Partially observable multistage stochastic programming

Oscar Dowson[a,*], David P. Morton[a], Bernardo K. Pagnoncelli[a,b]

[a]*Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA*
[b]*School of Business, Universidad Adolfo Ibáñez, Santiago, Chile*

Abstract

We propose a class of partially observable multistage stochastic programs and describe an algorithm for solving this class of problems. We provide a Bayesian update of a belief-state vector, extend the stochastic programming formulation to incorporate the belief state, and characterize saddle-function properties of the corresponding cost-to-go function. Our algorithm is a derivative of the stochastic dual dynamic programming method.

Keywords: Bayesian, multistage, partially observable, stochastic dual dynamic programming, stochastic programming

1. Introduction

Multistage stochastic programming is a framework for solving sequential decision problems under uncertainty. An algorithm for solving those problems is known as stochastic dual dynamic programming (SDDP) [1]. However, a critique of stochastic programming, including models solved by SDDP, is that the distribution of the uncertainty must be known *a priori* [2]. This distribution—referred to as the *nominal* distribution—is typically approximated from historical data and is an imperfect representation of the "true," *underlying* distribution. Such approximation can become a shortcoming when policies trained using the nominal distribution perform poorly when evaluated on the underlying distribution or, practically speaking, when evaluated on out-of-sample test data.

To circumvent this shortcoming, various approaches have been proposed in the literature. One approach is *distributionally robust* multistage stochastic programming; e.g., [2, 3, 4, 5]. In the distributionally robust setting, the agent optimizes with respect to the worst-case distribution over a set, often defined according to some distance metric:

$$\min_{x\in\mathcal{X}} \max_{P\in\mathcal{P}} \mathbb{E}_P[V(x,\omega)].$$

Here, $x \in \mathcal{X}$ captures feasible decisions that minimize the expectation of $V(x,\omega)$ with respect to a worst-case distribution P that lies in the *ambiguity set* $\mathcal{P}$. This formulation has the nice property that in choosing the ambiguity set, the modeler can impart domain knowledge about the likelihood of the underlying distribution. However, one critique of the distributionally robust approach in a multistage setting is that the agent does not revise the ambiguity set in response to new information. Some authors introduce adaptability via *decision-dependent* ambiguity sets in the two-stage case [6, 7], or provide multi-period uncertainty sets with here-and-now decisions [8]. However, we are unaware of analogous work exploring decision-dependent ambiguity sets in the multistage setting.

This paper addresses adaptability via a different approach. Instead of assuming a single nominal distribution, we assume that the modeler can construct a finite number of *candidate* distributions. Initially, the agent is unsure which of the candidate distributions is most similar to the underlying distribution. However, as time progresses, the agent observes realizations of the random vector from the underlying distribution and updates a belief about which of the candidates most accurately reflects the underlying distribution.

The adaptive approach just sketched has arisen in Markov decision processes (MDPs), where they are known as *contextual* MDPs (CMDPs) [9]. These have also been studied under the names *multi-model* MDPs [10] and *concurrent* MDPs [11]. CMDPs arise in a variety of settings. For example, the candidate distributions are possible rock strata in an oil drilling problem [12]; different types of visitors in online advertising [9]; and different treatment responses in healthcare [10]. CMDPs are a class of more general *partially observable* MDPs [13].

Our work is distinct from the MDP literature because we can compute provably optimal policies for problems with continuous state and control variables over a finite horizon. In contrast, to our knowledge, the relevant computational MDP literature focuses on problems with discrete state and control variables. The algorithm we describe is a form of value iteration [14], and it can also be viewed as approximate dynamic programming [15] or approximate linear programming [16] in which we exploit the structure of the problem through linear programming duality to find provably optimal basis functions.

Valladão et al. [17] discuss embedding a hidden Markov model in a stochastic program using SDDP, wherein the hidden Markov model corresponds to the nominal distribution switching between discrete states over time. However, because of the non-convex cost-to-go function, they relax the problem and formulate and solve the fully observable (convex) case. We also point to recent work [18, 19] in which a hidden Markov model

[*]Corresponding author
Email address: `oscar.dowson@northwestern.edu` (Oscar Dowson)

is incorporated in the approximate and stochastic dual dynamic programming algorithms. Our work differs in that we allow more general structures for the hidden Markov model, and we provide a proof of convergence.

To summarize, our contributions are: (i) introducing a general modeling framework for partial observability in stochastic programming; (ii) characterizing the saddle-function nature of the cost-to-go function; and (iii) showing how that characterization permits solution of the partially observable problem with continuous state and control variables by a saddle-cut variant of SDDP, which converges almost surely to an optimal policy.

The rest of the paper is laid out as follows. Section 2 introduces the policy graph as a way of modeling sequential decision problems, and Section 3 introduces the SDDP algorithm as a solution technique. Section 4 extends the modeling power of policy graphs to include partial observability and proves the key saddle property of the cost-to-go function. Section 5 exploits that property, presenting a saddle-cut version of the SDDP algorithm as a solution technique for partially observable multistage stochastic programs. We conclude in Section 6 with a numerical example to demonstrate the value of our approach.

2. Policy graphs

The policy graph framework of Dowson [20] models a multistage stochastic programming problem using a graph composed of *nodes* and *edges*. In each node i, the goal of the agent is to find a *decision rule*, $\pi_i(x, \omega)$, that maps the *incoming state variable* x and realization of a nodewise-independent *noise* ω to a feasible *control* u. We denote the set of feasible controls by $U_i(x, \omega)$, which depends on the state variable and noise realization. The noise ω is governed by a distribution with sample space Ω_i, and the term *nodewise-independent* means that the random variable ω is independent of x and of the realization of ω at all other nodes. We use $\mathbb{P}(\omega \in \Omega_i)$ to denote the probability of realizing ω at node i. After the agent chooses control u, the state variable transitions from the incoming state x, to the *outgoing state variable* x', according to a *transition function*, $x' = T_i(x, u, \omega)$. In addition, a cost of $C_i(x, u, \omega)$ is incurred.

Definition 1. In a *hazard-decision* node, the agent chooses a control u, according to the decision rule $\pi_i(x, \omega)$, *after* observing $\omega \in \Omega_i$. The state transitions from x to x' according to $T_i(x, u, \omega)$. The decision rule respects the set of admissible controls so that $u \in U_i(x, \omega)$, and cost $C_i(x, u, \omega)$ is incurred. A schematic of this is shown in Figure 1.

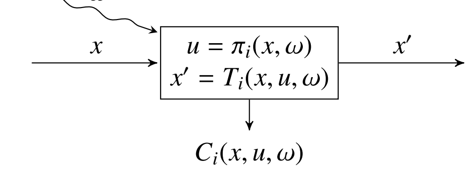

Figure 1: Schematic of a hazard-decision node.

Definition 2. A *policy graph*, $\mathcal{G} = (R, \mathcal{N}, \mathcal{E}, \Phi)$, is a tuple with a root node R, a further set of nodes $\mathcal{N}$, and directed edges, $\mathcal{E}$, connecting the nodes. An $|\mathcal{N}| + 1$ by $|\mathcal{N}|$ matrix Φ specifies transition probabilities between nodes, with entries ϕ_{ij}. Given a node i, for each edge $(i, j) \in \mathcal{E}$ we transition to node j with probability $\phi_{ij}>0$, where $i \in \mathcal{N} \cup \{R\}$, $j \in \mathcal{N}$, and $\sum_{j:(i,j)\in\mathcal{E}} \phi_{ij} \leq 1$. If no edge exists, then $\phi_{ij} = 0$.

Definition 3. The *children* of node i are the elements in the set $i^+ = \{j : \phi_{ij} > 0\}$, and i is a *leaf node* if $i^+ = \varnothing$.

The sum in Definition 2 is *at most* one to account for discount factors, cyclic policy graphs, and leaf nodes; see [20] for details. In what follows, we specialize to acyclic policy graphs.

Definition 4. A *multistage stochastic program* seeks an optimal policy π, which is a set of decision rules, $\pi_i(x, \omega)$, for each node $i \in \mathcal{N}$, that minimizes the expected cost at the root node:

$$\min_{\pi\in\Pi} \mathop{\mathbb{E}}_{i\in R^+;\ \omega\in\Omega_i} [V_i^\pi(x_R, \omega)], \tag{1}$$

where Π is the set of feasible policies, x_R is the initial state vector, and:

$$V_i^\pi(x, \omega) = \left\{ C_i(x, u, \omega) + \mathop{\mathbb{E}}_{j\in i^+;\ \varphi\in\Omega_j} \left[V_j^\pi(x', \varphi) \right] : \begin{array}{c} u = \pi_i(x, \omega) \\ x' = T_i(x, u, \omega) \end{array} \right\}.$$

The expectation operator, $\mathbb{E}[\cdot]$, accounts for transition probabilities ϕ_{ij} for nodes $j \in i^+$, and the probability mass function (pmf) governing the node-independent noise for each $\varphi \in \Omega_j$:

$$\mathop{\mathbb{E}}_{j\in i^+;\ \varphi\in\Omega_j} \left[V_j^\pi(x', \varphi) \right] = \sum_{j\in i^+} \phi_{ij} \sum_{\varphi\in\Omega_j} \mathbb{P}(\varphi \in \Omega_j) \cdot V_j^\pi(x', \varphi).$$

Definition 5. A *hazard-decision cost-to-go function* at node $i \in \mathcal{N}$, is given by:

$$\begin{aligned} \mathbf{HD}_i : V_i(x, \omega) = & \\ \min_{u,\bar{x},x'} \quad & C_i(\bar{x}, u, \omega) + \mathop{\mathbb{E}}_{j\in i^+;\ \varphi\in\Omega_j} \left[V_j(x', \varphi) \right] \\ \text{s.t.} \quad & \bar{x} = x \\ & x' = T_i(\bar{x}, u, \omega) \\ & u \in U_i(\bar{x}, \omega). \end{aligned} \tag{2}$$

We include the so-called *fishing constraint*, $\bar{x} = x$, to simplify an upcoming calculation of a subgradient with respect to x, and to simplify our upcoming assumption (A4).

Using Bellman's *principle of optimality* [21], we have that $V_i^{\pi^*}(x, \omega) = V_i(x, \omega)$ under an optimal policy π^* to (1) formed by choosing a control u in the arg min of (2):

$$\pi_i^*(x, \omega) \in \mathop{\arg\min}_{u\in U_i(x,\omega)} \left\{ C_i(x, u, \omega) + \mathop{\mathbb{E}}_{j\in i^+;\ \varphi\in\Omega_j} \left[V_j\left(T_i(x, u, \omega), \varphi\right) \right] \right\}.$$

Note that we have substituted out the transition function and fishing constraint for convenience.

A policy graph with $\mathcal{N} = \{1, 2, \ldots, |\mathcal{N}|\}$ and unit transition probabilities, $\phi_{R,1} = \phi_{i,i+1} = 1$, yields a standard multistage stochastic program with inter-stage independence [20], but other transition matrices Φ allow us to capture more general models such as those discussed in [20]. It is possible to replace $\mathbb{E}[\cdot]$ with an operator, $\mathbb{F}[\cdot]$, which represents a coherent risk measure [22], but we consider expectations for simplicity.

3. Stochastic dual dynamic programming

To facilitate a computationally tractable problem, which can be addressed by an SDDP-type algorithm, we make the following assumptions for the remainder of the paper. Given a policy graph $\mathcal{G} = (R, \mathcal{N}, \mathcal{E}, \Phi)$:

(A1) The number of nodes in $\mathcal{N}$ is finite.
(A2) Each $i \in \mathcal{N}$ is a hazard-decision node.
(A3) The sample space, Ω_i, is finite at each node $i \in \mathcal{N}$.
(A4) Given fixed x and ω, and excluding the expectation term, problem (2) at each node $i \in \mathcal{N}$ can be formulated as a linear program.
(A5) For each $i \in \mathcal{N}$, problem (2) is feasible and has a finite optimal solution for every incoming state x and $\omega \in \Omega_i$.
(A6) The graph $\mathcal{G}$ is acyclic.

Assumptions (A1), (A3), (A4), and (A6) imply model (1) is a multistage stochastic linear program of finite dimension. Assumption (A4) can be weakened to a convex program given additional assumptions such as an appropriate constraint qualification; see Girardeau et al. [23]. Assumption (A5) includes the notion of relatively complete recourse and ensures that problem (2) has a finite optimal solution. As a result, model (1) has a finite optimal solution.

The expected cost-to-go term in (2) is convex with respect to x'. Thus, it can be approximated from below by the maximum of a set of affine functions known as *cuts*:

$$\mathop{\mathbb{E}}_{j\in i^+;\ \varphi\in\Omega_j}\left[V_j(x',\varphi)\right] \geq \max_{k=1,2,\ldots,K}\left\{\alpha_k^{(i)} + \beta_k^{(i)\top}x'\right\}.$$

SDDP [1, 20, 24, 25, 26] is an algorithm that iteratively constructs the set of cuts for each node i. After K iterations the approximated version of $\mathbf{HD}_i$, which includes a scalar decision variable θ to linearize the cuts, is as follows:

$$\begin{aligned}
\mathbf{HD}_i^K : V_i^K(x,\omega) = \min_{u,\bar{x},x',\theta} \quad & C_i(\bar{x},u,\omega) + \theta \\
\text{s.t.} \quad & \bar{x} = x \qquad [\lambda] \\
& x' = T_i(\bar{x},u,\omega) \\
& u \in U_i(\bar{x},\omega) \\
& \theta \geq \alpha_k^{(i)} + \beta_k^{(i)\top}x', \quad k = 1,\ldots,K \\
& \theta \geq -M,
\end{aligned}$$

where M is sufficiently large. Each iteration of SDDP consists of two phases: (i) a forward pass, which samples a sequence of nodes and realizations of the node-independent noise terms, and generates a sequence of state variables at which new cuts are computed; and, (ii) a backward pass, which constructs a new cut at each realization of the state variables from the forward pass. Under mild sampling assumptions (notably, independence of the forward passes), this algorithm has been shown to converge to an optimal policy almost surely in a finite number of iterations [24]. Pseudo-code for the K^{th} iteration of a typical SDDP algorithm is given in Algorithm 1. The *rand*() function in the **while** loop yields a uniform random variable in [0, 1] and accounts for policy graphs that have discount factors. It also accounts for the situation $i^+ = \varnothing$, in which case the check is false and the forward pass is terminated. We iterate through list $\mathcal{S}$, from the last item to the first, via *reverse*($\mathcal{S}$). In addition to assumptions (A1)–(A6), execution of the algorithm requires that for each $i \in \mathcal{N}$, the size of $\sum_{j\in i^+}|\Omega_j|$ is modest so that the main step in the backward pass can be carried out.

Algorithm 1: K^{th} iteration of the SDDP algorithm.

```
/* Forward pass                                              */
set $x = x_R$, $\mathcal{S} = [\,]$, $i = R$
while ($rand() \in (0,1]) > 1 - \sum_{j\in i^+}\phi_{ij}$ do
    sample new $i$ from $i^+$ according to pmf, $\phi_{i,\cdot} / \sum_{j\in i^+}\phi_{ij}$
    sample $\omega$ from $\Omega_i$
    solve $\mathbf{HD}_i^K(x,\omega)$ and obtain $x'_i$
    append $(i, x'_i)$ to the list $\mathcal{S}$
    set $x = x'_i$
end
/* Backward pass                                             */
for $(i,\bar{x})$ in reverse($\mathcal{S}$) do
    if $i^+ = \varnothing$ then
        /* Handles leaf nodes                                */
        continue to next for-loop iterate
    end
    for $j \in i^+$, $\omega \in \Omega_j$ do
        solve $\mathbf{HD}_j^K(\bar{x},\omega)$
        set $\bar{\theta}_{j,\omega}^K$ to the optimal objective value
        set $\bar{\lambda}_{j,\omega}^K$ to an optimal dual vector $\lambda$
    end
    /* let $\mathbb{E}[\cdot]$ denote $\mathbb{E}_{j\in i^+;\omega\in\Omega_j}[\cdot]$     */
    set $\beta_{K+1}^{(i)} = \mathbb{E}\left[\bar{\lambda}_{j,\omega}^K\right]$
    set $\alpha_{K+1}^{(i)} = \mathbb{E}\left[\bar{\theta}_{j,\omega}^K\right] - \beta_{K+1}^\top\bar{x}$
    add the cut $\theta \geq \alpha_{K+1}^{(i)} + \beta_{K+1}^{(i)\top}x'$ to $\mathbf{HD}_i^K$
end
```

4. Policy graphs with imperfect node information

When formulating a multistage stochastic program as a policy graph, we assume that at each point in time the agent has perfect information regarding the current node at which a decision is to be made. In this section, we relax that modeling assumption.

Given a policy graph $\mathcal{G} = (R, \mathcal{N}, \mathcal{E}, \Phi)$, we replace the assumption of perfect node information by the assumption that the agent can partition the set of nodes $\mathcal{N}$ into *ambiguous subsets*. We denote the partition by $\mathcal{A}$, where $\bigcup_{A\in\mathcal{A}} A = \mathcal{N}$. By ambiguous subsets, we mean that the agent can observe the current ambiguity set, but is unsure of the current node within that set. So, in the case of perfect information from Section 2, we have $\mathcal{A} = \{\{i\} : i \in \mathcal{N}\}$, i.e., $|A| = 1$ for all $A \in \mathcal{A}$. Our generalization allows instead, e.g., for $|A| = 2$, where each $A \in \mathcal{A}$ could correspond to a particular time stage; i.e., the agent can observe the current stage, but not which node within the stage.

Definition 6. An *ambiguity partition* $\mathcal{A}$ partitions the nodes $\mathcal{N}$ into subsets of nodes that the agent cannot distinguish.

We extend a policy graph to $\mathcal{G} = (R, \mathcal{N}, \mathcal{E}, \Phi, \mathcal{A})$. We assume that the agent can observe the cost function C_i, transition function T_i, and set of feasible controls U_i when making a decision at $A \ni i$. Here, C_i, T_i, and U_i must be identical for all nodes in an ambiguity set because otherwise the agent could distinguish the nodes. The set of children i^+ and transition probabilities may differ between nodes within an ambiguity set, and the probability distribution governing realizations $\omega \in \Omega_i$ may also differ. Often the support of ω will be identical within an ambiguity set. That said, we allow $\Omega_i \neq \Omega_{i'}$ for $i, i' \in A$, and the agent can distinguish i from i' if $\omega \in \Omega_i \setminus \Omega_{i'}$ is observed.

Under imperfect node information, the agent models the probability of being in each node via a belief state.

Definition 7. At a fixed point in time, the *belief state b* is modeled by an $|\mathcal{N}|+1$ dimensional vector, where component b_i is the probability that the current node is i. The initial belief state is the unit vector with probability 1 on the root node R.

Given that the agent can observe the current ambiguity set, $A \in \mathcal{A}$, we have $b_i = 0$ for $i \notin A$, and of course, $\sum_{i \in A} b_i = 1$ and $b \geq 0$. Under perfect information, we have $|A| = 1$ for all A, meaning b is the unit vector corresponding to the current node.

Upon entering ambiguity set A and observing ω, the agent updates the belief state from b to b' using Bayes' theorem:

$$b'_i = \frac{[\mathbf{1}_{i \in A} \cdot \mathbb{P}(\omega \in \Omega_i)] \cdot \sum_{j \in \mathcal{N}_R} b_j \cdot \phi_{ji}}{\sum_{j \in \mathcal{N}_R} b_j \sum_{k \in A} \phi_{jk} \cdot \mathbb{P}(\omega \in \Omega_k)}, \tag{3}$$

where $\mathbf{1}_{i \in A} = 1$ if $i \in A$ and 0 otherwise, and $\mathcal{N}_R \equiv \mathcal{N} \cup \{R\}$. The numerator of (3) has two components. The first is the probability of observing A and ω conditional on the agent being in node i, and the second is the probability of transitioning to node i given previous belief b. The denominator is the probability of observing A and ω given the previous belief b and acts to normalize $\|b'\|_1 = 1$.

Equation (3) can be expressed in vector form as follows:

$$b' = B_k(b, \omega) = \frac{D^{\omega}_{A_k} \Phi^\top b}{\sum_{i \in \mathcal{N}_R} b_i \sum_{j \in A_k} \phi_{ij} \cdot \mathbb{P}(\omega \in \Omega_j)}, \tag{4}$$

where $D^{\omega}_{A_k}$ is an $|\mathcal{N}| \times |\mathcal{N}|$ diagonal matrix in which element (i, i) is $\mathbf{1}_{i \in A_k} \cdot \mathbb{P}(\omega \in \Omega_i)$. We have added subscript k to (unobservable) node k's ambiguity set A to facilitate subsequent node-specific calculations. In what follows we refer to (4) as the *Bayesian update*. This update hinges on our assumptions regarding: Markovian node-to-node transitions; the agent knowing the probability distribution governing ω for each $i \in \mathcal{N}$ and the transition probabilities Φ; and, the agent observing ω.

Analogous to model (1), we now define a partially observable multistage stochastic program.

Definition 8. A *partially observable multistage stochastic program* seeks an optimal policy $\pi \in \Pi$ that minimizes the expected cost at the root node:

$$\min_{\pi \in \Pi} \mathcal{V}^\pi(x_R, b_R),$$

where x_R is the initial state vector, b_R is the initial belief vector which places probability 1 on the root node R,

$$\mathcal{V}^\pi(x', b) = \sum_{j \in \mathcal{N}_R} b_j \sum_{k \in \mathcal{N}} \phi_{jk} \sum_{\varphi \in \Omega_k} \mathbb{P}(\varphi \in \Omega_k) \cdot V^\pi_k(x', B_k(b, \varphi), \varphi), \tag{5}$$

and:

$$V^\pi_i(x, b, \omega) = \left\{ C_i(x, u, \omega) + \mathcal{V}^\pi(x', b) : \begin{array}{l} u = \pi_i(x, b, \omega) \\ x' = T_i(x, u, \omega) \end{array} \right\}.$$

Here, $\mathcal{V}^\pi$ accounts for: (i) the probability the current node is j, b_j; (ii) the probability of transitioning from j to k, ϕ_{jk}; (iii) the probability of observing φ at k, $\mathbb{P}(\varphi \in \Omega_k)$; and (iv) the cost-to-go at node k given the updated belief $B_k(b, \varphi)$, $V^\pi_k(x', B_k(b, \varphi), \varphi)$. Note that the decision rule now includes the belief state vector. Analogous to the fully observable case, i.e., (2), we can represent an optimal policy, π^*, by selecting $\pi^*_i(x, b, \omega)$ as an element of the arg min of the following:

$$\textbf{Belief-SP} \quad \begin{array}{rl} V_i(x, b, \omega) = \min\limits_{u, \bar{x}, x'} & C_i(\bar{x}, u, \omega) + \mathcal{V}(x', b) \\ \text{s.t.} & \bar{x} = x \\ & x' = T_i(\bar{x}, u, \omega) \\ & u \in U_i(\bar{x}, \omega), \end{array} \tag{6}$$

with $V^{\pi^*}_i(x, b, \omega) = V_i(x, b, \omega)$.

Remark 1. *If the agent is in ambiguity set $A \in \mathcal{A}$, then $b_j = 0$ for all nodes $j \notin A$. Thus, at all nodes $i \in A$, $\mathcal{V}$ can be replaced in problem* (6) *by a function $\mathcal{V}_A$:*

$$\mathcal{V}_A(x', b) = \sum_{j \in A} b_j \sum_{k \in j^+} \phi_{jk} \sum_{\varphi \in \Omega_k} \mathbb{P}(\varphi \in \Omega_k) \cdot V_k(x', B_k(b, \varphi), \varphi).$$

We sometimes use the expression in equation (5), and other times it is more convenient to use the view in Remark 1. We now characterize properties of value function, $\mathcal{V}(x', b)$.

Lemma 1. *Assume (A1)-(A6), let $i \in \mathcal{N}$ and fix b and $\omega \in \Omega_i$. The function $V_i(x, b, \omega)$ is convex with respect to x.*

Proof. The node set $\mathcal{N}$ is finite by (A1). Thus, by (A6) there is a finite set of sample paths of nodes indexed, say, by $S \in \mathcal{S}$, rooted at node i that accounts for the sequence of nodes visited until the sample path terminates in a leaf node. The probability mass function governing $S \in \mathcal{S}$ is determined by the transition matrix Φ. The distribution of ω at each $j \in S$ has finite support by (A3). Hence, under (A4) and (A5) we can express $V_i(x, b, \omega)$ as the optimal value of a bounded and feasible linear program of finite dimension. Since x appears only on the right-hand side of the linear program, $V_i(\cdot, b, \omega)$ is convex. □

Lemma 2. *$B_i(b, \omega) = B_j(b, \omega)$ for all $j \in A_i$.*

Proof. The result is immediate because $A_i = A_j$. □

Lemma 3. *$V_i(x, b, \omega) = V_j(x, b, \omega)$ for all $j \in A_i$.*

Proof. The result is again immediate because all nodes $j \in A_i$ share the same cost functions C_j, transition functions T_j, and feasibility sets U_j. $\square$

Lemma 4. *Assume (A1)-(A6). The function $\mathcal{V}(x', b)$ is concave with respect to b for fixed x'.*

Proof. Construct a topological ordering of the nodes, $\mathcal{N}$, in the acyclic policy graph, so that nodes have higher precedence in the order than their children and so that all leaf nodes constitute the lowest precedence values. Then, given a node i, fix x' and $\varphi \in \Omega_k$, and assume that $V_k(x', b', \varphi)$ is a piecewise-linear, concave function of b' for all nodes k in the topological order subsequent to i. This assumption is trivially satisfied at leaf nodes of the policy graph, at which there is no cost-to-go term. We now show that for fixed x and $\omega \in \Omega_i$ that $V_i(x, b, \omega)$ is also a piecewise-linear concave function of b where:

$$V_i(x,b,\omega) = \min_{u,\bar{x},x'} \Bigg\{ C_i(\bar{x},u,\omega) + \sum_{j\in\mathcal{N}_R} b_j \sum_{A\in\mathcal{A}} \sum_{k\in A} \phi_{jk} \sum_{\varphi\in\Omega_k} \mathbb{P}(\varphi\in\Omega_k)\cdot V_k(x', B_k(b,\varphi),\varphi) \Bigg\}.$$

Note that we split the inner sum over $k \in \mathcal{N}$ from (5) into an equivalent sum over the ambiguity partition and then nodes within each element of the partition, and for simplicity we have suppressed the constraints of (6) that govern $u, \bar{x}, x'$. Re-arranging terms, using Lemmas 2 and 3, and with a slight abuse of notation so that $B_k = B_A$ and $V_k = V_A$, we obtain:

$$V_i(x,b,\omega) = \min_{u,\bar{x},x'} \Bigg\{ C_i(\bar{x},u,\omega) + \sum_{A\in\mathcal{A}} \sum_{\varphi\in\Omega_A} \sum_{j\in\mathcal{N}_R} b_j \sum_{k\in A} \phi_{jk}\cdot \mathbb{P}(\varphi\in\Omega_k)\cdot V_A(x', B_A(b,\varphi),\varphi) \Bigg\},$$

where $\Omega_A = \bigcup_{k\in A} \Omega_k$. Then, since the inner $V_A(x', \cdot, \varphi)$ is a piecewise-linear concave function, it can be represented by the minimum of a collection of affine functions:

$$V_i(x,b,\omega) = \min_{u,\bar{x},x'} \Bigg\{ C_i(\bar{x},u,\omega) + \sum_{A\in\mathcal{A}} \sum_{\varphi\in\Omega_A} \sum_{j\in\mathcal{N}_R} b_j \sum_{k\in A} \phi_{jk}\cdot \mathbb{P}(\varphi\in\Omega_k)\cdot \min_{\gamma\in\Gamma_A(x',\varphi)} \gamma^\top B_A(b,\varphi) \Bigg\}, \quad (7)$$

for an appropriately defined $\Gamma_A(x', \varphi)$. Here, $e^\top B_A(b, \varphi) = 1$, where e is the vector of all 1s, and so equation (7)'s piecewise-linear concave construct for $V_A(x', \cdot, \varphi)$ allows for an intercept in each of the affine pieces. Substituting the expression for the Bayesian update (4) yields:

$$V_i(x,b,\omega) = \min_{u,\bar{x},x'} \Bigg\{ C_i(\bar{x},u,\omega) + \sum_{A\in\mathcal{A}} \sum_{\varphi\in\Omega_A} \left[\sum_{j\in\mathcal{N}_R} b_j \sum_{k\in A} \phi_{jk}\cdot \mathbb{P}(\varphi\in\Omega_k) \right] \cdot \min_{\gamma\in\Gamma_A(x',\varphi)} \gamma^\top \left[\frac{D_A^\varphi \Phi^\top b}{\sum_{j\in\mathcal{N}_R} b_j \sum_{k\in A} \phi_{jk}\cdot \mathbb{P}(\varphi\in\Omega_k)} \right] \Bigg\},$$

which simplifies to:

$$V_i(x,b,\omega) = \min_{u,\bar{x},x'} \left\{ C_i(\bar{x},u,\omega) + \sum_{A\in\mathcal{A}} \sum_{\varphi\in\Omega_A} \min_{\gamma\in\Gamma_A(x',\varphi)} \gamma^\top D_A^\varphi \Phi^\top b \right\}.$$

The right-hand side of this expression for $V_i(x, b, \omega)$ is the minimum (over $u, \bar{x}, x'$) of a sum of piecewise-linear concave functions of b. Thus, $V_i(x, \cdot, \omega)$ is a piecewise-linear concave function. Applying this argument inductively up the topological order for all nodes $i \in \mathcal{N}$, we find that $\mathcal{V}(x', \cdot)$ is a piecewise-linear concave function. $\square$

Theorem 1. *Assume (A1)-(A6), let V be defined by* (6), *$\mathcal{V}(x', b)$ be defined by* (5), *and assume the belief state is updated according to* (4). *Then, $\mathcal{V}(x', b)$ is a saddle function, which is convex in x' for fixed b and concave in b for fixed x'.*

Proof. If we fix b then by Lemma 1 and equation (5), $\mathcal{V}(\cdot, b)$ is a convex combination of convex functions, $V_k(\cdot, B_k(b, \varphi), \varphi)$. Thus, $\mathcal{V}(x', b)$ is convex with respect to x'. If we fix x' then by Lemma 4, $\mathcal{V}(x', \cdot)$ is concave, yielding the desired result. $\square$

5. Solution method

Theorem 1 establishes that the cost-to-go function, $\mathcal{V}$, is a saddle function. Such saddle functions also arise in the work of Downward et al. [27] and Baucke et al. [28]. In contrast to our motivation to handle partial observability, their motivation, and the analogous concave component of $\mathcal{V}$, arises from modeling stagewise-dependent uncertainty in the objective function [27] and modeling a class of risk-averse optimization models [28]. While the form of our cost-to-go function differs due to the update for belief states, we borrow ideas from [27] to develop a variant of the SDDP algorithm that converges almost surely to an optimal policy in a finite number of iterations.

We begin by forming a lower-approximation of saddle function $\mathcal{V}(x', b)$ using an outer approximation for x' and an inner approximation for b. As in a standard SDDP algorithm, this approximation is iteratively refined. Each iteration consists of a forward pass, in which a sequence of nodes i, states x', and beliefs b are sampled, and a backward pass, in which the approximation is refined at the sampled points of the forward pass. After K iterations, model (6) is approximated by the following min-max subproblem:

$$\begin{array}{rll}
\textbf{Belief-SP}_i^K: & \bar{V}_i^K(x,b,\omega) = & \\
\min\limits_{u,\bar{x},x',\theta} \max\limits_{\gamma} & C_i(\bar{x},u,\omega) + \sum\limits_{k=1}^{K} \gamma_k\theta_k & \\
\text{s.t.} & \bar{x} = x & [\lambda] \\
& x' = T_i(\bar{x},u,\omega) & \\
& u \in U_i(\bar{x},\omega) & \\
& \sum\limits_{k=1}^{K} \gamma_k b_k = b & [\mu] \\
& \sum\limits_{k=1}^{K} \gamma_k = 1 & [\nu] \\
& \gamma_k \geq 0, \quad k = 1,\ldots,K & \\
& \theta_k \geq \alpha_k^{(i)} + \beta_k^{(i)\top} x', \quad k = 1,\ldots,K & \\
& \theta_k \geq -M, \quad k = 1,\ldots,K. &
\end{array} \quad (8)$$

The sequence of cut inequalities provides an outer approximation with respect to x' in the spirit of Benders' decomposition. The inner approximation with respect to b is achieved by adding a sequence of columns involving b_k with decision variables, γ_k, in the spirit of Dantzig-Wolfe decomposition. A visualization of the cost-to-go function is given in Figure 2.

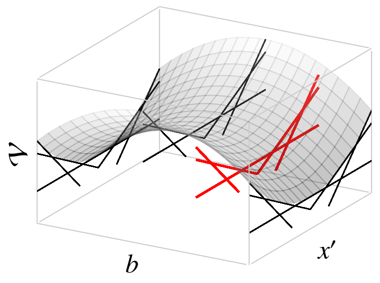

Figure 2: Example of a saddle function. Thin (black) straight lines are cuts w.r.t. x'. The γ variables handle interpolation of these functions to create the thick (red) straight lines of the interpolated approximate value function.

Feasibility of the inner maximization in (8) can be ensured by initializing with b_k equal, in turn, to each unit vector. Subproblem (8) has bilinear terms between γ_k and θ_k. However, because the Bayesian update does not depend on the state x or the control u, we can convert (8) into a single minimization model by taking the dual of the inner maximization problem:

$$
\begin{aligned}
\textbf{Belief-SP}_i^K:\quad & \bar{V}_i^K(x,b,\omega) = \\
\min_{u,\bar{x},x',\nu,\mu}\quad & C_i(\bar{x},u,\omega) + \mu^\top b + \nu \\
\text{s.t.}\quad & \bar{x} = x, && [\lambda] \\
& x' = T_i(\bar{x},u,\omega) \\
& u \in U_i(\bar{x},\omega) \\
& \mu^\top b_k + \nu \ge \alpha_k^{(i)} + \beta_k^{(i)\top} x', && k = 1,\ldots,K \\
& \mu^\top b_k + \nu \ge -M, && k = 1,\ldots,K.
\end{aligned}
\tag{9}
$$

Given fixed values for x, b, and ω, subproblem (9) is now tractable and can be solved as a linear program, or a convex program given an appropriate variant of assumption (A4).

Returning to Remark 1, given that the agent is currently in ambiguity set A, has belief b, and outgoing state variable x', a saddle cut that refines the approximation of $\mathcal{V}_A(x', b)$ can be computed using Algorithm 2. Note that compared to standard SDDP cut-computation algorithms, the only difference is that instead of knowing with certainty that we wish to add a cut to node i, we loop over all nodes in the current ambiguity set A and weigh these values by the agent's current belief. Pseudo-code for the full algorithm is given in Algorithm 3.

Theorem 2. *Assume (A1)-(A6), let V be defined by* (6), *$\mathcal{V}(x', b)$ be defined by* (5), *and assume the belief state is updated according to* (4). *Let the sample paths of Algorithm 3 in the forward pass be generated independently at each iteration K. Then, Algorithm 3 converges to an optimal policy almost surely in a finite number of iterations.*

Proof. The set of piecewise linear functions defining each function, $V_k(x', B_k(b,\varphi),\varphi)$, is finite in both x' and b. Given finite Ω_k and $\mathcal{N}$ it follows from equation (5) that $\mathcal{V}(x', b)$ is also a piecewise-linear saddle function, characterized by a finite number of pieces in both x' and b. Under Bayesian update (4) with finite Ω_k and an acyclic graph, there is a finite set of b_k values that can be generated across all forward passes of the algorithm. Convergence then follows via an argument, analogous to that of [24, 27], which uses the following: (i) there are a finite number of cuts that can be generated and hence after some iteration no more cuts are formed; (ii) if no new cuts are generated during a *deterministic* sequence of forward and backward passes that enumerates all possible sample paths then the algorithm has converged to an optimal policy; and, (iii) by the second Borel-Cantelli lemma [29], under independently sampling in the forward pass, we will almost surely sample the deterministic sequence of paths from (ii) after the last iteration in which a new cut is added. □

Algorithm 2: Cut calculation.

```
Given $K$, $A$, $x'_K$, $b_K$
for $i \in A$, $j \in i^+$, $\omega \in \Omega_j$ do
    solve Belief-SP$_j^K(x'_K, b_K, \omega)$ and obtain an extreme
      point solution
    set $\bar{\theta}_{i,j,\omega}^K$ to the optimal objective value
    set $\bar{\lambda}_{i,j,\omega}^K$ to an optimal dual vector $\lambda$
end
/* let $\mathbb{E}[\cdot]$ denote $\sum_{i\in A}(b_K)_i \mathbb{E}_{j\in i^+;\omega\in\Omega_j}[\cdot]$          */
set $\beta_{K+1}^{(i)\top} = \mathbb{E}\left[\bar{\lambda}_{i,j,\omega}^K\right]$
set $\alpha_{K+1}^{(i)} = \mathbb{E}\left[\bar{\theta}_{i,j,\omega}^K\right] - \beta_{K+1}^{(i)\top} x'_K$
obtain the saddle-cut $\mu^\top b_K + \nu \ge \alpha_{K+1}^{(i)} + \beta_{K+1}^{(i)\top} x'$
```

Algorithm 3: Belief-state SDDP.

```
Given $x_R$ and $b_R$
set $K = 0$
while true do
    set $x = x_R$, $b = b_R$, $\mathcal{S} = [\,]$, $i = R$
    while $(rand() \in (0,1]) > 1 - \sum_{j\in i^+}\phi_{ij}$ do
        sample new $i$ from $i^+$ according to pmf,
          $\phi_{i,\cdot} / \sum_{j\in i^+}\phi_{ij}$
        sample $\omega$ from $\Omega_i$
        set $A = A_i$
        set $b' = B_i(b,\omega)$
        solve Belief-SP$_i^K(x, b', \omega)$ and obtain $x'$, an
          extreme point solution
        append $(A, x', b')$ to the list $\mathcal{S}$
        set $x = x'$, $b = b'$
    end
    for $(A, \bar{x}', \bar{b}')$ in reverse($\mathcal{S}$) do
        obtain saddle-cut from Algorithm 2 given $K$, $A$,
          $\bar{x}'$, $\bar{b}'$
        add the saddle cut to nodes $i \in A$, where $i^+ \neq \varnothing$
    end
    set $K = K + 1$
end
```

6. Inventory management example

This section explores our method through a simple inventory management example, which has two stages in each cyclic period. In the first stage, the agent can purchase widgets at a price of \$1/widget, and there is no uncertainty. In the second stage, a demand of ω widgets is revealed and, if necessary, the agent must purchase additional widgets to meet demand at a cost of \$2/widget. Unsold widgets at the end of the stage incur a holding cost of \$1/widget. The agent has two candidate distributions for demand, which we index by $i = A, B$. The distribution of demand for each model is given in Table 1.

	Model A		Model B	
Demand ω	1	2	1	2
Probability	0.8	0.2	0.2	0.8

Table 1: Distribution of demand models A and B.

We assume that each demand model is equally likely. The policy graph of this problem is given in Figure 3.

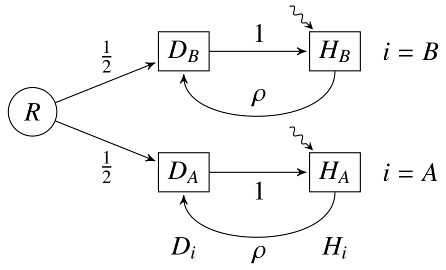

Figure 3: Policy graph of the inventory management problem.

For demand model $i = A, B$, the first-stage node can be formulated as:

$$\mathbf{D}_i : D_i(x) = \min_{u, x' \geq 0} \left\{ u + \mathop{\mathbb{E}}_{\omega}[H_i(x', \omega)] \mid x' = x + u \right\},$$

and the second-stage node as:

$$\mathbf{H}_i : H_i(x, \omega) = \min_{u, x' \geq 0} \left\{ 2u + x' + \rho D_i(x') \mid x' = x + u - \omega \right\},$$

where ρ is the probability the process continues (or equivalently, a discount factor), u is the quantity of widgets purchased, x is the number of widgets in inventory at the start of the stage, and x' is the number in inventory at the end of the stage.

To solve this problem, we implemented Algorithm 3 in the `SDDP.jl` package [30]. Then, we trained four policies using four different assumptions:

1. *fully observable*, where the agent knows the underlying distribution after departing R;
2. *partially observable*, where the ambiguity partition is $\{D_A, D_B\}, \{H_A, H_B\}$;
3. *risk-neutral average demand*, where demand is equally likely to be 1 or 2 units, and the agent is risk neutral; and,
4. *risk-averse average demand*, where demand is equally likely to be 1 or 2 units, and the agent is risk-averse, using a risk measure with the risk set formed by the convex hull of the candidate distributions listed in Table 1.

In each of the four cases, we assume $\rho = 0.9$. To satisfy the acyclic assumption (A6), we "unroll" each cycle 50 times to create an acyclic policy graph with 100 stages (50 two-stage periods). After training each policy using the appropriate variant of the SDDP algorithm, we performed a Monte Carlo simulation of the policy over 50 two-stage periods with 2000 replications. In each case, the same 2000 realizations of demand were used. Note that in the average demand case, we simulated demand using the original probability distribution, not the uniform demand model used to train the policy. The distributions of the discounted cumulative cost over the 50 periods are given by boxplots in Figure 4.

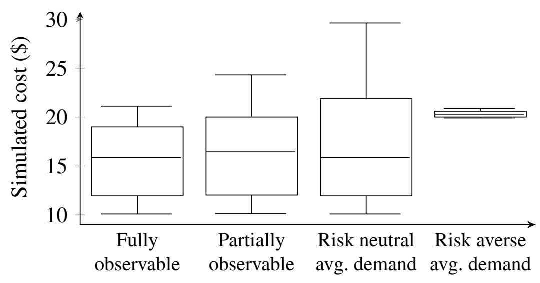

Figure 4: Distribution of discounted cumulative costs over 50 periods. Divisions in each boxplot are at 0, 25, 50, 75, and 100th empirical percentiles. The expected cost of the four policies grows from left-to-right in the plot.

As expected, the *fully observable* policy performs best because it can make an optimal decision at each stage. In model A, the policy buys widgets so that it has 1 unit in inventory at the end of the first stage. In model B, the policy buys widgets so that it has 2 units in inventory at the end of the first stage.

Compared with the *fully observable* policy, the *partially observable* policy has a higher median cost (\$18.9 compared with \$17.9), and a higher worst-case cost (\$24.3 compared with \$21.1). However, because the *partially observable* policy is able to learn over time which candidate distribution is the underlying distribution, it has minimum and 25th percentile costs that are identical to the *fully observable* policy.

Compared with the *partially observable* policy, the policy from assuming *risk-neutral average demand* is more expensive in the worst 50% of outcomes. This is consistent with the fact that the first-stage buying decision is based on an incorrect probability distribution of future demand. As a result, the policy always ends the first stage with 1 unit in inventory. This strategy is optimal for model A. However, if the the underlying distribution is model B, the agent is more likely to have to purchase widgets on the spot-market at \$2/widget. This results in a higher maximum cost of \$29.6.

To counter-act this risk, the *risk-averse average demand* policy minimizes the worst-case outcome over an ambiguity set. This reduces the maximum cost so that the worst-case outcome is similar to the worst-case outcome using the *fully observable* policy (\$20.9 compared with \$21.1). However, because the risk measure only considers the worst-case outcomes, the cost of the best-case outcomes is significantly increased (\$19.9 compared with \$10.1).

This example shows the benefit that solving the partially observable model can bring. By enabling the policy to adapt to distributional information as it is revealed, the partially observable policy can perform similarly to the fully observable policy, and better than either of the static average demand policies.

An example of the policy adapting to new information is given in Figure 5. When the agent's belief in model B exceeds 0.5, the first-stage buying decision ensures that 2 units of inventory enter the second stage. When the agent's belief is instead at most 0.5, the first-stage decision ensures only 1 unit of inventory enters the second stage.

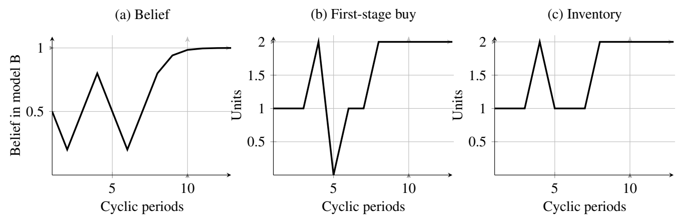

Figure 5: One sample path of the *partially observable* policy.

This section's example is simple for explanatory purposes. Real-world applications include hydro-thermal scheduling problems in which the sequence of inflows is modeled by a hidden Markov model with "wet" and "dry" hidden climate states and mixing between the hidden states over time. There are also applications in finance, in which the three distributions are "bear," "bull," and "neutral" market states, and the observed noise is the market return. See [17, 18, 19] for examples.

Supplementary material

We provide an implementation of Algorithm 3 in `SDDP.jl` [30], an open-source library for solving multistage stochastic programming problems using JuMP [31] and Julia [32]. See `https://github.com/odow.SDDP.jl`.

Acknowledgements

This article was developed, in part, based upon funding from the Alliance for Sustainable Energy, LLC, Managing and Operating Contractor for the National Renewable Energy Laboratory for the U.S. Department of Energy. The authors thank three referees whose comments improved the paper.

References

[1] M. Pereira, L. Pinto, Multi-Stage Stochastic Optimization Applied to Energy Planning, Mathematical Programming 52 (1991) 359–375.
[2] H. Rahimian, S. Mehrotra, Distributionally Robust Optimization: A Review, Optmization Online Http://www.optimization-online.org/DB_FILE/2019/08/7332.pdf.
[3] B. Analui, G. C. Pflug, On Distributionally Robust Multiperiod Stochastic Optimization, Computational Management Science 11 (3) (2014) 197–220.
[4] A. B. Philpott, V. de Matos, L. Kapelevich, Distributionally Robust SDDP, Computational Management Science 15 (3-4) (2018) 431–454.
[5] D. Bertsimas, S. Shtern, B. Sturt, A Data-Driven Approach for Multi-Stage Linear Optimization, Optimization Online Http://www.optimization-online.org/DB_FILE/2018/11/6907.pdf.
[6] F. Luo, S. Mehrotra, Distributionally Robust Optimization with Decision Dependent Ambiguity Sets, arXiv:1806.09215 [math] .
[7] N. Noyan, G. Rudolf, M. Lejeune, Distributionally Robust Optimization with Decision-Dependent Ambiguity Set, Optimization Online Http://www.optimization-online.org/DB_HTML/2018/09/6821.html.
[8] O. Nohadani, K. Sharma, Optimization under Connected Uncertainty, Working Paper, IEMS Department, Northwestern University .
[9] A. Hallak, D. Di Castro, S. Mannor, Contextual Markov Decision Processes, arXiv:1502.02259 .
[10] L. N. Steimle, D. L. Kaufman, B. T. Denton, Multi-Model Markov Decision Processes, Optimization Online Http://www.optimization-online.org/DB_FILE/2018/01/6434.pdf.
[11] P. Buchholz, D. Scheftelowitsch, Computation of Weighted Sums of Rewards for Concurrent MDPs, Mathematical Methods of Operations Research 89 (1) (2019) 1–42.
[12] R. R. Torrado, J. Rios, G. Tesauro, Optimal Sequential Drilling for Hydrocarbon Field Development Planning, in: Proceedings of the 29th AAAI Conference on Innovative Applications, San Francisco, 4734–4739, 2017.
[13] L. P. Kaelbling, M. L. Littmany, A. R. Cassandra, Planning and Acting in Partially Observable Stochastic Domains, Artificial Intelligence 101 (1998) 99–134.
[14] R. Howard, Dynamic Programming and Markov Processes, MIT Press, Cambridge, MA., 1960.
[15] W. B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality, Wiley Series in Probability and Statistics, Wiley, Hoboken, N.J, 2nd ed edn., 2011.
[16] D. de Farias, B. van Roy, The Linear Programming Approach to Approximate Dynamic Programming, Operations Research 51 (6) (2003) 850–865.
[17] D. Valladão, T. Silva, M. Poggi, Time-Consistent Risk-Constrained Dynamic Portfolio Optimization with Transactional Costs and Time-Dependent Returns, Annals of Operations Research 282 (1-2) (2019) 379–405.
[18] J. Durante, J. Nascimento, W. Powell, Backward Approximate Dynamic Programming with Hidden Semi-Markov Stochastic Models in Energy Storage Optimization, arXiv:1710.03914 [math] .
[19] J. Durante, J. Nascimento, W. Powell, Risk Directed Importance Sampling in Stochastic Dual Dynamic Programming with Hidden Markov Models for Grid Level Energy Storage, arXiv:2001.06026 [math] .
[20] O. Dowson, The policy graph decomposition of multistage stochastic programming problems, Networks 76 (2020) 3–23.
[21] R. Bellman, Dynamic Programming, Princeton University Press, Princeton, 1957.
[22] P. Artzner, F. Delbaen, J.-M. Eber, D. Heath, Coherent Measures of Risk, Mathematical Finance 9 (3) (1999) 203–228.
[23] P. Girardeau, V. Leclère, A. B. Philpott, On the Convergence of Decomposition Methods for Multistage Stochastic Convex Programs, Mathematics of Operations Research 40 (1) (2015) 130–145.
[24] A. B. Philpott, Z. Guan, On the Convergence of Sampling-Based Methods for Multi-Stage Stochastic Linear Programs, Operations Research Letters 36 (2008) 450–455.
[25] A. Shapiro, Analysis of Stochastic Dual Dynamic Programming Method, European Journal of Operational Research 209 (1) (2011) 63–72.
[26] V. Guigues, Convergence Analysis of Sampling-Based Decomposition Methods for Risk-Averse Multistage Stochastic Convex Programs, SIAM Journal on Optimization 26 (4) (2016) 2468–2494.
[27] A. Downward, O. Dowson, R. Baucke, Stochastic dual dynamic programming with stagewise-dependent objective uncertainty, Operations Research Letters 48 (2020) 33–39.
[28] R. Baucke, A. Downward, G. Zakeri, A Deterministic Algorithm for Solving Multistage Stochastic Minimax Dynamic Programmes, Optimization Online Http://www.optimization-online.org/DB_FILE/2018/02/6449.pdf.
[29] G. Grimmett, D. Stirzaker, Probability and Random Processes, Oxford University Press, Oxford, second edn., 1992.
[30] O. Dowson, L. Kapelevich, SDDP.Jl: A Julia Package for Stochastic Dual Dynamic Programming, INFORMS Journal on Computing In press.
[31] I. Dunning, J. Huchette, M. Lubin, JuMP: A Modeling Language for Mathematical Optimization, SIAM Review 59 (2) (2017) 295–320.
[32] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, Julia: A Fresh Approach to Numerical Computing, SIAM Review 59 (1) (2017) 65–98.