DECOMPOSITION METHODS FOR GLOBAL SOLUTION OF MIXED-INTEGER LINEAR PROGRAMS *

KAIZHAO SUN †, MOU SUN ‡, AND WOTAO YIN §

Abstract. This paper introduces two decomposition-based methods for two-block mixed-integer linear programs (MILPs), which aim to take advantage of separable structures of the original problem by solving a sequence of lower-dimensional MILPs. The first method is based on the ℓ_1-augmented Lagrangian method (ALM), and the second one is based on a modified alternating direction method of multipliers (ADMM). In the presence of certain block-angular structures, both methods create parallel subproblems in one block of variables, and add nonconvex cuts to update the other block; they converge to globally optimal solutions of the original MILP under proper conditions. Numerical experiments on three classes of MILPs demonstrate the advantages of the proposed methods on structured problems over the state-of-the-art MILP solvers.

Key words. Mixed-integer linear programs, decomposition method, augmented Lagrangian method, alternating direction method of multipliers

AMS subject classifications. 49M27, 90C11, 90C26

1. Introduction.

We consider the generic two-block mixed-integer linear program (MILP):

$$p^* := \min_{x,z}\{c^\top x + g^\top z : \ Ax + Bz = 0,\ x \in X,\ z \in Z\}, \tag{1.1}$$

with decision variables $x \in \mathbb{R}^n$ and $z \in \mathbb{R}^d$, rational parameters $c \in \mathbb{Q}^n$, $g \in \mathbb{Q}^d$, $A \in \mathbb{Q}^{m\times n}$, and $B \in \mathbb{Q}^{m\times d}$, and compact sets $X \subset \mathbb{R}^n$ and $Z \subset \mathbb{R}^d$. Interesting to us is when certain entries of x and z must be integers. The use of the zero vector in $Ax + Bz = 0$ has no loss of generality. [1].

Despite the fast development of general-purpose MILP solvers over the past few decades [1, 10], large MILPs still pose nontrivial challenges. This paper develops parallelizable methods for MILP (1.1) with a decomposable structure.

1.1. Decomposable Structure.

Consider the classic block-angular structure:

$$[A|B] = \left[\begin{array}{ccc|c} A_1 & & & B_1 \\ & \ddots & & \vdots \\ & & A_P & B_P \end{array}\right], \quad X = X_1 \times \cdots \times X_P, \tag{1.2}$$

where A is a block-diagonal matrix, rows of B are divided into groups accordingly, and the constraint $x \in X$ reduces to $x_1 \in X_1,\ \ldots,\ x_P \in X_P$. This structure naturally arises in two-stage optimization problems [7], where the first-stage variable z couples with the second-stage variables x_i via $A_i x_i + B_i z = 0$ in each scenario $i \in [P] := \{1, \cdots, P\}$. In decentralized optimization [61], the consensus between neighboring agents i and j in a graph is written as $x_i = z_{ij}$ and $x_j = z_{ij}$, which is a special case of (1.2).

*Submitted to the editors DATE.

Funding: Todo.

†School of ISyE, Georgia Institute of Technology (ksun46@gatech.edu).

‡Decision Intelligence Lab, DAMO Academy, Alibaba Group (mou.sunm@alibaba-inc.com).

§Decision Intelligence Lab, DAMO Academy, Alibaba Group (US) (wotao.yin@alibaba-inc.com).

[1] Any $Ax + Bz = b$ can be equivalently expressed as $Ax + \tilde{B}\tilde{z} = 0$ where $\tilde{B} = [B, b]$ and $\tilde{z} = [z^\top, -1]^\top$.

1.2. ALM and ADMM. When X and Z are convex, we can solve (1.1) by the augmented Lagrangian method (ALM) [33, 49] and the alternating direction method of multipliers (ADMM) [28, 31]. Define the augmented Lagrangian (AL) function:

$$L(x,z,\lambda,\rho) := c^\top x + g^\top z + \langle \lambda, Ax+Bz\rangle + \rho\sigma(Ax+Bz),\ \lambda\in\mathbb{R}^m,\ \rho>0. \tag{1.3}$$

Classic ALM and ADMM use the penalty function $\sigma(\cdot)=\frac{1}{2}\|\cdot\|_2^2$. The standard ALM is the iteration:

$$(x^{k+1},z^{k+1}) \in \underset{x\in X, z\in Z}{\operatorname{Argmin}} L(x,z,\lambda^k,\rho^k), \tag{1.4a}$$

$$\lambda^{k+1} = \lambda^k + \rho^k(Ax^{k+1}+Bz^{k+1}),\quad \text{update } \rho^{k+1} \text{ if needed.} \tag{1.4b}$$

This method cannot directly solve (1.1) or take advantage of structure (1.2). To make the former possible, a function $\sigma(\cdot)$ supporting the exact–penalty property is required. This paper will introduce ways to achieve the latter.

The standard ADMM is the iteration:

$$x^{k+1} \in \underset{x\in X}{\operatorname{Argmin}} L(x,z^k,\lambda^k,\rho), \tag{1.5a}$$

$$z^{k+1} \in \underset{z\in Z}{\operatorname{Argmin}} L(x^{k+1},z,\lambda^k,\rho), \tag{1.5b}$$

$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1}+Bz^{k+1}). \tag{1.5c}$$

If structure (1.2) is present, then the step (1.5a) decomposes into p independent lower-dimensional subproblems. However, (1.5) can diverge or only converge to an infeasible solution for (1.1).

1.3. Our Approach. This paper introduces two methods for (1.1) that can yield parallel subproblems by utilizing (1.2), i.e., they solve a set of independent, parallel MILP subproblems, each involving one $x_i \in X_i$. The methods also offer a decomposition between x and z when $P=1$. Both methods converge to global solutions of (1.1) under proper conditions. Below we present them at a high level.

We use the ℓ_1 penalty $\sigma(\cdot)=\|\cdot\|_1$ in AL (1.3) due to its exact penalization property and component-wise separability. Define the *AL dual function* and the *AL dual problem* for (1.1) as:

$$d(\lambda,\rho) := \min_{x\in X, z\in Z} L(x,z,\lambda,\rho), \tag{1.6}$$

$$\sup_{\lambda\in\mathbb{R}^m,\rho>0} d(\lambda,\rho). \tag{1.7}$$

Note that penalty parameter ρ is not fixed in (1.7). Feizollahi et al. [26] establishes that the primal (1.1) and the dual (1.7) problems have the same optimal objective.

Our *first method* solves (1.7) based on ALM:

Step 1: solve (1.6) by subroutine AUSAL : $(x^{k+1},z^{k+1}) \leftarrow \texttt{AUSAL}(\lambda^k,\rho^k,\epsilon)$;

Step 2: compute λ^{k+1},ρ^{k+1}.

AUSAL stands for "**A**lternating **U**pdate for the **S**harp **AL** function", where "sharp" refers to the ℓ_1 penalty. AUSAL is itself an iterative algorithm, alternating between the updates of x and z:

Step 1a: let $x^{t+1} \in \underset{x\in X}{\operatorname{Argmin}} L(x,z^t,\lambda^k,\rho^k)$;

Step 1b: add a $\texttt{ReverseNormCut}(x^{t+1},z^t)$ to z-subproblem and solve for z^{t+1}.

The x-subproblems and z-subproblems are MILPs in x and z, respectively. A *reverse norm cut* is a nonconvex minorant, which we borrow from global Lipschitz minimization [44]. At each iteration, we add a cut to the z-subproblem, making the subproblem closer to (1.6), so that the AUSAL iterations asymptotically solve (1.6). When a certain optimality gap falls under a tolerance $\epsilon > 0$, the iterations stop and return a pair (x^{k+1}, z^{k+1}) to complete ALM's Step 1. The pair (x^{k+1}, z^{k+1}) is then used to update $(\lambda^{k+1}, \rho^{k+1})$ in Step 2. Overall, the first method uses double loops, and each outer iteration calls AUSAL in Steps 1a-1b. Note that the reverse norm cuts generated in one call to AUSAL become useless after λ and ρ change, so we cannot use them in the next outer iteration. Details of this method are given in Section 3.

Our *second method* uses a single loop and a different kind of cut, which stays valid across iterations. The method resembles ADMM, cycling through the updates of x, z, and Lagrange multipliers and penalty parameter:

Step 1: let $x^{k+1} \in \operatorname*{Argmin}_{x \in X} L(x, z^k, \mu^k, \beta^k)$;

Step 2: add an $\texttt{ALCut}(x^{k+1}, z^k, \mu^k, \beta^k)$ to z-subproblem and solve for z^{k+1};

Step 3: compute μ^{k+1}, β^{k+1}.

The x-subproblems and z-subproblems are MILPs in x and z, respectively. We add *AL cuts* (despite its name, we use them only in this ADMM-like method) so that the z-subproblems' objectives approximate a certain value function with increasing accuracies. The dual pair in this method is denoted by (μ, β) instead of (λ, ρ) in the first method. Details of the second method are given in Section 4.

Compared to existing ALM and ADMM methods for convex and nonlinear optimization, our new treatments include: adding cuts to z-subproblems, properly updating penalty parameters, and different convergence analyses. Next, we review the related methods and compare them to ours.

1.4. Related Works.

1.4.1. Augmented Lagrangian Decomposition and Challenges. The classic ALM uses the squared penalty $\sigma(\cdot) = \frac{1}{2}\|\cdot\|_2^2$. Its convergence is well-established for convex programs [53, 54] and smooth nonlinear programs [5, 8]. We focus on works where X and Z contain discrete variables.

With MILP (1.1) and nonconvex nonsmooth problems, ALM algorithms work under the theory of nonconvex AL duality [30, 34, 55]. In particular, the theory states that $d(\lambda, \rho)$ in (1.6) is concave and upper-semicontinuous in (λ, ρ) [55, Exercise 11.56], so it can be maximized by a method for nonsmooth convex optimization. Specifically, works [14, 15, 16] introduce modified subgradient methods, and [23] describes an inexact bundle method. When (1.6) has a solution, dual convergence holds under proper conditions; convergence of the primal iterates can also be established. However, the challenge lies in solving (1.6) with its nonconvex mixed-integer constraint sets X or Z. All the methods in [14, 15, 16, 23] assume a (nearly) exact oracle of (1.6), that is, finding a (nearly) global minimizers of (1.6). Paper [23] proposes to alternatively update x and z variables, but this approach may get stuck at a non-stationary point due to the nonsmooth term $\|Ax + Bz\|_1$ in (1.3); see [69] for an example. For the special case of (1.1) with $B = I$, $g = 0$, and Z being a linear subspace, work [13] applies a proximal ALM to its convex relaxation, but it cannot guarantee primal feasibility. In comparison, our first method ensures solving (1.6) and thus finding a global solution to MILP (1.1) under proper conditions.

1.4.2. ADMM and Challenges. Since ADMM updates x and z separately, it tends to have cheaper updates than ALM and, further thanks to the structure (1.2), ADMM generates smaller parallelizable x_i-updates. Although ADMM is structually similar to ALM, its convergence analysis is different and trickier than that of ALM.

ADMM and its variants under convexity have matured in the last decade [25, 27, 46]. Their analyses are based on assembling convex inequalities, which clearly fail to hold for MILP subproblems. There is little we can borrow there to develop our method. Recently, ADMMs for discrete and mixed-integer problems started to appear. Paper [71] reformulates a vehicle routing problem as a multi-block MILP and applies ADMM with the typical quadratic penalty. Thanks to their binary variables, their mixed-integer quadratic program (MIQP) subproblems reduce to MILPs. The formulations studied in [4, 38, 39, 63, 64, 70] can all reduce to the form $\min_x\{f(x) \mid x \in X \cap Z\}$, where f and X are continuous and but Z is a discrete set. Introduce variable z for the reformulation $\min_{x,z}\{f(x) \mid x = z, x \in X, z \in Z\}$, and we can apply ADMM. The x-subproblem is convex (except in [39], where it is a nonlinear program), and the z-subproblem reduces to a projection onto a discrete set, which may admit a closed-form formula. These works report encouraging numerical results and discuss penalty parameter tuning and restart heuristics. Unfortunately, none of them comes with convergence guarantees to global solutions. In fact, there are examples on which ADMM either diverges or converges to local or infeasible solutions.

ADMMs for nonconvex continuous programs have also been studied [36, 45, 62, 67]. They show convergence to certain stationary points for problems satisfying certain *structural conditions*. For example, [67] assumes 1) $\mathrm{Im}(A) \subseteq \mathrm{Im}(B)$, and 2) $Z = \mathbb{R}^m$ (they fail to hold for the above reformulation when $Z \neq \mathbb{R}^m$). Paper [68] introduces an ADMM to solve problems involving binary constraints $x \in \{0,1\}^n$, which is reformulated as the intersection of $[0,1]^n$ and a shifted ℓ_p-sphere. The resulting ADMM converges to a stationary point of a perturbed problem that satisfies the aforementioned two conditions. Compared to the existing ADMMs, our second method is the first attempt towards obtaining global solutions of MILPs.

1.4.3. Two-stage stochastic MILP. Two-stage stochastic programs give rise to the block-angular structure (1.2). A recurring theme is Benders decomposition [7, 66]. In a typical iteration, given the first-stage variable z, the second-stage problem, or recourse problem, in x is solved, and valid inequalities, or *Benders cuts*, are generated to approximate the first stage's value function, $Q(z) = \min_{x \in X}\{c^\top x : Ax = -Bz\}$. When the problem has continuous recourse, i.e., when X is a polyhedron, Benders cuts are sufficient to guarantee convergence [37] due to the convexity and finite piece-wise linearity of Q. When X contains discrete decisions, Q is not convex or continuous in general [11]. Many works study binary Z [6, 29, 40, 47, 58, 59, 60], and establish finite convergence. When Z contains general mixed-integer variables, one usually needs to invoke a customized branch-and-cut algorithm for the first-stage problem [3, 12, 18, 20, 21, 22, 50] or introduce additional variables to formulate certain nonlinear approximation of the value function [2, 19]. The two approaches share some similarities. In theory, both linear and nonlinear cuts are trying to approximate the value function, and in practice, both approaches rely on the fast development of general-purpose MILP solvers and integration with modeling languages. The first one is usually implemented with solvers' callback functions, and the second one calls solvers a series of times. Our proposed methods take the second approach. Other approaches are also used to solve the first-stage problem, including heuristics, constraint program, and column generation; we refer the interested reader to the survey [52].

Besides Benders cuts, there are cuts based on the Lagrangian duality for mixed-integer recourse. Zou et al. [73] proposed Lagrangian cuts and strengthened Benders cuts for multistage stochastic programs with binary state variables. Chen and Luedtke [22] proposed a normalization perspective for generating Lagrangian cuts and accelerated the cut generation process by focusing on a restricted subspace. Benders cuts and Lagrangian cuts have also been combined [51] and extended to solve mixed-integer nonlinear programs [41, 42].

More recently, nonlinear cuts based on the augmented Lagrangian duality have been studied . Ahmed et al. [2] proposed reverse norm cuts and AL cuts to approximate Lipschitz value functions. Reverse norm cuts date back to global Lipschitz minimization [44], and AL cuts are derived from exact penalization established by Feizollahi et al. [26]. Later, Zhang and Sun [72] used the exact penalization property as a workaround to the complete recourse condition; they further proposed generalized conjugacy cuts, which unify reverse norm cuts and AL cuts, and studied iteration complexities of stochastic dual dynamic programs (SDDP).

Our proposed methods are motivated by [2, 72]. The proposed AUSAL subroutine applies reverse norm cuts to the augmented Lagrangian function, and the proposed ADMM variant uses AL cuts. Our iteration complexity analysis is inspired by [72]. We highlight some key differences and our contributions in the next subsection.

Another related work [65] uses *scaled cuts*, a class of linear cuts derived by scaling AL cuts, within the Benders framework. The sequence of scaled cut closures converges to the convex envelope of the value function. Despite promising theoretical properties, computing scaled cuts is tricky, involving the combination of fixed-point iterations with a row generation scheme or cutting plane techniques. To speed up convergence, a heuristic was proposed to generate cuts that are only locally valid.

1.5. Contributions. Building on and extending the tools developed in [2, 72], we propose two algorithms for general two-block MILPs that take advantage of the structure (1.2) with convergence guarantees to globally optimal solutions.

The first algorithm is an ALM where each subproblem is approximately solved by AUSAL, for which we give iteration complexity estimates. We further describe two approaches to obtain an approximate global solution of MILP (1.1): (i) applying AUSAL to the penalty formulation, i.e., keeping $\lambda = 0$ in (1.6) and increasing the penalty parameter to reach exact penalty, and (ii) using AUSAL as an oracle to solve the augmented Lagrangian dual problem and updating both Lagrange multipliers and the penalty parameter. Under (ii), we provide two variants of the subgradient method.

Our second algorithm is a variant of ADMM and utilizes AL cuts. Works [2, 72] both need the exact solutions to the augmented Lagrangian dual problem in x, which requires solving a sequence of augmented Lagrangian relaxations in every iteration. In contrast, we generate an AL cut by solving a *single* augmented Lagrangian relaxation in variable x in each iteration; see Section 2.4 for more details. Our ADMM approach is similar to the strengthened augmented Benders cut briefly described in [2, Section 4.1]; however, they did not update dual information, and their analysis cannot explain its convergence. We fill this gap by (i) giving conditions regarding the sequence of multipliers and penalty parameter that are sufficient for convergence, and (ii) establishing a finite convergence to an ϵ-solution of MILP (1.1). Our analysis of ADMM using AL cuts appears to be new, and our method also generalizes ADMM from convex and nonlinear optimization to discrete optimization.

Both methods take advantage of (1.2) to decompose the x-subproblem into smaller independent x_i-subproblems, which are amenable for parallel computing. We conduct

numerical experiments on three classes of MILPs to evaluate the practical performance of the proposed methods. Surprisingly, they exhibit advantages on structured problems over the state-of-the-art MILP solvers.

A disadvantage of our methods is that the z-subproblem grows in size as more cuts are added, making it increasingly difficult to solve. This is a common issue in cut-based MILP methods. We believe, however, we can alleviate this issue by generating more efficient cuts, applying row generation, or using other means to solve the z-subproblem efficiently. In fact, our numerical experiments suggest that this issue is manageable when the dimension of Z is mild. Note that we do not see the proposed methods as a replacement for MILP solvers but a means to scale them to larger MILP instances, especially for those with block separable substructures.

1.6. Notation and Organization. We let $\mathbb{R}$, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{N}$ denote the sets of real, integer, rational, and natural numbers. Write $[P] = \{1, \cdots, P\}$. For a vector $x \in \mathbb{R}^n$, use $\|x\|_p$ as the ℓ_p-norm of x for $1 \le p \le \infty$. The inner product of $x, y \in \mathbb{R}^n$ is denoted by $\langle x, y\rangle$ or $x^\top y$. For a matrix $A \in \mathbb{R}^{m\times n}$, $\|A\|_p$ denotes its induced (operator) norm, and $\mathrm{Im}(A)$ denotes its column space. We introduce $\overline{B}_p(x; R) = \{y \in \mathbb{R}^n : \ \|x - y\|_p \le R\}$ and $D_p(X) = \sup\{\|x - y\|_p : \ x, y \in X\}$. We call a set *mixed-integer-linear (MIL) representable* if it can be described by a finite number of mixed-integer variables and linear constraints; a function is MIL representable if its epigraph is MIL representable.

We state our assumption on problem (1.1) and provide a more detailed review of background materials in Section 2. We introduce the proposed ALM framework in Section 3 and the ADMM variant in Section 4, together with their convergence results. In Section 5, we discuss implementation issues and present numerical experiments. Finally, we give some concluding remarks in Section 6.

2. Preliminaries.

2.1. Assumptions and Approximate Solution. Throughout this paper, we make the following assumption on MILP (1.1).

Assumption 2.1. Problem (1.1) is feasible, and constraint sets X and Z in (2.1) are compact and mixed-integer representable, i.e.,

$$X = \{x \in \mathbb{R}^{n_1}_+ \times \mathbb{Z}^{n-n_1}_+ : \ Ex = f\}, \quad Z = \{z \in \mathbb{R}^{d_1}_+ \times \mathbb{Z}^{d-d_1}_+ : \ Gz = h\}, \tag{2.1}$$

for rational matrices (E, G) and vectors (f, h) of proper dimensions.

We measure the accuracy of an approximate solution as follows.

Definition 2.2. *Let $\epsilon > 0$. We say (x^*, z^*) is an ϵ-solution of the MIP (1.1) if $x^* \in X$, $z^* \in Z$, $c^\top x^* + g^\top z^* \le p^* + \epsilon$, and $\|Ax^* + Bz^*\|_1 \le \epsilon$.*

Note that infeasibility is measured in the ℓ_1-norm due to our analysis. Since (x^*, z^*) may be infeasible, it is possible that $c^\top x^* + g^\top z^* < p^*$.

2.2. Exact Penalization. Since the AL dual problem (1.7) has weak duality $\sup_{\lambda\in\mathbb{R}^m,\rho\ge 0} d(\lambda, \rho) \le p^*$, two follow-up questions are: 1) how to obtain strong duality, i.e., for "=" to hold, and 2) how to obtain an optimal primal solution by solving the dual problem? Feizollahi et al. [26] provides positive answers to both questions for MILP. Recall the definition of exact penalization.

Definition 2.3 (Exact penalization [55, Definition 11.60]). *A dual variable $\lambda \in \mathbb{R}^m$ is said to support exact penalization if, for all sufficiently large $\rho > 0$,*

$$\operatorname*{Argmin}_{x\in X, z\in Z}\{c^\top x + g^\top z : \ Ax + Bz = 0\} = \operatorname*{Argmin}_{x\in X, z\in Z} L(x, z, \lambda, \rho).$$

We say a pair (λ, ρ) supports exact penalization if the above equation holds. We simply say ρ supports exact penalization when $(0, \rho)$ does so.

THEOREM 2.4 (Exact penalization for MILP [26]). *Under Assumption 2.1, strong duality holds for (1.7): $\sup_{\lambda\in\mathbb{R}^m,\rho\geq 0} d(\lambda,\rho) = p^*$. For any $\lambda \in \mathbb{R}^m$, there exists $\underline{\rho} > 0$ such that for every $\rho \in [\underline{\rho}, +\infty)$, (λ, ρ) supports exact penalization.*

The results proved in [26] applies to a broader class of penalty functions, including all norms in $\mathbb{R}^m$. This paper focuses on the ℓ_1-norm for component-wise separability. The proximal augmented Lagrangian where $\sigma(\cdot) = \frac{1}{2}\|\cdot\|_2^2$ does not support exact penalization; in general, no finite penalty ρ can close the duality gap [26, Proposition 7]. A complete characterization of exact penalization is given as follows.

THEOREM 2.5 (Criterion for exact penalization [55, Theorem 11.61]). *Suppose Assumption 2.1 holds. The following statements are equivalent:*

1. *The pair (λ, ρ) supports exact penalization.*
2. *The pair (λ, ρ) solves the dual problem (1.7).*
3. *There exists $r > 0$ such that $p(u) \geq p(0) + \langle\lambda, u\rangle - \rho\|u\|_1$ for all $u \in \overline{B}_1(0; r)$, where $p(u) := \min_{x,z}\{c^\top x + g^\top z \mid Ax + Bz + u = 0, x \in X, z \in Z\}$.*

2.3. Nonconvex Cuts. In this subsection, we review the main tools used in our algorithmic development, namely, reverse norm cuts and AL cuts.

2.3.1. Global Lipschitz Minimization and Reverse Norm Cuts. Consider

$$v^* = \min_{z\in Z} f(z) + Q(z), \tag{2.2}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a simple function, e.g., a linear one, and $Q : \mathbb{R}^d \to \mathbb{R}$ is K-Lipschitz with respect to the ℓ_1-norm over the compact set $Z \subseteq \mathbb{R}^d$. In particular, Q can be nonconvex and we only have access to its zero-order oracle, i.e., given $z \in Z$, we can evaluate $Q(z)$. Since Q is K-Lipschitz, given $\bar{z} \in Z$, the following inequality

$$Q(z) \geq r(z; \bar{z}) := Q(\bar{z}) - K\|z - \bar{z}\|_1 \tag{2.3}$$

holds for all $z \in Z$. Due to the term "$-\|z - \bar{z}\|_1$" in its right-hand side, inequality (2.3) is called a reverse norm cut by [2]. The function $r(z; \bar{z})$ is a lower approximation of Q that is tight at $\bar{z}$, i.e., $Q(\bar{z}) = r(\bar{z}; \bar{z})$. Given a set $\bar{Z} \subset Z$, further consider

$$\underline{R}(z; \bar{Z}) := \sup\{r(z; \bar{z}) : \ \bar{z} \in \bar{Z}\}, \tag{2.4}$$

which remains a lower approximation of Q and gets closer to Q as the set $\bar{Z}$ gets larger. If $\bar{Z} = Z$, we have $\underline{R}(z; \bar{Z}) = Q(z)$ over all $z \in Z$. As a result, we can use $\underline{R}$ as an approximation of Q, and iteratively refine $\underline{R}$ by expanding $\bar{Z}$. This idea was firstly proposed by Mayne and Polak [44] in the 80s (to our knowledge) and revisited recently [2, 43]. We formalize the procedure in Algorithm 2.1.

Remark 2.6. We give some remarks regarding Algorithm 2.1.

1. The initial point $z^0 \in Z$ is required only for notation consistency. A pre-solve step where $\underline{R}$ is replaced by a finite lower bound of Q over Z yields z^0.
2. The minimal Lipschitz constant K can be replaced by any over-estimators.
3. Quantities `UB` and $f(x^k) + t^k$ keep track of upper and lower bounds of v^*, respectively. Another termination criteria is to check whether $Q(x^k) - t^k \leq \epsilon$.

Algorithm 2.1 : Global Minimization by Reverse Norm Cuts

1: **Input**: $(z^0, K, \epsilon) \in Z \times \mathbb{R}_{++} \times \mathbb{R}_+$;
2: set $Z_0 \leftarrow \{z^0\}$ and initialize $\texttt{UB} \leftarrow +\infty$;
3: **for** $k = 1, 2, \cdots$ **do**
4: compute $z^k \in \operatorname{Argmin}_{z\in Z} f(z) + \underline{R}(z; Z_{k-1})$, and let $t^k \leftarrow \underline{R}(z^k; Z_{k-1})$;
5: $\texttt{UB} \leftarrow \min\{\texttt{UB}, f(z^k) + Q(z^k)\}$;
6: **if** $\texttt{UB} - f(x^k) - t^k \le \epsilon$ **then**
7: **return** z^k.
8: **end if**
9: $Z_k \leftarrow Z_{k-1} \cup \{z_k\}$;
10: **end for**

4. The problem in line 4 can be cast as

$$\min_{z\in Z}\{f(z) + t : \ t \ge Q(z^j) - K\|z - z^j\|, \ j = 0, \cdots, k-1\}. \tag{2.5}$$

When f and Z are MIL representable, the above problem can be formulated as a MILP by introducing additional $k \times d$ binary variables and $k \times 2d$ nonnegative variables; e.g., see [2].

The convergence of Algorithm 2.1 is stated in the next theorem.

THEOREM 2.7 ([2, 44]). *If $\epsilon > 0$, then Algorithm 2.1 terminates in a finite number of iterations with an ϵ-optimal solution to problem (2.2), i.e., a point $z^k \in Z$ such that $f(z^k) + Q(z^k) \le v^* + \epsilon$. If $\epsilon = 0$, then either Algorithm 2.1 terminates in a finite number of iterations with a global optimal solution to problem (2.2), or the sequence $\{f(z^k) + t^k\}_{k\in\mathbb{N}}$ converges to v^* monotonically from below, and any limit point z^* of $\{z^k\}_{k\in\mathbb{N}}$ is a global optimal solution to problem (2.2).*

In addition, inspired by the analysis in [72], we further derive an iteration complexity estimate for Algorithm 2.1, which complements the results in [2, 44].

THEOREM 2.8. *Let $\epsilon > 0$, and suppose $Z \subseteq \overline{B}_1(\bar{z}; R)$ for some $\bar{z} \in \mathbb{R}^d$ and radius $R > 0$. Then Algorithm 2.1 terminates in no more than $(1 + 4KR\epsilon^{-1})^d$ iterations.*

Proof. See Appendix A.1. ∎

2.4. Augmented Lagrangian Cuts. Augmented Lagrangian cuts were introduced in [2] and generalized in [72]. In this subsection, we review and modify AL cuts in the context of problem (1.1). Consider the penalty formulation of problem (1.1):

$$\min_{z\in Z} g^\top x + R_\rho(z), \tag{2.6}$$

where $R_\rho : \mathbb{R}^d \to \mathbb{R}$ is the optimal value of a partial minimization with respect to x:

$$R_\rho(z) := \min_{x\in X} c^\top x + \rho\|Ax + Bz\|_1. \tag{2.7}$$

Algorithm 2.1 can be applied to solve problem (2.6) by generating reverse norm cuts of R_ρ; we elaborate this idea in Section 3.2. We consider another class of nonconvex cuts, augmented Lagrangian (AL) cuts [2] that can be used to approximate R_ρ.

Let $\mu \in \mathbb{R}^m$ and $\beta \geq 0$, and define

$$P(z, \mu, \beta) := \min_x c^\top x + \langle \mu, Ax + Bz \rangle + \beta \|Ax + Bz\|_1. \tag{2.8}$$

Notice that

$$P(z, \mu, \beta) \leq \min_x c^\top x + (\beta + \|\mu\|_\infty)\|Ax + Bz\|_1 \leq R_\rho(z) \tag{2.9}$$

for all $(\mu, \beta) \in \Lambda(\rho) := \{(\mu, \beta) \in \mathbb{R}^{m+1} \mid \beta \geq 0, \beta + \|\mu\|_\infty \leq \rho\}$. Inequality (2.9) is a *weak duality* result in the sense that $P(z, \mu, \beta)$ provides a lower bound for $R_\rho(z)$ when the pair (μ, β) is constrained in $\Lambda(\rho)$. It turns out that *strong duality* also holds:

$$\begin{aligned} R_\rho(z) &= \max_{(\mu,\beta)\in\Lambda(\rho)} P(z, \mu, \beta) \\ &= \max_{(\mu,\beta)\in\Lambda(\rho)} \min_{x\in X} c^\top x + \langle \mu, Ax + Bz \rangle + \beta \|Ax + Bz\|_1, \end{aligned} \tag{2.10}$$

simply due to the fact that $(0, \rho) \in \Lambda(\rho)$ and $P(z, 0, \rho) = R_\rho(z)$ by definition. Given $\bar{z}$ and $(\bar{\mu}, \bar{\beta}) \in \Lambda(\rho)$, suppose we solve the problem (2.8), then for any $z \in Z$,

$$\begin{aligned} R_\rho(z) \geq & P(z, \bar{\mu}, \bar{\beta}) = \min_{x\in X} f(x) + \langle \bar{\mu}, Ax + Bz \rangle + \bar{\beta}\|Ax + Bz\|_1 \\ \geq & \min_{x\in X} f(x) + \langle \bar{\mu}, Ax + B\bar{z} \rangle + \bar{\beta}\|Ax + B\bar{z}\|_1 + \langle \bar{\mu}, Bz - B\bar{z} \rangle - \bar{\beta}\|Bz - B\bar{z}\|_1 \\ = & P(\bar{z}, \bar{\mu}, \bar{\beta}) + \langle \bar{\mu}, Bz - B\bar{z} \rangle - \bar{\beta}\|Bz - B\bar{z}\|_1. \end{aligned}$$

Therefore, the function defined by

$$\tilde{r}(z; \bar{z}, \bar{\mu}, \bar{\beta}) := P(\bar{z}, \bar{\mu}, \bar{\beta}) + \langle \bar{\mu}, Bz - B\bar{z} \rangle - \bar{\beta}\|Bz - B\bar{z}\|_1 \tag{2.11}$$

is a lower approximation of $R_\rho(z)$.

DEFINITION 2.9. *We call the inequality $R_\rho(z) \geq \tilde{r}(z; \bar{z}, \bar{\mu}, \bar{\beta})$ an augmented Lagrangian (AL) cut at $\bar{z}$ parameterized by $(\bar{\mu}, \bar{\beta})$. We say the cut is tight at z if $R_\rho(z) = \tilde{r}(z; \bar{z}, \bar{\mu}, \bar{\beta})$.*

We note that an AL cut is not necessarily tight since $P(\bar{z}, \bar{\mu}, \bar{\beta}) < R_\rho(\bar{z})$ when $(\bar{\mu}, \bar{\beta}) \in \Lambda(\rho)$ is not optimal for (2.10). The additional linear term $\langle \bar{\mu}, Bz - B\bar{z} \rangle$ corresponds to a rotation around the pivot $(\bar{z}, P(\bar{z}, \bar{\mu}, \bar{\beta})) \in \mathbb{R}^{d+1}$. In addition, since $\|\bar{\mu}\|_\infty + \bar{\beta} \leq \rho$, the AL cut may have a smaller Lipschitz constant than R_ρ. Geometrically, the rotation effect and smaller Lipschitz constant allow an AL cut to be "fatter" than R_ρ and thus covers a wider range in Z than a reverse norm cut. Moreover, a smaller value of $\bar{\beta} + \|\bar{\mu}\|_\infty$ can be preferable for optimization solvers.

Both algorithms in [2, 72] assume the AL cut generated in each iteration is tight in the sense that a pair (μ, β) optimal to the maximization problem in (2.10) is available. In practice, one needs to call a subgradient method to solve the max-min problem (2.10) in a double-looped fashion, and hence in each iteration, multiple MILPs in the form of (2.8) needs to be solved until convergence. In contrast, the proposed ADMM in Section 4 generates an AL cut in iteration k by solving a *single* problem (2.8) with $(\mu, \beta) = (\mu^k, \beta^k)$, and guide convergence through proper updates of (μ^{k+1}, β^{k+1}). We note that the strengthened augmented Benders cut in [2] is also generated by solving a single MILP in x and is implemented to accelerate convergence of multistage SDDP. In the context of SIP, we provide theoretical justification to this computationally favorable scheme in the two-stage case. Compared to the ADMM literature, our method finds global solutions of nonconvex problems with convergence guarantees.

3. An ALM Method empowered by AUSAL. In this section, we introduce an ALM framework for MILP (1.1). In Section 3.1, we present the AUSAL algorithm for subproblem (1.6). In order to find an ϵ-solution of MILP (1.1), AUSAL is further applied to the penalty formulation in Section 3.2 or embedded in the ALM in Section 3.3. We present two variants of ALM based on different subgradient updates.

3.1. AUSAL. Consider the augmented Lagrangian relaxation (1.6):

$$d(\lambda,\rho)=\min_{x\in X,z\in Z}\ c^\top x+g^\top z+\langle\lambda,Ax+Bz\rangle+\rho\|Ax+Bz\|_1,$$

where $\lambda\in\mathbb{R}^m$ and ρ are fixed as constants. We decompose the minimization into two stages: the inner stage minimizes over $x\in X$ with z fixed, and the outer stage minimizes over Z:

$$R(z)=R(z;\lambda,\rho):=\min_{x\in X}\ \langle c+A^\top\lambda,x\rangle+\rho\|Ax+Bz\|_1, \tag{3.1}$$

$$d(\lambda,\rho)=\min_{z\in Z}\ \langle g+B^\top\lambda,z\rangle+R(z). \tag{3.2}$$

Note that $R(z)$ is well defined over $z\in\mathbb{R}^d$ due to the compactness of Z. The function $R(z)$ is known as the value function or cost-to-go function in the context of sequential decision making. We omit the dependency on (λ,ρ) for simplicity in this section. The next lemma suggests that R is Lipschitz.

LEMMA 3.1. *Suppose Assumption 2.1 holds, and X is compact for any right-hand side vector f (possibly empty). Then $R(z)$ is piecewise-linear and K_ρ-Lipschitz continuous with respect to the ℓ_1-norm over $\mathbb{R}^d$, where $K_\rho:=\rho\|B\|_1$.*

Proof. We show $R(z)$ is piecewise linear by considering the standard-form MILP:

$$v(b):=\min_x\left\{c^\top x\,\middle|\,Ax:=\begin{bmatrix}A_1x\\A_2x\end{bmatrix}=\begin{bmatrix}b_1\\b_2\end{bmatrix}=:b,\ x\in\mathbb{R}^p_+\times\mathbb{Z}^q_+\right\}.$$

Let $v(b)=+\infty$ if it is infeasible and $-\infty$ if it is unbounded. Define $D:=\{b\mid v(b)<+\infty\}$. It is shown in [11] that if the MILP is described by rational data and $v(b)>-\infty$ for all $b\in D$, then $v(b)$ is a piecewise linear function over D. Now fix $b_2\in\mathrm{Im}(A_2)$ and define $v_1(b_1)=v([b_1^\top,b_2^\top]^\top)$. Then $v_1(b_1)$ is also piecewise linear over its domain $D_1:=\{b_1\mid[b_1^\top,b_2^\top]^\top\in D\}$. Notice that the premise of [11] is satisfied for problem (3.1), and we can equivalently write $R(z)=\min_{x\in X,u}\{\langle c+A^\top\lambda,x\rangle+\rho\|Ax+Bu\|_1\mid u=z\}$, where z plays the role of b_1. Since the rest of the problem defining $R(z)$ can be cast as a standard MILP, we conclude that $R(z)$ is a piecewise linear function in $\mathbb{R}^d$.

To prove $R(z)$ is Lipschitz, take any $z^1,z^2\in\mathbb{R}^d$, and let x^1,x^2 be the optimal solution to (3.1) with $z=z^1$ and $z=z^2$, respectively. It holds that $R(z^1)-R(z^2)\le\langle c+A^\top\lambda,x^2\rangle+\rho\|Ax^2+Bz^1\|_1-\langle c+A^\top\lambda,x^2\rangle-\rho\|Ax^2+Bz^2\|_1\le\rho\|B\|_1\|z^1-z^2\|_1$, where the first inequality is due to the optimality of x^1 in the definition of $R(z^1)$, and the second inequality is by the triangle inequality. Similarly we have $R(z^2)-R(z^1)\le\rho\|B\|_1\|z^1-z^2\|_1$, which concludes the proof with the claimed modulus K_ρ. ∎

As a result, we can apply Algorithm 2.1 to solve problem (1.6), and we name the resulting scheme Alternating Update for the Sharp Augmented Lagrangian function, dubbed AUSAL. See Algorithm 3.1.

Remark 3.2. In view of the block-angular structure (1.2), the evaluation of $R(z)$ can be decomposed into P parallel subproblems thanks to the component-wise separability of ℓ_1-norm.

Algorithm 3.1 : AUSAL

1: **Input** $(\lambda, \rho, \epsilon) \in \mathbb{R}^m \times \mathbb{R}_{++} \times \mathbb{R}_+$;
2: initialize $f(\cdot) \leftarrow \langle g + B^\top \lambda, \cdot \rangle$, $Q(\cdot) \leftarrow R(\cdot)$, $z_0 \in Z$, and $K_\rho \leftarrow \rho \|B\|_1$;
3: compute $z^* \in Z$ by calling Algorithm 2.1 with input (z_0, K_ρ, ϵ);
4: **return** (x^*, z^*) where $x^* \in X$ is a minimizer in (3.1) with $z = z^*$.

We immediately have the following corollary of Theorem 2.8.

COROLLARY 3.3. *Suppose Assumption 2.1 holds, $\epsilon > 0$, and $Z \subseteq \overline{B}_1(\bar{z}; R)$ for some $\bar{z} \in \mathbb{R}^d$ and $R > 0$. AUSAL terminates in no more than $(1 + 4\rho\|B\|_1 R\epsilon^{-1})^d = \mathcal{O}(\rho^d \epsilon^{-d})$ iterations*

Remark 3.4. We do not expect this upper bound in to be practically informative since it depends exponentially on the dimension of Z. The results indicates that ASUAL is a parallelizable algorithm with finite convergence to some ϵ-solution.

3.2. Penalty Approach. We present a penalty method that solves the original problem (1.1) using AUSAL as a subroutine. We first present a helpful lemma.

LEMMA 3.5. *Let $(\lambda, \rho) \in \mathbb{R}^m \times \mathbb{R}_{++}$ satisfy $\rho \geq \|\lambda\|_\infty$, and $(\bar{x}, \bar{z})$ be an ϵ-solution to the augmented Lagrangian relaxation $\min_{x \in X, z \in Z} L(x, y, \lambda, \rho)$. Then we have the following chain of inequalities:*

$$\begin{aligned} c^\top \bar{x} + g^\top \bar{x} \leq & c^\top \bar{x} + g^\top \bar{x} + (\rho - \|\lambda\|_\infty)\|A\bar{x} + B\bar{z}\|_1 \\ \leq & c^\top \bar{x} + g^\top \bar{x} + \langle \lambda, A\bar{x} + B\bar{z} \rangle + \rho \|A\bar{x} + B\bar{z}\|_1 \leq p^* + \epsilon. \end{aligned} \tag{3.3}$$

Proof. The inequalities are due to $\rho \geq \|\lambda\|_\infty$, Hölder's inequality applied to $\langle \lambda, A\bar{x} + B\bar{z} \rangle$, and $(\bar{x}, \bar{z})$ being ϵ-optimal for (1.6) and $d(\lambda, \rho) \leq p^*$, respectively. ∎

Suppose that we have a pair (λ, ρ) that supports exact penalization; then we only need to call AUSAL to solve the augmented Lagrangian relaxation (1.6) once. Our first result suggests that, with such a pair (λ, ρ), we can expect to obtain an ϵ-solution of (1.1) upon the termination of AUSAL.

THEOREM 3.6. *Suppose $(\lambda, \rho) \in \mathbb{R}^{m+1}$ satisfies that 1) $\rho \geq \|\lambda\|_\infty$, and 2) $(\lambda, \rho - 1)$ supports exact penalization. Then AUSAL applied to the augmented Lagrangian relaxation (1.6) returns an ϵ-solution of MILP (1.1).*

Proof. Let $(\bar{x}, \bar{z})$ denote the solution returned by Algorithm 3.1, which is ϵ-optional to the augmented Lagrangian relaxation $\min_{x \in X, z \in Z} L(x, y, \lambda, \rho)$. By Lemma 3.5, we have $c^\top \bar{x} + g^\top \bar{x} \leq p + \epsilon$. Furthermore, the last inequality in (3.3) implies that $\rho\|A\bar{x} + B\bar{z}\|_1 \leq p^* + \epsilon - c^\top \bar{x} - g^\top \bar{x} - \langle \lambda, A\bar{x} + B\bar{z} \rangle$; since $(\lambda, \rho - 1)$ also supports exact penalization, it holds that $p^* \leq c^\top \bar{x} + g^\top \bar{x} + \langle \lambda, A\bar{x} + B\bar{z} \rangle + (\rho - 1)\|A\bar{x} + B\bar{z}\|_1$. The above two inequalities together imply $\|A\bar{x} + B\bar{z}\|_1 \leq \epsilon$. ∎

Since the current (λ, ρ) may not support exact penalization, we must address the question: given any $\lambda \in \mathbb{R}^m$, what is the sufficiently large ρ for AUSAL to deliver an ϵ-solution of MILP (1.1). Note that $\lambda = 0$ is possible, and if this is the case, the primal subproblem (1.6) becomes the penalty formulation of (1.1).

THEOREM 3.7. *Let $\lambda \in \mathbb{R}^m$ and $\epsilon > 0$. Then AUSAL applied to the primal subproblem (1.6) with $\rho = (\|c\|_1 D_\infty(X) + \|g\|_1 D_\infty(Z))\,\epsilon^{-1} + \|\lambda\|_\infty + 1$ returns an ϵ-solution $(\bar{x}, \bar{z})$ of the MILP (1.1) where $D_1(\cdot)$ returns the diameter of the argument, and R is the radius of Z. In particular, AUSAL terminates in at most $(1 + 4\rho\|B\|_1 R\epsilon^{-1})^d = \mathcal{O}(\epsilon^{-2d})$ iterations.*

Proof. By (3.3) and Lemma 3.5, $(\rho - \|\lambda\|_\infty)\|A\bar{x} + B\bar{x}\|_1 \le p^* + \epsilon - c^\top \bar{x} - g^\top \bar{z}$ and $c^\top \bar{x} + g^\top \bar{x} \le p + \epsilon$. Let (x^*, z^*) be an optimal solution of MILP (1.1), then

$$\|A\bar{x} + B\bar{x}\|_1 \le \frac{c^\top(x^* - \bar{x}) + g^\top(z^* - \bar{z}) + \epsilon}{\rho - \|\lambda\|_\infty} \le \frac{\|c\|_1 D_\infty(X) + \|g\|_1 D_\infty(Z) + \epsilon}{\rho - \|\lambda\|_\infty}.$$

It is straightforward to verify the choice of ρ ensures $\|Ax^* + Bz^*\|_1 \le \epsilon$, and finally the complexity result is a direct consequence of Corollary 3.3. ∎

The choice of ρ in Theorem 3.7 serves as a sufficient condition for AUSAL to deliver an ϵ-solution of (1.1) given any $\lambda \in \mathbb{R}^m$. However, we observed that a constant large penalty $\rho = \mathcal{O}(\epsilon^{-1})$ numerically slows down the algorithm, even in its early stages. Hence in practice, instead of calling AUSAL just once with some fixed (λ, ρ), one can also update them in an iterative fashion, which we discuss in the next section.

3.3. ALM and Dual Updates. We present an ALM that uses AUSAL for its subproblem and two different subgradient methods for the update of (λ, ρ). This method is appropriate when a pair (λ, ρ) that supports exact penalization is not known initially.

The dual function $d(\lambda, \rho)$ is a concave and upper-semicontinuous function in (λ, ρ). Let $(\bar{x}, \bar{z})$ be a pair of solution returned by AUSAL with input $(\bar{\lambda}, \bar{\rho}, \epsilon)$. Then for all $(\lambda, \rho) \in \mathbb{R}^m \times \mathbb{R}_+$, it is straightforward to see that

$$(-d)(\lambda, \rho) \ge (-d)(\bar{\lambda}, \bar{\rho}) - \begin{bmatrix} A\bar{x} + B\bar{z} \\ \|A\bar{x} + B\bar{z}\|_1 \end{bmatrix}^\top \begin{bmatrix} \lambda - \bar{\lambda} \\ \rho - \bar{\rho} \end{bmatrix} - \epsilon,$$

or equivalently,

$$-\begin{bmatrix} A\bar{x} + B\bar{x} \\ \|A\bar{x} + B\bar{z}\|_1 \end{bmatrix} \in \partial_\epsilon(-d)(\bar{\lambda}, \bar{\rho}). \tag{3.4}$$

When $\epsilon = 0$, the above inclusion yields a convex subgradient. Therefore, we can use the primal information $(\bar{x}, \bar{z})$ to construct an ϵ-subgradient of $(-d)$ at $(\bar{\lambda}, \bar{\rho})$ and apply an inexact subgradient method to solve the dual problem (1.7). The compactness on X and Z ensures that $d(\lambda, \rho)$ is well defined for all $\lambda \in \mathbb{R}^m$ and $\rho \ge 0$; the exact penalization of MILP from Theorem 2.4 guarantees that $\operatorname{Argmax} d(\lambda, \rho)$ is nonempty.

The subgradient method has many variants, such as the exactness of subgradient, choices of step size, and stopping criteria. Below we provide two specific implementations. Since the analysis is standard, we defer the proof in Appendix A only for completeness. Beyond the two implementations below, we can apply methods [14, 15, 16, 23] with AUSAL as a subroutine to solve the dual problem (1.7).

3.3.1. First Subgradient Method and Iteration Complexity. The first subgradient algorithm is presented as Algorithm 3.2. The constant $\epsilon_p \ge 0$ controls the exactness of the subgradient in (3.4). If AUSAL returns (x^k, z^k) with $Ax^k + Bz^k = 0$, (x^k, z^k) is an ϵ_p-solution to MILP (1.1). To study convergence and complexity, we assume $\|Ax^k + Bz^k\|_1 > 0$ for all $k \in \mathbb{N}$, so that Algorithm 3.2 will produce an infinite sequence $\{(\lambda^k, \rho^k)\}_{k \in \mathbb{N}}$. We introduce the following constants for our analysis:

$$D^* := \operatorname*{Argmax}_{\lambda, \rho \ge 0} d(\lambda, \rho), d_0 := \min_{(\lambda, \rho) \in D^*} \|\lambda^1 - \lambda\|_1 + |\rho^1 - \rho|, M := \max_{x \in X, z \in Z} \|Ax + Bz\|_1.$$

Consider the step size α_k chosen as in line 5 of Algorithm 3.2. Other standard choices include $\alpha_k = \epsilon_d/(2M^2)$ or $\alpha_k = \epsilon_d/(2\|Ax^k + Bz^k\|_1^2)$ for some tolerance $\epsilon_d > 0$; their convergence results are similar and thus omitted.

Algorithm 3.2 : A Subgradient Variant with Iteration Estimate on Objective Gap

Input $\epsilon_p \geq 0$;
initialize (λ^1, ρ^1), and a sequence $\{\tau_k\}_{k\in\mathbb{N}}$ such that $\tau_k > 0$, $\tau_k \to 0$, and $\sum_{k\in\mathbb{N}} \tau_k = +\infty$;
for $k = 1, 2, \cdots$ **do**
  $(x^k, z^k) \leftarrow \text{AUSAL}(\lambda^k, \rho^k, \epsilon_p)$;
  set $\alpha_k = \tau_k/(\sqrt{2}\|Ax^k + Bz^k\|_1)$;
  $\lambda^{k+1} \leftarrow \lambda^k + \alpha_k(Ax^k + Bz^k)$, $\rho^{k+1} \leftarrow \rho^k + \alpha_k\|Ax^k + Bz^k\|_1$;
end for

THEOREM 3.8. *The following statements hold.*

1. *Let $\epsilon_d > 0$. Suppose Algorithm 3.2 performs $K = \lceil 2M^2 d_0^2/\epsilon_d^2 \rceil$ iterations with $\tau_k = d_0/\sqrt{K}$ for all $k \in [K]$. It holds that $\min_{k\in[K]} p^* - d(\lambda^k, \rho^k) \leq \epsilon_p + \epsilon_d$.*
2. *Suppose $\epsilon_p = 0$, and the sequence $\{\tau_k\}_{k\in\mathbb{N}}$ also satisfies $0 < \tau_k \leq \tau$ for all $k \in \mathbb{N}$ and $\sum_{k\in\mathbb{N}} \tau_k^2 < +\infty$. Then (λ^k, ρ^k) converges to some $(\lambda^*, \rho^*) \in D^*$.*

Proof. See Appendix A.2. ∎

Remark 3.9. We give some remarks regarding the first case of Theorem 3.8.

1. The outer-level subgradient method can be preferred over solving a single penalty problem with a large $\rho = \mathcal{O}(\epsilon^{-1})$ suggested in Theorem 3.7 when ϵ_d is allowed to be larger than ϵ_p . Suppose $\rho^1 = \frac{d_0}{\sqrt{2K}}$ so that $\rho^k = \frac{kd_0}{\sqrt{2K}}$ for all $k \in [K]$. By Theorem 3.3 and the fact that $K = \mathcal{O}(\epsilon_d^{-2})$, the total number of AUSAL iterations required in Algorithm 3.2 can be bounded by

$$\sum_{k=1}^{K} \mathcal{O}\left[\left(\frac{\rho^k}{\epsilon_p}\right)^d\right] = \mathcal{O}\left[\epsilon_p^{-d}\left(\frac{1}{d+1}K^{d/2+1} + K^{d/2}\right)\right] = \mathcal{O}\left(\epsilon_p^{-d}\epsilon_d^{-d-2}\right), \tag{3.5}$$

   where the first equality is due to

$$\sum_{k=1}^{K}\left(\frac{k}{\sqrt{K}}\right)^d \leq K^{-d/2}\left(\int_1^K x^d\, dx + K^d\right) \leq \frac{1}{d+1}K^{d/2+1} + K^{d/2}.$$

   The $\mathcal{O}$-notation hides some constant dependent on d. Suppose $\epsilon_p = \epsilon \in (0, 1)$, then (3.5) is no worse than $\mathcal{O}(\epsilon^{-2d})$ presented in Theorem 3.7 as long as $\epsilon_d \geq \epsilon^{\frac{d}{d+2}}$. For example, choosing $\epsilon_d = \epsilon^{\frac{d}{2d+4}}$ reduces (3.5) to $\mathcal{O}(\epsilon^{-1.5d})$.
2. If Algorithm 3.2 does not find the optimal dual variable in K iterations, then by part one of Theorem 3.8, we can estimate the duality gap in the objective, and expect the best-so-far iterate (λ^k, ρ^k) to be close to some optimal dual solution (λ^*, ρ^*). Consequently, we can post-process to recover an optimal dual solution as shown in the next corollary.

COROLLARY 3.10. *Suppose Algorithm 3.2 generates a pair (λ^k, ρ^k) that satisfies $\|(\lambda^k, \rho^k) - (\lambda^*, \rho^*)\|_\infty \leq l$ for some $(\lambda^*, \rho^*) \in D^*$. Then $(\lambda^k, \rho^k + 2l)$ supports exact penalization. Applying AUSAL with $\lambda = \lambda^k$, $\rho = \max\{\|\lambda_k\|_\infty, \rho^k + 2l + 1\}$, and $\epsilon > 0$ returns an ϵ-solution of the MILP (1.1).*

Proof. The pair $(\lambda^k, \rho^k + 2l)$ supports exact penalization since, by Theorem 2.5,

there exists $r > 0$ such that for all $u \in \overline{B}_1(0; r)$, it holds that

$$\begin{aligned} p(u) \geq & p(0) + \langle \lambda^*, u \rangle - \rho^* \|u\|_1 \\ \geq & p(0) + \langle \lambda^k, u \rangle - \rho^k \|u\|_1 - \|\lambda^k - \lambda^*\|_\infty \|u\|_1 - |\rho^k - \rho^*| \|u\|_1 \\ \geq & p(0) + \langle \lambda^k, u \rangle - (\rho^k + 2l) \|u\|_1, \end{aligned}$$

where the third inequality is due to $\max\{\|\lambda^k - \lambda^*\|_\infty, \|\rho^k - \rho^*\|_\infty\} \leq l$. The second claim follows directly from Theorem 3.6. ∎

3.3.2. Second Subgradient Method with Finite Convergence to an Approximate Solution. The subgradient method proposed in the previous subsection maximizes the augmented Lagrangian dual function $d(\lambda, \rho)$ using inexact subgradients of $-d$; however, an approximate global solution to the primal problem (1.1) may not be readily available from Algorithm 3.2, and some post-processing step needs to be invoked. In this subsection, we present the second subgradient variant in Algorithm 3.3, which directly returns an ϵ_p-solution in a finite number of calls of AUSAL.

Algorithm 3.3 : A Subgradient Variant with Finite Convergence

Input $\epsilon_p > 0$;
initialize (λ^1, ρ^1) with $\rho^1 \geq \|\lambda^1\|_\infty$, and some $\tau > 0$;
for $k = 1, 2, \cdots$ **do**
  $(x^k, z^k) \leftarrow \text{AUSAL}(\lambda^k, \rho^k, \epsilon_p)$;
  if $\|Ax^k + Bz^k\|_1 \leq \epsilon_p$ **then**
    return (x^k, z^k).
  end if
  set $\alpha_k = \tau / \|Ax^k + Bz^k\|_1$;
  $\lambda^{k+1} \leftarrow \lambda^k + \alpha_k (Ax^k + Bz^k)$, $\rho^{k+1} \leftarrow \max\{\|\lambda^{k+1}\|_\infty, \rho^k + \alpha_k \|Ax^k + Bz^k\|_1\}$;
end for

If Algorithm 3.3 terminates with (x^k, z^k), then $x^k \in X$, $z^k \in Z$, and $\|Ax^k + Bz^k\|_1 \leq \epsilon_p$. Since $\rho^k \geq \|\lambda^k\|_\infty$, lemma 3.5 ensures $c^\top x^k + g^\top z^k \leq p^* + \epsilon_p$. Therefore, (x^k, z^k) is indeed an ϵ_p-solution of the MILP (1.1).

THEOREM 3.11. *Let $\epsilon_p > 0$. Algorithm 3.3 returns an ϵ_p-solution of MILP (1.1) in a finite number of iterations.*

Proof. See Appendix A.3. ∎

4. An ADMM-Based Method. In this section, we present a variant of ADMM that uses AL cuts introduced in Section 2.4. In iteration k, the method first approximately evaluate $R_\rho(z^{k-1})$ by solving the augmented Lagrangian relaxation (2.8) with some fixed (μ^k, β^k). Then an AL cut is generated to the z-subproblem where z^k is computed. Finally, we update (μ^{k+1}, β^{k+1}) and proceed to iteration $k+1$. Recall the AL cut defined in (2.11):

$$\tilde{r}(z; \bar{z}, \bar{\mu}, \bar{\beta}) := P(\bar{z}, \bar{\mu}, \bar{\beta}) + \langle \bar{\mu}, Bz - B\bar{z} \rangle - \bar{\beta} \|Bz - B\bar{z}\|_1.$$

A conceptual ADMM is described in Algorithm 4.1.

Remark 4.1. We give some remarks regarding the ADMM variant.

1. The x-subproblem (4.1) has the same form as in AUSAL, and can be decomposed into P parallel smaller MILPs in view of the block structure (1.2).

Algorithm 4.1 : An ADMM Framework using AL Cuts

1: **Initialize** $(z^0, \mu^1, \beta^1) \in Z \times \mathbb{R}^m \times \mathbb{R}_{++}$;
2: **for** $k = 1, 2, \cdots$ **do**
3: compute

$$x^k \in \operatorname*{Argmin}_{x \in X} c^\top x + \langle \mu^k, Ax + Bz^{k-1} \rangle + \beta^k \|Ax + Bz^{k-1}\|_1; \tag{4.1}$$

4: compute

$$(z^k, t^k) \in \operatorname*{Argmin}_{z \in Z, t \in \mathbb{R}} \{ g^\top z + t : \ t \geq \tilde{r}(z; z^{j-1}, \mu^j, \beta^j) \ \forall j \in [k] \}; \tag{4.2}$$

5: update $(\mu^{k+1}, \beta^{k+1}) \in \mathbb{R}^m \times \mathbb{R}_{++}$;
6: **end for**

2. We do not specify how (μ^{k+1}, β^{k+1}) is updated in Algorithm 4.1. Instead, we present a set of assumptions on the selection of (μ^{k+1}, β^{k+1}) to establish convergence results. Any updates that meet the assumptions will ensure convergence. We provide specific examples and some geometric intuitions in Section 4.1.

One major difference between our ADMM variant and the classic ADMM lies in the z-subproblem. In a traditional ADMM framework, the z-subproblem has the following structure:

$$\min_{z \in Z} \ g^\top z + \left(c^\top x^k + \langle \mu^k, Ax^k + Bz \rangle + \beta^k \sigma(Ax^k + Bz) \right), \tag{4.3}$$

where $\sigma(\cdot) = \frac{1}{2}\|\cdot\|_2^2$ (proximal Lagrangian) or $\sigma(\cdot) = \|\cdot\|_1$ (sharp Lagrangian) is usually used. Recall the original problem is equivalent to $\min_{x \in Z} g^\top z + R_\rho(z)$ for some sufficiently large ρ. Update (4.3) can be viewed as a local search scheme, where in each iteration, $R_\rho(z)$ is replaced by a *local* approximation inside the parenthesis. In contrast, problem (4.2) consists of *global* lower approximation for $R_\rho(z)$, which is refined over iterations. This might shed some light on why the classic ADMM cannot converge to global optimal solutions.

4.1. Assumptions on Dual Variables. Our analysis builds upon the following main requirement on the sequence of dual variables.

Assumption 4.2. Suppose $\underline{\rho} > 0$ supports exact penalization for MILP (1.1), and (μ^k, β^k) are chosen such that

1. $\beta^k - \|\mu^k\|_\infty \geq \underline{\rho}$ for all sufficiently large $k \in \mathbb{N}$;
2. $\beta^k + \|\mu^k\|_\infty \leq \overline{\rho}$ for all $k \in \mathbb{N}$ for some $\overline{\rho} > 0$.

Remark 4.3. We provide some geometric intuition regarding Assumption 4.2.

1. Part 1 avoids too many loose cuts. It guarantees that for sufficiently large k, the peak of the AL cut reaches at least $R_{\underline{\rho}}$. Otherwise the objective of the z-subproblem is always a strict lower bound of p^*.
2. Part 1 can be satisfied if β^k is bounded away from $\|\mu^k\|_\infty$ by some constant. However, we do not want β^k to go to infinity, as the resulting generalized cut is very "slim". This is ensured by part 2. "Slim" cuts are not desirable since

we will need a lot more such cuts to construct a good approximation. The constant $\overline{\rho}$ also appears in the complexity result in Theorem 4.5.

One trivial example of Assumption 4.2 is to set $\mu^k = 0$ and $\beta^k = \underline{\rho}$ for all k, in which case the ADMM variant reduces to AUSAL applied to the penalty problem (Theorem 3.7). Another update scheme, following the classic AL-based methods, is to set

$$\mu^{k+1} = \Pi_{[\underline{\mu},\overline{\mu}]}\left(\mu^k + \beta^k(Ax^k + Bz^k)\right), \tag{4.4}$$

where $\Pi_{[\underline{\mu},\overline{\mu}]}$ denotes the projection onto some hypercube $[\underline{\mu},\overline{\mu}] \subseteq \mathbb{R}^m$, and then let

$$\beta^{k+1} = \min\{\overline{\beta}, \gamma\beta^k\} \text{ or } \beta^{k+1} = \overline{\beta} \text{ for all } k \in \mathbb{N}, \tag{4.5}$$

for some constants $\overline{\beta} > 0$ and $\gamma \geq 1$. Clearly, Assumption 4.2 can be satisfied if $\overline{\beta} \geq \underline{\rho} + \max\{\|\underline{\mu}\|_\infty, \|\overline{\mu}\|_\infty\}$. Updates (4.4) and (4.5) generate a non-trivial AL cut, which admits an additional rotation with a potentially smaller Lipschitz constant compared to a reverse norm cut. Hence the ADMM variant is proposed in the hope that such geometric effects of AL cuts are able to shape the true value functions $R_{\underline{\rho}}$ and $R_{\overline{\rho}}$ faster and cut off regions in Z that do not contain optimal solutions.

We acknowledge that in general it is hard to verify Assumption 4.2 at every iteration since $\underline{\rho}$ is unknown, and hence the ADMM variant is rather a conceptual framework. We provide some convergence properties in the next two theorems when Assumption 4.2 and a slightly stronger version of it can be satisfied. The results aim to justify the usage of this conceptual framework to a reasonable level, where dual variables can be selected in a flexible way. In fact, in our numerical experiments, the projection step in (4.4) is skipped to encourage more diverse AL cuts.

4.2. Convergence and Complexity. Recall p^* is the optimal value of the MILP (1.1) and that $t^k = \max_{j\in[k]}\{\tilde{r}(z^k; z^{j-1}, \mu^j, \beta^j)\}$. We first state the convergence of ADMM in the following theorem.

THEOREM 4.4. *Suppose Assumption 4.2 holds. Let $\{(x^{k+1}, z^k, t^k)\}_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 4.1, and (x^*, z^*) be a limit point of $\{(x^{k+1}, z^k)\}_{k\in\mathbb{N}}$. The following claims hold.*

1. *$\{g^\top z^k + t^k\}_{k\in\mathbb{N}}$ converges to p^* monotonically from below.*
2. *$p^* = g^\top z^* + R_{\underline{\rho}}(z^*) = g^\top z^* + R_{\overline{\rho}}(z^*)$.*
3. *(x^*, z^*) is an optimal solution to MILP (1.1).*

Proof.

1. We firstly prove the sequence $\{g^\top z^k + t^k\}_{k\in\mathbb{N}}$ converges to p^* monotonically from below. Since $g^\top z^k + t^k$ is the optimal value of problem (4.2), whose feasible region is shrinking over $k \in \mathbb{N}$, we know $\{g^\top z^k + t^k\}_{k\in\mathbb{N}}$ is monotone non-decreasing. Since $\|\mu^k\|_\infty + \beta^k \leq \overline{\rho}$, it follows that $(\mu^k, \beta^k) \in \Lambda(\overline{\rho})$ for all for all $k \in \mathbb{N}$, i.e., all AL cuts are valid lower approximations of $R_{\overline{\rho}}(z)$. As a result, we have $g^\top z^k + t^k \leq \min_{z\in Z}\ g^\top z + R_{\overline{\rho}}(z) = p^*$, where the equality is due to $\overline{\rho}$ supports exact penalization. The sequence $\{g^\top z^k + t^k\}_{k\in\mathbb{N}}$ is non-decreasing and bounded from above by p^*, so it converges to some $\bar{p} \leq p^*$. Next let $\{z^{k_j}\}_{j\in\mathbb{N}}$ be a subsequence of $\{z^k\}_{k\in\mathbb{N}}$ convergent to $z^* \in Z$. By the Hölder's inequality and Assumption 4.2, we have

$$\begin{aligned} & P(z^{k_{j-1}}, \mu^{k_{j-1}+1}, \beta^{k_{j-1}+1}) \\ \geq & \min_{x\in X} c^\top x + (\beta^{k_{j-1}+1} - \|\mu^{k_{j-1}+1}\|_\infty)\|Ax + Bz^{k_{j-1}}\|_1 \\ \geq & \min_{x\in X} c^\top x + \underline{\rho}\|Ax + Bz^{k_{j-1}}\|_1 = R_{\underline{\rho}}(z^{k_{j-1}}) \end{aligned} \tag{4.6}$$

for large enough j. In addition,

$$\begin{aligned} g^\top z^{k_j} + t^{k_j} \geq & g^\top z^{k_j} + P(z^{k_{j-1}}, \mu^{k_{j-1}+1}, \beta^{k_{j-1}+1}) + \langle \mu^{k_{j-1}+1}, Bz^{k_j} - Bz^{k_{j-1}} \rangle \\ & - \beta^{k_{j-1}+1} \|Bz^{k_j} - Bz^{k_{j-1}}\|_1 \\ \geq & g^\top z^{k_j} + R_{\underline{\rho}}(z^{k_{j-1}}) + \langle \mu^{k_{j-1}+1}, Bz^{k_j} - Bz^{k_{j-1}} \rangle \\ & - \beta^{k_{j-1}+1} \|Bz^{k_j} - Bz^{k_{j-1}}\|_1. \end{aligned}$$

where the first inequality is due to constraints in (4.2) and the second inequality is due to (4.6). Taking limit on both sides gives $\bar{p} = \lim_{j\to\infty} g^\top z^{k_j} + t^{k_j} \geq g^\top z^* + R_{\underline{\rho}}(z^*) \geq p^*$, where the first inequality is due to the continuity of $R_{\underline{\rho}}$, the fact that $Bz^{k_j} - Bz^{k_{j-1}}$ vanishes, and the boundedness of $\{(\mu^k, \beta^k)\}_{k\in\mathbb{N}}$ by Assumption 4.2. So we conclude that $g^\top z^* + R_{\underline{\rho}}(z^*) = p^* = \bar{p}$.

2. We have shown the first equality. Let $x(z^*, \underline{\rho})$ be optimal to the penalty problem $\min_{x\in X} c^\top x + \underline{\rho}\|Ax + Bz^*\|_1$. Since $\underline{\rho}$ supports exact penalization, we have $(x(z^*, \underline{\rho}), z^*) \in= \operatorname{Argmin}_{x\in X, z\in Z}\{c^\top x + g^\top z \mid Ax + Bz = 0\}$. So we know $Ax(z^*, \underline{\rho}) + Bz^* = 0$ and $c^\top x(z^*, \underline{\rho}) + g^\top z^* = p^*$. Consequently,

$$\begin{aligned} p^* = g^\top z^* + R_{\underline{\rho}}(z^*) \leq & g^\top z^* + R_{\overline{\rho}}(z^*) = g^\top z^* + \min_{x\in X} c^\top x + \overline{\rho}\|Ax + Bz^*\|_1 \\ \leq & g^\top z^* + c^\top x(z^*, \underline{\rho}) = p^*. \end{aligned}$$

This proves the second equality.

3. Let $\{(x^{k_j+1}, z^{k_j})\}_{j\in\mathbb{N}}$ be the subsequence convergent to (x^*, z^*). Notice that the value of $g^\top z^{k_j} + c^\top x^{k_j+1} + \langle \mu^{k_j+1}, Ax^{k_j+1} + Bz^{k_j} \rangle + \beta^{k_j+1}\|Ax^{k_j+1} + Bz^{k_j}\|_1$ is bounded from below by $g^\top z^{k_j} + R_{\underline{\rho}}(z^{k_j})$ and from above by $g^\top z^{k_j} + R_{\overline{\rho}}(z^{k_j})$. Assuming without loss generality that $\lim_{j\to\infty}(\mu^{k_j+1}, \beta^{k_j+1}) = (\mu^*, \beta^*)$ and taking limit on both sides of the above two inequalities give $c^\top x^* + g^\top z^* + \langle \mu^*, Ax^* + Bz^* \rangle + \beta^*\|Ax^* + Bz^*\|_1 = p^*$, where the equality holds due to the second claim. It suffices to show $Ax^* + Bz^* = 0$. Suppose not, then by Theorem 2.4, $(x^*, z^*) \notin \operatorname{Argmin}_{x\in X, z\in Z} c^\top x + g^\top z + \underline{\rho}\|Ax + Bz\|_1$; therefore,

$$\begin{aligned} p^* = & c^\top x^* + g^\top z^* + \langle \mu^*, Ax^* + Bz^* \rangle + \beta^*\|Ax^* + Bz^*\|_1 \\ \geq & c^\top x^* + g^\top z^* + \underline{\rho}\|Ax^* + Bz^*\|_1 \\ > & \min_{x\in X, z\in Z} c^\top x + g^\top z + \underline{\rho}\|Ax + Bz\|_1 = p^*, \end{aligned}$$

where is a desired contradiction. This completes the proof. ∎

In order to establish iteration complexity of Algorithm 4.1, we need a slightly stronger version of Assumption 4.2.

THEOREM 4.5. *Suppose in addition to Assumption 4.2, $\underline{\rho} - 1 > 0$ supports exact penalization, and $\beta^k - \|\mu^k\|_\infty \geq \underline{\rho}$ for all $k \in \mathbb{N}$. Let $Z \subseteq \overline{B}_1(\bar{z}; R)$ for some $\bar{z} \in \mathbb{R}^d$ and radius $R > 0$. Given $\epsilon > 0$, algorithm 4.1 finds a solution (z^K, t^K) of problem (4.2) satisfying $p^* - (g^\top z^K + t^K) \leq \epsilon$ in no more than $K \leq (1 + 2(\underline{\rho} + \overline{\rho})\|B\|_1 R\epsilon^{-1})^d$ iterations. Moreover, the following claims hold.*

1. *Let $x(z^K, \underline{\rho}) \in \operatorname{Argmin}_{x\in X} c^\top x + \underline{\rho}\|Ax + Bz^K\|_1$. The pair $(x(z^K, \underline{\rho}), z^K)$ is an ϵ-solution to MILP (1.1).*
2. *If $\underline{\rho}$ and $\overline{\rho}$ also satisfy that*

$$\underline{\rho} \geq 2(\|c\|_1 D_\infty(X) + \|g\|_1 D_\infty(Z))\epsilon^{-1} \text{ and } \overline{\rho} \leq \frac{3}{2}\underline{\rho} - 1, \tag{4.7}$$

then (x^{K+1}, z^K) *is an approximate solution to* (1.1) *in the sense that*

$$\|Ax^{K+1} + Bz^K\|_1 \le \epsilon, \text{ and } c^\top x^{K+1} + g^\top z^K \le p^* + \epsilon + \|c\|_1 D_\infty(X). \tag{4.8}$$

Proof. For all nonnegative integers $i < j$, we have

$$\begin{aligned}\max\{p^* - (g^\top z^j + t^j), g^\top z^j + R_{\underline{\rho}}(z^j) - p^*\} &\le R_{\underline{\rho}}(z^j) - t^j \\ &\le R_{\underline{\rho}}(z^j) - P(z^i, \mu^{i+1}, \beta^{i+1}) - \langle \mu^{i+1}, Bz^j - Bz^i\rangle + \beta^{i+1}\|Bz^j - Bz^i\|_1 \\ &\le R_{\underline{\rho}}(z^j) - R_{\underline{\rho}}(z^i) + (\|\mu^{i+1}\|_\infty + \beta^{i+1})\|Bz^j - Bz^i\|_1 \le (\underline{\rho} + \overline{\rho})\|B\|_1\|z^j - z^i\|_1.\end{aligned} \tag{4.9}$$

The first inequality is due to $g^\top z^j + t^j \le p^* \le g^\top z^j + R_{\underline{\rho}}(z^j)$, the second inequality is due to (4.2) (note that $j \ge i+1$), the third inequality is due to (4.6), and the last inequality is due to $\beta^k + \|\mu^k\|_\infty \le \overline{\rho}$ and $R_{\underline{\rho}}$ being $\underline{\rho}\|B\|_1$-Lipschitz. Let K be the smallest index such that $\max\{p^* - (g^\top z^K + t^K), g^\top z^K + R_{\underline{\rho}}(z^K) - p^*\} \le \epsilon$. Then we must have $\|z^i - z^j\|_1 > \epsilon/[(\underline{\rho} + \overline{\rho})\|B\|_1]$ for all $0 \le i < j \le K-1$, since otherwise (4.9) implies that

$$\max\{p^* - (g^\top z^j + t^j), g^\top z^j + R_{\underline{\rho}}(z^j) - p^*\} \le (\underline{\rho} + \overline{\rho})\|B\|_1\|z^j - z^i\|_1 \le \epsilon,$$

contradicting to the choice of K. By the same argument as in the proof of Theorem 2.8, we can bound K as claimed. Next we prove the two claims.

1. Since $g^\top z^K + R_{\underline{\rho}}(z^K) - p^* \le \epsilon$, we know $(x(z^K, \underline{\rho}), z^K)$ is an ϵ-optimal solution to the problem $\min_{x\in X, z\in Z} c^\top x + g^\top z + \underline{\rho}\|Ax + Bz\|_1$. By the same proof of Theorem 3.6, we conclude that $(\tilde{x}^K, z^K)$ is an ϵ-solution of MILP (1.1).
2. Since $g^\top z^K + R_{\underline{\rho}}(z^K) - p^* \le \epsilon$, we know

$$g^\top z^K + c^\top x(z^K, \underline{\rho}) + \underline{\rho}\|Ax(z^K, \underline{\rho}) + Bz^K\|_1 \le p^* + \epsilon, \tag{4.10}$$

Since $\underline{\rho} - 1$ also supports exact penalization,

$$p^* \le c^\top x(z^K, \underline{\rho}) + g^\top z^K + (\underline{\rho} - 1)\|Ax(z^K, \underline{\rho}) + Bz^K\|_1.$$

The above two inequalities together implies that $\|Ax(z^K, \underline{\rho}) + Bz^K\|_1 \le \epsilon$. By the definition of $R_{\overline{\rho}}$, we then have

$$R_{\overline{\rho}}(z^k) \le c^\top x(z^K, \underline{\rho}) + \overline{\rho}\|Ax(z^K, \underline{\rho}) + Bz^K\|_1 \le R_{\underline{\rho}}(z^K) + (\overline{\rho} - \underline{\rho})\epsilon. \tag{4.11}$$

Denote $(\mu, \beta) = (\mu^{K+1}, \beta^{K+1})$. Invoking Hölder's inequality, it holds that

$$\begin{aligned} & c^\top x^{K+1} + \underline{\rho}\|Ax^{K+1} + Bz^K\|_1 \\ \le & c^\top x^{K+1} + \langle \mu, Ax^{K+1} + Bz^K\rangle + \beta\|Ax^{K+1} + Bz^K\|_1 \\ \le & R_{\overline{\rho}}(z^K) \le R_{\underline{\rho}}(z^K) + (\overline{\rho} - \underline{\rho})\epsilon \le p^* - g^\top z^K + (\overline{\rho} + 1 - \underline{\rho})\epsilon, \end{aligned} \tag{4.12}$$

where the inequalities are due to $\beta - \|\mu\| \ge \underline{\rho}$, $\beta + \|\mu\| \le \overline{\rho}$, (4.11), and (4.10), respectively. It then follows that

$$\|Ax^{K+1} + Bz^K\|_1 \le \frac{p^* - c^\top x^{K+1} - g^\top z^K}{\underline{\rho}} + \frac{(\overline{\rho} + 1 - \underline{\rho})\epsilon}{\underline{\rho}} \le \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon,$$

where the second inequality is due to(4.7). Finally, by (4.10), we see that $c^\top x^{K+1} + g^\top z^K \le p^* + \epsilon + c^\top(x^{K+1} - x(z^K, \underline{\rho})) \le p^* + \epsilon + \|c\|_1 D_\infty(X)$. ∎

Remark 4.6. Some remarks follow.

1. Theorem 4.5 provides an estimate on the number of ADMM iterations in order for the objective value of z-subproblem (4.2) to get close to p^*. In particular, it is ensured that $z^K + t^K \le p^* + \epsilon$ if we let ADMM run $K = (1 + 2(\underline{\rho} + \overline{\rho})\|B\|_1 R\epsilon^{-1})^d$ iterations.
2. Theorem 4.5 also provides two ways to recover approximate solutions to MILP (1.1). In the first case, we invoke a post-processing step by evaluating $R_{\underline{\rho}}$ (if a good estimate of $\underline{\rho}$ is available). In the second approach, we solve another x-subproblem (4.1) to get x^{K+1}; under additional requirements on $\underline{\rho}$ and $\overline{\rho}$, (x^{K+1}, z^k) is ϵ-feasible with a sub-optimality bound provided in (4.8).
3. We comment on the term $\|c\|_1 D_\infty(X)$ in the right-hand side of (4.8), which comes from the inner product $c^\top(x^{K+1} - x(z^K, \underline{\rho}))$. Although x^{K+1} and $x(z^K, \underline{\rho})$ are both closely related to z^K, we do not think it is easy to derive a better bound on their distance, especially when tools such as error bounds from convex analysis are not applicable here. Nevertheless, when $X = \prod_{i=1}^P X_i$, the diameter $D_\infty(X)$ is equal to $\max_{i\in[P]} D_\infty(X_i)$, which could be independent of P for many applications.

Theorem 4.5 does not answer the question: can ADMM itself produce an ϵ-solution to MILP (1.1). We provide an affirmative answer in the next theorem.

THEOREM 4.7. *Let $\epsilon > 0$. Suppose in addition to Assumption 4.2, $\underline{\rho}$ also satisfies that $\underline{\rho} \ge (\|c\|_1 D_\infty(X) + \|g\|_1 D_\infty(Z))\epsilon^{-1} + 1$. Then Algorithm 4.1 finds an ϵ-solution to MILP (1.1) in a finite number of iterations.*

Proof. By the second claim in Theorem 4.4, there exists an index K such that $g^\top z^K + R_{\overline{\rho}}(z^K) \le p^* + \epsilon$; together with the first two inequalities in (4.12), we have $c^\top x^{K+1} + \underline{\rho}\|Ax^{K+1} + Bz^K\|_1 \le R_{\overline{\rho}}(z^K) \le p^* + \epsilon - g^\top z^K$. Now the assumed lower bound of $\underline{\rho}$ implies that (x^{K+1}, z^K) is an ϵ-solution to MILP (1.1). ∎

Finally we note that the objective of the z-subproblem (4.2) remains a valid lower bound of p^*, and if ADMM finds a feasible solution (x^{k+1}, z^k), then $c^\top x^{k+1} + g^\top z^k$ will be a valid upper bound of p^*. Our implementation monitors these two quantities.

5. Numerical Experiments.

We demonstrate the performance of the proposed ALM and ADMM on three classes of problems: variants of an investment planning problem [57] (Section 5.2), the stochastic server location problem (SSLP) [22, 48] (Section 5.3), and a class of structured MILPs generated with random data (Section 5.4). We first discuss some implementation details in Section 5.1.

5.1. Implementation Details.

The goal of our experiments is to demonstrate the correctness and efficiency of the proposed algorithms and assess the extent to which the scheme of adding nonconvex cuts can serve as a practical solution method. To this end, our comparisons are straightforward: we first solve a MILP instance with a given solver, then we run ALM and ADMM on the instance, where each x/z-subproblem is solved by the same solver. In order to evaluate the effect of nonconvex cuts alone, we do not add other well-known cuts, e.g., Benders cuts and Lagrangian cuts, though a combined implementation might improve the overall performance.

5.1.1. Experiment Setup.

Experiments in Sections 5.2 and 5.4 are performed

on a personal laptop with a 2.6GHz 6-Core Intel Core i7 processor and 16GB RAM, with Gurobi 9.5.2 [32] under default settings (12 threads) as the underlying MILP solver. Since large-scale SSLP instances require long solution time, experiments in Section 5.3 are performed on an elastic compute service (ECS) server with one Intel Xeon Platinum 8269CY CPU (104 virtual CPUs) and 768GB memory. Due to the unavailability of Gurobi on the ECS server, we use HiGHS 1.2.2 [35] with default settings (single thread) as the underlying MILP solver. Both Gurobi and HiGHS report optimality when the relative gap is less than 0.01%. Our codes are written in Julia 1.6.6 [9], and solvers are interfaced through JuMP 1.3.0 [24].

5.1.2. Implementation and Modification of Nonconvex Cuts. Consider the ALM framework introduced in Section 3. Fixing the current dual pair (λ, ρ), we use AUSAL (Algorithm 3.1) to solve the primal subproblem (1.6): given $\bar{z} \in Z$, we approximate the function $R(z; \lambda, \rho)$ by adding a reverse-norm cut of the form

$$R(z; \lambda, \rho) \geq R(\bar{z}; \lambda, \rho) - K_\rho \|z - \bar{z}\|_1 \tag{5.1}$$

to the z-subproblem, where $R(z; \lambda, \rho)$ is defined in (3.1). When the current AUSAL terminates, we update a new pair of dual variables (λ^+, ρ^+), and start the next AUSAL. Notice that (5.1) generated with (λ, ρ) is not valid anymore for $R(z; \lambda^+, \rho^+)$. Naively, we can remove all previous cuts when starting a new AUSAL, but this will cause a loss of historical information. Instead, we modify old cuts so that they always stay valid for the latest dual information (λ^+, ρ^+). Assume $\rho^+ \geq \rho$. It is easy to see $R(\bar{z}; \lambda^+, \rho^+) \geq R(\bar{z}; \lambda, \rho) - \|\lambda^+ - \lambda\|_\infty (\max_{x \in X} \|Ax\|_1)$ and hence the following inequality is valid or all $R(z; \lambda, \rho)$ over $z \in Z$:

$$R(z; \lambda^+, \rho^+) \geq R(\bar{z}; \lambda, \rho) - \|\lambda^+ - \lambda\|_\infty \left(\max_{x \in X} \|Ax\|_1 \right) - K_{\rho^+} \|z - \bar{z}\|_1. \tag{5.2}$$

The new cut (5.2) can be obtained by modifying the coefficient K_ρ and constant $R(\bar{z}; \lambda, \rho)$ in (5.1). Though these modified cuts may not be tight, empirically we observe that they help ALM to maintain a more stable lower bound. In addition, in our experiments on SSLP instances, we only maintain the latest 200 nonconvex cuts.

5.1.3. MILP Subproblems, Dual Updates, and Parameters. In our tested problems, the x-subproblem in ALM and ADMM consists of a series of parallel subproblems. For ease of implementation, we solve these problems sequentially in every iteration of ALM and ADMM. Therefore, in view of solution time reported in our experiments, we expect further acceleration when more computational budgets are available. For both ALM and ADMM, the dual multiplies are initialized with zeros, and the penalty is initialized with $\rho_0 > 0$. For ALM, we update dual information when the gap of AUSAL is less than 0.01%, or `innerALM` iterations are reached. Then we set the penalty by $\rho \leftarrow \gamma\rho$ for some $\gamma > 1$, and update multiplies by $\lambda \leftarrow \lambda + \alpha_k(Ax^k + Bz^k)$ with k being the number of AUSAL calls and

$$\alpha_k = \texttt{almDualStepSize}/(k \times \sqrt{2} \times \max\{1, r^k\}),$$

where `almDualStepSize` > 0 and r^k is the incumbent primal residual meansured in ℓ_1-norm. For ADMM, we update the penalty $\beta \leftarrow \gamma\beta$ every `innerADMM` iterations, and update dual multipliers in every iteration by

$$\mu^{k+1} \leftarrow \mu^k + \texttt{admmDualStepSize} \times \beta \times (Ax^k + Bz^k).$$

Notice that we do not explicitly project multipliers onto some bounded set to encourage more diverse AL cuts. In the follows, we report parameters as a 6-tuple:

$$(\rho_0, \gamma, \texttt{innerALM}, \texttt{innerADMM}, \texttt{almDualStepSize}, \texttt{admmDualStepSize}).$$

5.2. Investment Planning Problems. We consider the following investment planning problem:

$$\min_z \ \{-1.5z_1 - 4z_2 + \mathbb{E}_\omega[v_\omega(z)] : \ z \in Z\}, \tag{5.3}$$

where $Z \subseteq \mathbb{R}^2$ and $v_\omega(z)$ is defined as

$$\min_{x \in \{0,1\}^4} \left\{ -16x_1 - 19x_2 - 23x_3 - 28x_4 : \ \begin{bmatrix} 2 & 3 & 4 & 5 \\ 6 & 1 & 3 & 1 \end{bmatrix} x \le h_\omega - Tz \right\}$$

with $h_\omega \in \mathbb{R}^2$ and $T \in \mathbb{R}^{2\times 2}$. The problem was firstly introduced by Schultz et al. [57] with $Z = [0,5]^2$ and $T = I_{2\times 2}$, and its variants have been used as benchmark instances in the literature [2, 29, 65]. We first consider the following variants tested in [65]. Given an integer $S > 1$, let the two components of h_ω correspond to a lattice point over the 2-dimensional grid $\{(5 + 10\frac{s_1-1}{S-1}, 5 + 10\frac{s_2-1}{S-1}) : \ s_1, s_2 \in [S]\}$ with equal probability. In the context of two-stage stochastic programs, the second stage contains a total of S^2 recourse problems. Though each subproblem in x is relatively simple, in the largest tested instance where $S = 101$, there are 40,804 binary variables in the second stage, plus two general-integer variables in the first stage. We further consider two choices of the technology matrix T: either $T = I_{2\times 2}$ (denoted by `I`), or $T = [\frac{2}{3} \ \frac{1}{3}; \frac{1}{3} \ \frac{2}{3}]$ (denoted by `T`). The results are presented in Tables 1 and 2.

Table 1: Comparison with Gurobi for $Z = [0,5]^2 \cap \mathbb{Z}^2$ and $T = \texttt{I}$ with parameters (1, 1.1, 100, 50, 200, 200)

	ALM			ADMM			Gurobi		
S	Gap	Iter.	Time (s)	Gap	Iter.	Time (s)	Gap	Node	Time (s)
21	0.00%	37	9.88	0.00%	37	9.06	0.00%	1	0.35
31	0.00%	37	19.21	0.00%	37	18.47	0.00%	1	0.78
41	0.00%	37	33.81	0.00%	37	33.11	0.00%	1	1.55
51	0.00%	37	49.75	0.00%	37	50.12	0.00%	1	2.77
61	0.00%	37	72.92	0.00%	37	71.95	0.00%	1	4.88
71	0.00%	37	95.15	0.00%	37	98.80	0.00%	1	8.35
81	0.00%	37	123.69	0.00%	37	127.82	0.00%	1	15.98
91	0.00%	37	157.53	0.00%	37	161.15	0.00%	1	22.04
101	0.00%	37	195.81	0.00%	37	201.96	0.00%	1	31.13
Avg.	0.00%	37	84.19	0.00%	37	85.83	0.00%	1	9.76

For $T = \texttt{I}$, Gurobi is able to find the optimal solution at the root node, and hence the number of explored node is 1 across all instances. We observe that Gurobi invokes many heuristics at the root node, while another solver, HiGHS, does not report optimality at the root node for any instance in Table 1. The proposed ALM and ADMM are slower, partly because a large number of subproblems in x are solved sequentially in our implementation. Nevertheless, for all instances, both ALM and

Table 2: Comparison with Gurobi for $Z = [0,5]^2 \cap \mathbb{Z}^2$ and $T = \mathtt{T}$ with parameters (1, 1.1, 100, 50, 200, 200)

	ALM			ADMM			Gurobi		
S	Gap	Iter.	Time (s)	Gap	Iter.	Time (s)	Gap	Node	Time (s)
21	0.00%	37	10.16	0.00%	37	9.26	0.00%	2987	68.91
31	0.00%	37	21.76	0.00%	37	20.96	0.01%	2874	53.59
41	0.00%	37	32.84	0.00%	37	32.94	0.00%	3859	129.19
51	0.00%	37	49.64	0.00%	37	50.10	0.01%	3617	289.76
61	0.00%	37	74.19	0.00%	37	74.81	0.01%	3825	150.14
71	0.00%	37	95.11	0.00%	37	99.47	0.00%	2875	268.72
81	0.00%	37	123.02	0.00%	37	126.08	0.01%	3746	228.46
91	0.00%	37	162.84	0.00%	37	173.63	0.01%	2578	214.24
101	0.00%	37	190.75	0.00%	37	208.48	0.01%	2444	351.39
Avg.	0.00%	37	84.48	0.00%	37	88.41	0.01%	3201	194.93

ADMM close the gap with the same optimal objective value as Gurobi. For $T = \mathtt{H}$, the problems become harder. Gurobi explored a few thousands of nodes before reporting optimal solutions. In contrast, ALM and ADMM are able to locate and verify optimality around 55% faster than Gurobi on average, even though recourse subproblems are solved sequentially.

Another interesting observation is that, ALM and ADMM always terminate at the 37th iteration. Note that the feasible region $Z = [0,5]^2 \cap \mathbb{Z}^2$ consists of 36 points in $\mathbb{R}^2$. The proposed algorithms enumerate the feasible region in the first 36 iterations, and use one more iteration to verify optimality. Though this worst-case complexity is not very promising, we further enlarge the feasible region of z to $Z = [0,10]^2 \cap \mathbb{Z}^2$ so that there are 121 feasible solutions. We report the same metrics in Table 3. The proposed ALM and ADMM terminate successfully in less than 121 iterations for all instances and take 265.08 and 357.30 seconds on average, respectively. In contrast, without structural knowledge on the problem data, Gurobi only solves the smallest instance within 1800 seconds.

Table 3: Comparison with Gurobi for $Z = [0,10]^2 \cap \mathbb{Z}^2$ and $T = \mathtt{T}$ with parameters (1, 1.1, 100, 100, 200, 0.01)

	ALM			ADMM			Gurobi		
S	Gap	Iter.	Time (s)	Gap	Iter.	Time (s)	Gap	Node	Time (s)
21	0.00%	107	38.21	0.00%	99	42.07	0.00%	2103	14.71
31	0.00%	107	74.03	0.00%	105	88.35	0.62%	12353	1800.01
41	0.00%	107	113.77	0.00%	101	137.30	0.05%	43435	1800.01
51	0.00%	107	161.41	0.00%	102	209.29	0.16%	30359	1800.02
61	0.00%	107	234.30	0.00%	105	308.62	0.51%	10681	1800.01
71	0.00%	107	298.71	0.00%	102	396.42	0.02%	37002	1800.03
81	0.00%	107	382.68	0.00%	102	514.50	0.46%	11571	1800.01
91	0.00%	107	497.18	0.00%	105	692.33	0.69%	11114	1800.02
101	0.00%	107	585.47	0.00%	102	826.80	0.03%	39720	1800.05
Avg.	0.00%	107	265.08	0.00%	103	357.30	0.28%	22038	$\geq$ 1601.65

5.3. Stochastic Server Location Problems . The stochastic server location problem (SSLP) [22, 48] is a classic two-stage MILP that can be cast as follows:

$$\begin{aligned}
\min_{z,\{x^p,s^p\}_{p\in[P]}} \quad & \sum_{j=1}^{m} c_j z_j + \sum_{p\in[P]} \text{prob}_p \left(\sum_{j=1}^{m} q_{0j} s_j^p - \sum_{i=1}^{n}\sum_{j=1}^{m} q_{ij} x_{ij}^p \right) \\
\text{s.t.} \quad & \sum_{i=1}^{n} d_{ij} x_{ij}^p \le u x_j + s_j^p, \quad j \in [m],\ p \in [P], \\
& \sum_{j=1}^{m} x_{ij}^p = h_i^p, \qquad\qquad i \in [n],\ p \in [P], \\
& z_j, x_{ij}^p \in \{0,1\}, s_j^p \ge 0, \quad i \in [n],\ j \in [m],\ p \in [P].
\end{aligned}$$

The problem aims to allocate servers to m potential sites and meet the demand of n potential clients. In the first stage, a decision maker needs to allocate servers at m potential sites ($z_j = 1$ if and only if a server is located at site j) associated with a cost c_j for $j \in [m]$. Then in each scenario p in the second stage, the availability of client i is observed and expressed by a vector h^p with $h_i^p = 1$ if and only if client i is present in scenario p. The variable $x_{ij}^p = 1$ if and only if client i is served at site j in scenario p. Each allocated server has u units of resource, and client i uses d_{ij} units of resource at site j and generate revenue q_{ij}; shortage of resource at site j is modeled by a continuous variable $s_j^p \ge 0$ and penalized by a cost q_{0j}.

We generate data the same way as in [22]. Each c_j is uniformly sampled from $\{40, 41, \cdots, 80\}$, each d_{ij} is uniformly sampled from $\{0, 1, \cdots, 25\}$, and each h_i^p is 0 or 1 with equal probability; we then set $\text{prob}_p = 1/P$, $q_{0j} = 1000$, and $u = \frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{n} d_{ij}$. We fix the number of second-stage scenarios to $P = 50$, and experiment with two sets of (m, n) pairs. In the first set, the number of potential servers is relatively small: `mSmall` $= \{(5, 100), (10, 100), (15, 100)\}$; in the second set, we test on problem scales considered in [22]: `mLarge` $= \{(20, 100), (30, 70), (40, 50), (50, 40)\}$. We limit ALM and ADMM iterations by 2000, and set a time limit of 7200 seconds for all solution methods.

The results are presented in Tables 4 and 5. On `mSmall` instances, the average gap of 9 instances is 1.81% for ALM and 1.65% for ADMM, while HiGHS has a slightly higher gap of 2.2%. HiGHS also solves two more instances to optimality within the time limit. The `mLarge` instances are more challenging for the proposed methods. ALM and ADMM obtain an average of gap of 22.09% and 14.41%, respectively, and HiGHS achieves a better average gap of 6.45%. We also observe that ADMM tends to perform better than ALM. Such an advantage might due to 1) historical cuts are effectively preserved in ADMM, and 2) linear terms in AL cuts can help the method escape non-optimal regions faster.

We note that the state-of-the-art approach by Chen and Luedtke [22] reports optimality on `mLarge` instances with an average solution time of 1469 seconds for $P = 50$ and 4348 seconds for $P = 200$. While not immediately comparable to theirs, our results suggest that the proposed methods can be preferable over a black-box MILP solver, at least when the dimension or feasible region of Z is not too large. Directions for further acceleration include parallel implementation of x-subproblems, instance-specific parameter tuning (note that we fix a set of parameters in each table, while dual updates can affect the performance of AL-based methods drastically), and combination with existing linear cuts.

Table 4: Comparison with HiGHS on `mSmall` instances with parameters (1, 1.25, 50, 50, 50, 50)

		ALM			ADMM			HiGHS		
m-n-P	Instance	Gap	Iter.	Time (s)	Gap	Iter.	Time (s)	Gap	Node	Time (s)
5-100-50	1	0.00%	86	225.73	0.00%	17	25.65	0.00%	176	78.56
	2	0.00%	81	221.86	0.00%	42	81.25	0.00%	955	213.43
	3	0.00%	135	570.40	0.00%	54	82.59	0.00%	735	111.68
10-100-50	1	1.53%	2000	4381.55	1.85%	2000	3531.80	0.00%	3266	1958.28
	2	3.15%	2000	4728.14	2.37%	2000	3900.31	6.31%	17918	7200.39
	3	2.18%	2000	4822.01	0.55%	2000	3769.56	0.01%	3513	1782.54
15-100-50	1	3.90%	1529	7200.00	3.68%	2000	7002.99	5.35%	9178	7200.00
	2	2.28%	1490	7200.00	1.49%	1387	7200.00	4.81%	9781	7200.08
	3	3.22%	1790	7200.00	4.89%	1776	7200.00	3.50%	14453	7200.01
Avg.		1.81%	1235	4061.08	1.65%	1253	3643.79	2.22%	6664	3660.55

Table 5: Comparison with HiGHS on `mLarge` instances with parameters (1, 1.25, 50, 50, 20, 20)

		ALM			ADMM			HiGHS		
m-n-P	Instance	Gap	Iter.	Time (s)	Gap	Iter.	Time (s)	Gap	Node	Time (s)
20-100-50	1	7.41%	269	7200.00	5.48%	516	7200.00	8.94%	6680	7200.05
	2	8.59%	640	7200.00	7.37%	1244	7200.00	7.29%	5276	7200.07
	3	9.37%	351	7200.00	7.64%	895	7200.00	6.89%	5460	7200.04
30-70-50	1	25.45%	290	7200.00	15.89%	738	7200.00	8.32%	3546	7200.03
	2	20.69%	209	7200.00	17.99%	558	7200.00	13.08%	2892	7200.01
	3	24.75%	229	7200.00	8.57%	189	7200.00	3.44%	2604	7200.02
40-50-50	1	21.87%	197	7200.00	24.26%	382	7200.00	5.66%	1634	7200.04
	2	36.30%	187	7200.00	22.17%	359	7200.00	9.68%	1942	7200.20
	3	18.79%	196	7200.00	9.25%	332	7200.00	2.71%	1779	7200.21
50-40-50	1	44.22%	170	7200.00	27.23%	286	7200.00	8.14%	1192	7200.10
	2	30.11%	71	7200.00	7.98%	306	7200.00	0.00%	117	2538.02
	3	17.59%	167	7200.00	19.07%	314	7200.00	3.23%	1435	7200.01
Avg.		22.09%	248	7200.00	14.41%	510	7200.00	6.45%	2880	$\geq$ 6811.57

5.4. A Class of Random MILPs. Our experiments in the previous two subsections suggest that the scheme of adding nonconvex cuts is indeed practical, and can be even preferable, when the feasible region of Z is not too large and subproblems in x are relatively easy. To further demonstrate how the proposed methods can take advantages of such structures, we consider a class of MILPs in the form of (1.1) generated as follows. Given an integer $P > 0$, for each $p \in [P]$, we create a polytope $X_p = \{x \in [0,2]^{50} : \ E_p x = f_p, x_p \in \mathbb{Z}^{30} \times \mathbb{R}^{20}\}$ and an objective vector $c_p \in \mathbb{R}^{50}$. The matrix $E_p \in \mathbb{R}^{30\times 50}$ and the vector c_p have standard Gaussian entries, and $f_p = E_p \bar{x}_p \in \mathbb{R}^{30}$, where $\bar{x}_i$ is uniformly sampled from $\{0,1,2\}$ for $i \in \{1,\cdots,30\}$ and $[0,2]$ for $i \in \{31,\cdots,50\}$. Then from each block p, we introduce a copy of the first three components of x_p, denoted by (z_{p1}, z_{p2}, z_{p3}). We then denote all copied variables by z, and let $Z = \{z \in \{0,1\}^{3P} : \ Gz = h\}$, where $G \in \mathbb{R}^{(3P-50)\times 3P}$ has standard Gaussian entries, and $h = G\bar{z}$, where each component of $\bar{z}$ has the same value as its copy in $\bar{x}_p$ so that the problem is feasible. By construction, the coupling constraints $Ax + Bz = 0$ has the form $-[x_p]_i + z_{pi} = 0$ for $i \in [3]$, $p \in [P]$, where $[x_p]_i$ denotes the i-th component of x_p. The objective vector of z is set to zeros. Written in its extensive form (1.1), the problem has $20P$ continuous variables, $33P$ integer variables, and $36P - 50$ equality constraints.

For each $P \in \{50, 100, 200, 500\}$, we generate 3 instances and report results in Table 6. The proposed ALM and ADMM successfully terminate for all 12 instances with zero duality gaps, while Gurobi fails to find feasible solutions when $P \in \{200, 500\}$ in 1800 seconds. We do not report iterations of ALM and ADMM because both of them terminate in exactly 2 iterations for all generated instances: the first iteration finds the optimal solution, and the second iteration verifies the solution is indeed globally optimal by adding a nonconvex cut at the same point.

Note that the matrix G is nearly square as P increases. Our construction deliberately reduces the number of feasible solutions in Z so that once a feasible (probably optimal as well) z is found, the rest problem in x can be easily solved. We note that other linear cuts, i.e., Benders and Lagrangian cuts, should also terminate the algorithm in two iterations: since the optimal solution is found in the first iteration, which cannot be cut off by valid cuts, the second iteration should close the gap. On the other hand, Gurobi as a general-purpose solver does not pass such structural information to branch-and-bound. The results have no intention to claim the superiority of ALM and ADMM over Gurobi, or any MILP solvers, but rather demonstrate that the proposed methods can significantly benefit from problem structures.

Table 6: Comparison with Gurobi on a class of MILPs with random data with parameters (1, 1.1, 100, 50, 200, 200)

		ALM		ADMM		Gurobi		
P	Instance	Gap	Time (s)	Gap	Time (s)	Gap	Node	Time (s)
50	1	0.00%	8.88	0.00%	8.69	0.00%	38369	146.88
	2	0.00%	8.97	0.00%	8.71	0.00%	50905	163.85
	3	0.00%	7.98	0.00%	7.51	0.00%	37550	264.91
100	1	0.00%	18.78	0.00%	19.15	0.00%	31679	674.83
	2	0.00%	20.90	0.00%	21.38	0.00%	40892	863.40
	3	0.00%	17.78	0.00%	17.78	0.00%	28346	507.21
200	1	0.00%	55.71	0.00%	53.47	-	20005	1800.27
	2	0.00%	52.24	0.00%	51.68	-	30363	1800.17
	3	0.00%	53.93	0.00%	53.84	-	25722	1800.56
500	1	0.00%	1108.85	0.00%	1108.05	-	1317	1801.52
	2	0.00%	326.92	0.00%	328.01	-	1447	1801.45
	3	0.00%	448.33	0.00%	442.67	-	3945	1800.36
Avg.		0.00%	177.44	0.00%	176.75	-	25878	$\geq$ 1118.78

6. Concluding Remarks. In this paper, we study generic MILP problems with two blocks of variables x and z. We propose an algorithm named AUSAL that alternatively updates x and z in the augmented Lagrangian function, which can be further directly embedded into the penalty method or ALM. We also propose a single-looped ADMM variant, which is built upon the AL cut introduced in [2, 72]. Different from the procedure used in the previous two references, we obtain an AL cut by solving a single augmented Lagrangian relaxation in variable x; compared to existing ADMM works, our ADMM variant allows a more flexible update scheme for the dual variable and penalty, and is guaranteed to converge to a global optimal solution with iteration complexity estimates. When certain block-angular structure is present, the update of x can be further decomposed and solved in parallel in both algorithms.

We conduct numerical experiments on three classes of MILPs and demonstrate

that the proposed methods exhibit advantages on structured problems over the state-of-the-art MILP solvers. Admittedly, the update of z variable in both algorithms requires solving a MILP problem with an increasing size of variables and constraints, which can be the computational bottleneck for large and dense problems. We are interested in investigating more practical subproblem oracles, i.e., managing a controllable size of cuts, or new methodologies to approximate the dependency between x and z. We leave these in the future work.

Appendix A. Missing Proofs.

A.1. Proof of Theorem 2.8.

We first present a useful lemma.

LEMMA A.1. *Let z^k be an iterate generated by Algorithm 2.1. Then $Q(z) - \underline{R}(z; Z_k) \le \epsilon$ for all $z \in Z$ such that $\|z - z^k\|_1 \le \epsilon/(2K)$.*

Proof. For all $z \in Z$ and $\|z - z^k\|_1 \le \epsilon/(2K)$, we have $Q(z) - \underline{R}(z; Z_k) \le Q(z) - Q(z^k) + K\|z - z^k\|_1 \le 2K\|z - z^k\|_1 \le \epsilon$, where the first inequality is due to $\underline{R}(z; Z_k) \ge Q(z^k) - K\|z - z^k\|_1$, and the second inequality is due to Q being K-Lipschitz. ∎

Proof of Theorem 2.8. Let $\{z^{k_j}\}_{j\in\mathbb{N}}$ be a subsequence convergent to some optimal solution z^* of (2.2) if we do not terminate Algorithm 2.1. Since $\lim_{j\to\infty} z^{k_j} = z^*$, we have $\|z^{k_{j+1}} - z^{k_j}\|_1 \le \epsilon/(2K_\rho)$ for all large enough $j \in \mathbb{N}$. Notice $k_{j+1} \ge k_j + 1$; by Lemma A.1, it follows $Q(z^{k_{j+1}}) - t^{k_{j+1}} \le Q(z^{k_{j+1}}) - \underline{R}(z^{k_{j+1}}; Z_{k_j}) \le \epsilon$. Now let T be the first index such that $Q(z^T) - t^T \le \epsilon$, which is sufficient to ensure the termination of Algorithm 2.1. For all $0 \le i < j \le T-1$, we claim that $\|z^i - z^j\|_1 > \epsilon/(2K)$: suppose not, then Lemma A.1 suggests that $Q(z^j) - t^j \le Q(z^j) - \underline{R}(z^j; Z_i) \le \epsilon$, contradicting to the choice of T. Let $r = \epsilon/(4K)$. Since $z^i \in Z$ for all $0 \le i \le T-1$ and $Z \subseteq \overline{B}_1(\bar{z}; R)$, $\bigcup_{i=0}^{T-1} \overline{B}_1(z^i; r) \subseteq Z + \overline{B}_1(0; r) \subseteq \overline{B}_1(\bar{z}; R + r)$; since the balls $\{\overline{B}_1(z^i; r)\}_{0\le i\le T-1}$ are disjoint, it follows

$$\mathrm{Vol}\left(\bigcup_{i=0}^{T-1} \overline{B}_1(z^i; r)\right) = T\mathrm{Vol}\left(\left(\overline{B}_1(0; r)\right)\right) \le \mathrm{Vol}\left(\overline{B}_1(\bar{z}; R + r)\right),$$

where $\mathrm{Vol}(\cdot)$ returns the volume of the argument, and thus

$$T \le \frac{\mathrm{Vol}(\overline{B}_1(\bar{z}; R + r))}{\mathrm{Vol}(\overline{B}_1(0; r))} = \left(\frac{R + r}{r}\right)^d = (1 + 4KR\epsilon^{-1})^d.$$

This completes the proof. ∎

A.2. Proof of Theorem 3.8.

We first state a standard lemma regarding the progress in objective value.

LEMMA A.2. *Let $\{(\lambda^k, \rho^k)\}_{k\in\mathbb{N}}$ be the sequence of iterates generated by Algorithm 3.2. Then for all $K \ge 1$, it holds that*

$$\min_{k\in[K]} p^* - d(\lambda^k, \rho^k) \le \frac{M}{\sqrt{2}} \frac{d_0^2 + \sum_{k=1}^K \tau_k^2}{\sum_{k=1}^K \tau_k} + \epsilon_p.$$

Proof. To simplify notation, we denote $w^k = (\lambda^k, \rho^k)$, and denote $w^* = (\lambda^*, \rho^*)$ to be the maximizer in the definition of d_0. We also denote the ϵ-subgradient of $(-d)(w^k)$ by g^k, and it holds

$$\|g^k\|_2^2 = \|Ax^k + Bz^k\|_2^2 + \|Ax^k + Bz^k\|_1^2 \le 2\|Ax^k + Bz^k\|_1^2 \le 2M^2. \tag{A.1}$$

Use the fact that $g^k \in \partial_\epsilon(-d)(w^k)$, we have $\|w^{k+1} - w^*\|_2^2 = \|w^k - \alpha_k g^k - w^*\|_2^2 \le \|w^k - w^*\|_2^2 + \alpha_k^2\|g_k\|_2^2 - 2\alpha_k(p^* - d(w^k) - \epsilon_p)$. Summing over $k = 1, \cdots, K$, we see $2\left(\min_{k\in[K]} p^* - d(\lambda^k, \rho^k) - \epsilon_p\right)\sum_{k=1}^K \alpha_k$ is bounded from above by

$$\begin{aligned} &2\sum_{k=1}^K \alpha_k(p^* - d(\lambda^k, \rho^k) - \epsilon_d) \le \|w^1 - w^*\|_2^2 + \sum_{k=1}^K \alpha_k^2\|g^k\|_2^2 \\ \le &\|w^1 - w^*\|_1^2 + \sum_{k=1}^K \alpha_k^2 2\|Ax^k + Bz^k\|_1^2 = d_0^2 + \sum_{k=1}^K \tau_k^2, \end{aligned}$$

which further implies

$$\min_{k\in[K]} p^* - d(\lambda^k, \rho^k) \le \frac{d_0^2 + \sum_{k=1}^K \tau_k^2}{2\sum_{k=1}^K \alpha_k} + \epsilon_p \le \frac{M}{\sqrt{2}} \frac{d_0^2 + \sum_{k=1}^K \tau_k^2}{\sum_{k=1}^K \tau_k} + \epsilon_p. \qquad \square$$

Proof of Theorem 3.8. The first claim follows from Lemma A.2 and the choices of τ_i and K so that:

$$\frac{M}{\sqrt{2}} \frac{d_0^2 + \sum_{k=1}^K \tau_k^2}{\sum_{k=1}^K \tau_k} = \frac{M}{\sqrt{2}} \frac{2d_0^2}{\sqrt{K} d_0} = \frac{\sqrt{2} M d_0}{\sqrt{K}} \le \epsilon_d.$$

Let $\gamma_k = \alpha_k/\tau_k$. Recall the bound in (A.1), and we have $\gamma_k\|g^k\|_2 \le 1$; the second claim is then proved in [56, Theorem 7.4]. $\square$

A.3. Proof of Theorem 3.11.

Proof of Theorem 3.11. Firstly notice that according to the penalty update, we have $\rho^k = \rho^1 + \sum_{j=2}^k \rho^j - \rho^{j-1} = \rho^1 + \sum_{j=2}^k \max\{\|\lambda^j\|_\infty, \alpha_j\|Ax^j + Bz^j\|_1\} \ge \rho^1 + \sum_{j=2}^k \alpha_j\|Ax^j + Bz^j\|_1 = \rho^1 + (k-1)\tau$. For the purpose of contradiction, suppose $\|Ax^k + Bz^k\|_1 > \epsilon_p$ for all $k \in \mathbb{N}$, and thus Algorithm 3.3 will generate an unbounded sequence $\{\rho^k\}_{k\in\mathbb{N}}$. Let (λ^*, ρ^*) be an optimal solution to the dual problem (1.7). Then we have $\|\lambda^{k+1} - \lambda^*\|_2^2$ is bounded from above by

$$\begin{aligned} &\|\lambda^k - \lambda^*\|_2^2 + \alpha_k^2\|Ax^k + Bz^k\|_2^2 + 2\alpha_k\left(d(\lambda^k, \rho^k) - p^* + \epsilon_p + \|Ax^k + Bz^k\|_1(\rho^* - \rho^k)\right) \\ &\le \|\lambda^k - \lambda^*\|_2^2 + \alpha_k^2\|Ax^k + Bz^k\|_2^2 + 2\alpha_k\left(\epsilon_p + \|Ax^k + Bz^k\|_1(\rho^* - \rho^k)\right), \end{aligned}$$

where the inequality is due to (3.4) and $d(\lambda^k, \rho^k) \le p^*$. By the definition of α_k and the fact that $\|Ax^k + Bz^k\|_1 > \epsilon_p$, we further have

$$\|\lambda^{k+1} - \lambda^*\|_2^2 \le \|\lambda^k - \lambda^*\|_2^2 + \tau^2 + 2\tau + 2\tau\rho^* - 2\tau\rho^k. \tag{A.2}$$

Notice that when $\rho^k \ge \rho^* + \tau/2 + 1$, we have $\|\lambda^{k+1} - \lambda^*\|_2 \le \|\lambda^k - \lambda^*\|_2$, and thus the dual sequence λ^k stays bounded; now letting $k \to \infty$ on (A.2), the left-hand side is nonnegative while the right-hand side goes to $-\infty$, which is a desired contradiction. $\square$

Acknowledgments. We would like to thank the authors of [17] for making their primal-decomposition code available and the authors of [72] for the discussion on AL cuts. A part of this work was done during an internship of Alibaba (US) Innovation Research.

[1] T. Achterberg and R. Wunderling, *Mixed integer programming: Analyzing 12 years of progress*, in Facets of combinatorial optimization, Springer, 2013, pp. 449–481.
[2] S. Ahmed, F. G. Cabral, and B. F. P. da Costa, *Stochastic lipschitz dynamic programming*, Mathematical Programming, (2020), pp. 1–39.
[3] S. Ahmed, M. Tawarmalani, and N. V. Sahinidis, *A finite branch-and-bound algorithm for two-stage stochastic integer programs*, Mathematical Programming, 100 (2004), pp. 355–377.
[4] A. Alavian and M. C. Rotkowitz, *Improving ADMM-based optimization of mixed integer objectives*, in 2017 51st Annual Conference on Information Sciences and Systems (CISS), IEEE, 2017.
[5] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, *On augmented Lagrangian methods with general lower-level constraints*, SIAM Journal on Optimization, 18 (2007), pp. 1286–1309.
[6] G. Angulo, S. Ahmed, and S. S. Dey, *Improving the integer l-shaped method*, INFORMS Journal on Computing, 28 (2016), pp. 483–499.
[7] J. F. Benders, *Partitioning procedures for solving mixed-variables programming problems*, Numerische mathematik, 4 (1962), pp. 238–252.
[8] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Academic press, 2014.
[9] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM review, 59 (2017), pp. 65–98.
[10] R. E. Bixby, *A brief history of linear and mixed-integer programming computation*, Documenta Mathematica, 2012 (2012), pp. 107–121.
[11] C. E. Blair and R. G. Jeroslow, *The value function of a mixed integer program: I*, Discrete Mathematics, 19 (1977), pp. 121–138.
[12] M. Bodur, S. Dash, O. Günlük, and J. Luedtke, *Strengthened benders cuts for stochastic integer programs with continuous recourse*, INFORMS Journal on Computing, 29 (2017), pp. 77–91.
[13] N. Boland, J. Christiansen, B. Dandurand, A. Eberhard, and F. Oliveira, *A parallelizable augmented Lagrangian method applied to large-scale non-convex-constrained optimization problems*, Mathematical Programming, 175 (2019), pp. 503–536.
[14] R. S. Burachik, R. N. Gasimov, N. A. Ismayilova, and C. Y. Kaya, *On a modified subgradient algorithm for dual problems via sharp augmented Lagrangian*, Journal of Global Optimization, 34 (2006), pp. 55–78.
[15] R. S. Burachik, A. N. Iusem, and J. G. Melo, *A primal dual modified subgradient algorithm with sharp Lagrangian*, Journal of Global Optimization, 46 (2010), pp. 347–361.
[16] R. S. Burachik, A. N. Iusem, and J. G. Melo, *An inexact modified subgradient algorithm for primal-dual problems via augmented Lagrangians*, Journal of Optimization Theory and Applications, 157 (2013), pp. 108–131.
[17] A. Camisa, I. Notarnicola, and G. Notarstefano, *A primal decomposition method with suboptimality bounds for distributed mixed-integer linear programming*, in 2018 IEEE Conference on Decision and Control (CDC), IEEE, 2018, pp. 3391–3396.
[18] C. C. Carøe and R. Schultz, *Dual decomposition in stochastic integer programming*, Operations Research Letters, 24 (1999), pp. 37–45.
[19] C. C. Carøe and J. Tind, *L-shaped decomposition of two-stage stochastic programs with integer recourse*, Mathematical Programming, 83 (1998), pp. 451–464.
[20] B. Chen, S. Küçükyavuz, and S. Sen, *Finite disjunctive programming characterizations for general mixed-integer linear programs*, Operations Research, 59 (2011), pp. 202–210.
[21] B. Chen, S. Küçükyavuz, and S. Sen, *A computational study of the cutting plane tree algorithm for general mixed-integer linear programs*, Operations research letters, 40 (2012), pp. 15–19.
[22] R. Chen and J. Luedtke, *On generating lagrangian cuts for two-stage stochastic integer programs*, INFORMS Journal on Computing, (2022).
[23] M. Cordova, W. de Oliveira, and C. Sagastizábal, *Revisiting augmented Lagrangian duals*, Available on: http://www.optimization-online.org/DB_HTML/2020/03/7709.html, (2020).
[24] I. Dunning, J. Huchette, and M. Lubin, *Jump: A modeling language for mathematical optimization*, SIAM review, 59 (2017), pp. 295–320.
[25] J. Eckstein and D. P. Bertsekas, *On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators*, Mathematical Programming, 55 (1992), pp. 293–318.
[26] M. J. Feizollahi, S. Ahmed, and A. Sun, *Exact augmented Lagrangian duality for mixed*

integer linear programming, Mathematical Programming, 161 (2017), pp. 365–387.
[27] D. Gabay, *Applications of the method of multipliers to variational inequalities, in,(1983), 299. doi: 10.1016*, S0168-2024 (08), pp. 70034–1.
[28] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers & Mathematics with Applications, 2 (1976), pp. 17–40.
[29] D. Gade, S. Küçükyavuz, and S. Sen, *Decomposition algorithms with parametric gomory cuts for two-stage stochastic integer programs*, Mathematical Programming, 144 (2014), pp. 39–64.
[30] R. N. Gasimov, *Augmented Lagrangian duality and nondifferentiable optimization methods in nonconvex programming*, Journal of Global Optimization, 24 (2002), pp. 187–203.
[31] R. Glowinski and A. Marroco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires*, Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique, 9 (1975), pp. 41–76.
[32] L. Gurobi Optimization, *Gurobi optimizer reference manual*, 2022.
[33] M. R. Hestenes, *Multiplier and gradient methods*, Journal of optimization theory and applications, 4 (1969), pp. 303–320.
[34] X. Huang and X. Yang, *A unified augmented Lagrangian approach to duality and exact penalization*, Mathematics of Operations Research, 28 (2003), pp. 533–552.
[35] Q. Huangfu and J. J. Hall, *Parallelizing the dual revised simplex method*, Mathematical Programming Computation, 10 (2018), pp. 119–142.
[36] B. Jiang, T. Lin, S. Ma, and S. Zhang, *Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis*, Computational Optimization and Applications, 72 (2019), pp. 115–157.
[37] P. Kall, S. W. Wallace, and P. Kall, *Stochastic programming*, vol. 6, Springer, 1994.
[38] Y. Kanno and S. Fujita, *Alternating direction method of multipliers for truss topology optimization with limited number of nodes: A cardinality-constrained second-order cone programming approach*, Optimization and Engineering, 19 (2018), pp. 327–358.
[39] Y. Kanno and S. Kitayama, *Alternating direction method of multipliers as a simple effective heuristic for mixed-integer nonlinear optimization*, Structural and Multidisciplinary Optimization, 58 (2018), pp. 1291–1295.
[40] G. Laporte and F. V. Louveaux, *The integer l-shaped method for stochastic integer programs with complete recourse*, Operations research letters, 13 (1993), pp. 133–142.
[41] C. Li and I. E. Grossmann, *An improved l-shaped method for two-stage convex 0–1 mixed integer nonlinear stochastic programs*, Computers & Chemical Engineering, 112 (2018), pp. 165–179.
[42] C. Li and I. E. Grossmann, *A generalized benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables*, Journal of Global Optimization, 75 (2019), pp. 247–272.
[43] C. Malherbe and N. Vayatis, *Global optimization of lipschitz functions*, in International Conference on Machine Learning, PMLR, 2017, pp. 2314–2323.
[44] D. Q. Mayne and E. Polak, *Outer approximation algorithm for nondifferentiable optimization problems*, Journal of Optimization Theory and Applications, 42 (1984), pp. 19–30.
[45] J. G. Melo and R. D. Monteiro, *Iteration-complexity of a linearized proximal multi-block admm class for linearly constrained nonconvex optimization problems*, Available on: http://www.optimization-online.org/DB_HTML/2017/04/5964.html, (2017).
[46] R. D. Monteiro and B. F. Svaiter, *Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers*, SIAM Journal on Optimization, 23 (2013), pp. 475–507.
[47] L. Ntaimo, *Fenchel decomposition for stochastic mixed-integer programming*, Journal of Global Optimization, 55 (2013), pp. 141–163.
[48] L. Ntaimo and S. Sen, *The million-variable "march" for stochastic combinatorial optimization*, Journal of Global Optimization, 32 (2005), pp. 385–400.
[49] M. J. Powell, *A method for nonlinear constraints in minimization problems*, R. Fletcher, ed., Optimization, (1969), pp. 283–298.
[50] Y. Qi and S. Sen, *The ancestral benders cutting plane algorithm with multi-term disjunctions for mixed-integer recourse decisions in stochastic programming*, Mathematical Programming, 161 (2017), pp. 193–235.
[51] R. Rahmaniani, S. Ahmed, T. G. Crainic, M. Gendreau, and W. Rei, *The benders dual decomposition method*, Operations Research, 68 (2020), pp. 878–895.
[52] R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei, *The benders decomposition*

algorithm: A literature review, European Journal of Operational Research, 259 (2017), pp. 801–817.

[53] R. T. Rockafellar, *The multiplier method of Hestenes and Powell applied to convex programming*, Journal of Optimization Theory and applications, 12 (1973), pp. 555–562.

[54] R. T. Rockafellar, *Augmented Lagrangians and applications of the proximal point algorithm in convex programming*, Mathematics of operations research, 1 (1976), pp. 97–116.

[55] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, vol. 317, Springer Science & Business Media, 2009.

[56] A. Ruszczynski, *Nonlinear Optimization*, Princeton university press, 2011.

[57] R. Schultz, L. Stougie, and M. H. Van Der Vlerk, *Solving stochastic programs with integer recourse by enumeration: A framework using gröbner basis*, Mathematical Programming, 83 (1998), pp. 229–252.

[58] S. Sen and J. L. Higle, *The c 3 theorem and a d 2 algorithm for large scale stochastic mixed-integer programming: Set convexification*, Mathematical Programming, 104 (2005), pp. 1–20.

[59] S. Sen and H. D. Sherali, *Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming*, Mathematical Programming, 106 (2006), pp. 203–223.

[60] H. D. Sherali and B. M. Fraticelli, *A modification of benders' decomposition algorithm for discrete subproblems: An approach for stochastic programs with integer recourse*, Journal of Global Optimization, 22 (2002), pp. 319–342.

[61] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, *On the linear convergence of the admm in decentralized consensus optimization*, IEEE Transactions on Signal Processing, 62 (2014), pp. 1750–1761.

[62] K. Sun and X. A. Sun, *A two-level distributed algorithm for nonconvex constrained optimization*, Computational Optimization and Applications, (2022), pp. 1–41.

[63] R. Takapoui, *The alternating direction method of multipliers for mixed-integer optimization applications*, PhD thesis, Stanford University, 2017.

[64] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, *A simple effective heuristic for embedded mixed-integer quadratic programming*, International Journal of Control, 93 (2020), pp. 2–12.

[65] N. van der Laan and W. Romeijnders, *A converging benders' decomposition algorithm for two-stage mixed-integer recourse models*, 2020.

[66] R. M. Van Slyke and R. Wets, *L-shaped linear programs with applications to optimal control and stochastic programming*, SIAM journal on applied mathematics, 17 (1969), pp. 638–663.

[67] Y. Wang, W. Yin, and J. Zeng, *Global convergence of ADMM in nonconvex nonsmooth optimization*, Journal of Scientific Computing, 78 (2019), pp. 29–63.

[68] B. Wu and B. Ghanem, *ℓ_p-box ADMM: A versatile framework for integer programming*, IEEE transactions on pattern analysis and machine intelligence, 41 (2018), pp. 1695–1708.

[69] Y. Xu and W. Yin, *A globally convergent algorithm for nonconvex optimization based on block coordinate update*, Journal of Scientific Computing, 72 (2017), pp. 700–734.

[70] A. K. Yadav, R. Ranjan, U. Mahbub, and M. C. Rotkowitz, *New methods for handling binary constraints*, in 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), IEEE, 2016, pp. 1074–1080.

[71] Y. Yao, X. Zhu, H. Dong, S. Wu, H. Wu, L. C. Tong, and X. Zhou, *ADMM-based problem decomposition scheme for vehicle routing problem with time windows*, Transportation Research Part B: Methodological, 129 (2019), pp. 156–174.

[72] S. Zhang and X. A. Sun, *Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization*, Mathematical Programming, 196 (2022), pp. 935–985.

[73] J. Zou, S. Ahmed, and X. A. Sun, *Stochastic dual dynamic integer programming*, Mathematical Programming, 175 (2019), pp. 461–502.