)]
\]

s.t. \( \bar{x} = x \)

\[
u \in U_i(\bar{x}, \omega)
\]

\[
x' = T_i(\bar{x}, u, \omega)
\]

Goal: Find \( \pi_i(x, \omega) \) that solves (1) for each \( i \in \mathcal{N}, x, \text{and } \omega \)

(A1) \( \mathcal{N} \) is finite

(A2) \( \Omega_i \) is finite and \( \omega_i \) is node-wise independent \( \forall i \in \mathcal{N} \)

(A3) Excluding cost-to-go term, subproblem (2) is an LP

(A4) Subproblem (2) has finite optimal solution

(A5) Hit leaf node with probability 1 (or graph \( G \) is acyclic)
Policy Graphs with Partial Observability

Extend policy graph to:

\[ \mathcal{G} = (R, \mathcal{N}, \mathcal{E}, \Phi, \mathcal{A}) \]

where \( \mathcal{A} \) partitions \( \mathcal{N} \):

\[ \bigcup_{A \in \mathcal{A}} A = \mathcal{N} \quad A \cap A' = \emptyset, A \neq A' \]

We know the current ambiguity set, \( \mathcal{A} \), but not which node

Full observability:

\[ \mathcal{A} = \{\{i\} : i \in \mathcal{N}\}, \text{ i.e., } |\mathcal{A}| = 1 \]

But, could have \( |\mathcal{A}| = 2 \), where we know the stage but not the node
Updates to the Belief State

\[ \mathcal{A} = \{ A_1, A_2 \}, \text{ with } A_1 = \{ D_A, D_B \} \text{ and } A_2 = \{ H_A, H_B \} \]

\[ \mathbb{P}\{\text{Node} = k \mid \omega, A\} = \frac{1_{k \in A} \cdot \mathbb{P}\{\omega \mid \text{Node} = k\} \mathbb{P}\{\text{Node} = k\}}{\mathbb{P}\{\omega\}} \]

\[ b_k \leftarrow \frac{\left[ 1_{k \in A} \cdot \mathbb{P}(\omega \in \Omega_k) \right] \sum_{i \in N} b_i \phi_{ik}}{\sum_{i \in N} b_i \sum_{j \in A} \phi_{ij} \mathbb{P}(\omega \in \Omega_j)} \]

\[ b \leftarrow B(b, \omega) = \frac{D_A^\omega \Phi^\top b}{\sum_{i \in N} b_i \sum_{j \in A} \phi_{ij} \mathbb{P}(\omega \in \Omega_j)} \]
Policy Graphs with Partial Observability

Each node:

\[ b \leftarrow B(b, \omega) \]
\[ u = \pi_i(x, \omega, b) \]
\[ x' = T_i(x, u, \omega) \]

\[ C_i(x, u, \omega) \]

- All nodes in an ambiguity set have the same \( C_i, T_i, \) and \( U_i \)
- Children \( i^+ \), transition probabilities \( \phi_{ij} \), even \( \Omega_i \) may differ
Policy Graphs with Partial Observability

\[
\min_{\pi} \mathbb{E}_{i \in R^+; \omega \in \Omega_i} \left[ V_i(x_R, B_i(b_R, \omega), \omega) \right]
\]

(3)

where

\[
V_i(x, b, \omega) = \min_{u, \bar{x}, x'} C_i(\bar{x}, u, \omega) + \mathcal{V}(x', b)
\]

s.t. \( \bar{x} = x \)

\( u \in U_i(\bar{x}, \omega) \)

\( x' = T_i(\bar{x}, u, \omega) \)

and where

\[
\mathcal{V}(x', b) = \sum_{j \in \mathcal{N}} b_j \sum_{k \in \mathcal{N}} \phi_{jk} \sum_{\varphi \in \Omega_k} \mathbb{P}(\varphi \in \Omega_k) \cdot V_k(x', B_k(b, \varphi), \varphi)
\]

Goal: Find \( \pi_A(x, b, \omega) \) that solves (3) for each \( A \in \mathcal{A}, x, b, \) and \( \omega \)
Saddle Property of Cost-to-go Function

\[ V_i(x, b, \omega) = \min_{u, \bar{x}, x'} C_i(\bar{x}, u, \omega) + \mathcal{V}(x', b) \]
\[
\text{s.t. } \bar{x} = x \\
u \in U_i(\bar{x}, \omega) \\
x' = T_i(x, u, \omega)
\]

where

\[ \mathcal{V}(x', b) = \sum_{j \in \mathcal{N}} b_j \sum_{k \in \mathcal{N}} \phi_{jk} \sum_{\varphi \in \Omega_k} \mathbb{P}(\varphi \in \Omega_k) \cdot V_k(x', B_k(b, \varphi), \varphi) \]

Assume (A1)-(A5) with \( G \) acyclic

**Lemma 1.** Fix \( i, b, \omega \). Then, \( V_i(x, b, \omega) \) is piecewise linear convex in \( x \).

**Lemma 2.** Fix \( x' \). Then, \( \mathcal{V}(x', b) \) is piecewise linear concave in \( b \).

**Theorem 1.** \( \mathcal{V}(x', b) \) is a piecewise linear saddle function, which is convex in \( x' \) for fixed \( b \) and concave in \( b \) for fixed \( x' \).
Linear Interpolation: Towards an SDDP Algorithm

\[ V(b) = \max_{\gamma \geq 0} \sum_{k=1}^{K} \gamma_k V(\bar{b}_k) \]

s.t.

\[ \sum_{k=1}^{K} \gamma_k = 1 \]

\[ \sum_{k=1}^{K} \gamma_k \bar{b}_k = b \]
Saddle Function with Interpolated Cuts

\[ V(x', b) \]
Computing Cuts for What?

\[ V_i(x, b, \omega) = \min_{u, \bar{x}, x'} C_i(\bar{x}, u, \omega) + V_A(x', b) \]

s.t. \( \bar{x} = x \)
\( u \in U_i(\bar{x}, \omega) \)
\( x' = T_i(\bar{x}, u, \omega) \)

where

\[ V_A(x', b) = \sum_{j \in A} b_j \sum_{k \in j^+} \phi_{jk} \sum_{\varphi \in \Omega_k} \mathbb{P}(\varphi \in \Omega_k) \cdot V_k(x', B_k(b, \varphi), \varphi) \]
SDDP Master Program

\[ V^K_i(x, b, \omega) = \min_{u, \bar{x}, x', \theta} \max_{\gamma \geq 0} C_i(\bar{x}, u, \omega) + \sum_{k=1}^{K} \gamma_k \theta_k \]

s.t. \[ \bar{x} = x \]
    \[ u \in U_i(\bar{x}, \omega) \]
    \[ x' = T_i(\bar{x}, u, \omega) \]
    \[ \sum_{k=1}^{K} \gamma_k b_k = b \]
    \[ \sum_{k=1}^{K} \gamma_k = 1 \]
    \[ \theta_k \geq G_k x' + g_k, \quad k = 1, \ldots, K \]
SDDP Master Program

\[ V^K_i(x, b, \omega) = \min_{u, \bar{x}, x', \nu, \mu} C_i(\bar{x}, u, \omega) + \mu^\top b + \nu \]
\[ \text{s.t. } \bar{x} = x, \quad \mu^\top b_k + \nu \geq G_k x' + g_k, \quad k = 1, \ldots, K \]

**Theorem 2.** Assume (A1)-(A5) with \( \mathcal{G} \) acyclic. Let the sample paths of the “obvious” SDDP algorithm be generated independently at each iteration. Then, the algorithm converges to an optimal policy almost surely in a finite number of iterations.
Inventory Example

Demand model $A$: $\mathbb{P}(\omega = 1) = 0.2 \quad \mathbb{P}(\omega = 2) = 0.8$

Demand model $B$: $\mathbb{P}(\omega = 1) = 0.8 \quad \mathbb{P}(\omega = 2) = 0.2$

\[D_i : \quad D_i(x) = \min_{u, x' \geq 0} \left[ u + \mathbb{E}_{\omega}[H_i(x', \omega)] \right] \]
\[\text{s.t. } x' = x + u\]

\[H_i : \quad H_i(x, \omega) = \min_{u, x' \geq 0} \left[ 2u + x' + \rho D_i(x) \right] \]
\[\text{s.t. } x' = x + u - \omega\]
Inventory Example: Train Four Policies

1. *fully observable*: distribution known upon departing $R$
2. *partially observable*: ambiguity partition \( \{D_A, D_B\}, \{H_A, H_B\} \)
3. *risk-neutral average demand*: demand equally likely to be 1 or 2
4. *DRO average demand*: modified $\chi^2$ method with radius 0.25
Inventory Example: Train Four Policies

- 2000 out-of-sample costs over 50 periods; quartiles; $\rho = 0.9$
Inventory Example
One Sample Path of the Partially Observable Policy

(a) Belief
(b) First-stage buy
(c) Inventory
Concluding Thoughts

- Partially observable multistage stochastic programs
  - Saddle-cut SDDP algorithm
  - SDDP.jl (Dowson and Kapelevich)
- Related saddle-function work in stochastic programming
  - Baucke et al. (2018): risk measures
  - Downward et al. (2018): stage-wise dependent obj. coefficients
- Closely related ideas are well known in POMDPs
  - Contextual, multi-model, concurrent MDPs
  - We allow continuous state and action spaces via convexity
- Countably infinite LPs for cyclic case
- We did not handle decision-dependent learning
  - $b \leftarrow B(b, \omega)$ versus $b \leftarrow B(b, \omega, u)$
Concluding Thoughts

http://www.optimization-online.org/DB_HTML/2019/03/7141.html