Stochastic Mixed-Integer Programming: A Survey

Ward Romeijnders[a], Yihang Zhang[b], Suvrajeet Sen[b]

[a]*University of Groningen, Department of Operations, Nettelbosje 2, 9747 AE, Groningen, The Netherlands*
[b]*University of Southern California, Epstein Department of Industrial and Systems Engineering, 3715 McClintock Ave, Los Angeles, 90089, CA, USA*

Abstract

The goal of this survey is to provide a road-map for exploring the growing area of stochastic mixed-integer programming (SMIP) models and algorithms. We provide a comprehensive overview of existing decomposition algorithms for two-stage SMIPs, including Dantzig-Wolfe decomposition, dual decomposition, Lagrangian cuts, and decomposition approaches using parametric cutting planes and scaled cuts. Moreover, we explicitly discuss the relationship between these methods. Furthermore, building on these two-stage results, we summarize recent developments in the emerging field of multistage stochastic mixed-integer programming, and we present directions for future research.

Keywords: Stochastic programming, Integer programming

1. Introduction

Stochastic Programming (SP) emerged from the seminal work of Dantzig (1955) and Beale (1955), extending deterministic optimization, particularly linear programming, to accommodate circumstances where uncertainty in the underlying data is a critical factor. When integer constraints are imposed on a subset of decision variables within this uncertain environment, Stochastic Mixed-Integer Programming (SMIP) models arise. Conceptually, SMIP extends both stochastic linear programming by incorporating discrete choices and deterministic mixed-integer programming by accounting for significant data uncertainty.

The inherent difficulty in SMIP models stems from the dual challenge of: (a) managing discrete decision variables, and (b) accommodating the random variables that represent uncertainty. For instance, unit commitment problems (Philpott et al., 2000; Hobbs et al., 2001; Huang et al., 2017), which involve discrete first-stage decisions on committing indivisible power generation units, frequently lead to very large-scale SMIPs. SMIP models are also applied in healthcare settings for problems such as surgical scheduling (Pham and Klinkert, 2008; Bovim et al., 2020). Integer restrictions can also extend to future decision stages, as seen in stochastic server location problems (Ntaimo and Sen, 2005). In multi-stage SMIPs, future decisions may also need to satisfy integer constraints, often building upon or adapting to earlier choices. The intractability of such problems is particularly pronounced due to the coupling of these integer-constrained decisions across multiple stages. This linking creates complex interdependencies which, combined with the discrete nature of some decision variables, significantly amplifies the overall difficulty.

While traditional stochastic linear programs often exhibit a nested convex structure, allowing for approximation via sequences of convex optimization problems, this advantageous property is lost in SMIP.

Email addresses: w.romeijnders@rug.nl (Ward Romeijnders), zhangyih@usc.edu (Yihang Zhang), suvrajes@usc.edu (Suvrajeet Sen)

The introduction of discrete decisions leads to non-convex value functions, fundamentally altering the solution landscape. This survey focuses on the intricate challenges of bridging the theoretical foundations of stochastic linear programming with the complexities arising from the discrete, sequential nature of decisions in multistage SMIPs. With advances in SMIP algorithms, computing platforms, and data resources, there is an increased need to understand the current boundaries of SMIP formulations, and algorithms. With this survey we wish to provide a road-map for a reader with modest SP background to explore the growing area of SMIP models and algorithms.

The main contributions of this survey are as follows.

- We provide a comprehensive overview of existing decomposition algorithms for two-stage SMIPs. Included in this study are a variety of cuts that are used for decomposition algorithms. In particular we present parametric cutting planes, Lagrangian cuts and scaled cuts.
- We also present decomposition algorithms including dual decomposition and Dantzig-Wolfe decomposition, and we discuss the relationship between them.
- We summarize emerging developments in the area of multistage stochastic mixed-integer programming.

The remainder of this paper is organized as follows. In Section 2 we define two-stage SMIP and present some of its properties. In Section 3 we discuss several decomposition algorithms for two-stage SMIP, and discuss their relationship. Unless all first-stage variables are binary, these approaches typically need to be embedded in a (spatial) B&B tree, which is the topic of Section 4. We also discuss scaled cuts in this section. In Section 5, we move to multistage SMIP, and provide its definition. Building on the two-stage results, we next discuss dual and primal decomposition approaches for multistage SMIP in Sections 6 and 7, respectively. We end with some conclusions in Section 8 which includes some directions for future research.

2. Properties of two-stage SMIP

We consider two-stage SMIPs of the form

$$\eta^* = \min_{x \in X} \left\{ c^\top x + Q(x) \right\},$$

where $Q(x)$ represents the expected value function, defined as

$$Q(x) = \mathbb{E}_\omega[v(\omega, x)], \qquad x \in X,$$

where the second-stage value function v is defined as

$$v(\omega, x) = \min_{y \in Y} \left\{ q(\omega)^\top y : T(\omega)x + W(\omega)y \geq h(\omega) \right\}, \qquad \omega \in \Omega, x \in X.$$

In these models, the variables $x \in X \subseteq \mathbb{R}^{n_1}$ represent the first-stage decisions, also referred to as here-and-now decisions, that have to be made before the realizations of the random parameters $\omega \in \Omega \subseteq \mathbb{R}^m$ become known. In the second stage, we decide upon the second-stage (recourse) decisions $y \in Y \subset \mathbb{R}^{n_2}$,

after we observe the realization of ω. We assume that X and Y may impose non-negativity and integrality restrictions on the first- and second-stage decisions x and y, respectively. Moreover, we assume that any linear constraints on x and y, such as $Ax \geq b$, are included in X and Y, respectively.[1] Furthermore, we assume that the probability distribution $\mathbb{P}_\omega$ of ω is known.

In addition to the structures imposed by decision variables, SP models also impose some structure on requirements imposed on the representation of uncertainty. In this regard, we will often assume that ω can take on at most finitely many realizations, which we will denote by $\omega_1, \ldots, \omega_S$ with probability $\mathbb{P}_\omega(\omega = \omega_s) = p_s$, $s = 1, \ldots, S$. These scenarios $\omega_1, \ldots, \omega_S$ may, for example, be obtained by sampling from the true probability distribution $\mathbb{P}_\omega$, yielding the well-known sample average approximation (SAA), see Kleywegt et al. (2002).

Assumption 1. *Unless stated otherwise, we assume throughout this survey that the recourse is relatively complete and sufficiently expensive. That is, we have $-\infty < v(\omega, x) < +\infty$ for all $\omega \in \Omega$ and $x \in X$. Similarly, we assume that $\mathbb{E}_\omega[|\omega_i|] < +\infty$ for all $i = 1, \ldots, m$, so that $Q(x)$ is finite for all $x \in X$. Moreover, we assume that X and Y are compact and that all parameters in the model are rational.*

2.1. Simple integer recourse

Simple integer recourse (SIR) is the integer extension of the continuous simple recourse model (Wets, 1983), and can be considered as the simplest second-stage structure that is possible for a two-stage SMIP. Its one-sided one-dimensional version has the following form, see Louveaux and van der Vlerk (1993); Haneveld and van der Vlerk (1999),

$$v(\omega, x) = \min_{y \in \mathbb{Z}_+} \left\{ y : y \geq \omega - x \right\} = \lceil \omega - x \rceil^+, \qquad \omega \in \Omega, x \in X.$$

For every $\omega \in \Omega$, this value function $v(\omega, x)$ is non-convex and discontinuous in x. This also implies that in general, for two-stage SMIPs, the second-stage value function $v(\omega, x)$ and expected value function $Q(x)$ may be neither convex nor continuous. However, Klein Haneveld et al. (2006) and van Beesten et al. (2024) show that exceptions do exist. In particular, they show that the SIR expected value function Q is convex if the distribution of ω can be represented as a convex combination of possibly infinitely-many uniform distributions with support of integer length.

2.2. General mixed-integer recourse

In general, similar to simple integer recourse models, it holds that both the second-stage mixed-integer value function v and expected value function Q of a two-stage SMIP are discontinuous if ω is discretely distributed with finitely many realizations. If ω is continuously distributed, then Q may be continuous as well, for example if only the right-hand side is random, see Schultz (2003b). Under all circumstances, it does hold that both v and Q are lower semicontinuous, see Proposition 1 below. We refer to Schultz (2003a) and Sen (2005) for more detailed properties of v and Q.

Proposition 1. *Consider the second-stage mixed-integer value function v and expected value function Q of a two-stage SMIP. Then, under Assumption 1 it holds that*

(i) $v(\omega, \cdot)$ is lower semicontinuous for all $\omega \in \Omega$, and

(ii) Q is lower semicontinuous.

[1] For nonlinear SMIPs, see, e.g., Li et al. (2011).

3. Algorithms for two-stage SMIP

In this section, we discuss algorithms for solving two-stage SMIP. Section 3.1 introduces the large-scale deterministic equivalent formulation, which allows us to solve a two-stage SMIP as a large-scale mixed-integer linear program. The remaining sections are dedicated to decomposition algorithms for two-stage SMIPs. These algorithms may be classified in one of the following forms: (i) primal decomposition or (ii) dual decomposition. The former algorithms decompose the SMIP problem by time stages, whereas the latter decompose the problem by scenarios.

Primal decomposition approaches.. In primal decomposition approaches the problem is decomposed into a master problem and several subproblems, where the master problem is given by

$$\text{(MP)} \quad \min_{x\in X,\theta\in\mathbb{R}} \left\{c^\top x + \theta : \theta \geq \alpha_k^\top x + \beta_k \ \forall k \in K\right\}. \tag{1}$$

Here, θ is an auxiliary variable approximating the epigraph of the expected value function $Q(x)$, where $\theta \geq \alpha_k^\top x + \beta_k$, $k \in K$, are known as the optimality cuts. That is, they are affine lower bounds for the expected value function Q, so that $Q(x) \geq \alpha_k^\top x + \beta_k$ for all $k \in K$ and $x \in X$.

Remark 1. *Problem (MP) in (1) is known as the single-cut master problem since we only use a single auxiliary variable $\theta \in \mathbb{R}$ and a single collection of optimality cuts to lower bound the expected value function Q. Alternatively, we may use a multi-cut master problem in which each optimality cut provides a lower bound for a scenario s, and because there are multiple scenarios indexed by $s = 1, \ldots, S$, one associates sets of optimality cuts corresponding to auxiliary variables θ^s, $s = 1, \ldots, S$, each lower bounding the second-stage value function $v_s := v(\omega_s, \cdot)$ for each scenario s, separately.*

Dual decomposition approaches. In dual decomposition approaches we introduce copies x_s, $s = 1, \ldots, S$ of the first-stage decisions for every scenario $s = 1, \ldots, S$. The problem then can be written in so-called split-variable form as

$$\begin{aligned}
\eta^* = \min_{x_1,\ldots,x_S,y_1,\ldots,y_S} \quad & \sum_{s=1}^{S} p_s\left(c^\top x_s + q_s^\top y_s\right) && && (2)\\
\text{s.t.} \quad & T_s x_s + W_s y_s \geq h_s && \forall s = 1, \ldots, S,\\
& x_s \in X, y_s \in Y && \forall s = 1, \ldots, S,\\
& x_1 = \ldots = x_S,
\end{aligned}$$

where $q_s := q(\omega_s)$, $T_s := T(\omega_s)$, $W_s := W(\omega_s)$, and $h_s := h(\omega_s)$ for all $s = 1, \ldots, S$.

The last set of constraints are called the *non-anticipativity constraints*. Without these constraints the problem separates into smaller scenario subproblems, leading for example to the wait-and-see lower bound (Birge and Louveaux, 1997; Klein Haneveld et al., 2020; Shapiro et al., 2021), or stronger bounds when only a subset of the non-anticipativity constraints are relaxed (Sandıkçı et al., 2013; Maggioni and Pflug, 2016; Sandikçi and Özaltin, 2017). Alternatively, approaches such as dual decomposition are based on Lagrangian relaxation over these non-anticipativity constraints.

Remark 2. *There exist several alternative but theoretically equivalent formulations of the non-anticipativity constraints in the literature. For example, in this section we will use $x = x_s$ for all $s = 1, \ldots, S$ for some artificial auxiliary variable x.*

Scenario set convexification. The challenge with both primal and dual decomposition approaches is that the second-stage value functions v_s, $s = 1, \dots, S$, and the expected value function Q are typically non-convex when some of the second-stage decision variables y are integer. For example, for primal decomposition algorithms this implies that the affine optimality cuts $\theta \geq \alpha_k^\top x + \beta_k$, $k \in K$, are not tight everywhere, and at best may yield $\text{co}_X(Q)$, the convex envelope of Q with respect to X. Similarly, in a multi-cut master problem the variables θ^s may at best represent $\text{co}_X(v_s)$. It turns out that these convex envelopes $\text{co}_X(v_s)$ of v_s may also be obtained by *separately convexifying the scenario sets* $\mathcal{F}_s$, with $\mathcal{F}_s$ defined for all $s = 1, \dots, S$, as

$$\mathcal{F}_s = \Big\{(x, y_s) \in X \times Y : T_s x + W_s y_s \geq h_s\Big\}. \tag{3}$$

Lemma 1. *For every $s = 1, \dots, S$ and $x \in \text{conv}(X)$ it holds that*

$$co_X(v_s)(x) = \min_{y_s} \Big\{q_s^\top y_s : (x, y_s) \in \text{conv}(\mathcal{F}_s)\Big\}.$$

Proof. See the Appendix. □

Thus, many primal and dual decomposition approaches may be unable to obtain η^* but instead are able to retrieve the lower bound $\bar{\eta}$ defined as

$$\begin{aligned} \bar{\eta} = \min_{x, y_1, \dots, y_S} \quad & c^\top x + \sum_{s=1}^{S} p_s q_s^\top y_s & & \\ \text{s.t.} \quad & (x, y_s) \in \text{conv}(\mathcal{F}_s) & & \forall s = 1, \dots, S. \end{aligned} \tag{4}$$

Note that in this problem above the decisions $x, y_1, \dots, y_S$ are continuous since the scenario sets $\mathcal{F}_s$ are replaced by their convex hulls $\text{conv}(\mathcal{F}_s)$.

There are several ways to represent the convex hull $\text{conv}(\mathcal{F}_s)$: (i) by adding cutting planes, and (ii) by enumerating the extreme points, or vertices, of $\text{conv}(\mathcal{F}_s)$. Moreover, (iii) we can represent $\text{co}_X(v_s)$ by its supporting hyperplanes $\mathcal{L}_s(\pi_s) - \pi_s^\top x$, which can be obtained via Lagrangian relaxation of the non-anticipativity constraints.

In the following subsections we discuss several decomposition methods that use one of these approaches to obtain a lower bound $\bar{\eta}$. They include parametric cutting planes, Dantzig-Wolfe (DW) decomposition, dual decompositions, and Lagrangian cuts. It turns out that these methods, although independently presented in the literature, are intimately related. In this survey we will discuss their relationships.

Remark 3. *In Section 4.1, we show that when the first-stage decisions x are restricted to be pure binary that then $\text{co}_X(v_s)(x) = v_s(x)$ for all $s = 1, \dots, S$ and $x \in X$, so that the lower bounding minimization problem in (4) with the additional constraints $x \in X$ yields the exact optimal objective value η^* and exact optimal first-stage decision x^*. In other cases, the lower bound $\bar{\eta}$ is not necessarily tight but may be embedded in a Branch-and-Bound scheme.*

Remark 4. *For computational purposes, it may be more efficient to use a set $\hat{\mathcal{F}}_s \supset \text{conv}(\mathcal{F}_s)$ in (4) if this only leads to a slightly weaker lower bound that can be computed much faster.*

3.1. The large-scale deterministic equivalent formulation

Under the finite scenario assumption, we can, in principle, obtain a recourse vector y_s for each for each scenario $s = 1, \ldots, S$. The corresponding large-scale deterministic equivalent formulation is given by

$$\begin{aligned} \eta^* = \min_{x, y_1, \ldots, y_S} \quad & c^\top x + \sum_{s=1}^{S} p_s q_s^\top y_s & \\ \text{s.t.} \quad & T_s x + W_s y_s \geq h_s & \forall s = 1, \ldots, S, \\ & x \in X, y_s \in Y & \forall s = 1, \ldots, S. \end{aligned}$$

This is a large deterministic MILP for which both the variables as well as the constraints grow with the number of scenarios S. Consequently, when the number of scenarios are very large, such all-in-one deterministic equivalent formulation can easily become too large to be solved by using deterministic MIP solvers.

3.2. Affine parametric cutting planes

In view of the final remark above, a natural approach to separately constructing convex polyhedral lower bounds for the second-stage value functions v_s, $s = 1, \ldots, S$, given by

$$v_s(x) = \min_{y_s \in Y} \left\{ q_s^\top y_s : W_s y_s \geq h_s - T_s x \right\}, \tag{5}$$

is to relax the integrality constraints on y_s, while bearing in mind that the second-stage mixed-integer feasible region

$$\mathcal{Y}_s(x) := \{ y_s \in Y : W_s y_s \geq h_s - T_s x \} \tag{6}$$

must be strengthened via cutting planes which are parametrized by x. Indeed, since $v_s(x)$ is the value function of a mixed-integer linear program for every $x \in X$, we can equivalently solve it by replacing the feasible region $\mathcal{Y}_s(x)$ by its convex hull $\text{conv}(\mathcal{Y}_s(x))$. That is,

$$v_s(x) := \min_{y_s \in \text{conv}(\mathcal{Y}_s(x))} q_s^\top y_s, \qquad x \in X, s = 1, \ldots, S. \tag{7}$$

The problem with this approach, however, is that the cutting planes with which $\mathcal{Y}_s(x)$ need to be strengthened to obtain $\text{conv}(\mathcal{Y}_s(x))$ differ for every x, and thus cannot be reused in a decomposition algorithm. Hence, instead we may strengthen $\mathcal{Y}_s(x)$ using *parametric cutting planes* in x. However, the form of these parametric cutting planes has significant impact on the computational tractability of the master problem, since non-linear parametric cutting planes for the feasible region of v_s may translate to non-linear lower bounds for v_s. A sufficient condition to obtain convex polyhedral lower bounds for v_s is to use parametric cuts for $\mathcal{Y}_s(x)$ that are *affine* in x.

Definition 1. *An affine parametric cutting plane $\pi^\top(x, y_s) \geq \pi_0$ is valid for the feasible region $\mathcal{Y}_s(x)$ with respect to X if and only if for every $\bar{x} \in X$ and $\bar{y}_s \in \mathcal{Y}_s(\bar{x})$ it holds that $\pi^\top(\bar{x}, \bar{y}_s) \geq \pi_0$.*

An important observation is that any affine parametric cut in x for $\mathcal{Y}_s(x)$ is also valid for the scenario set $\mathcal{F}_s := \{(x, y_s) \in X \times Y : T_s x + W_s y_s \geq h_s\}$, and since it is an affine cut, also for $\text{conv}(\mathcal{F}_s)$.

Proposition 2. *Let $s = 1, \dots, S$, be given, and consider the feasible region $\mathcal{Y}_s(x)$ in (5) for some $x \in X$. Then, every affine parametric cutting plane for $\mathcal{Y}_s(x)$ with respect to X is valid for*

$$\text{conv}(\mathcal{F}_s) = conv\Big\{(x, y_s) \in X \times Y : T_s x + W_s y_s \geq h_s\Big\}.$$

Proposition 2 implies that the best lower bound for v_s that we may obtain using affine parametric cuts, is to use all of those required to construct $\text{conv}(\mathcal{F}_s)$. Let $\bar{T}_s x + \bar{W}_s y_s \geq \bar{h}_s$ denote the system of inequalities that describes $\text{conv}(\mathcal{F}_s)$, which includes $\bar{A}x \geq \bar{b}$ to describe $\text{conv}(X)$. That is, $\text{conv}(\mathcal{F}_s) = \{(x, y_s) : \bar{T}_s x + \bar{W}_s y_s \geq \bar{h}_s\}$. Then, by Lemma 1 it holds that

$$\text{co}_X(v_s)(x) = \min_{y_s} \Big\{q_s^\top y_s : \bar{T}_s x + \bar{W}_s y_s \geq \bar{h}_s\Big\}.$$

In a practical algorithm, we do not intend to add all those cuts at once. Instead, we iteratively add them, leading to increasingly strengthened sets $\hat{\mathcal{F}}_s^k := \{(x, y_s) : T_s^k x + W_s^k y_s \geq h_s^k\}$, $k = 1, \dots, K$, with $\hat{\mathcal{F}}_s^1 \supseteq \cdots \supseteq \hat{\mathcal{F}}_s^K \supseteq \text{conv}(\mathcal{F}_s)$ for every $s = 1, \dots, S$. Here, typically, $\hat{\mathcal{F}}_s^1 := \{(x, y_s) : T_s x + W_s y_s \geq h_s\}$, that is, the original set $\mathcal{F}_s$ with all integrality constraints relaxed.

Corresponding to each set $\hat{\mathcal{F}}_s^k$, we define a convex polyhedral outer approximation $\hat{v}_s^k$, i.e., a lower bound, of v_s, defined for every $x \in \text{conv}(X)$ as

$$\begin{aligned}\hat{v}_s^k(x) &= \min_{y_s} \Big\{q_s^\top y_s : (x, y_s) \in \hat{\mathcal{F}}_s^k\Big\} \\ &= \min_{y_s} \Big\{q_s^\top y_s : T_s^k x + W_s^k y_s \geq h_s^k\Big\}. \end{aligned} \tag{8}$$

Since the sets $\hat{\mathcal{F}}_s^k$ are iteratively strengthened, it holds that $\hat{v}_s^1 \leq \ldots \leq \hat{v}_s^K \leq \text{co}_X(v_s)$. Moreover, for each $k = 1, \dots, K$,

$$\begin{aligned}\hat{\eta}^k &:= \min_{x \in \text{conv}(X)} \Big\{c^\top x + \sum_{s=1}^S p_s \hat{v}_s^k(x)\Big\} \\ &= \min_{x, y_1, \dots, y_S} \Big\{c^\top x + \sum_{s=1}^S p_s q_s^\top y_s : (x, y_s) \in \hat{\mathcal{F}}_s^k, \quad \forall s = 1, \dots, S\Big\}, \end{aligned} \tag{9}$$

yields a lower bound $\hat{\eta}^k$ of $\bar{\eta}$ with $\hat{\eta}^1 \leq \ldots \leq \hat{\eta}^K \leq \bar{\eta}$.

The minimization problem in (9) can be interpreted as the large-scale deterministic formulation of a two-stage stochastic program with continuous decision variables only, and can thus be solved by decomposition algorithms such as the L-shaped algorithm (Van Slyke and Wets, 1969), i.e., Benders decomposition. Conceptually, we may obtain $\bar{\eta}$ in iterative fashion, by solving (9) in each iteration, with or without decomposition, to determine the current solution $(x^k, y_1^k, \dots, y_S^k)$, and by strenghtening the scenario set $\hat{\mathcal{F}}_s^k$ for every $s = 1, \dots, S$, by separating (x^k, y_s^k) from $\text{conv}(\mathcal{F}_s)$ using an affine parametric cutting plane if $(x^k, y_s^k) \notin \text{conv}(\mathcal{F}_s)$. However, it is more efficient to run a single decomposition algorithm and to embed the scenario set strengthening within the decomposition algorithm. In such an algorithm, we iteratively solve a single master problem of the form

$$\bar{\eta}^t := \min_{x, \theta_1, \dots, \theta_S} \Big\{c^\top x + \sum_{s=1}^S p_s \theta_s : \theta_s \geq \alpha_s^\tau + (\beta_s^\tau)^\top x, \quad \forall \tau = 1, \dots, t, \ \forall s = 1, \dots, S\Big\},$$

yielding the current solution $(x^t, \theta_1^t, \ldots, \theta_S^t)$. Next, for each scenario $s = 1, \ldots, S$, given the current scenario set $\hat{\mathcal{F}}_s^k$, we solve the LP subproblem $\hat{v}_s^k(x^t)$ in (8) yielding an optimal primal solution y_s^t and an optimal dual solution λ_s^t. Indeed, by strong LP duality, we have

$$\hat{v}_s^k(x^t) = \max_{\lambda_s \geq 0} \left\{ \lambda_s^\top (h_s^k - T_s^k x^t) : \lambda_s^\top W_s^k \leq q_s^\top \right\},$$

and thus the optimal dual solution λ_s^t may be used to derive an optimality cut $\theta_s \geq (\lambda_s^t)^\top (h_s^k - T_s^k x)$ for the master problem if $\theta_s^t < \hat{v}_s^k(x^t)$. Note that this optimality cut is affine since the parametric cutting planes $T_s^k x + W_s^k y_s \geq h_s^k$ are affine in x.

At the same time, the primal solution y_s^t may be used to strengthen the scenario set $\hat{\mathcal{F}}_s^k$ if $(x^t, y_s^t) \notin \text{conv}(\mathcal{F}_s)$ by separating (x^t, y_s^t) from $\text{conv}(\mathcal{F}_s)$ using an affine parametric cutting plane. If at some iteration t it holds that $(x^t, y_s^t) \in \text{conv}(\mathcal{F}_s)$ and $\theta_s^t \geq \hat{v}_s^k(x^t)$ for all $s = 1, \ldots, S$, then we can stop. It holds that $\bar{\eta}^t = \bar{\eta}$.

The main challenges in designing such decomposition algorithms are (i) to determine when to add optimality cuts and when to add affine parametric cutting planes, and (ii) to find an affine parametric cutting plane that separates $(x^t, y_s^t) \notin \text{conv}(\mathcal{F}_s)$ from $\text{conv}(\mathcal{F}_s)$. For example, if the first-stage decision variables are pure binary, then we may decide to only do the separation when $x^t \in X$. In this case, $(x^t, y_s^t) \in \text{conv}(\mathcal{F}_s)$ if and only if $y_s^t \in \mathcal{Y}_s(x^t)$, i.e., if y_s^t satisfies its integrality restrictions. To guarantee obtaining a current binary first-stage solution it is possible to sometimes solve the master problem as a MIP with binary restrictions on x.

There exist several types of affine parametric cutting planes in the literature, most of them inspired by cutting plane techniques for deterministic mixed-integer linear programming, including disjunctive programming (Sen and Higle, 2005), Lift-and-Project cuts (Carøe and Tind, 1997), Fenchel cuts (Ntaimo, 2013), Gomory cuts (Gade et al., 2014; Zhang and Küçükyavuz, 2014), and mixed-integer rounding cuts (Kim and Mehrotra, 2015; Bodur and Luedtke, 2017). A review of these methods can be found in the survey by Küçükyavuz and Sen (2017). We refer to these references for details on these types of cuts.

3.3. Dantzig-Wolfe decomposition

Another possible way to represent $\text{conv}(\mathcal{F}_s)$ is to enumerate its extreme points (x_s^k, y_s^k), $k \in K_s$, for every $s = 1, \ldots, S$. Then, any $(x_s, y_s) \in \text{conv}(\mathcal{F}_s)$ can be written as a convex combination of these extreme points. That is, for every $(x_s, y_s) \in \text{conv}(\mathcal{F}_s)$, there exist $\lambda_s^k \in [0, 1]$ with $\sum_{k \in K_s} \lambda_s^k = 1$ so that

$$(x_s, y_s) = \sum_{k \in K_s} \lambda_s^k (x_s^k, y_s^k).$$

Using this representation of $\text{conv}(\mathcal{F}_s)$, we may solve the lower bounding problem in (4), presented here in split-variable form with $\mathcal{X} \supseteq \text{conv}(X)$ as

$$\begin{aligned}
\bar{\eta} = \min_{x \in \mathcal{X}} \min_{\substack{x_1, \ldots, x_S \\ y_1, \ldots, y_S}} \quad & \sum_{s=1}^{S} p_s (c^\top x_s + q_s^\top y_s) & \\
\text{s.t.} \quad & (x_s, y_s) \in \text{conv}(\mathcal{F}_s) & \forall s = 1, \ldots, S, \\
& x_s = x & \forall s = 1, \ldots, S,
\end{aligned}$$

by optimizing over the variables λ_s^k, $k \in K_s$, $s = 1, \ldots, S$, instead of over (x_s, y_s), $s = 1, \ldots, S$. Defining $c_s^k := c^\top x_s^k + q_s^\top y_s^k$, $k \in K_s$, $s = 1, \ldots, S$, as the costs associated with extreme point (x_s^k, y_s^k), the resulting

Dantzig-Wolfe master problem (Lulli and Sen, 2004; Singh et al., 2009; Schulze et al., 2017) becomes

$$\begin{aligned} \bar{\eta} = \min_{x \in \mathcal{X}} \min_{\lambda \geq 0} \quad & \sum_{s=1}^{S} \sum_{k \in K_s} p_s \lambda_s^k c_s^k && && (10) \\ \text{s.t.} \quad & \sum_{k \in K_s} \lambda_s^k = 1 && \forall s = 1, \dots, S, \\ & \sum_{k \in K_s} \lambda_s^k x_s^k = x && \forall s = 1, \dots, S. \end{aligned}$$

In general, $\text{conv}(\mathcal{F}_s)$ may have exponentially many extreme points for every scenario $s = 1, \dots, S$, and thus the Dantzig-Wolfe master problem in (10) may have exponentially many variables. That is why, in practice we initially only include a few extreme points and use column generation to iteratively add extreme points with negative reduced costs, i.e., that help improve the objective value, until no such extreme points can be found.

Given a subset $\bar{K}_s \subseteq K_s$ of extreme points of $\text{conv}(\mathcal{F}_s)$ for every $s = 1, \dots, S$, the restricted DW master problem is given by

$$\begin{aligned} \bar{\eta}_{RMP} = \min_{x \in \mathcal{X}} \min_{\lambda \geq 0} \quad & \sum_{s=1}^{S} \sum_{k \in \bar{K}_s} p_s \lambda_s^k c_s^k \\ \text{s.t.} \quad & \sum_{k \in \bar{K}_s} \lambda_s^k = 1 && \forall s = 1, \dots, S, && (p_s \theta_s) \\ & \sum_{k \in \bar{K}_s} \lambda_s^k x_s^k = x && \forall s = 1, \dots, S, && (p_s \pi_s) \end{aligned}$$

with optimal objective value $\bar{\eta}_{RMP}$ yielding an upper bound on $\bar{\eta}$. Letting $p_s\theta_s$ and $p_s\pi_s$, $s = 1, \dots, S$, denote the dual variables corresponding to the first and second set of constraints, respectively, the reduced costs of any decision variable λ_s^k corresponding to extreme point (x_s^k, y_s^k) with $k \notin \bar{K}_s$, $s = 1, \dots, S$, are given by $p_s(c_s^k - \theta_s + \pi_s^\top x_s^k)$. Hence, λ_s^k has negative reduced costs if

$$c_s^k - \theta_s + \pi_s^\top x_s^k = c^\top x_s^k + q_s^\top y_s^k - \theta_s + \pi_s^\top x_s^k < 0.$$

Moreover, note that the optimality conditions of the resticted DW master problem imply that any variable λ_s^k with $k \in \bar{K}_s$ cannot have negative reduced costs. Hence, by minimizing the reduced costs over $k \in K_s$, $s = 1, \dots, S$, we can determine for which variable λ_s^k these costs are lowest, and thus also determine whether any λ_k^s, $k \notin \bar{K}_s$, $s = 1, \dots, S$, has negative reduced costs. If not, then the optimal solution to the restricted DW master problem is also optimal for the full DW master problem and $\bar{\eta}_{RMP} = \bar{\eta}$. This minimization problem over the reduced costs, also called the pricing problem, is for every $s = 1, \dots, S$, equal to

$$\begin{aligned} \min_{k \in K_s} \left\{ c^\top x_s^k + q_s^\top y_s^k - \theta_s + \pi_s^\top x_s^k \right\} &= \min_{k \in K_s} \left\{ c^\top x_s^k + q_s^\top y_s^k + \pi_s^\top x_s^k \right\} - \theta_s \\ &= \min_{x_s, y_s} \left\{ c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \text{conv}(\mathcal{F}_s) \right\} - \theta_s \\ &= \min_{x_s, y_s} \left\{ c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \mathcal{F}_s \right\} - \theta_s \\ &= \mathcal{L}_s(\pi_s) - \theta_s, \end{aligned}$$

where

$$\mathcal{L}_s(\pi_s) := \min_{x_s, y_s} \Big\{ c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \mathcal{F}_s \Big\},$$

and where the second equality holds since (x_s^k, y_s^k), $k \in K_s$, are the extreme points of $\text{conv}(\mathcal{F}_s)$, and the third equality holds since the objective value is linear.

Summarizing, we iteratively solve the restricted master problem $\bar{\eta}_{RMP}$ with columns indexed by $\bar{K}_s$ for all $s = 1, \ldots, S$, to obtain prices (θ_s, π_s) for all $s = 1, \ldots, S$. Next, we solve the pricing problem $\mathcal{L}_s(\pi_s) - \theta_s$ for all $s = 1, \ldots, S$ to obtain columns with negative reduced costs. Expand the index sets $\bar{K}_s$ with any such columns and reiterate until no columns with negative reduced costs are obtained.

3.4. Dual decomposition

In dual decomposition our starting point is the split-variable model in (2), which is equivalent to

$$\begin{aligned} \eta^* := \min_{x \in \mathcal{X}} \min_{x_1, \ldots, x_S, y_1, \ldots, y_S} \quad & \sum_{s=1}^{S} p_s \Big(c^\top x_s + q_s^\top y_s \Big) && (11) \\ \text{s.t.} \quad & (x_s, y_s) \in \mathcal{F}_s && \forall s = 1, \ldots, S, \\ & x = x_s && \forall s = 1, \ldots, S. \end{aligned}$$

We apply Lagrangian relaxation to the non-anticipativity constraints $x = x_s$ for all $s = 1, \ldots, S$. That is, we relax these constraints but add a penalty to the objective function when these constraints are violated. For a given set of Lagrangian multipliers $\pi = (\pi_s)_{s=1,\ldots,S}$, the Lagrangian is given by

$$L(\pi) := \min_{x \in \mathcal{X}} \min_{\substack{x_1, \ldots, x_S \\ y_1, \ldots, y_S}} \left\{ \sum_{s=1}^{S} p_s \Big(c^\top x_s + q_s^\top y_s + \pi_s^\top x_s - \pi_s^\top x \Big) : (x_s, y_s) \in \mathcal{F}_s \right\}.$$

For every value of π, the Lagrangian $L(\pi)$ represents a lower bound on η^*, the optimal objective value of the original problem in (11). That is, $L(\pi) \leq \eta^*$ for all π. Moreover, we will show that the best possible lower bound yields $\bar{\eta}$.

The main reason why it may be advantageous to obtain $\bar{\eta}$ in this way is that for fixed π and for fixed $x \in \mathcal{X}$, the inner minimization problem in $L(\pi)$ is separable in the scenarios $s = 1, \ldots, S$, so that by defining

$$\mathcal{L}_s(\pi_s) := \min_{x_s, y_s} \Big\{ c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \mathcal{F}_s \Big\}, \qquad \forall s = 1, \ldots, S,$$

this Lagrangian lower bound equals

$$L(\pi) = \min_{x \in \mathcal{X}} \sum_{s=1}^{S} p_s \Big(\mathcal{L}_s(\pi_s) - \pi_s^\top x \Big).$$

The Lagrangian dual is the problem of finding the best possible Lagrangian lower bound by maximizing over the Lagrangian multipliers $\pi = (\pi_s)_{s=1,\ldots,S}$. That is, the Lagrangian dual is given by

$$\bar{\eta}_{DD} := \sup_{\pi} L(\pi) = \sup_{\pi_1, \ldots, \pi_S} \min_{x \in \mathcal{X}} \sum_{s=1}^{S} p_s \Big(\mathcal{L}_s(\pi_s) - \pi_s^\top x \Big). \tag{12}$$

Theorem 1. *The dual decomposition lower bound $\bar{\eta}_{DD}$ equals the lower bound $\bar{\eta}$ from (4). That is, $\bar{\eta}_{DD} = \bar{\eta}$.*

Proof. See Proposition 2 in Carøe and Schultz (1999). □

A common approach, see, e.g., Lubin et al. (2013), to compute the Lagrangian dual in (12) is to take $\mathcal{X} = \mathbb{R}^{n_1}$, so that the inner minimization problem is unbounded, and equal to $-\infty$ unless $\sum_{s=1}^{S} p_s \pi_s = 0$. Then, the problem can be rewritten as

$$\begin{aligned}\bar{\eta}_{DD} &= \sup_{\pi_1,\ldots,\pi_S} \left\{ \min_{x\in\mathcal{X}} \sum_{s=1}^{S} p_s\Big(\mathcal{L}_s(\pi_s) - \pi_s^\top x\Big) : \sum_{s=1}^{S} p_s\pi_s = 0 \right\} \\ &= \sup_{\pi_1,\ldots,\pi_S} \left\{ \sum_{s=1}^{S} p_s\mathcal{L}_s(\pi_s) : \sum_{s=1}^{S} p_s\pi_s = 0 \right\}. \end{aligned} \tag{13}$$

Here, the minimization problem over x disappears since its contribution to the objective, $\sum_{s=1}^{S} p_s\pi_s^\top x$, equals zero for any $x \in \mathbb{R}^{n_1}$ if π satisfies the constraint $\sum_{s=1}^{S} p_s\pi_s = 0$.

Using that $\mathcal{L}_s(\pi_s) = \min_{x_s,y_s}\{c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \mathcal{F}_s\}$, it follows directly that $\mathcal{L}_s(\pi_s)$ is the pointwise minimum over infinitely many affine functions, and thus is concave. Hence, the maximization problem in (13) can potentially be solved by any convex nonsmooth optimization algorithm, see, e.g., Kim and Zavala (2018), Kim et al. (2019), and Kim and Dandurand (2022). Moreover, since the objective in $\mathcal{L}_s(\pi_s)$ is linear in (x_s, y_s) for every π_s, it follows that at least one of the extreme points (x_s^k, y_s^k), $k \in K_s$, $s = 1,\ldots,S$, of $\mathrm{conv}(\mathcal{F}_s)$ is optimal. In other words,

$$\mathcal{L}_s(\pi_s) = \min_{k\in K_s} \left\{ c_s^k + \pi_s^\top x_s^k \right\}, \tag{14}$$

and thus we can rewrite the master problem corresponding to (13) as

$$\begin{aligned} \sup_{\pi,\theta} \quad & \sum_{s=1}^{S} p_s\theta_s \\ \text{s.t.} \quad & \theta_s \le c_s^k + \pi_s^\top x_s^k && \forall k \in K_s, \forall s = 1,\ldots,S \\ & \sum_{s=1}^{S} p_s\pi_s = 0. \end{aligned}$$

Instead of adding all supergradient inequalities directly, we may add them iteratively where we start with only a subset $\bar{K}_s \subseteq K_s$, $s = 1,\ldots,S$, of those. The restricted master problem becomes

$$\begin{aligned} \bar{\eta}_{RMP} = \sup_{\pi,\theta} \quad & \sum_{s=1}^{S} p_s\theta_s \\ \text{s.t.} \quad & \theta_s \le c_s^k + \pi_s^\top x_s^k && \forall k \in \bar{K}_s, \forall s = 1,\ldots,S && (p_s\lambda_s^k) \\ & \sum_{s=1}^{S} p_s\pi_s = 0, && && (x) \end{aligned}$$

with $p_s\lambda_s^k$ and x representing the dual variables corresponding to each constraint. Solving this restricted master problem yields current solutions $(\bar{\pi}_s, \bar{\theta}_s)$. Next, we check for optimality by solving the subproblems $\mathcal{L}_s(\bar{\pi}_s)$ for every $s = 1,\ldots,S$, yielding optimal extreme points $(x_s^{\bar{k}_s}, y_s^{\bar{k}_s})$, $s = 1,\ldots,S$. If $\bar{\theta}_s \le \mathcal{L}_s(\bar{\pi}_s)$

for every $s = 1, \ldots, S$, then we stop, since it holds that $\bar{\eta}_{RMP} = \bar{\eta}$. If not, then for every $s = 1, \ldots, S$, we append the index set $\bar{K}_s$ in the restricted master problem with $\bar{k}_s$ if $k_s \notin K_s$, and resolve the restricted master problem.

Remark 5. *The Dantzig-Wolfe decomposition from Section 3.3 and dual decomposition are theoretically equivalent in the sense that their restricted master problems are LP duals, and the DW pricing problems are equal to the DD subproblems.*

3.5. Lagrangian cuts

Lagrangian cuts are constructed using an approach that is very similar to dual decomposition. They are cuts of the form $\mathcal{L}_s(\pi_s) - \pi_s^\top x$, representing for every π_s a supporting hyperplane of the convex function $c^\top x + \mathrm{co}_X(v_s)(x)$, that can be used to construct optimality cuts in a primal decomposition algorithm. To derive these Lagrangian cuts, note that the inner minimization of the split-variable model in (11) is separable in the scenarios $s = 1, \ldots, S$, and can be written as

$$\eta^* = \min_{x \in X} \sum_{s=1}^{S} p_s \min_{x_s, y_s} \left\{ c^\top x_s + q_s^\top y_s : x = x_s,\ (x_s, y_s) \in \mathcal{F}_s \right\}.$$

Instead of applying Lagrangian relaxation to the non-anticipativity constraints in the joint minimization problem over x and (x_s, y_s), $s = 1, \ldots, S$, we can instead apply Lagrangian relaxation to the non-anticipitivity constraints $x = x_s$ in each scenario minimization problem over x_s and y_s for every $s = 1, \ldots, S$, separately. Replacing these minimization problems by their Lagrangian duals yields the lower bound

$$\begin{aligned} \bar{\eta}_{LC} &= \min_{x \in X} \sum_{s=1}^{S} p_s \sup_{\pi_s} \min_{x_s, y_s} \left\{ c^\top x_s + q_s^\top y_s + \pi_s^\top x_s - \pi_s^\top x : (x_s, y_s) \in \mathcal{F}_s \right\} \\ &= \min_{x \in X} \sum_{s=1}^{S} p_s \sup_{\pi_s} \left\{ \mathcal{L}_s(\pi_s) - \pi_s^\top x \right\}. \end{aligned} \tag{15}$$

Remark 6. *Even in the case that $X = \mathbb{R}^{n_1}$, the minimizer $\bar{x}$ in (15) will be contained in* $\mathrm{conv}(X)$*. If not, then the maximizations over π_s go to $+\infty$, because if $\bar{x} \notin \mathrm{conv}(X)$, then there exists a separating hyperplane, defined by its normal vector $\bar{\pi}$, such that $\bar{\pi}^\top \bar{x} > \bar{\pi}^\top x$ for all $x \in \mathrm{conv}(X)$. By scaling this normal vector $\bar{\pi}$ by α and letting $\alpha \to +\infty$, we obtain a sequence of π whose corresponding Lagrangian $\mathcal{L}_s(\pi)$ diverges to $+\infty$ for every $s = 1, \ldots, S$.*

For every π_s, the affine function $\mathcal{L}_s(\pi_s) - \pi_s^\top x$ may be interpreted as a supporting hyperplane of the function $c^\top x + \mathrm{co}_X(v_s)(x)$. Indeed, by definition, $\mathcal{L}_s(\pi_s)$ is the largest intercept of any affine function in x with slope $-\pi_s$ that lower bounds $c^\top x + \mathrm{co}_X(v_s)(x)$ on $\mathrm{conv}(X)$. Hence, $\sup_{\pi_s}\{\mathcal{L}_s(\pi_s) - \pi_s^\top x\}$ is a dual representation of $c^\top x + \mathrm{co}_X(v_s)(x)$, and thus it holds that $\bar{\eta}_{LC} = \bar{\eta}$. The next two results contain these observations. Their proofs can be found in the Appendix.

Theorem 2. *For every $x \in \mathrm{conv}(X)$ and $s = 1, \ldots, S$, it holds that*

$$c^\top x + co_X(v_s)(x) = \sup_{\pi_s} \left\{ \mathcal{L}_s(\pi_s) - \pi_s^\top x \right\}.$$

Corollary 1. *The lower bound $\bar{\eta}_{LC}$, obtained using Lagrangian cuts, equals the lower bound $\bar{\eta}$ from (4). That is, $\bar{\eta}_{LC} = \bar{\eta}$.*

Since the Lagrangian $\mathcal{L}_s(\pi_s)$ equals $\mathcal{L}_s(\pi_s) = \min_{k\in K_s}\{c_s^k + \pi_s^\top x_s^k\}$, see (14), it follows that $\mathcal{L}(\pi_s)$ is the pointwise minimum of finitely many affine functions, and thus concave polyhedral. This means that there exist finitely many points, denoted π_s^r, $r \in R_s$, so that $\sup_{\pi_s}\{\mathcal{L}_s(\pi_s) - \pi_s^\top x\} = \sup_{r\in R_s}\{\mathcal{L}_s(\pi_s^r) - (\pi_s^r)^\top x\}$ for all $x \in \text{conv}(X)$. If the hypograph $\text{Hypo}(\mathcal{L}_s) := \{(\pi_s, \eta_s) : \eta_s \le \mathcal{L}_s(\pi_s)\}$ of $\mathcal{L}_s$ is pointed, then $(\pi_s^r, \mathcal{L}_s(\pi_s^r))$, $r \in R_s$, correspond to the extreme points of $\text{Hypo}(\mathcal{L}_s)$ for every $s = 1, \dots, S$.

Remark 7. *If dim*$(\text{conv}(X)) = n_1$*, then* $\text{Hypo}(\mathcal{L}_s)$ *is pointed, i.e., then* $\text{Hypo}(\mathcal{L}_s)$ *has at least one extreme point.*

It follows that

$$\bar{\eta}_{LC} = \min_{x\in X} \sum_{s=1}^{S} p_s \sup_{r\in R_s} \left\{\mathcal{L}_s(\pi_s^r) - (\pi_s^r)^\top x\right\}.$$

A primal decomposition algorithm to obtain $\bar{\eta}_{LC}$ maintains a restricted master problem of the form

$$\begin{aligned}
\bar{\eta}_{RMP} := \min_{x,\theta_1,\dots,\theta_S} \quad & \sum_{s=1}^{S} p_s\theta_s & & \\
\text{s.t.} \quad & \theta_s \ge \left(\mathcal{L}_s(\pi_s^r) - (\pi_s^r)^\top x\right) \qquad \forall r \in \bar{R}_s,\ s = 1, \dots, S & & (p_s\mu_s^r) \\
& \bar{A}x \ge \bar{b}, & & (\nu)
\end{aligned}$$

where the system $\bar{A}x \ge \bar{b}$ represents $\text{conv}(X)$, and where $(\pi_s^r, \mathcal{L}_s(\pi_s^r))$, $r \in \bar{R}_s$, denote the currently available extreme points of $\text{Hypo}(\mathcal{L}_s)$.

Remark 8. *If* $\bar{A}x \ge \bar{b}$ *is initially not available, these cuts may be iteratively added throughout the decomposition algorithm using a cutting plane scheme.*

The solution of the restricted master problem yields the current solution $(\bar{x}, \bar{\theta}_1, \dots, \bar{\theta}_S)$. For every $s = 1, \dots, S$, we solve the subproblem $\sup_{\pi_s}\{\mathcal{L}_s(\pi_s) - \pi_s^\top \bar{x}\}$, yielding $\pi_s^{\bar{r}_s}$ for all $s = 1, \dots, S$. If $\bar{\theta}_s \ge \sup_{\pi_s}\{\mathcal{L}_s(\pi_s) - \pi_s^\top \bar{x}\}$, then we stop. It holds that $\bar{\eta}_{RMP} = \bar{\eta}$. Otherwise, for every $s = 1, \dots, S$, we append $\bar{r}_s$ to $\bar{R}_s$ in the restricted master problem if $\bar{\theta}_s < \mathcal{L}_s(\pi_s^{\bar{r}_s}) - (\pi_s^{\bar{r}_s})^\top \bar{x}$, and resolve the restricted master problem.

Remark 9. *It is often efficient to first determine weaker optimality cuts than Lagrangian cuts that can be computed faster. For example, strengthened Benders cuts (Zou et al., 2019) may be obtained for a given* $x \in \text{conv}(X)$ *and* $s = 1, \dots, S$*, by first solving the standard Benders cut based on the LP-relaxation of the second-stage subproblem* v_s*, to obtain its slope* $-\hat{\pi}_s$*, and then to compute* $\mathcal{L}_s(\hat{\pi}_s)$ *to derive the optimality cut* $\theta_s \ge \mathcal{L}_s(\hat{\pi}_s) - \hat{\pi}_s^\top x$*.*

For efficient computation of Lagrangian cuts and efficient decomposition schemes with Lagrangian cuts, we refer to Chen and Luedtke (2022) and Yang and Yang (2025). See also Deng and Xie (2024) for nonlinear, so-called ReLU Lagrangian cuts.

3.6. Dantzig-Wolfe Dual Decomposition (DW-DD)

In this section we present a Dantzig-Wolfe version of dual decomposition (DW-DD). To the best of our knowledge this approach has not been proposed yet in the SMIP literature. We present it here for completeness, since it is intricately linked to both dual decomposition and Lagrangian cuts, see also Section 3.7.

The starting point of DW-DD is the DD model from (13):

$$\sup_{\pi_1,\dots,\pi_S} \Big\{ \sum_{s=1}^{S} p_s \mathcal{L}_s(\pi_s) : \sum_{s=1}^{S} p_s \pi_s = 0 \Big\}.$$

Instead of maximizing over $\pi_s, s = 1,\dots,S$, however, we will maximize over all points (π_s, η_s), $s = 1,\dots,S$, in the hypographs of $\mathcal{L}_s$ with $\sum_{s=1}^{S} p_s \pi_s = 0$. That is,

$$\begin{aligned} \sup_{\pi,\eta} \quad & \sum_{s=1}^{S} p_s \eta_s \\ \text{s.t.} \quad & (\pi_s, \eta_s) \in \mathrm{Hypo}(\mathcal{L}_s) \qquad \forall s = 1,\dots,S \\ & \sum_{s=1}^{S} p_s \pi_s = 0. \end{aligned}$$

For every $s = 1,\dots,S$, the set $\mathrm{Hypo}(\mathcal{L}_s)$ is a polyhedron, since by (14),

$$\mathrm{Hypo}(\mathcal{L}_s) = \Big\{ (\pi_s, \eta_s) : \eta_s \le c_s^k + \pi_s^\top x_s^k \quad \forall k \in K_s \Big\}, \qquad s = 1,\dots,S.$$

By the Minkowski-Weyl theorem, there exists points (π_s^r, η_s^r), $r \in R_s$, with $\eta_s^r := \mathcal{L}_s(\pi_s^r)$, such that

$$\mathrm{Hypo}(\mathcal{L}_s) = \mathrm{conv}\Big\{ (\pi_s^r, \eta_s^r) : r \in R_s \Big\} + \mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s)),$$

where $\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$ represents the recession cone of $\mathrm{Hypo}(\mathcal{L}_s)$, i.e., the set of rays of $\mathrm{Hypo}(\mathcal{L}_s)$. Moreover, if $\mathrm{Hypo}(\mathcal{L}_s)$ is a pointed polyhedron, i.e., if $\mathrm{Hypo}(\mathcal{L}_s)$ has extreme points, then (π_s^r, η_s^r), $r \in R_s$, are the extreme points of $\mathrm{Hypo}(\mathcal{L}_s)$, see Remark 7.

It turns out that the recession cones $\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$ of $\mathrm{Hypo}(\mathcal{L}_s)$ are identical for every $s = 1,\dots,S$.

Proposition 3. *For every $s = 1,\dots,S$,*

$$\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s)) = \Big\{ (\pi_s, \eta_s) : \eta_s \le \pi_s^\top x \quad \forall x \in \mathrm{conv}(X) \Big\}.$$

Proof. See the Appendix. □

Since we may redefine $\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$, $s = 1,\dots,S$, in Proposition 3 using constraints corresponding to the extreme points of $\mathrm{conv}(X)$ only, it follws that $\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$ is a finitely generated cone. Its vectors $(\pi_s, \eta_s) \in \mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$, $s = 1,\dots,S$, correspond to coefficients of halfspaces that cover $\mathrm{conv}(X)$. Since we represent $\mathrm{conv}(X)$ by $\bar{A}x \ge \bar{b}$, the coefficients $(\bar{a}_i, \bar{b}_i)$ of each row of this system of inequalities are the extreme rays of $\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s))$ for every $s = 1,\dots,S$. That is,

$$\begin{pmatrix} \bar{A}^\top \\ \bar{b}^\top \end{pmatrix} \nu_s \in \mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s)) \qquad \Leftrightarrow \qquad \nu_s \ge 0.$$

Using this result it follows that $(\pi_s, \eta_s) \in \mathrm{Hypo}(\mathcal{L}_s)$ if and only if there exist $\mu_s^r \ge 0$, $r \in R_s$, with $\sum_{r \in R_s} \mu_s^r = 1$, and $\nu_s \ge 0$, such that

$$(\pi_s, \eta_s) = \sum_{r \in R_s} \mu_s^r (\pi_s^r, \mathcal{L}(\pi_s^r)) + \begin{pmatrix} \bar{A}^\top \\ \bar{b}^\top \end{pmatrix} \nu_s.$$

Thus, the full DW-DD master problem is given by

$$\bar{\eta} = \sup_{\mu,\nu\geq 0} \sum_{s=1}^{S} \sum_{r\in R_s} p_s \mu_s^r \mathcal{L}_s(\pi_s^r) + \nu^\top \bar{b} : \quad \sum_{r\in R_s} \mu_s^r = 1 \qquad \forall s = 1,\dots,S$$
$$\sum_{s=1}^{S} \sum_{r\in R_s} p_s \mu_s^r \pi_s^r + \bar{A}^\top \nu = 0,$$

where we use ν instead of the weighted average $\sum_{s=1}^{S} p_s \nu_s$ since all hypographs $\mathrm{Hypo}(\mathcal{L}_s)$, $s = 1,\dots,S$, are identical.

Since there may be many extreme points $(\pi_s^r, \mathcal{L}_s(\pi_s^r))$, $r \in R_s$, of $\mathrm{Hypo}(\mathcal{L}_s)$, we first include only a subset $\bar{R}_s \subseteq R_s$, $s = 1,\dots,S$, of these points in a restricted master problem, and add extreme points using column generation. The restricted master problem becomes

$$\bar{\eta}_{RMP} = \sup_{\mu,\nu\geq 0} \sum_{s=1}^{S} \sum_{r\in \bar{R}_s} p_s \mu_s^r \mathcal{L}_s(\pi_s^r) + \nu^\top \bar{b} : \quad \sum_{r\in \bar{R}_s} \mu_s^r = 1 \quad \forall s = 1,\dots,S \qquad (p_s\theta_s)$$
$$\sum_{s=1}^{S} \sum_{r\in \bar{R}_s} p_s \mu_s^r \pi_s^r + \bar{A}^\top \nu = 0. \qquad (x)$$

Let $\bar{x}$ and $p_s\bar{\theta}_s$ denote the optimal dual variables of the current restricted master problem. Then, the pricing problem to determine which variable has the largest positive reduced costs —we are maximizing instead of minimizing here— is for every $s = 1,\dots,S$, given by

$$\sup_{\pi_s} \left\{ \mathcal{L}_s(\pi_s) - \pi_s^\top \bar{x} - \bar{\theta}_s \right\}.$$

If none of these pricing problems yield positive reduced costs, then we may stop. It holds that $\bar{\eta}_{RMP} = \bar{\eta}$.

Remark 10. *The decomposition with Lagrangian cuts from Section 3.5 and DW-DD are theoretically equivalent in the sense that their restricted master problems are LP duals, and the DW-DD pricing problems are equal to the LC subproblems.*

3.7. The relationship between the decomposition algorithms

In this section we will discuss the relationship between the decomposition algorithms described in Sections 3.2–3.6. We classify affine parametric cutting planes, DW decomposition, and Lagrangian cuts as primal decomposition, and DD and DW-DD as dual decomposition approaches. We will argue that these primal and dual approaches can be interpreted as computing opposite sides of the same saddle point. Moreover, DW and DD are theoretically equivalent since the restricted DW and DD master problems are LP duals, and the DW pricing problems are identical to the DD subproblems, see Remark 5. A similar relationship holds between LC and DW-DD, see Remark 10.

One of the key elements that binds these decomposition approaches together are the concave polyhedral functions $\mathcal{L}_s(\pi_s)$, $s = 1,\dots,S$. They can be interpreted as the intercepts of the supporting hyperplanes $\mathcal{L}_s(\pi_s) - \pi_s^\top x$ of $c^\top x + \mathrm{co}_X(v_s)(x)$. Thus, from a primal perspective, see Theorem 2 for Langrangian cuts,

$$\bar{\eta} = \min_{x\in X} \sum_{s=1}^{S} p_s \left(c^\top x + \mathrm{co}_X(v_s)(x) \right) = \min_{x\in X} \sup_{\pi_1,\dots,\pi_S} \sum_{s=1}^{S} p_s \left(\mathcal{L}_s(\pi_s) - \pi_s^\top x \right),$$

whereas from a dual perspective, see (12),

$$\bar{\eta} = \sup_{\pi_1,\dots,\pi_S} \min_{x\in X} \sum_{s=1}^{S} p_s\Big(\mathcal{L}_s(\pi_s) - \pi_s^\top x\Big).$$

Hence, $\bar{\eta}$ can be interpreted as the objective value of the saddle point of the function $G(x,\pi) = \sum_{s=1}^{S} p_s(\mathcal{L}_s(\pi_s) - \pi_s^\top x)$, and LC and DD compute this value from opposite sides.

Further inspection shows that DW-DD can be interpreted as a Dantzig-Wolfe version of DD where we do not optimize

$$\bar{\eta}_{DD} = \sup_{\pi_1,\dots,\pi_S} \left\{ \sum_{s=1}^{S} p_s\mathcal{L}_s(\pi_s) : \sum_{s=1}^{S} p_s\pi_s = 0 \right\}$$

over π_s, but instead optimize over convex combinations of extreme points and extreme rays of the hypographs $\mathrm{Hypo}(\mathcal{L}_s)$ of $\mathcal{L}_s$. Similarly, DW can be interpreted as a Dantzig-Wolfe version of LC, where we do not optimize

$$\bar{\eta}_{LC} = \min_{x\in \mathrm{conv}(X)} \sum_{s=1}^{S} p_s\Big(c^\top x + \mathrm{co}_X(v_s)(x)\Big)$$

over x, but instead optimize over a convex combination of extreme points of the epigraphs of $c^\top x + \mathrm{co}_X(v_s)(x)$. Indeed, it turns out that any such extreme point in the epigraph corresponds to an extreme point of $\mathrm{conv}(\mathcal{F}_s)$. Another similarity between these decomposition algorithms is that both LC and DD are row generation algorithms and DW and DW-DD are column generation algorithms, where in each iteration of the algorithm a constraint or variable, respectively, is added to the restricted master problem. In Bertsekas and Yu (2011) such algorithms are classified as outer and inner approximations, respectively. See Figure 1 for a schematic overview of these relationships between DW, DD, LC, and DW-DD.

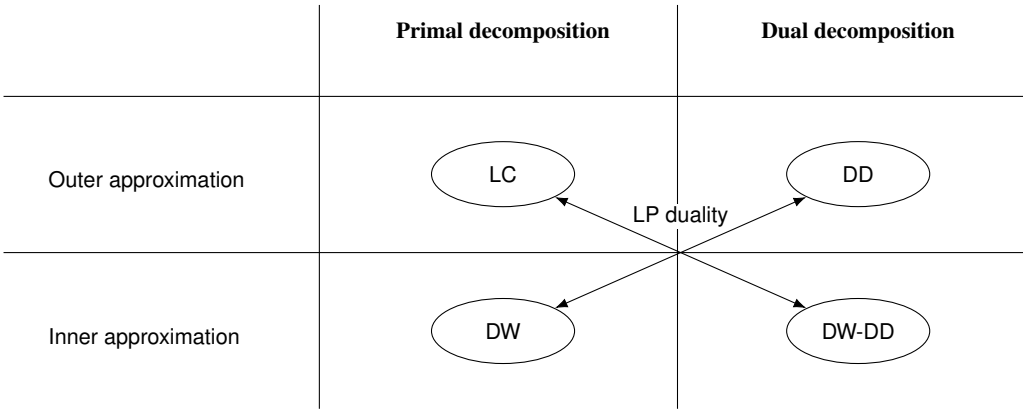

Figure 1: A schematic overview of the relationship between the decomposition methods DW, LC, DD, and DW-DD.

A surprising insight that we obtain from this overview in Figure 1 is that LC, although derived in very similar fashion, can be interpreted as a Dantzig-Wolfe decomposition version of DD. Which of the two approaches is more efficient will thus depend on the characteristics of the problem instance, but this dependence can be expected to be similar as when applying Dantzig-Wolfe decomposition to deterministic

optimization problems.

Affine parametric cutting planes stand out from the other decomposition approaches since they do not require solving any MIP subproblems. Indeed, all other methods need to determine the values of the Lagrangian functions $\mathcal{L}_s(\pi_s) := \min_{x_s,y_s}\left\{c^\top x_s + q_s^\top y_s + \pi_s^\top x_s : (x_s, y_s) \in \mathcal{F}_s\right\}$, $s = 1, \ldots, S$, in their subproblems, with integer restrictions on x_s and y_s, whereas in decomposition algorithms with affine parametric cutting planes, subproblems $\hat{v}_s^k(x) = \min_{y_s}\{q_s^\top y_s : (x, y_s) \in \hat{\mathcal{F}}_s^k\}$ are solved as LPs. The relationship with other primal decomposition methods, such as, e.g., LC, is as follows.

If all affine parametric cutting planes $\bar{T}_s x + \bar{W}_s y_s \geq \bar{h}_s$ would be present at the start of the decomposition algorithm, then any optimality cut $\theta_s \geq (\lambda_s^t)^\top(\bar{h}_s - \bar{T}_s x)$ generated by affine parametric cutting planes using an optimal vertex of the dual feasible region $\{\lambda_s \geq 0 : \lambda_s^\top \bar{W}_s \leq q_s^\top\}$ corresponds to an optimal extreme point $(\pi_s^t, \mathcal{L}_s(\pi_s^t))$ of Hypo$(\mathcal{L}_s)$ with

$$(\lambda_s^t)^\top \bar{h}_s = \mathcal{L}_s(\pi_s^t) \qquad \text{and} \qquad (\lambda_s^t)^\top \bar{T}_s = \pi_s^t.$$

In practice, however, affine parametric cutting planes are added iteratively, and thus the resulting optimality cuts are typically weaker than Lagrangian cuts. Moreover, these optimality cuts may be generated faster because one needs to only solve LP subproblems, rather than MIP subproblems.

4. B&B schemes, binary first-stage variables, and scaled cuts

In Section 3 we have presented several primal and dual decomposition approaches that yield the lower bound

$$\bar{\eta} = \min_{x \in \text{conv}(X)} \left\{c^\top x + \sum_{s=1}^{S} p_s \text{co}_X(v_s)(x)\right\} \tag{16}$$

of the optimal objective value η^*. In general, this lower bound $\bar{\eta}$ is not necessarily tight since

$$\eta^* = \min_{x \in \text{conv}(X)} \left\{c^\top x + \text{co}_X(Q)(x)\right\}, \tag{17}$$

and the inequality

$$\sum_{s=1}^{S} p_s \text{co}_X(v_s)(x) \leq \text{co}_X\Big(\sum_{s=1}^{S} p_s v_s\Big)(x) = \text{co}_X(Q)(x) \qquad \forall x \in \text{conv}(X), \tag{18}$$

is typically strict, see, e.g., Example 1 in van der Laan and Romeijnders (2024). Moreover, note that since we are minimizing in (16) over conv(X) to obtain $\bar{\eta}$, it is possible that we obtain a solution $\bar{x} \notin X$ that does not satisfy the first-stage integrality conditions. Below we discuss two approaches for strengthening this bound to η^*. The first is to explicitly enforce the first-stage integrality constraints, within a (spatial) branch-and-bound scheme if necessary, see Section 4.1, and the second is to use a decomposition approach with so-called scaled cuts, see Section 4.2.

4.1. Binary first-stage decisions and B&B schemes

A natural approach to strengthen the lower bound $\bar{\eta}$ is to explicitly enforce the first-stage integrality restrictions, if present. That is, to consider the lower bound

$$\hat{\bar{\eta}} = \min_{x \in X} \left\{ c^\top x + \sum_{s=1}^{S} p_s \mathrm{co}_X(v_s)(x) \right\} \tag{19}$$

In general, it holds that $\bar{\eta} \leq \hat{\bar{\eta}} \leq \eta^*$ and both inequalities may be strict. However, it turns out that the lower bound $\hat{\bar{\eta}}$ is tight in an important special case, namely when all first-stage variables are pure binary. This is true since when $X \subseteq \mathbb{B}^{n_1}$, it holds that $\mathrm{co}_X(v_s)(x) = v_s(x)$ for all $x \in X$, and thus $\hat{\bar{\eta}}$ reduces to

$$\hat{\bar{\eta}} = \min_{x \in X} \left\{ c^\top x + \sum_{s=1}^{S} p_s v_s(x) \right\} = \eta^*.$$

This result is a direct consequence of the following geometrical insight. Let $\bar{x} \in X$ denote an extreme point of $\mathrm{conv}(X)$. Then,

$$F_s(\bar{x}) = \left\{ (x, y_s) \in \mathrm{conv}(\mathcal{F}_s) : x = \bar{x} \right\},$$

represents a face of $\mathrm{conv}(\mathcal{F}_s)$, satisfying $(\bar{x}, y_s) \in F_s(\bar{x})$ if and only if $y_s \in \mathrm{conv}(\mathcal{Y}_s(\bar{x}))$ with $\mathcal{Y}_s(\bar{x})$ denoting the second-stage feasible region as defined in (6). Hence,

$$\mathrm{co}_X(v_s)(\bar{x}) = \min_{y_s} \left\{ q_s^\top y_s : (\bar{x}, y_s) \in \mathrm{conv}(\mathcal{F}_s) \right\} = \min_{y_s} \left\{ q_s^\top y_s : y_s \in \mathrm{conv}(\mathcal{Y}_s(\bar{x})) \right\} = v_s(\bar{x}).$$

Of course, if $X \subseteq \mathbb{B}^{n_1}$, then any binary point $\bar{x} \in X$ is an extreme point of $\mathrm{conv}(X)$.

The following results summarize these observations.

Proposition 4. *If $\bar{x} \in X$ is an extreme point of* $\mathrm{conv}(X)$*, then $v_s(\bar{x}) = \mathrm{co}_x(v_s)(\bar{x})$ for all $s = 1, \ldots, S$.*

Proof. See, e.g., Sherali and Zhu (2006). □

Theorem 3. *If $\bar{x}$ is an extreme point of* $\mathrm{conv}(X)$ *for all $\bar{x} \in X$, then $\hat{\bar{\eta}} = \eta^*$ and $\hat{\bar{x}} = x^*$, where $\hat{\bar{x}}$ is an optimal solution of (19).*

Proof. Follows directly from Proposition 4. □

It follows from Theorem 3 that if all first-stage variables are binary, that then it suffices to solve the optimization problem in (19). In general, this can be done by embedding the decomposition approaches from Sections 3.2–3.6 in a branch-and-bound scheme, where $\bar{\eta}$ provides the lower bound in any node of the B&B tree, and branching can be done on first-stage components x_i that need to be binary but currently have a fractional solution $\bar{x}_i$.

Remark 11. *This B&B scheme requires running a separate decomposition within each node of the B&B tree. However, some decomposition methods from Section 3 can be easily adapted to obtain $\hat{\bar{\eta}}$ using only a single decomposition. In particular, for affine parametric cutting planes and Lagrangian cuts we may simply enforce the integrality restrictions $x \in X$ for each restricted master problem and solve each restricted master problem as a MIP. Alternatively, we may solve the problem using a single B&B tree in which optimality cuts are iteratively added at solutions encountered within the B&B solve.*

For two-stage SMIP that do not satisfy the assumption of Theorem 3, e.g., for problems having continuous first-stage variables, however, it does not necessarily suffice to solve (19). In this case, we may strengthen the lower bound $\hat{\bar{\eta}}$ in (19) as follows. First of all, we may binarize pure integer and even continuous variables, see Zou et al. (2019). Alternatively, we may partition the first-stage feasible region X into regions $X_k \subseteq X$, $k = 1, \ldots, K$, with $\cup_{k=1}^{K} X_k = X$, for example by (spatial) branching on the first-stage decision variables, so that the X_k correspond to integer feasible regions in nodes of a B&B tree. Then,

$$\hat{\bar{\eta}}_K := \min_{k=1,\ldots,K} \min_{x \in X_k} \left\{ c^\top x + \sum_{s=1}^{S} p_s \mathrm{co}_{X_k}(v_s)(x) \right\}$$

is a strengthened version of $\hat{\bar{\eta}}$, i.e., $\hat{\bar{\eta}}_K \geq \hat{\bar{\eta}}$, since for every $k = 1, \ldots, K$, $X_k \subseteq X$, so that $\mathrm{co}_{X_k}(v_s)(x) \geq \mathrm{co}_X(v_s)(x)$ for all $x \in X_k$. Hence, by iteratively further partitioning, i.e., branching, we may obtain (approximate) optimal solutions.

Remark 12. *Both binarization and branching can be considered as a special type of lifting, which is another way in which the lower bound $\bar{\eta}$ in (16) may be improved. See Sherali and Zhu (2006) for more details on lifting in two-stage SMIPs.*

4.2. Scaled cuts

A primal decomposition method that is able to derive a stronger bound than $\bar{\eta}$ in (4) is the decomposition method using scaled cuts from van der Laan and Romeijnders (2024). These scaled cuts are affine optimality cuts for the expected value function Q, and in the limit are able to recover $\mathrm{co}_X(Q)$. Hence, from (17) it follows that scaled cuts are able to derive an arbitrarily tight lower bound on the optimal objective value η^*.

A key characteristic of scaled cuts is that the scaled cut decomposition algorithm maintains a master problem with affine optimality cuts for Q. No nonlinear cuts or additional binary variables are added. The strength of these affine cuts comes from the fact that nonlinear cuts for the second-stage value functions v_s, $s = 1, \ldots, S$, are translated to affine optimality cuts for Q. This works as follows. Let Q_{out} denote a convex polyhedral outer approximation of Q, and suppose that for every $s = 1, \ldots, S$, there exist cut coefficients $(\alpha_s, \beta_s, \tau_s)$, $s = 1, \ldots, S$, with $\tau_s \geq 0$, such that

$$v_s(x) \geq \alpha_s - \beta_s^\top x - \tau_s Q_{\mathrm{out}}(x) \qquad \forall x \in X, \tag{20}$$

that is, the right-hand side in (20 is a nonlinear lower bound for v_s. Then, it follows directly that

$$Q(x) \geq \alpha - \beta^\top x - \tau Q_{\mathrm{out}}(x) \qquad \forall x \in X, \tag{21}$$

with $\alpha := \sum_{s=1}^{S} p_s \alpha_s$, $\beta := \sum_{s=1}^{S} p_s \beta_s$, and $\tau := \sum_{s=1}^{S} p_s \tau_s$, is a are nonlinear lower bound for Q as well. We do not add this nonlinear optimality cut to the master problem but instead derive a scaled cut using that

$$(1 + \tau) Q(x) \geq Q(x) + \tau Q_{\mathrm{out}}(x) \geq \alpha - \beta^\top(x) \qquad \forall x \in X,$$

where the first inequality holds since Q_{out} is a lower bound of Q, and the second inequality holds by (21).

Dividing by $1+\tau$, we obtain the scaled cut

$$Q(x) \geq \frac{\alpha - \beta^\top x}{1+\tau} \qquad \forall x \in \text{conv}(X).$$

Remark 13. *We note that we can derive scaled cuts in this fashion since the nonlinear lower bounds on v_s have a very particular form. The only nonlinear part of the lower bound is defined by the current outer approximation Q_{out} of Q.*

Conceptually, we may use a current outer approximation Q_{out} of Q to derive many scaled cuts for Q. We define the scaled cut closure $\text{SCC}(Q_{\text{out}})$ of Q_{out} with respect to Q as the pointwise maximum of all scaled cuts that can be generated using Q_{out}. That is,

$$\text{SCC}(Q_{\text{out}}) := \sup_{(\alpha_s,\beta_s,\tau_s)_{s=1}^S} \left\{ \frac{\sum_{s=1}^S p_s\alpha_s - \sum_{s=1}^S p_s\beta_s^\top x}{1+\sum_{s=1}^S p_s\tau_s} : (\alpha_s,\beta_s,\tau_s) \in \Pi_s(Q_{\text{out}}) \ \ \forall s \right\},$$

where $\Pi_s(Q_{\text{out}})$ denotes the set of feasible cut coefficients $(\alpha_s,\beta_s,\tau_s)$, defined for every $s=1,\dots,S$ as

$$\Pi_s(Q_{\text{out}}) = \left\{(\alpha_s,\beta_s,\tau_s) : v_s(x) \geq \alpha_s - \beta_s^\top x - \tau_s Q_{\text{out}}(x) \quad \forall x \in X, \ \tau_s \geq 0\right\}.$$

Starting with any initial convex polyhedral outer approximation Q_{out}^1, iteratively applying the scaled cut closure operation yields a sequence of outer approximations that converges uniformly to $\text{co}_X(Q)$, the best possible lower bound of Q.

Theorem 4. *Let Q_{out}^1 denote a convex polyhedral outer approximation of the expected value function Q, and consider the sequence $\{Q_{out}^k\}_{k=1}^\infty$, defined as $Q_{out}^{k+1} := SCC(Q_{out}^k)$ for all $k \in \mathbb{N}$. Then, $\{Q_{out}^k\}_{k=1}^\infty$ converges uniformly to $\text{co}_X(Q)$.*

Proof. See van der Laan and Romeijnders (2024). □

In practice, we do not intend to compute all scaled cuts from the scaled cut closure but instead we iteratively add scaled cuts one at a time. In particular, for a given current solution $\bar{x} \in \text{conv}(X)$ of the current master problem and given the current outer approximation Q_{out} of Q, we add a dominating scaled cut using cut coefficients $(\alpha_s,\beta_s,\tau_s)$, $s=1,\dots,S$, maximizing the scaled cut at $\bar{x}$, that is, maximizing

$$\sup_{(\alpha_s,\beta_s,\tau_s)_{s=1}^S} \left\{ \frac{\sum_{s=1}^S p_s\alpha_s - \sum_{s=1}^S p_s\beta_s^\top \bar{x}}{1+\sum_{s=1}^S p_s\tau_s} : (\alpha_s,\beta_s,\tau_s) \in \Pi_s(Q_{\text{out}}) \ \ \forall s \right\}.$$

Dominating scaled cuts may be determined by a fixed-point iteration algorithm in which scenario subproblems can be solved in (van der Laan and Romeijnders, 2024). See also Romeijnders and van der Laan (2024) for scaled cuts in a multistage setting.

Remark 14. *Observe that if we restrict $\tau_s = 0$ for all $s=1,\dots,S$, then for every $s=1,\dots,S$, we optimize $\alpha_s - \beta_s^\top \bar{x}$ such that $\alpha_s - \beta_s^\top x$ is an affine lower bound for v_s. That is, we derive Lagrangian cuts. By allowing*

$\tau_s > 0$ for some $s = 1, \ldots, S$, we may expect scaled cuts to yield stronger cuts than Lagrangian cuts. We do note that this comes at a higher computational cost, see van der Laan and Romeijnders (2024).

5. Definition of Multistage stochastic mixed-integer programming (MSMIP)

Multistage stochastic programming addresses decision/optimization problems in which decisions must be made sequentially over time as uncertainty unfolds. This uncertainty is modeled as a stochastic process, which is assumed to be stagewise independent in most cases.[2] A fundamental principle in these problems is the notion of **non-anticipativity**, which ensures that decisions made at any given stage can only depend on information available up to that stage, and we cannot use knowledge of future outcomes of the random process.

The structure of uncertainty and decision epochs in a multistage problem is commonly represented by a **scenario tree**, which is a rooted graph denoted by $\mathcal{T} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is a set of **nodes** where each node represents a specific state of the world at a particular stage (point in time). There is a unique **root node**, denoted as $n = 1$, representing a given initial state at stage 0. The set $\mathcal{E}$ is the set of edges connecting nodes between consecutive stages. The tree structure implies that every node $n \in \mathcal{N} \setminus \{1\}$ has a unique **ancestor** (parent) node, denoted by $a(n)$. We use the mapping $a : \mathcal{N} \setminus \{1\} \to \mathcal{N}$ to indicate this relationship. The root node $n = 1$ has no predecessor within the tree. Nodes can be grouped into **stages** $t = 0, 1, ..., T$. The root node is at stage $t = 0$. A node n is at stage t if the path from the root to n has t edges. Each node $n \in \mathcal{N}$ is associated with a **probability** p_n, representing the probability that the specific sequence of events leading from the root node to node n occurs. Therefore, $p_1 = 1$, and for any node $n \neq 1$, p_n is the product of conditional probabilities along the path from the root to n (because of stagewise independence). The sum of probabilities of all nodes at any given stage t is 1.

Associated with each node $n \in \mathcal{N}$ is a **state variable** vector x_n. This vector is used to record the relevant information about the system's state when node n is reached. Also in each node $n \in \mathcal{N}$ we have a **control variable** (or local decision variable) vector u_n. This represents the action or decision taken by the decision-maker at node n, having observed the history leading to node n.

Remark 15. *The key distinction between state variables and control variables is that the state vector x_n encapsulates all the information needed by subsequent stages, while the influence of the control variable u_n on the future is only carried forward through its effect on the state variable x_n.*

To maintain consistent notation in the formulation, it is often convenient to introduce a conceptual "dummy" node 0 such that $a(1) = 0$. This node represents the point before the decision process begins, and x_0 is a deterministic initial state vector of the system before any decisions are made or uncertainty is revealed.

The extensive form (all-in-one form) of MSMIP lists the objective and constraints for all nodes in the scenario tree into a large-scale finite-dimensional mathematical program:

[2] It is possible to model dependencies in the stochastic process. Scenario-based modeling (Sections 5 and 6) can accommodate arbitrary dependency structures, and the recursive dynamic programming formulation (Section 7) can handle Markovian dependency.

$$\min_{x,u} \sum_{n\in\mathcal{N}} p_n(c_n^\top x_n + d_n^\top u_n) \tag{22}$$

$$\text{subject to } x_n = A_n x_{a(n)} + B_n u_n + e_n \qquad \forall n \in \mathcal{N} \tag{23}$$

$$D_n u_n \le h_n - C_n x_{a(n)} \qquad \forall n \in \mathcal{N} \tag{24}$$

$$(x_n, u_n) \in Z_n \qquad \forall n \in \mathcal{N}. \tag{25}$$

Constraint (23) defines the **system dynamics**, linking the state at a node to the state of its parent and the local control action. Constraint (24) defines feasible controls. The set Z_n in (25) defines the feasible region for the state and decision variables at node n. This may include bounds (e.g., $x_n \ge 0$, $\underline{u}_n \le u_n \le \overline{u}_n$). For an MSMIP, Z_n also enforces integrality constraints on some components of x_n and u_n. For example, Z_n might be a subset of $\mathbb{R}^{d_x^r} \times \mathbb{Z}^{d_x^z} \times \mathbb{R}^{d_u^r} \times \mathbb{Z}^{d_u^z}$. This presence of integer variables is what distinguishes MSMIPs from multistage stochastic linear programs (MSLP).

The notion of non-anticipativity is *implicitly* enforced by this formulation. Because there is only a single decision variable vector u_n associated with each node n, the adopted decision depends on having reached node n (i.e., based on the history represented by the path from the root to n). It cannot depend on uncertain outcomes revealed after node n.

Remark 16. *The extensive form formulation is correct only when $(\mathcal{N}, \mathcal{E})$ forms a tree structure. If the paths are allowed to merge, a recursive formulation must be used, as described in Section 7.1. This is unique to multistage problems.*

For algorithmic discussions, it is convenient to group constraints (23)–(24) into the following general structure which is similar to the two-stage case:

$$W_n x_n + T_n x_{a(n)} + V_n u_n \ge h_n. \tag{26}$$

Notationwise, we encapsulate all of a node's constraints within a single set, X_n. This set contains all feasible triplets of an input state variable (z_n), a control variable (u_n), and a resulting state variable (x_n), defined as follows:

$$X_n := \{(z_n, u_n, x_n) \mid W_n x_n + T_n z_n + V_n u_n \ge h_n,\ (x_n, u_n) \in Z_n\}. \tag{27}$$

In this definition, z_n can be viewed as a "placeholder" for the state $x_{a(n)}$ passed from the parent node.

6. Dual Decomposition (Scenario decomposition) for MSMIP

In the multistage setup, dual decomposition works by unrolling the scenario tree and explicitly writing out the non-anticipativity constraints. Because it decomposes the problem into individual scenarios, it is also called scenario decomposition. Formally, we define a scenario s as a complete path from the root node to a leaf node in the scenario tree. Let $\mathcal{S}$ be the set of all such scenarios. Each scenario $s \in \mathcal{S}$ occurs with probability p_s, representing the probability of the sequence of random outcomes defining that path. (If the scenario tree has a uniform depth T, each scenario s corresponds uniquely to a leaf node l, and $p_s = p_l$, the

probability of reaching node l.) To link scenarios back to the original tree structure, we define $n(s,t)$ as the unique node in $\mathcal{N}$ that lies on scenario path s at stage t, for $t = 0, 1, ..., T$.

We introduce scenario-specific state variables $x_{s,t}$ and control variables $u_{s,t}$ for each scenario $s \in \mathcal{S}$ and stage $t = 0 \dots T$. This sequence of variables must satisfy the initial condition $x_{s,0} = \bar{x}_0$, and furthermore, the system dynamics and local constraints must be satisfied at each step along the path s. These step-wise constraints are defined by the sets $X_{n(s,t)}$ (from (27)). The set containing all such feasible sequences for scenario s is the scenario polyhedron $\mathcal{F}_s$:

$$\mathcal{F}_s := \Big\{(x_{s,0}, u_{s,0}, \dots, u_{s,T-1}, x_{s,T}, u_{s,T}) \;\Big|\; x_{s,0} = \bar{x}_0, \\ (x_{s,t-1}, u_{s,t}, x_{s,t}) \in X_{n(s,t)} \quad \forall t = 1, \dots, T\Big\} \tag{28}$$

Under this transformation the extensive form (22) is equivalent to the following split variable model:

$$\begin{aligned} \min \sum_s p_s \sum_{t=0}^{T} c_{s,t} x_{s,t} + d_{s,t} u_{s,t} \\ (x_{s,0}, u_{s,0}, \dots, x_{s,T}) \in \mathcal{F}_s \quad \forall s \in \mathcal{S} \\ x_{n(s,t)} = x_{s,t} \qquad u_{n(s,t)} = u_{s,t} \quad \forall s \in \mathcal{S}, t = 0 \dots T \end{aligned} \tag{29}$$

The non-anticipativity constraints link the scenario-specific variables ($x_{s,t}$, $u_{s,t}$) to the corresponding bundle variables ($x_{n(s,t)}$, $u_{n(s,t)}$). If multiple scenarios pass through the same node n at stage t, they are forced to share the same state x_n and control u_n via these constraints. This formulation is the multistage analogue to the two-stage non-anticipativity constraint (e.g., $x_s = x$ for the first-stage variables) shown in (11).

Once the multistage problem is formulated using scenario-specific variables and coupling constraints (as in (29), the "split-variable model"), the dual decomposition techniques in two-stage SMIP readily extends to the multistage case. For details, see Section 3.4.

While the fundamental decomposition strategies extend from the two-stage case, the computational challenges in multistage stochastic programming are significantly amplified. The main issue is the exponential growth in the number of scenarios relative to the number of stages. The scale of the problem is typically significantly larger compared to typical two-stage instances.

1. Different algorithms exist for solving the Lagrangian dual problem (or its equivalent convexified formulation) that arises from the decomposition. For instance, the Branch-and-Price method in Lulli and Sen (2004) uses a column generation procedure for the master problem. In contrast, DDSIP (Carøe and Schultz, 1999) originally proposed subgradient methods, but there are variants that use bundle methods. Solving these problems can become a bottleneck, particularly in large-scale multistage setting.

2. The feasible set $\mathcal{F}_s$ for each scenario represents the feasible set of a deterministic mixed-integer program. For many applications, such as the unit commitment problem, strong formulations for these underlying deterministic problems are well-studied. Using these tighter formulations directly into the definition or handling of $\mathcal{F}_s$ within the decomposition algorithm leads to stronger relaxations.

Alternatively, the split-variable formulation also serves as a basis for methods using Progressive Hedging (PH). PH iteratively penalizes deviations from the bundle variables. For convex problems, it takes far less iterations than subgradient based methods because of its linear convergence. While standard Progressive Hedging (PH) lacks convergence guarantees for SMIPs, it can be embedded within a Branch-and-Bound (B&B) framework to obtain optimality, as demonstrated by the PH-BAB algorithm developed by Atakan and Sen (2018). This B&B procedure requires lower bounds for pruning. PH-BAB obtains these by applying PH to the convex nodal relaxations and then using the resulting dual price information to compute lower bounds as presented by Gade et al. (2016). Upper bounds are derived from integer feasible solutions found during the search. Furthermore, there are refinements that combine PH and Dual Decomposition. Guo et al. (2015) propose using PH to obtain reasonable dual multipliers as the starting point for DDSIP which demonstrates potential speedups.

7. Primal decomposition (Resource-directive Decomposition) for MSMIP

If the randomness is stagewise independent, or at least Markovian (i.e., future uncertainty depends only on the uncertainty at the current stage but not the path taken to reach it) and the state dynamics only depend on the current action and the previous state variable, the MSMIP can be posed in a recursive form, similar in spirit to dynamic programming.

As an overview, first we will introduce an equivalent, recursive description of MSMIP, in a form that is suitable for primal decomposition algorithms. Then we will introduce the framework to describe these SDDP-like algorithms, which involves incrementally building lower approximations of the value function Q_n. Then we will use this framework to describe the SDDiP algorithm, which uses Lagrangian cuts (the same technique as in the two-stage case in Section 3.4), and its convergence sketch when the state variables x_n are all restricted to be pure binary. This case is analogous to the two-stage case when the first stage is purely binary, see Section 4.1.

Because SDDiP builds its approximation from linear cuts, the approximation is always convex. At best, it converges to the convex envelope of the true value function. This reliance on convex approximation can cause SDDiP to stall when state variables include general integers or are mixed-binary. Therefore, we discuss strategies designed to overcome this limitation.

(i) **State Variable Binarization / Lifting:** Approximating general integer or continuous state variables with binary expansions, as originally proposed within SDDiP by Zou et al. (2019).

(ii) **Mixed-Integer Representable Cuts:** Use stronger, non-convex under-approximations derived from optimality conditions or relaxations that better capture the non-convexity, such as reverse norm cuts[3] (Ahmed et al., 2022) or ReLU cuts (Deng and Xie, 2024).

7.1. Recursive Formulation

We start by defining a set of nodes $\mathcal{N}$ on a scenario tree. For $n \in \mathcal{N}$, let $\mathcal{S}(n)$ denote the set of immediate successors (children) of node n. Note $\mathcal{S}(n) = \emptyset$ if n is a terminal stage node. The **value function** Q_n at node n, representing the minimum expected cost from stage $t(n)$ onwards given the state $x_{a(n)}$ at its predecessor $a(n)$, is defined recursively:

[3]Following the cited literature, these non-convex under-approximations are also referred to as nonlinear cuts. They are called "cuts" although they are not necessarily affine.

$$
\begin{aligned}
Q_n(x_{a(n)}) := \min_{x_n, u_n} & \left\{ c_n^\top x_n + d_n^\top u_n + \sum_{m \in \mathcal{S}(n)} q_{nm} Q_m(x_n) \right\} \\
& \text{subject to} \\
& (x_{a(n)}, u_n, x_n) \in X_n.
\end{aligned}
\tag{30}
$$

Here q_{nm} is the conditional probability of transitioning from node n to node m, given that node n has been reached. If $(\mathcal{N}, \mathcal{E})$ is a scenario tree, then this conditional probability is $q_{nm} = p_m / p_n$ if $m \in \mathcal{S}(n)$ (where p_n and p_m are the unconditional probabilities of reaching nodes n and m from the root node, respectively), and $q_{nm} = 0$, otherwise. The **expected cost-to-go function** at node n is defined as $\mathcal{Q}_n(x_n) := \sum_{m \in \mathcal{S}(n)} q_{nm} Q_m(x_n)$.

It is important to view the value function $Q_n(x_{a(n)})$ as providing the optimal expected cost from node n and onwards, given an input state $x_{a(n)}$ obtained from the previous stage, where, $x_{a(n)}$ serves as a parameter for this input state. Interpreted in this way, the recursive structure (minimizing immediate cost plus the expected future cost $\mathcal{Q}_n(x_n)$ based on successor nodes) shares the same spirit as Approximate Dynamic Programming (Powell, 2011). It can be defined more generally on structures like **scenario lattices** (also called **recombining scenario trees**) which are directed acyclic graphs where paths merge, or compactly represent stagewise independent stochastic processes. Such a view of the problem can be very powerful because the scenario lattice structure allows re-using approximation on different branches, potentially allowing us to work on problems that have too many scenarios even to pose within scenario decomposition. However, because our primary focus here is on techniques for dealing with methodology of approximating Q_m with integrality restrictions, we do not dwell further into such modeling choices. For a more rigorous treatment of multistage modeling, we refer the reader to Pflug and Pichler (2014). For perspectives on the connection to Dynamic Programming, see the literature on Approximate Dynamic Programming (ADP) by Powell (2011) and related work on non-tree structures such as policy graphs (Dowson, 2020). Methods for modeling Markovian dependencies include using a recursive formulation with scenario lattices (Philpott and De Matos, 2012) or expanding state variables (Löhndorf and Shapiro, 2019). Additionally, Chapter 14 of Füllner and Rebennack (2025) provides a comprehensive review of dependency modeling in the continuous SDDP setup, featuring techniques that are also applicable here.

Using the set X_n defined in (27), the value function $Q_n(x_{a(n)})$ can be rewritten by explicitly adding a constraint that equates the input state placeholder z_n (from the definition of X_n) to the actual state $x_{a(n)}$ passed from the parent node:

$$
\begin{aligned}
Q_n(x_{a(n)}) = \min_{z_n, u_n, x_n} & \left\{ c_n^\top x_n + d_n^\top u_n + \sum_{m \in \mathcal{S}(n)} q_{nm} Q_m(x_n) \right\} \\
& \text{subject to} \\
& (z_n, u_n, x_n) \in X_n \\
& z_n = x_{a(n)}.
\end{aligned}
\tag{31}
$$

Here, the optimization is performed over the variables (z_n, u_n, x_n). The dual variables associated with the added constraint $z_n = x_{a(n)}$ will be particularly important in such formulations.

7.2. SDDP-like algorithms

Table 1: Summary of Notations

Notation	Meaning	Definition / Context
$Q_n(x_{a(n)})$	True value function at node n.	Eq. (30), Eq. (31) (Recursive Formulation)
$\mathcal{Q}_n(x_n)$	True expected cost-to-go function at node n.	$\sum_{m\in\mathcal{S}(n)} q_{nm} Q_m(x_n)$
$\underline{Q}_n(x_{a(n)})$(also $\underline{Q}_n^k$)	Lower approximation of the true value function $Q_n(x_{a(n)})$.	Updated via $g_n^k(\cdot)$ in Eq. (33) (Backward Pass).
$\underline{\mathcal{Q}}_n(x_n)$(also $\underline{\mathcal{Q}}_n^{k-1}$)	Approximation of the true expected cost-to-go function $\mathcal{Q}_n(x_n)$. Aggregates approximations $\underline{Q}_m$ from successor nodes $m \in \mathcal{S}(n)$. Used in the forward pass problem.	Defined as $\sum_{m\in\mathcal{S}(n)} q_{nm}\underline{Q}_m(x_n)$ (using $\underline{Q}_m^{k-1}$ for $\underline{\mathcal{Q}}_n^{k-1}$).
$\hat{Q}_n(\bar{x}_{a(n)}^k; \underline{\mathcal{Q}}_n^{k-1})$	Optimal objective value of the forward problem P_n (Eq. (32)) solved in iteration k with input state $\bar{x}_{a(n)}^k$ and using the future cost approximation $\underline{\mathcal{Q}}_n^{k-1}$.	Defined by Eq. (32) (Forward Pass, Step 4).

For multistage stochastic mixed-integer programs (MSMIPs) posed in the form of (31), methods like Stochastic Dual Dynamic integer Programming (SDDiP) (Zou et al., 2019) and Stochastic Lipschitz Dynamic Programming (SLDP) (Ahmed et al., 2022) extend the ideas originally developed for linear problems in the Stochastic Dual Dynamic Programming (SDDP) algorithm by Pereira and Pinto (1991); see Füllner and Rebennack (2025) for a recent review. We will refer to these extensions collectively as SDDP-like algorithms for MSMIPs.

These algorithms share the iterative structure of SDDP. At each iteration, the algorithm runs forward and backward passes along selected paths on the scenario tree or lattices. A key element is that the algorithm incrementally builds approximations, denoted by $\underline{Q}_n(x_{a(n)})$, for the true value function $Q_n(x_{a(n)})$ at each node n. Consequently, its aggregation $\underline{\mathcal{Q}}_n(x_n) := \sum_{m\in\mathcal{S}(n)} q_{nm}\underline{Q}_m(x_n)$ is an approximation for the true cost-to-go function $\mathcal{Q}_n(x_n)$.

This section outlines a general iterative decomposition framework for resource-directive decomposition under multistage settings. These SDDP-like algorithms usually share the following key features:

1. The algorithm maintains a lower approximation $\underline{Q}_n^k(\cdot)$ to the value function $Q_n(\cdot)$ for $n \in \mathcal{N}$.

2. **Initialization** Set iteration counter $k = 1$. Initialize the approximations $\underline{Q}_n^0(\cdot)$ for all nodes $n \in \mathcal{N}$. A common choice is $\underline{Q}_n^0(\cdot) = L_n$, where L_n is a simple, known lower bound (e.g., 0 if costs are non-negative) on the value function at node n.

3. **Path Selection** Sample (or select) one (or more) paths from the root to a leaf node from the scenario tree/lattice $\mathcal{T}$. Let $\omega^k = (n_0, n_1, \ldots, n_T)$ where $n_0 = 1$ is the root node, and n_t is the node visited at stage t along this specific path in iteration k. (Note: In SDDiP under stagewise independence, sampling involves selecting realizations ξ_t^k for each stage t, which is equivalent to defining paths).

4. **Forward Pass** Traverse the sampled path(s) forward from stage $t = 0$ to T. At each node $n \in \omega^k$ (let $n = n_t$), the aggregated function $\underline{\mathcal{Q}}_n^{k-1}(x_n) := \sum_{m\in\mathcal{S}(n)} q_{nm}\underline{Q}_m^{k-1}(x_n)$. Solve the following forward

problem $P_n(x_{a(n)}; \underline{Q}_n^{k-1})$, which involves replacing the expected cost-to-go function $Q_n(\cdot)$ with its lower approximation $\underline{Q}_n^{k-1}(\cdot)$, and using the state $\bar{x}_{a(n)}^k$ computed at its parent node $a(n)$ in the current iteration's forward pass (with $\bar{x}_{a(1)}^k = x_0$, the initial state):

$$\begin{aligned} P_n(\bar{x}_{a(n)}^k; \underline{Q}_n^{k-1}) : \hat{Q}_n(\bar{x}_{a(n)}^k; \underline{Q}_n^{k-1}) := \min_{z_n, u_n, x_n, \theta_n} & \left\{ c_n^\top x_n + d_n^\top u_n + \theta_n \right\} \\ & \text{subject to} \\ & (z_n, u_n, x_n) \in X_n \\ & z_n = \bar{x}_{a(n)}^k \\ & \theta_n \geq \underline{Q}_n^{k-1}(x_n). \end{aligned} \tag{32}$$

Let $(\bar{u}_n^k, \bar{x}_n^k, \bar{\theta}_n^k)$ denote the optimal values for the decision variables (u_n, x_n, θ_n) obtained by solving $P_n(\bar{x}_{a(n)}^k; \underline{Q}_n^{k-1})$. The state solution $\bar{x}_n^k$ is then passed forward as input for the problem at the next node n_{t+1} along the path ω^k.

5. **Backward Pass** Traverse the sampled path(s) ω^k backward from stage $t = T - 1$ down to $t = 0$. For each node $n \in \omega^k$ visited at stage t (i.e., $n = n_t$), solve a certain relaxation of problem (32) and generate one or more cuts based on the forward pass state solution $\bar{x}_n^k$ obtained at node n. These cuts serve as under-estimators for the true value function $Q_n(\cdot)$. Let $g_n^k(\cdot)$ denote the generated lower approximation for Q_n during the backward pass of iteration k. If multiple cuts are generated for node n in this iteration (e.g., from different sampled paths visiting n, or multiple cut types are allowed), then $g_n^k(x_{a(n)})$ represents their pointwise maximum. The specific details of the exact form of $g_n^k(\cdot)$ and the type of relaxation solved depend on the particular algorithm variant (like SDDiP, SLDP) and are discussed in subsequent sections. The newly generated cut(s) are then added to the approximation for node n. The general update rule is

$$\underline{Q}_n^k(\cdot) = \max \left\{ \underline{Q}_n^{k-1}(\cdot), g_n^k(\cdot) \right\}. \tag{33}$$

For nodes n not visited by any sampled path ω^k during the forward pass of iteration k, no new cuts are generated, so the approximation remains unchanged, i.e., $\underline{Q}_n^k = \underline{Q}_n^{k-1}$.

7.3. The SDDiP algorithm

We now focus on the Stochastic Dual Dynamic integer Programming (SDDiP) algorithm developed by Zou et al. (2019), which is an important algorithm for MSMIPs with binary state variables. While the forward pass follows the general scheme outlined previously, the backward pass employs a specific cut generation strategy.

The backward pass at iteration k attempts to improve the lower approximation of the value function for $\underline{Q}_n^{k-1}$ at each visited node n. It uses $\bar{x}_{a(n)}^k$ obtained from the forward pass as the evaluation point. In SDDiP, a relaxation (depending on the cut type used, stated later) of the forward problem at node n is solved during the backward pass to derive the cut at node n:

$$\begin{aligned}\hat{Q}_n(\bar{x}^k_{a(n)}; \underline{Q}^k_n) := \min_{z_n, u_n, x_n, \theta_n} \left\{ c_n^\top x_n + d_n^\top u_n + \theta_n \right\} \\ \text{subject to} \\ (z_n, u_n, x_n) \in X_n \\ z_n = \bar{x}^k_{a(n)} \\ z_n \in \{0,1\}^d \\ \theta_n \geq \underline{Q}^k_n(x_n).\end{aligned} \tag{34}$$

This is essentially the forward problem at node n, with $\underline{Q}^{k-1}_n$ replaced by $\underline{Q}^k_n$, because the backward pass goes from the last stage to the first stage, and some of its successor children nodes $\underline{Q}_m$ for $m \in \mathcal{S}_n$ may have been updated.

A key insight is that the process of generating cuts in the SDDiP backward pass mirrors the strategies used in two-stage stochastic integer programming. In the two-stage setting, given a first-stage solution $\bar{x}$, various techniques are employed to generate cuts approximating the second-stage value function $Q_s(\cdot)$ for each scenario s evaluated at $\bar{x}$. SDDiP applies these same cut generation techniques within its recursive, multistage framework. Specifically, for a node n visited in the backward pass with state $\bar{x}^k_{a(n)}$, these techniques are used to generate cuts for the approximate value function $\hat{Q}_n(\cdot; \underline{Q}^k_n)$ at the point $\bar{x}^k_{a(n)}$. Since the true expected future cost $Q_n(x_n)$ is always greater than or equal to its approximation $\underline{Q}^k_n(x_n)$ if validity is maintained throughout, any cut that validly underestimates $\hat{Q}_n(\cdot; \underline{Q}^k_n)$ also serves as a valid underestimator for the true value function $Q_n(\cdot)$.

Cut families. The following cuts are proposed in the SDDiP paper. The primary distinction between these cut families is how they handle the constraint $z_n = \bar{x}^k_{a(n)}$ and the integrality requirements of inputs z_n within the nodal subproblem. This is a trade-off between the computational effort needed to generate a cut and the quality (e.g., tightness) of the resulting value function approximation. Since they share the same technique as the two-stage resource decomposition, please refer to the appropriate sections in the two-stage case or the SDDiP paper (Zou et al., 2019):

1. Benders Cut, obtained by relaxing the integrality (replacing them with bounds) at node n using the LP dual variables on the constraint $z_n = \bar{x}^k_{a(n)}$, in the same way as SDDP.

2. Strengthened Benders Cut. The idea is to shift the Benders Cut so that it is tight at some point, but it is not guaranteed to be tight at the current reference point $\bar{x}^k_{a(n)}$, see Remark 9. The cut is obtained by solving an mixed-integer program.

3. Integer Optimality Cut, as in Laporte and Louveaux (1993), De La Vega et al. (2023), and Parada et al. (2024), that produce a cut tight at $\bar{x}^k_{a(n)}$, but provides no information at all other points. Such cuts are enumerative in nature.

4. Lagrangian Cut, which involves dualizing $z_n = \bar{x}^k_{a(n)}$ in a Lagrangian relaxation of (34) to obtain the cut coefficients.

Properties of Cuts. In the SDDiP setup, ideally we want the cuts to have the following properties:

1. *Valid* with respect to the objective function used for its generation ($\hat{Q}_n(\cdot; \underline{Q}_n^k)$ derived in the backward pass). Because $\hat{Q}_n(\cdot; \underline{Q}_n^k) \leq Q_n(\cdot)$ (assuming validity of $\underline{Q}_n^k$), this ensures the cut g_n^k is also valid with respect to the true value function Q_n. This property guarantees the correctness of the algorithm by maintaining $\underline{Q}_n \leq Q_n$.

2. *Tight* means the cut is strong enough to ensure algorithmic progress. An ideal cut is facet-defining for the epigraph of the value function it approximates. However the minimum requirement for binary state problems from a theoretical perspective is that it matches the objective value at the point of its generation. (See Remark 17 for detailed discussion.) That is, the generated cut $g_n^k(\cdot)$ satisfies $g_n^k(\bar{x}_{a(n)}^k) = \hat{Q}_n(\bar{x}_{a(n)}^k; \underline{Q}_n^k)$, meaning the approximation $\underline{Q}_n$ becomes exact (w.r.t. $\hat{Q}_n(\bar{x}_{a(n)}^k; \underline{Q}_n^k)$) at the reference point $\bar{x}_{a(n)}^k$ after the update. Note that we do not require the cut to be tight with respect to the true value function Q_n.

3. *Finite* means the algorithm could only generate a finite number of distinct cut coefficients. This can be used for finite termination of the algorithm.

Remark 17. *While the theoretical convergence of the algorithm only requires a cut to be tight at its generation point, practical performance is highly dependent on the cut's overall strength. A stronger cut improves the value function approximation over a wider region of the state space. This global improvement can significantly reduce the number of cuts and iterations required for convergence. In contrast, relying on only locally effective cuts (e.g., Integer Optimality Cuts) can cause the algorithm to behave more like an enumerative scheme. Accumulating a large number of such cuts in the nodal subproblems could significantly slow down or cause issues in the LP/MIP solver.*

However, this emphasis on strength should be balanced with computational effort. Non-tight cuts, such as Strengthened Benders cuts, can still be valuable. Because they are often computationally cheaper to generate, they can be effective for making rapid initial progress, especially in early iterations when the approximation in the next stage is still far from the true value function. Furthermore, as discussed for the two-stage case (Remark 9), the coefficients from these weaker cuts can be used to warm-start the more computationally demanding subproblems required to generate stronger cuts.

Remark 18. *When generating a Lagrangian cut, the dual problem may have multiple optimal multipliers. The literature discusses strategies for selecting among these multipliers to generate "deep" cuts in the mixed-integer state variable setting (Füllner et al., 2024a,b).*

Finite Convergence of SDDiP for Binary State MSMIP. The finite convergence of SDDiP for MSMIPs with purely binary state variables ($x_n \in \{0,1\}^d$) is a consequence of the *Valid*, *Tight*, and *Finite* property of the cuts used in the backward pass, and the path selection scheme visits any path $\omega = (n_0, \ldots, n_T)$ infinitely often, with probability one.

1. Monotonicity via Validity: By the Validity property, the generated cuts $g_n^k(\cdot)$ are valid lower bounds on the value function computed in the backward pass (i.e., $g_n^k(x) \leq \hat{Q}_n(x; \underline{Q}_n^k)$), which itself is a lower bound on the true value function $Q_n(x)$. The update rule $\underline{Q}_n^k(\cdot) = \max\{\underline{Q}_n^{k-1}(\cdot), g_n^k(\cdot)\}$ ensures that the approximation functions $\underline{Q}_n^k(\cdot)$ are non-decreasing point-wise as the iteration k increases. Consequently, the overall lower bound on the problem's optimal value, derived from the root node's approximation, is non-decreasing throughout the algorithm's execution.

2. Progress via Tightness: The tightness property of the cuts used implies that the algorithm cannot get stuck with suboptimal approximations indefinitely. Suppose the approximations $\{\underline{Q}_n^k\}$ were to stabilize prematurely after some iteration K (i.e., $\underline{Q}_n^k = \underline{Q}_n^K$ for all n and all $k \geq K$). Assuming the same subproblem in the forward pass solves to the same optimal solution (the deterministic subproblem solver assumption), if a specific path $\omega = (n_0, \ldots, n_T)$ is sampled repeatedly after iteration K, the sequence of forward states generated along that path, $\{\bar{x}_{n_t}^k\}_{t=0}^T$, will also stabilize. Let us denote this stabilized path-dependent solution as $\{\bar{x}_{n_t}^*(\omega)\}_{t=0}^T$. If the stabilized approximations $\{\underline{Q}_n^K\}$ do not correspond to the true optimal value functions, then there must exist at least one path ω^* such that its stabilized solution $\{\bar{x}_{n_t}^*(\omega^*)\}_{t=0}^T$ does not satisfy the Bellman optimality conditions with respect to these stabilized approximations. This implies there exists at least one node $n^* \in \omega^*$ along this path where the stabilized approximation is strictly lower than the value calculated in the backward pass: $\underline{Q}_{n^*}^K(\bar{x}_{a(n^*)}^*(\omega^*)) < \hat{Q}_{n^*}(\bar{x}_{a(n^*)}^*(\omega^*); \underline{Q}_{n^*}^K)$. The sampling assumption ensures that this specific path ω^* will eventually be selected again (with probability 1) in some iteration $k' > K$. When the backward pass reaches n^* along path ω^* in iteration k', the *Tightness* property guarantees the newly generated cut $g_{n^*}^{k'}$ matches this higher value exactly at the evaluated point: $g_{n^*}^{k'}(\bar{x}_{a(n^*)}^*(\omega^*)) = \hat{Q}_{n^*}(\bar{x}_{a(n^*)}^*(\omega^*); \underline{Q}_{n^*}^K)$. Because this value is strictly greater than $\underline{Q}_{n^*}^K(\bar{x}_{a(n^*)}^*(\omega^*))$, adding this cut via the max operation forces $\underline{Q}_{n^*}^{k'}(\bar{x}_{a(n^*)}^*(\omega^*)) > \underline{Q}_{n^*}^K(\bar{x}_{a(n^*)}^*(\omega^*))$. This contradicts the assumption that the approximations had stabilized. Therefore, the algorithm is guaranteed (w.p. 1) to improve its approximation at the evaluated points along specific paths whenever a suboptimal state is encountered during sampling for that path. This can be viewed as progress towards optimality.

3. Finite Termination via Finite Number of Cut Coefficents: In SDDiP, only a finite number of distinct cuts (defined by their coefficients) can be generated by the algorithm, if the same subproblem generates the same Lagrangian Cut. Combined with point (2), which shows the algorithm must make progress (change the approximation) w.p. 1 if not yet optimal, this property says only a finite number of such improving steps can be made. Therefore, it must converge to a state where the approximations define an optimal policy within a finite number of iterations, with probability one.

Despite the similarity of cut generation techniques to those used in two-stage primal decomposition, and the existence of theoretical finite convergence results for the binary case, practical implementations of SDDP-like algorithms for MSMIPs require significantly more computational effort than their two-stage counterparts. One reason is the exponential growth in the number of scenario paths, which makes the theoretical sampling requirement underpinning convergence proofs (visiting all paths infinitely often) practically impossible to satisfy; indeed, even visiting each distinct path once can often require a prohibitive number of iterations, let alone visiting each path infinitely often. Furthermore, the recursive nature of value function approximations in the multistage setting presents a distinct challenge: errors or weak initial estimates in later-stage value functions directly degrade the quality of decisions made and cuts generated in earlier stages. Unlike the two-stage setting which involves only one level of future cost approximation, accurately building these nested approximations across many stages can significantly slow the overall stabilization of the algorithm, especially during early iterations.

Tightness of Lagrangian Cuts in SDDiP, Binarization. In the standard SDDiP algorithm using Lagrangian cuts, the generated cuts $g_n^k(\cdot)$ are affine. As the approximation $\underline{Q}_n^k(\cdot)$ is constructed as the pointwise maximum of these affine cuts (via Eq. (33)), it is inherently convex and piecewise linear. Consequently, the

aggregated expected cost-to-go approximation $\underline{Q}_n^k(x_n) = \sum_{m \in \mathcal{S}(n)} q_{nm} \underline{Q}_m^k(x_n)$ is also convex. The best approximation this framework can build for the true expected cost-to-go function Q_n is its convex envelope, $co(Q_n)$.

This convex approximation approach works effectively for MSMIPs with purely binary state variables[4] ($x_n \in \{0,1\}^d$) because in the binary case, any feasible input state is an extreme point of the set of feasible input states, hence the value of the function at that point coincides with the value of its convex envelope, i.e., $Q_n(x_n) = co(Q_n)(x_n)$, similar to Proposition 4 in Section 4.1 for the two-stage case. When combined with cuts (like Lagrangian cuts) that provide tightness with respect to the backward pass objective, this allows the algorithm's lower approximation $\underline{Q}_n^k$ to eventually converge to the true optimal value function over the relevant domain.

However, when state variables x_n include general integers or continuous components, the property that the function matches its convex envelope at all feasible integer points generally *does not* hold. There can exist feasible integer states x_n where the true value is strictly greater than the value of its convex envelope: $Q_n(x_n) > co(Q_n)(x_n)$. Because the standard SDDiP algorithm builds its convex approximation $\underline{Q}_n^k$ using affine cuts, the generated cuts are generally not tight with respect to the true value function Q_n at these critical integer points where the gap $Q_n > co(Q_n)$ exists. Consequently, even though valid cuts (e.g., Lagrangian cuts) can still be computed for these problems, the algorithm can stall without converging to the true optimal solution for general MSMIPs. Examples illustrating this discrepancy between Q_n and $co(Q_n)$ for problems involving general integer or continuous state variables can be found, for instance, in Zou et al. (2019) Ahmed et al. (2022), and Deng and Xie (2024).

Binarization, as proposed in the SDDiP paper (Zou et al., 2019), partially resolves the issue of non-binary state variables by transforming the MSMIP. General integer components of the state variable are replaced with their exact binary expansions, while bounded continuous components are approximated using binary (fractional) expansions. For example, to replace a non-negative general integer component x_{nj}, one declares new binary state variables $b_{j0}, \ldots, b_{jL} \in \{0,1\}$, and replaces x_{nj} with the expansion $x_{nj} = \sum_{l=0}^{L} 2^l b_{jl}$. Here, L is chosen such that $2^{L+1} - 1$ is a sufficiently large upper bound for the range of x_{nj}. Continuous components x_{nj} within a bounded range $[X_{min}, X_{max}]$ can be similarly approximated to a desired precision using a binary fractional expansion, such as $x_{nj} = X_{min} + (X_{max} - X_{min}) \sum_{l=1}^{L} 2^{-l} b_{jl}$. All appearances of the original state variable component x_{nj} in the model constraints and objective are then replaced by its corresponding binary expansion. As a consequence of this transformation, the state space of the reformulated problem consists entirely of binary variables. This allows the standard SDDiP algorithm to be applied to the transformed problem and recovers the optimal policy at each node with respect to the original general integer state variables. For problems involving continuous state variables, the binarization serves as an approximation, and the quality of the solution depends on the precision used (i.e., the number of binary expansion variables L).

From a geometric viewpoint, the strategies used to handle general integer or continuous state variables can be interpreted as forms of **lifting**.[5] Generally, lifting in this context refers to augmenting the original state variables x_n by adding new binary variables and constraints, thereby mapping the problem into a

[4] It is worth noting that this discussion is on the nature of the *state variables* (x_n) as they determine the domain of the value function Q_n. The *control variables* (u_n) optimized within each node's problem can be continuous, integer, binary, or mixed-integer without altering the fundamental challenges discussed here related to approximating Q_n when x_n involves general integers or continuous components.

[5] This concept should not be confused with traditional lift-and-project methods from deterministic MIP, which typically involve generating cuts in a higher-dimensional space and then projecting them back into the original variable space.

higher-dimensional representation to allow for separation. The specific mapping can be different. For instance, binary variables might be used as indicators for logical disjunctions. The state variable binarization can be viewed as a particular type of lifting: it introduces binary expansion variables (e.g., $b_{j0}, \ldots, b_{jL}$) and linking constraints (e.g., $x_{nj} = \sum_{l=0}^{L} 2^l b_{jl}$) that equate original components to these expansions. With this binary expansion approach, each feasible integer value for a component x_{nj} corresponds one-to-one with an extreme point in the higher-dimensional space defined by the binary b_{jl} variables. This explicit representation allows cuts generated with respect to these binary state variables (e.g., Lagrangian cuts) to be stronger than in the original space. Such binarization or lifting schemes are tightly connected to the non-convex cuts described in the next section (mixed-integer representable cuts). In fact, Füllner and Rebennack (2022) shows that each cut in the binarized space corresponds to a large number of cuts in the original space.

A practical consideration for the binarization technique from SDDiP is the potentially large increase in the number of binary state variables required, particularly when approximating continuous variables. Füllner and Rebennack (2022) projects the cut in the binary space back to the original space, with the help of auxiliary binary variables.

Adaptive lifting schemes also exist, such as the approach in Qi and Sen (2017, 2024) for two-stage problems, and more recently Yang and Yang (2025), which perform lifting by adding binary variables incrementally to enforce disjunctions, typically only when the cut derived fails to separate the current point, or in a bisection manner.

7.4. Mixed-Integer Representable Cuts

An alternative approach to state variable binarization for handling general integer or continuous state variables in MSMIPs is to directly construct non-convex under-approximations $g_n^k(\cdot)$ for the value function $Q_n(\cdot)$. Since g_n^k is no longer restricted to be affine, the resulting overall approximation $\underline{Q}_n^k$ can also be non-convex, so the approximation can capture the true value function more accurately than its convex envelope, removing the key limitation of standard SDDiP with linear cuts.

Reverse Norm Cuts, Augmented Lagrangian Cuts. Ahmed et al. (2022) introduced the Reverse Norm cut and the Augmented Lagrangian cut. Assume the true value function $Q_n(x_{a(n)})$ is Lipschitz continuous with respect to a norm $\|\cdot\|$ (typically the L_1 or L_∞ norm for computational tractability in MIPs) with a known or estimated Lipschitz constant ρ_n. The definition of Lipschitz continuity, $|Q_n(x_{a(n)}) - Q_n(\bar{x}_{a(n)}^k)| \leq \rho_n \|x_{a(n)} - \bar{x}_{a(n)}^k\|$, directly implies the lower bound relationship which forms the basis for the reverse norm cut:

$$Q_n(x_{a(n)}) \geq g_n^k(x_{a(n)}) := Q_n(\bar{x}_{a(n)}^k) - \rho_n \|x_{a(n)} - \bar{x}_{a(n)}^k\|. \tag{35}$$

By construction in (35), the reverse norm cut is tight at the point $\bar{x}_{a(n)}^k$ where it is generated. However, similar to integer optimality cuts, its strength is primarily local; the subtracted norm term $\|x_{a(n)} - \bar{x}_{a(n)}^k\|$ means the cut provides a weak lower bound for points $x_{a(n)}$ far from $\bar{x}_{a(n)}^k$.

Augmented Lagrangian cuts extend the idea of Lagrangian cuts by adding an additional penalty term to the objective of the subproblem. This generates a stronger approximation of the value function. An Augmented Lagrangian cut is generated by solving the following problem:

$$\max_{\pi_n} \min_{z_n, u_n, x_n} c_n^\top x_n + d_n^\top u_n + \underline{Q}_n^k(x_n) \; - \pi_n^\top (z_n - \bar{x}_{a(n)}^k) + \rho_n \|z_n - \bar{x}_{a(n)}^k\|$$
$$\text{subject to} \tag{36}$$
$$(z_n, u_n, x_n) \in X_n.$$

A valid cut for the true value function Q_n is derived from the optimal value v_n^* and dual multiplier π_n^* of the augmented Lagrangian problem (36). The derivation relies on weak duality, applying the dual lower bound to the solution that defines $Q_n(x_{a(n)})$ in (31), and using the relaxation $Q_n(x_n) \geq \underline{Q}_n^k(x_n)$. We obtain

$$Q_n(x_{a(n)}) \geq v_n^* + \pi_n^{*\top}(x_{a(n)} - \bar{x}_{a(n)}^k) - \rho_n \|x_{a(n)} - \bar{x}_{a(n)}^k\|. \tag{37}$$

The resulting cut is non-convex because of the concave term $-\rho_n \|x_{a(n)} - \bar{x}_{a(n)}^k\|$, but it remains Mixed-Integer Representable. We refer to Ahmed et al. (2022) for a complete derivation.

ReLU Cuts. More recent work by Deng and Xie (2024) proposed ReLU cuts, which is a reformulation of the constraints $z_n = \bar{x}_{a(n)}$ into $(z_n - \bar{x}_{a(n)})_+ = 0$ and $(\bar{x}_{a(n)} - z_n)_+ = 0$. Such construction leads to non-linear cuts. Because of space constraints, we refer the reader to the cited paper for the details.

All the above cut families share several desirable properties: (i) They can be both *valid* and *tight*. (ii) They are typically chosen to be mixed-integer representable to ensure that the constraint $\theta_n \geq \underline{Q}_n^k(x_n)$ added to the forward pass subproblem (32) can be modeled within a standard MIP solver's capabilities using auxiliary variables and constraints. (iii) By operating directly on the original state variables, these methods avoid the state space explosion that can result from upfront binarization techniques. However, now additional integer or binary variables are needed to represent the cuts.

Unlike standard SDDiP convergence analysis, which relies on generating a finite number of cuts, schemes using mixed-integer representable cuts for mixed-integer states lack this *finiteness* guarantee. Because the algorithm can potentially create an infinite sequence of distinct cuts, proving convergence requires an alternative to the standard finiteness arguments.

8. Conclusion and Future directions

This survey has detailed the two dominant paradigms for solving multistage stochastic mixed-integer programs (MSMIPs): dual decomposition and primal decomposition. Dual (or scenario) decomposition methods reformulate the problem by creating copies of variables for each scenario and relaxing the coupling non-anticipativity constraints. This approach excels at leveraging the structure of individual scenario subproblems but can face challenges with the sheer scale of the formulation as the number of scenarios grows. In contrast, primal (or stagewise) decomposition, exemplified by the SDDP family of algorithms, uses a recursive, dynamic programming-like formulation. These methods are powerful for problems with stagewise independence and can handle a vast number of potential scenarios implicitly, but their effectiveness hinges on the quality of the approximations (cuts) built for the non-convex value functions, which is challenging when state variables are not purely binary.

A critical observation is that all the detailed algorithms—from DDSIP to SDDiP and its variants—operate within the Sample Average Approximation (SAA) framework. They assume the stochastic process is represented by a fixed, predetermined scenario tree or lattice and proceed to solve that deterministic equivalent

model to optimality. This approach provides a solution that is optimal only for the given sample of scenarios, which may or may not be close to the true optimal policy of the underlying stochastic process.

A promising direction for future research is the development of algorithms for MSMIPs that move beyond the SAA paradigm. For multistage stochastic linear programs, online sampling-based methods like Stochastic Decomposition (Higle and Sen, 1991, 1994) and multistage extensions like SDLP (Gangammanavar and Sen, 2021) do not require a full scenario tree upfront. Instead, they iteratively sample scenarios and update the solution, providing statistical, rather than deterministic, guarantees of convergence to the true optimum.

Extending these incremental sampling approaches to the mixed-integer domain remains a challenging open question. When we move beyond a fixed SAA scenario tree, we are dealing with a problem that has a potentially infinite number of scenarios. As a result, the true expected future cost at any stage cannot be calculated exactly; it can only be estimated by solving a problem with a finite sample of scenarios. A cut generated from such a sample is therefore only a statistical estimate, and sampling error can "corrupt" its validity. That is, the cut is not guaranteed to be a true lower bound on the value function of the real problem. This breaks the core assumption of methods like SDDiP, which require every cut to be valid. The challenge is to develop a fix that does not depend on per-iteration validity. Such an algorithm would have to handle occasionally "incorrect" cuts, relying on the principle of asymptotic validity—the idea that the cuts become statistically accurate as more samples are incorporated—which is a significant departure from the current SIP literature.

Future work could also focus on enhancing the existing SAA-based methods. The development of stronger, yet computationally tractable, mixed-integer representable non-convex cuts (Ahmed et al., 2022; Deng and Xie, 2024) is important for improving the performance of primal decomposition on general MSMIPs. Similarly, integrating machine learning techniques to learn approximate value functions could offer a powerful hybrid approach, such as Larsen et al. (2024). This could potentially allow combining the recursive structure of SDDP-like algorithms with the expressive power of neural networks to better navigate the complexities of integer-variable state spaces. However, it is important to recognize that inherent with SAA are issues related to variability which can introduce variance in the recommended decisions. This aspect has received some study for algorithms in SLP, but there have been very few papers which recognize the need to control variability which result from SAA-based approaches. While the simulation optimization literature has studied such issues for decision models with a few discrete choices, the complications are much more challenging in the SMIP realm. We refer to Xu and Sen (2023) for some initial guidance towards variance reduction, and the need for regularization tools within SMIP algorithms. This approach provides an initial direction which combines simulation-optimizarion with stochastic mixed-binary optimization.

Acknowledgments. Suvrajeet Sen and Yihang Zhang are grateful for the support provided by the Advanced Scientific Computing Research (ASCR) program under the U.S. Department of Energy, Award Number DE-SC0023361. The research of Ward Romeijnders has been supported by the Netherlands Organisation for Scientific Research [Grant VI.Vidi.211.066].

References

Ahmed, S., Cabral, F., da Costa, B., 2022. Stochastic Lipschitz dynamic programming. Mathematical Programming 191, 755–793.

Atakan, S., Sen, S., 2018. A Progressive Hedging based branch-and-bound algorithm for mixed-integer stochastic programs. Computational Management Science 15, 501–540. URL: http://link.springer.com/10.1007/s10287-018-0311-3, doi:10.1007/s10287-018-0311-3.

Beale, E.M.L., 1955. On Minimizing a Convex Function Subject to Linear Inequalities. Journal of the Royal Statistical Society Series B: Statistical Methodology 17, 173–184. URL: https://academic.oup.com/jrsssb/article/17/2/173/7026715, doi:10.1111/j.2517-6161.1955.tb00191.x.

van Beesten, E., Romeijnders, W., Morton, D., 2024. Pragmatic distributionally robust optimization for simple integer recourse models. SIAM Journal on Optimization 34, 1755–1783.

Bertsekas, D.P., Yu, H., 2011. A unifying polyhedral approximation framework for convex optimization. SIAM Journal on Optimization 21, 333–360.

Birge, J., Louveaux, F., 1997. Introduction to Stochastic Programming. Springer.

Bodur, M., Luedtke, J.R., 2017. Mixed-integer rounding enhanced benders decomposition for multiclass service-system staffing and scheduling with arrival rate uncertainty. Management Science 63, 2073–2091.

Bovim, T.R., Christiansen, M., Gullhav, A.N., Range, T.M., Hellemo, L., 2020. Stochastic master surgery scheduling. European Journal of Operational Research 285, 695–711. URL: https://www.sciencedirect.com/science/article/pii/S0377221720301041, doi:https://doi.org/10.1016/j.ejor.2020.02.001.

Carøe, C., Schultz, R., 1999. Dual decomposition in stochastic integer programming. Operations Research Letters 24, 37–45.

Carøe, C.C., Tind, J., 1997. A cutting-plane approach to mixed 0–1 stochastic integer programs. European Journal of Operational Research 101, 306–316.

Chen, R., Luedtke, J., 2022. On generating Lagrangian cuts for two-stage stochastic integer programs. INFORMS Journal on Computing 34, 2332–2349. URL: https://doi.org/10.1287/ijoc.2022.1185, doi:10.1287/ijoc.2022.1185, arXiv:https://doi.org/10.1287/ijoc.2022.1185.

Dantzig, G.B., 1955. Linear Programming under Uncertainty. Management Science 1, 197–206. URL: https://pubsonline.informs.org/doi/10.1287/mnsc.1.3-4.197, doi:10.1287/mnsc.1.3-4.197.

De La Vega, J., Gendreau, M., Morabito, R., Munari, P., Ordóñez, F., 2023. An integer L-shaped algorithm for the vehicle routing problem with time windows and stochastic demands. European Journal of Operational Research 308, 676–695.

Deng, H., Xie, W., 2024. On the ReLU Lagrangian Cuts for Stochastic Mixed Integer Programming. URL: http://arxiv.org/abs/2411.01229, doi:10.48550/arXiv.2411.01229. arXiv:2411.01229 [math].

Dowson, O., 2020. The policy graph decomposition of multistage stochastic programming problems. Networks 76, 3–23. URL: https://onlinelibrary.wiley.com/doi/10.1002/net.21932, doi:10.1002/net.21932.

Füllner, C., Rebennack, S., 2022. Non-convex nested Benders decomposition. Mathematical Programming 196, 987–1024.

Füllner, C., Rebennack, S., 2025. Stochastic dual dynamic programming and its variants: A review. SIAM Review 67, 415–539. URL: https://doi.org/10.1137/23M1575093, doi:10.1137/23M1575093, arXiv:https://doi.org/10.1137/23M1575093.

Füllner, C., Sun, X.A., Rebennack, S., 2024a. A new framework to generate Lagrangian cuts in multistage stochastic mixed-integer programming. Optimization Online. URL: https://optimization-online.org/?p=27312. published: 2024/08/08, Updated: 2024/11/18, Accessed: 2025/05/15.

Füllner, C., Sun, X.A., Rebennack, S., 2024b. On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming. Optimization Online. URL: https://optimization-online.org/?p=27295. published: 2024/08/07, Updated: 2025/02/19, Accessed: 2025/05/15.

Gade, D., Hackebeil, G., Ryan, S.M., Watson, J.P., Wets, R.J.B., Woodruff, D.L., 2016. Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. Mathematical Programming 157, 47–67. URL: https://link.springer.com/article/10.1007/s10107-016-1000-z, doi:10.1007/s10107-016-1000-z. company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 1 Publisher: Springer Berlin Heidelberg.

Gade, D., Küçükyavuz, S., Sen, S., 2014. Decomposition algorithms with parametric Gomory cuts for two-stage stochastic integer programs. Mathematical Programming 144, 39–64.

Gangammanavar, H., Sen, S., 2021. Stochastic dynamic linear programming: A sequential sampling algorithm for multistage stochastic linear programming. SIAM Journal on Optimization 31, 2111–2140. URL: https://epubs.siam.org/doi/10.1137/19M1290735, doi:10.1137/19M1290735.

Guo, G., Hackebeil, G., Ryan, S.M., Watson, J.P., Woodruff, D.L., 2015. Integration of progressive hedging and dual decomposition in stochastic integer programs. Operations Research Letters 43, 311–316. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167637715000462, doi:10.1016/j.orl.2015.03.008.

Haneveld, W.K.K., van der Vlerk, M.H., 1999. Stochastic integer programming: General models and algorithms. Annals of operations research 85, 39.

Higle, J.L., Sen, S., 1991. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. Mathematics of Operations Research 16, 650–669. URL: https://pubsonline.informs.org/doi/10.1287/moor.16.3.650, doi:10.1287/moor.16.3.650. publisher: INFORMS.

Higle, J.L., Sen, S., 1994. Finite master programs in regularized stochastic decomposition. Mathematical Programming 67, 143–168. URL: https://link.springer.com/article/10.1007/BF01582219, doi:10.1007/BF01582219.

Hobbs, B.F., Rothkopf, M.H., O'Neill, R.P., Chao, H.p., Hillier, F.S. (Eds.), 2001. The Next Generation of Electric Power Unit Commitment Models. volume 36 of *International Series in Operations Research & Management Science*. Springer US, Boston, MA. URL: http://link.springer.com/10.1007/b108628, doi:10.1007/b108628.

Huang, Y., Pardalos, P.M., Zheng, Q.P., 2017. Electrical Power Unit Commitment. SpringerBriefs in Energy, Springer US, Boston, MA. URL: http://link.springer.com/10.1007/978-1-4939-6768-1, doi:10.1007/978-1-4939-6768-1.

Kim, K., Dandurand, B., 2022. Scalable branching on dual decomposition of stochastic mixed-integer programming problems. Mathematical Programming Computation 14, 1–41. URL: https://doi.org/10.1007/s12532-021-00212-y, doi:10.1007/s12532-021-00212-y.

Kim, K., Mehrotra, S., 2015. A two-stage stochastic integer programming approach to integrated staffing and scheduling with application to nurse management. Operations Research 63, 1431–1451.

Kim, K., Petra, C.G., Zavala, V.M., 2019. An asynchronous bundle-trust-region method for dual decomposition of stochastic mixed-integer programming. SIAM Journal on Optimization 29, 318–342. URL: https://doi.org/10.1137/17M1148189, doi:10.1137/17M1148189, arXiv:https://doi.org/10.1137/17M1148189.

Kim, K., Zavala, V.M., 2018. Algorithmic innovations and software for the dual decomposition method applied to stochastic mixed-integer programs. Mathematical Programming Computation 10, 225–266. URL: https://doi.org/10.1007/s12532-017-0128-z, doi:10.1007/s12532-017-0128-z.

Klein Haneveld, W., Stougie, L., van der Vlerk, M., 2006. Simple integer recourse models: convexity and convex approximations. Mathematical Programming 108, 435–473.

Klein Haneveld, W., van der Vlerk, M., Romeijnders, W., 2020. Stochastic Programming: Modeling Decision Problems Under Uncertainty. Springer.

Kleywegt, A., Shapiro, A., Homem-de Mello, T., 2002. The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization 12, 479–502.

Küçükyavuz, S., Sen, S., 2017. An introduction to two-stage stochastic mixed-integer programming, in: INFORMS TutORials in Operations Research, pp. 1–27.

van der Laan, N., Romeijnders, W., 2024. A converging benders' decomposition algorithm for two-stage mixed-integer recourse models. Operations Research 72, 2190–2214. URL: https://doi.org/10.1287/opre.2021.2223, doi:10.1287/opre.2021.2223, arXiv:https://doi.org/10.1287/opre.2021.2223.

Laporte, G., Louveaux, F., 1993. The integer L-shaped method for stochastic integer programs with complete recourse. Operations Research Letters 13, 133–142.

Larsen, E., Frejinger, E., Gendron, B., Lodi, A., 2024. Fast continuous and integer L-shaped heuristics through supervised learning. INFORMS Journal on Computing 36, 203–223. URL: https://pubsonline.informs.org/doi/10.1287/ijoc.2022.0175, doi:10.1287/ijoc.2022.0175.

Li, X., Tomasgard, A., Barton, P.I., 2011. Nonconvex generalized benders decomposition for stochastic separable mixed-integer nonlinear programs. Journal of Optimization Theory and Applications 151, 425–454. doi:10.1007/s10957-011-9888-1.

Louveaux, F., van der Vlerk, M., 1993. Stochastic programming with simple integer recourse. Mathematical Programming 61, 301–325.

Lubin, M., Martin, K., Petra, C., Sandıkçı, B., 2013. On parallelizing dual decomposition in stochastic integer programming. Operations Research Letters 41, 252–258. URL: https://www.sciencedirect.com/science/article/pii/S0167637713000242, doi:https://doi.org/10.1016/j.orl.2013.02.003.

Lulli, G., Sen, S., 2004. A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. Management Science 50, 786–796.

Löhndorf, N., Shapiro, A., 2019. Modeling time-dependent randomness in stochastic dual dynamic programming. European Journal of Operational Research 273, 650–661. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221718306787, doi:10.1016/j.ejor.2018.08.001.

Maggioni, F., Pflug, G.C., 2016. Bounds and approximations for multistage stochastic programs. SIAM Journal on Optimization 26, 831–855.

Ntaimo, L., 2013. Fenchel decomposition for stochastic mixed-integer programming. Journal of Global Optimization 55, 141–163.

Ntaimo, L., Sen, S., 2005. The Million-Variable "March" for Stochastic Combinatorial Optimization. Journal of Global Optimization

32, 385–400. URL: http://link.springer.com/10.1007/s10898-004-5910-6, doi:10.1007/s10898-004-5910-6.

Parada, L., Legault, R., Côté, J.F., Gendreau, M., 2024. A disaggregated integer L-shaped method for stochastic vehicle routing problems with monotonic recourse. European Journal of Operational Research 318, 520–533.

Pereira, M., Pinto, L., 1991. Multi-stage stochastic optimization applied to energy planning. mathematical programming. Mathematical Programming 52, 359–375.

Pflug, G.C., Pichler, A., 2014. Multistage Stochastic Optimization. Springer Series in Operations Research and Financial Engineering, Springer International Publishing, Cham. URL: https://link.springer.com/10.1007/978-3-319-08843-3, doi:10.1007/978-3-319-08843-3.

Pham, D.N., Klinkert, A., 2008. Surgical case scheduling as a generalized job shop scheduling problem. European Journal of Operational Research 185, 1011–1025. URL: https://www.sciencedirect.com/science/article/pii/S0377221706005820, doi:https://doi.org/10.1016/j.ejor.2006.03.059.

Philpott, A.B., Craddock, M., Waterer, H., 2000. Hydro-electric unit commitment subject to uncertain demand. European Journal of Operational Research 125, 410–424.

Philpott, A.B., De Matos, V.L., 2012. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. European Journal of Operational Research 218, 470–483. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221711010332, doi:10.1016/j.ejor.2011.10.056.

Powell, W.B., 2011. Approximate dynamic programming : solving the curses of dimensionality. 2nd ed. ed., Wiley, Hoboken, N.J.

Qi, Y., Sen, S., 2017. The ancestral Benders' cutting plane algorithm with multi-term disjunctions for mixed-integer recourse decisions in stochastic programming. Mathematical Programming 161, 193–235.

Qi, Y., Sen, S., 2024. Correction to: The ancestral Benders cutting-plane algorithm with multi-term disjunctions for mixed-integer recourse decisions in stochastic programming. Mathematical Programming 205, 841–845. URL: https://link.springer.com/10.1007/s10107-023-02039-y, doi:10.1007/s10107-023-02039-y.

Romeijnders, W., van der Laan, N., 2024. Benders decomposition with scaled cuts for multistage stochastic mixed-integer programs. URL: https://optimization-online.org/?p=26876.

Sandikçi, B., Özaltin, O., 2017. A scalable bounding method for multistage stochastic programs. SIAM Journal on Optimization 27, 1772–1800.

Sandıkçı, B., Kong, N., Schaefer, A.J., 2013. A hierarchy of bounds for stochastic mixed-integer programs. Mathematical Programming 138, 253–272. URL: https://doi.org/10.1007/s10107-012-0526-y, doi:10.1007/s10107-012-0526-y.

Schultz, R., 2003a. Mixed-integer value functions in stochastic programming, in: Combinatorial Optimization–Eureka, You Shrink!. Springer, pp. 171–184.

Schultz, R., 2003b. Stochastic programming with integer variables. Mathematical Programming 97, 285–309.

Schulze, T., Grothey, A., McKinnon, K., 2017. A stabilised scenario decomposition algorithm applied to stochastic unit commitment problems. European Journal of Operational Research 261, 247–259.

Sen, S., 2005. Algorithms for stochastic mixed-integer programming models, in: Aardal, K., Nemhauser, G., Weismantel, R. (Eds.), Discrete Optimization. Elsevier. volume 12 of *Handbooks in Operations Research and Management Science*, pp. 515–558. URL: https://www.sciencedirect.com/science/article/pii/S092705070512009X, doi:https://doi.org/10.1016/S0927-0507(05)12009-X.

Sen, S., Higle, J., 2005. The C^3 theorem and a D^2 algorithm for large scale stochastic mixed-integer programming: Set convexification. Mathematical Programming 104, 1–20.

Shapiro, A., Dentcheva, D., Ruszczynski, A., 2021. Lectures on stochastic programming: modeling and theory. SIAM.

Sherali, H., Zhu, X., 2006. On solving discrete two-stage stochastic programs having mixed-integer first-and second-stage variables. Mathematical Programming 108, 597–616.

Singh, K.J., Philpott, A.B., Wood, R.K., 2009. Dantzig-wolfe decomposition for solving multistage stochastic capacity-planning problems. Operations Research 57, 1271–1286.

Van Slyke, R., Wets, R., 1969. L-shaped linear programs with applications to optimal control and stochastic programming. SIAM Journal on Applied Mathematics 17, 638–663.

Wets, R.J.B., 1983. Solving stochastic programs with simple recourse. Stochastics 10, 219–242.

Xu, J., Sen, S., 2023. Ensemble variance reduction methods for stochastic mixed-integer programming and their application to the stochastic facility location problem. INFORMS J. on Comuting 36, 587–599. URL: https://doi.org/10.1287/ijoc.2021.0324. publisher: INFORMS.

Yang, H., Yang, H., 2025. Globally converging algorithm for multistage stochastic mixed-integer programs via enhanced Lagrangian cuts. Optimization Online URL: https://optimization-online.org/?p=29960.

Zhang, M., Küçükyavuz, S., 2014. Finitely convergent decomposition algorithms for two-stage stochastic pure integer programs. SIAM Journal on Optimization 24, 1933–1951.

Appendix

Proof of Lemma 1. Observe that for every $s = 1, \dots, S$, the function $\varphi_s : \text{conv}(X) \to \mathbb{R}$, defined as

$$\varphi_s(x) = \min_{y_s} \left\{ q_s^\top y_s : (x, y_s) \in \text{conv}(\mathcal{F}_s) \right\}, \qquad x \in \text{conv}(X),$$

is convex polyhedral on $\text{conv}(X)$, and a lower bound of v_s on X since $\text{conv}(\mathcal{F}_s) \supseteq \mathcal{F}_s$. Hence, $\varphi_s(x) \leq \text{co}_X(v_s)(x)$ for all $x \in \text{conv}(X)$.

Suppose for contradiction that there exists $\bar{x} \in \text{conv}(X)$, so that $\varphi_s(\bar{x}) < \text{co}_X(v_s)(\bar{x})$. Then, there exists $\bar{y}_s$ such that $(\bar{x}, \bar{y}_s) \in \text{conv}(\mathcal{F}_s)$ and $q_s^\top \bar{y}_s < \text{co}_X(v_s)(\bar{x}$. However, since $\text{co}_X(v_s)$ is finite and convex on $\text{conv}(X)$, we can write $\text{co}_X(v_s)$ as the supremum over its supporting hyperplanes $\alpha + \beta^\top x$. That is,

$$\begin{aligned}
\text{co}_X(v_s)(\bar{x}) &= \sup_{\alpha,\beta} \left\{ \alpha + \beta^\top \bar{x} : \alpha + \beta^\top x \leq v_s(x) \quad \forall x \in X \right\} \\
&= \sup_{\alpha,\beta} \left\{ \alpha + \beta^\top \bar{x} : \alpha + \beta^\top x \leq q_s^\top y_s \quad \forall (x, y_s) \in \mathcal{F}_s \right\} \\
&= \sup_{\alpha,\beta} \left\{ \alpha + \beta^\top \bar{x} : \alpha + \beta^\top x \leq q_s^\top y_s \quad \forall (x, y_s) \in \text{conv}(\mathcal{F}_s) \right\},
\end{aligned}$$

where the last equality holds since the constraints are linear in (x, y_s). Let $(\bar{\alpha}, \bar{\beta})$ denote the maximizers in the supremum above for $\bar{x}$, so that

$$\text{co}_X(v_s)(\bar{x}) = \bar{\alpha} + \bar{\beta}^\top \bar{x} \qquad \text{and} \qquad \bar{\alpha} + \bar{\beta}^\top \bar{x} \leq q_s^\top y_s \quad \forall (\bar{x}, y_s) \in \text{conv}(\mathcal{F}_s).$$

This, however, contradicts that $\text{co}_X(v_s)(\bar{x}) = \bar{a} + \bar{\beta}^\top \bar{x} > q_s^\top \bar{y}_s$ with $(\bar{x}, \bar{y}_s) \in \text{conv}(\mathcal{F}_s)$. We conclude that $\varphi_s(x) = \text{co}_X(v_s)(x)$ for all $x \in \text{conv}(X)$. □

Proof of Theorem 2. Let $s = 1, \dots, S$ be given, and consider the convex function $c^\top x + \text{co}(v_s)(x)$, with

$$\text{co}(v_s)(x) = \min_{y_s} \left\{ q_s^\top y_s : (x, y_s) \in \text{conv}(\mathcal{F}_s) \right\}, \qquad x \in \text{conv}(X),$$

by Lemma 1. Let slope $-\pi_s$ be given, and define $\mathcal{L}_s(\pi_s)$ for all π_s as the largest intercept so that $\mathcal{L}_s(\pi_s) - \pi_s^\top x$ is a lower bound for $\text{co}_X(v_s)(x)$ on $\text{conv}(X)$. Then, we require that $\mathcal{L}_s(\pi_s)$ is the largest value such that $\mathcal{L}_s(\pi_s) \leq c^\top x + \text{co}(v_s)(x) + \pi_s^\top x$ for all $x \in \text{conv}(X)$. Hence,

$$\begin{aligned}
\mathcal{L}_s(\pi_s) &= \min_{x \in \text{conv}(X)} \left\{ c^\top x + \min\{ q_s^\top y_s : (x, y_s) \in \text{conv}(\mathcal{F}_s) \} + \pi_s^\top x \right\} \\
&= \min_{x, y_s} \left\{ c^\top x + q_s^\top y_s + \pi_s^\top x : (x, y_s) \in \text{conv}(\mathcal{F}_s) \right\}.
\end{aligned}$$

By maximizing over all supporting hyperplanes, we conclude that for all $x \in \text{conv}(X)$, it holds that $c^\top x + \text{co}_X(v_s)(x) = \sup_{\pi_s} \{ \mathcal{L}_s(\pi_s) - \pi_s^\top x \}$. □

Proof of Corollary 1. It follows directly from Theorem 2 and (15) that

$$\begin{aligned}\bar{\eta}_{LC} &= \min_{x\in X} \sum_{s=1}^{S} p_s\Big(c^\top x + \mathrm{co}_X(v_s)(x)\Big) \\ &= \min_{x\in X} \Big\{c^\top x + \sum_{s=1}^{S} p_s \mathrm{co}_X(v_s)(x)\Big\}.\end{aligned}$$

Next, applying Lemma 1 to $\mathrm{co}_X(v_s)(x)$, we conclude by definition of $\bar{\eta}$ in (4) that $\bar{\eta}_{LC} = \bar{\eta}$. □

Proof of Proposition 3. For every $s = 1, \ldots, S$

$$\mathrm{recc}(\mathrm{Hypo}((\mathcal{L}_s)) = \Big\{(\pi_s, \eta_s) : \eta_s \le \pi_s^\top x_s^k \quad \forall k \in K_s\Big\}, \tag{38}$$

since in general for any polyhedron $\mathcal{P} = \{z \in \mathbb{R}^n : Az \le b\}$, it holds that $\mathrm{recc}(\mathcal{P}) = \{z \in \mathbb{R}^n : Az \le 0\}$. Since X is compact, it has finitely many extreme points $\bar{x}^k$, $k \in \bar{K}$. Moreover, for any such vertex $\bar{x}$ of $\mathrm{conv}(X)$ and for every $s = 1, \ldots, S$, there exists $k \in K_s$ such that $\bar{x} = x_s^k$. Hence, for every $s = 1, \ldots, S$,

$$\mathrm{recc}(\mathrm{Hypo}(\mathcal{L}_s)) \subseteq \Big\{(\pi_s, \eta_s) : \eta_s \le \pi_s^\top \bar{x}^k \quad \forall k \in \bar{K}\Big\}. \tag{39}$$

In fact, the inclusion in (39) holds with equality since any remaining inequalities in (38) are redundant. Indeed, if x_s^κ, $\kappa \in K_s$, is not an extreme point of $\mathrm{conv}(X)$, then it can be written as a convex combination of those extreme points. That is, there exist $\lambda_k \ge 0$, $k \in \bar{K}$, with $\sum_{k\in\bar{K}} \lambda_k = 1$ such that $x_s^\kappa = \sum_{k\in\bar{K}} \lambda_k \bar{x}^k$. The weighted sum over the constraints $\eta_s \le \pi_s^\top \bar{x}^k$ with weights λ_k yields

$$\sum_{k\in\bar{K}} \lambda_k \eta_s \le \sum_{k\in\bar{K}} \lambda_k \pi_s^\top \bar{x}^k \qquad \Leftrightarrow \qquad \eta_s \le \pi_s^\top x_s^\kappa.$$

The desired result follows since the robust constraint $\eta_s \le \pi_s^\top \bar{x}^k$ for all $k \in \bar{K}$, and $\eta_s \le \pi_s^\top x$ for all $x \in \mathrm{conv}(X)$, are equivalent. □