Multiblock ADMM Heuristics for Mixed-Binary Optimization on Classical and Quantum Computers

CLAUDIO GAMBELLA [ID] AND ANDREA SIMONETTO [ID] (Member, IEEE)

IBM Quantum, IBM Research Ireland, Mulhuddart D15 HN66, Ireland

Corresponding author: Claudio Gambella (claudio.gambella1@ie.ibm.com).

ABSTRACT Solving combinatorial optimization problems on current noisy quantum devices is currently being advocated for (and restricted to) binary polynomial optimization with equality constraints via quantum heuristic approaches. This is achieved using, for example, the variational quantum eigensolver (VQE) and the quantum approximate optimization algorithm (QAOA). In this article, we present a decomposition-based approach to extend the applicability of current approaches to "quadratic plus convex" mixed binary optimization (MBO) problems, so as to solve a broad class of real-world optimization problems. In the MBO framework, we show that the alternating direction method of multipliers (ADMM) can split the MBO into a binary unconstrained problem (that can be solved with quantum algorithms), and continuous constrained convex subproblems (that can be solved cheaply with classical optimization solvers). The validity of the approach is then showcased by numerical results obtained on several optimization problems via simulations with VQE and QAOA on the quantum circuits implemented in Qiskit, an open-source quantum computing software development framework.

INDEX TERMS Mathematical programming, optimization, quantum computing.

I. INTRODUCTION

Mixed-binary optimization (MBO) has been studied for decades in mathematical programming, because of the widespread range of applications in several domains [1]–[3], and the inherent difficulties posed by integer variables. MBO is known to be NP-hard in the general case.

The MBO class is very broad, and tailored exact or heuristic solution approaches have been devised in classical computation, depending on the nature and structure of the specific formulation [4], [5]. Recently, the advances in universal quantum computing [6]–[9] fostered efforts to understand whether this alternative computing paradigm could offer advantages (e.g., faster exact algorithms, more reliable heuristics) to solving combinatorial optimization problems [10]. Research directed to apply the resulting algorithms to early generation of universal quantum computers has mainly focused on quantum variational approaches [6], which have been applied to chemistry [11], [12], machine learning [13], [14], and mathematical optimization [7], [15], [16]. In broad terms, a variational approach works by choosing a parametrization of the space of quantum states that depends on a relatively small set of parameters, then using classical optimization routines to determine values of the parameters corresponding to a quantum state that maximizes or minimizes a given utility function. Typically, the utility function is given by a Hamiltonian encoding the total energy of the system, to be minimized. The variational theorem ensures that the expectation value of the Hamiltonian is greater than or equal to the minimum eigenvalue of the Hamiltonian. Such variational approaches can be applied for solving combinatorial optimization problems, provided that we can construct a Hamiltonian encoding the objective function of the optimization problem. See [17] and [18]. In the mathematical optimization context, research has been directed mainly to quadratic unconstrained binary optimization (QUBO) problems

$$\underset{x}{\text{minimize}} \quad c^{\mathsf{T}}x + x^{\mathsf{T}}Qx$$

$$\text{subject to:} \quad x \in \{0, 1\}^n, \quad \text{with } c \in \mathbb{R}^n, Q \in \mathbb{R}^{n \times n}$$

which can be transformed into an Ising model with Hamiltonian constituted as a summation of weighted tensor products of Z Pauli operators. In case equality constraints $Ax = b$ are required to be modeled, a QUBO can still be devised by

adding a quadratic penalization $\alpha\|Ax - b\|^2$ of the equality constraints to the objective function, as a soft-constraint in an augmented Lagrangian fashion [18]–[20].

A typical quantum variational approach, such as VQE [11] would involve two key steps in solving a QUBO, given its Ising Hamiltonian H formulation. First, one would parametrize the quantum state via a small set of rotation parameters θ: the state can then be expressed as $|\psi(\theta)\rangle = U(\theta)|0\rangle$, where $U(\theta)$ is the parametrized quantum circuit applied to the initial state $|0\rangle$. The variational approach would then aim at solving $\min_\theta \langle\psi(\theta)|H|\psi(\theta)\rangle$. Such optimization can be performed in a hybrid setting that uses a classical computer running an iterative algorithm to select θ, and a quantum computer to compute information about $\langle\psi(\theta)|H|\psi(\theta)\rangle$ for given θ (e.g., its gradient).

In the MBO formulations, continuous variables and inequality constraints are typically both required to be modeled. Tackling a general MBO problem with quantum variational approaches is at its early stages. One possibility is to introduce slack-based formulations, and consider the slacks as additional continuous parameters for the quantum QUBO solvers [21]. While Grover searches have been applied for some continuous optimization problems [22], and quantum annealing allows for tackling inequality constraints [23], the potential of quantum optimization algorithms for subclasses of MBO problems has not been investigated with a principled approach yet.

In this article, we aim at extending the quantum optimization methodologies to be able to cope with MBOs on current quantum devices. As a matter of fact, we start in a bit more general context and we pose as an assumption that an approximate or noisy QUBO solver oracle is available to approximately solve QUBOs with some degree of suboptimality; then, we ask ourselves where we can go from there. In particular, we explore ways to (heuristically) approximately solve certain classes of MBOs with the assumed noisy QUBO solver. The aim of this article is as follows:

1) to extend the quantum optimization methodologies to cope with MBOs;
2) to propose new heuristics to solve MBOs on quantum computers, having the potential to scale, in the future, to larger sizes than heuristics on classical computers;
3) to offer a glimpse on current research in combinatorial optimization in quantum computing, along with assumptions, challenges, and open problems.

The proposed heuristics are based on the celebrated alternating direction method of multipliers (ADMM), see [24]–[35]. ADMM is an operator splitting algorithm that has a long history in convex optimization. ADMM is known to have residual, objective, and dual variable convergence properties, provided that convexity assumptions are holding [24]. Recently, nonconvex variants of ADMM have been developed (see [36]–[42] for nonconvex/combinatorial theoretical results and ADMM-inspired heuristics). In the heuristic

framework of [36] for problems with convex objective and decision variables from a nonconvex set, ADMM relies on a (eventually approximate) projection on the nonconvex feasibility set, and the ADMM iterates are improved via local search (LS) methods. Our method instead does not involve a projection step and makes use of the ADMM operator-splitting procedure to devise a decomposition for certain classes of MBOs into the following:

1) a QUBO subproblem to be solved by a QUBO (approximate) solver, e.g., on noisy quantum devices via quantum variational algorithms, such as VQE [11], QAOA [8], or with Grover-search-based algorithms [43], or quantum-based semidefinite programming (SDP) relaxations [44].
2) a convex constrained subproblem, which can be efficiently solved with classical optimization solvers [45].

Our method builds upon the recent results of [37] and [38], which propose global convergence guarantees for nonconvex and nonsmooth optimization problems. For MBO, the convergence results of [37] would not hold because of the requirement on the Lipschitz continuity of specific components of the objective function. However, the authors in [38] and [46] observed that a third block can be added to the two-block decomposition of ADMM in order to gain convergence properties to stationary points. Our method also leverages the recent results of [47] on related tame problems to ensure that convergence is attained to a unique stationary point, by taking advantage of the semialgebraic structure of the problem. Possible extensions to our current setting might be offered by [48] and [49], but not pursued here.

The mathematical contribution of this article is a multiblock ADMM heuristic (M-ADMM-H) algorithm for MBO, for which we present the following points:

1) a decomposition approach suitable for computation on current quantum devices;
2) conditions for convergence, feasibility, and optimality.

The quantum computing contribution of this article is instead the analysis of the mentioned heuristic on quantum devices, and in particular, of the following:

1) computational results for classical and quantum implementations, including comments on the solution quality achieved;
2) analysis of errors coming from an approximate solution of the underlying QUBO problem, which are typical in noisy (quantum) machines.

Despite the quantum-oriented angle of this article, our results applied also to classical computing, whenever a classical QUBO solver is available, or whenever the QUBO subproblems are easy or trivially solvable.

This article is organized as follows. Section II outlines current research effort in optimization in quantum computing, especially for QUBO solvers and the possible speed-up with respect to classical computing. Section III reviews the

ADMM proposed for convex optimization and introduces a two and three-block implementation of ADMM when binary variables are present. In Section IV, the two and three-block implementations are then specified for MBO problems and convergence properties are illustrated. To get a better picture of the proposed algorithm, Section V discusses small-sized examples. The two illustrative MBO formulations in Section VI are solved with the proposed ADMM-based algorithms in Section VII, discussing the results obtained on simulated quantum devices. Finally, conclusions are drawn in Section VIII.

Notation: Notation is whenever possible standard and borrowed from convex analysis [45], [50]. Vectors $x \in \mathbb{R}^n$, matrices $A \in \mathbb{R}^{n \times m}$, and sets $\mathcal{X} \subseteq \mathbb{R}^n$. For vectors and matrices, $(\cdot)^{\mathsf{T}}$ indicates the transpose operation. Functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, whose codomain is the extended real line. A function is convex iff its epigraph is a convex set. A convex function is closed if its epigraph is closed, and it is proper if its effective domain is nonempty and it never attains $-\infty$. A function is lower semicontinuous if and only if all of its sublevel sets $\{x \in \mathcal{X} : f(x) \leq y\}$ are closed. A proper convex function is closed iff it is lower semicontinuous. The indicator function of the set $\mathcal{X}$ is a function $\iota_{\mathcal{X}} : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, for which $\iota_{\mathcal{X}}(x) = 0$ if $x \in \mathcal{X}$ and $+\infty$ otherwise. The indicator function of any closed set is lower semicontinuous.

II. QUANTUM COMPUTING FOR QUBOS

We briefly review here the current efforts in solving QUBOs via quantum/classical approaches. The aim of this section is to further expand on the state of the art, discuss what it is meant with potential quantum advantage of quantum solvers versus classical solvers and why this is important for MBOs.

First of all, we look at current noisy quantum computers and quantum optimization algorithms, in particular VQE and QAOA. We do not touch upon quantum annealing, but the interested reader can find studies in the works of [51]–[54] and references therein. VQE and QAOA are currently under major scrutiny and discussions on their performance, e.g., [9], [18], [55]–[58], especially for the solution of unconstrained binary optimization problems, of which MaxCut is one important embodiment. On one hand, it seems that, in general, the QAOA performance is still not well understood under all circumstances, and (even) some local classical algorithms could outperform it in some cases [59]. On the other hand, some encouraging results in terms of performance advantage of QAOA with respect to the classical Goemans–Williamson limit have appeared [60]. In addition, the combination of hyperparametrization and multistart strategies has shown promising results in escaping local optima [61]. These considerations, together with the fact that QAOA is not efficiently simulatable by classical computers, make QAOA an appealing algorithm to explore on noisy quantum machines.

Second, looking ahead to less noisy quantum computers, quadratic speedup for Grover-based quantum QUBO solvers has been demonstrated in [43], with respect to a classical unstructured search. While solving a large QUBO with an unstructured search is unrealistic for classical computers, we can envision employing such Grover-based quantum QUBO solvers for subproblems coming from classical branch-and-bound methodologies. This would generate a quantifiable speedup.

Third, looking further ahead to even less noisy quantum computers, quadratic speedup for quantum semidefinite programming (Q-SDP) relaxations arising in QUBO solving has been shown in [44]. Given that currently SDP relaxations are one of the workhorses for QUBO solving, this new Q-SDP relaxations have the real potential of offering tangible speedup in solving QUBOs.

Given such wealth of results and studies, there is a significant interest to see if and how one can leverage quantum QUBO solvers to tackle more complicated constrained problems. This is what we are going to do next.

III. FROM THE STANDARD ADMM TO A THREE-BLOCK STRUCTURE

A. CONVEX ADMM

We start with some background on ADMM and the known results in the case of nonconvex and combinatorial problems. Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $h : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ be closed convex proper functions, and let $A \in \mathbb{R}^{n \times p}$, $B \in \mathbb{R}^{m \times p}$ be given matrices. The prototypical problem we are interested in is of the form

$$\underset{x \in \mathbb{R}^n, y \in \mathbb{R}^m}{\text{minimize}} \quad f(x) + h(y) \tag{1a}$$

$$\text{subject to:} \quad Ax + By = 0. \tag{1b}$$

Then, the ADMM is the following algorithm:

1) initialize the sequences $(x_k)_{k \in \mathbb{N}}$, $(y_k)_{k \in \mathbb{N}}$, $(\lambda_k)_{k \in \mathbb{N}}$ as $x_0 \in \mathbb{R}^n$, $y_0 \in \mathbb{R}^m$, $\lambda_0 \in \mathbb{R}^p$. Choose a penalty parameter $\varrho > 0$;
2) for $k = 1, 2, \ldots,$ do:
    a) First block update

$$\begin{aligned} x_k = \underset{x \in \mathbb{R}^n}{\arg\min} \ & f(x) \\ & + \lambda_{k-1}^{\mathsf{T}} (Ax + By_{k-1}) \\ & + \frac{\varrho}{2} \|Ax + By_{k-1}\|_2^2. \end{aligned} \tag{2a}$$

    b) Second block update

$$\begin{aligned} y_k = \underset{y \in \mathbb{R}^m}{\arg\min} \ & h(y) \\ & + \lambda_{k-1}^{\mathsf{T}} (Ax_k + By) \\ & + \frac{\varrho}{2} \|Ax_k + By\|_2^2. \end{aligned} \tag{2b}$$

    c) Dual variable update

$$\lambda_k = \lambda_{k-1} + \varrho(Ax_k + By_k). \tag{2c}$$

For the classical ADMM, we have various convergence and convergence rate results. For an ample classes of convex

costs, ADMM converges for any ϱ, that is, starting from any x_0, y_0, λ_0, it generates a sequence for which we have the following.

1) *Residual convergence:* $Ax_k + By_k \rightarrow 0$ as $k \rightarrow \infty$, i.e., the iterates approach feasibility.
2) *Objective convergence:* $f(x_k) + h(y_k) \rightarrow p^*$ as $k \rightarrow \infty$, i.e., the objective function of the iterates approaches the optimal value.
3) *Dual variable convergence:* $\lambda_k \rightarrow \lambda^*$ as $k \rightarrow \infty$, where λ^* is a dual optimal point.

See, for instance, [24] and [62], while for convergence rate analysis see, for example, [30] and [35].

Nonconvex results (when the cost functions are nonconvex) are less ubiquitous in the literature and, in general, more restrictive in terms of assumptions. However, ADMM still behaves quite favourably in nonconvex cases and attracts a considerable amount of attention from the research community.

B. MIXED-BINARY ADMM

In this article, we start by modifying (1) by considering that x is now constrained to live in the nonconvex set $\{0, 1\}^n$, or equivalently that each of the component of the vector x, i.e., $x_{(i)}, i \in \{1, \ldots, n\}$, is constrained as $x_{(i)}(1 - x_{(i)}) = 0$. We compactly write this as requiring $x \in \mathcal{X}$, where $\mathcal{X}$ represents the said binary set.

Let now $\iota_{\mathcal{X}} : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ be the indicator function of the set $\mathcal{X}$, which is by construction closed and proper (but obviously nonconvex), and consider the new function

$$f^{NC}(x) = f(x) + \iota_{\mathcal{X}}(x).$$

The function $f^{NC}(x)$ is nonconvex by construction, yet one could still attempt at using the ADMM approach (2) with the new function $f^{NC}(x)$ in lieu of the "old" one $f(x)$, with the goal of solving the MBO

$$\underset{x \in \mathcal{X}, y \in \mathbb{R}^m}{\text{minimize}} \quad f(x) + h(y) \tag{3a}$$

$$\text{subject to:} \quad Ax + By = 0. \tag{3b}$$

This is in general a heuristic. However, under some more restricting conditions the sequence generated by ADMM converges also in this case as follows.

Theorem 1 (Convergence of mixed-binary ADMM [37]): Consider the following assumptions.

A1) (Coercivity). The objective function $f^{NC}(x) + h(y)$ is coercive over the set $Ax + By = 0$.
A2) (Feasibility). $\text{Im}(A) \subseteq \text{Im}(B)$, where $\text{Im}(\cdot)$ returns the image of a matrix.
A3) (Lipschitz subminimization paths). There exists a positive constant $\bar{M}$, such that for any iterate counters k_1 and k_2, we have

$$\|x_{k_1} - x_{k_2}\| \leq \bar{M}\|Ax_{k_1} - Ax_{k_2}\|$$

$$\|y_{k_1} - y_{k_2}\| \leq \bar{M}\|By_{k_1} - By_{k_2}\|. \tag{4}$$

A4) (Objective f-regularity). Function $f^{NC}(x)$ is lower semicontinuous.
A5) (Objective h-regularity). Function $h(y)$ is Lipschitz differentiable with constant L_h.

Define the augmented Lagrangian

$$\mathcal{L}_\varrho(x, y, \lambda) = f^{NC}(x) + h(y) + \lambda^{\mathsf{T}}(Ax + By)$$

$$+ \frac{\varrho}{2}\|Ax + By\|_2^2.$$

Then, Binary ADMM converges subsequently for any sufficiently large ϱ, that is, starting from any x_0, y_0, λ_0, it generates a sequence that is bounded, has at least one limit point, and that each limit point (x^*, y^*, λ^*) is a stationary point of $\mathcal{L}_\varrho$, namely, $0 \in \partial \mathcal{L}_\varrho(x^*, y^*, \lambda^*)$.

In addition, if $\mathcal{L}_\varrho$ is a Kurdyka–Łojasiewicz (KŁ) function [63]–[65], then (x_k, y_k, λ_k) converges globally to the unique limit point (x^*, y^*, λ^*).

Theorem 1 is a special case of the more general [37, Th. 1] adapted to our problem setting (and where we have chosen to use a stronger version of A3) for sake of clarity and ease of implementation). Functions satisfying the KŁproperty are, for example, semialgebraic functions and locally strongly convex functions. We recall that a semialgebraic function can be defined based on its graph as follows.

Definition 1 (See [66]): A subset of $\mathbb{R}^n$ is called semialgebraic if it can be written as a finite union of sets of the form

$$\{x \in \mathbb{R}^n : p_i(x) = 0, q_i(x) < 0, i = 1, \ldots, p\}$$

where p_i, q_i are real polynomial functions.

A function $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is semialgebraic if its graph is a semialgebraic subset of $\mathbb{R}^{n+1}$.

The following hold:

1) finite sums and products of semialgebraic functions are semialgebraic;
2) scalar products are semialgebraic;
3) indicator functions of semialgebraic sets are semialgebraic;
4) generalized inverse of semialgebraic mappings are semialgebraic;
5) composition of semialgebraic functions or mappings are semialgebraic; see [66].

From this discussion, $\iota_{\mathcal{X}}(x)$ (besides being lower semicontinuous) is semialgebraic, since it is the indicator functions of semialgebraic sets $\{x_{(i)}(1 - x_{(i)}) = 0\} \, \forall \, i$, and $f^{NC}(x)$ is semialgebraic if $f(x)$ is semialgebraic.

Theorem 1 (or its broader version) is fairly tight, and counter-examples exists in which some of the assumptions are not verified and ADMM fails to converge. Relaxing some of the assumptions, for example, A3), is a topic of current research, e.g., by leveraging the slightly different setting in [48].

To understand better the implications of 1, we consider a toy example, which verifies all the assumptions of the theorem.

Example 1: Consider the problem

$$\min_{v\in\{0,1\},w\in\mathbb{R}} -2v + w^2, \text{ subject to: } v = w. \quad (5)$$

The unique optimal solution is $v^* = w^* = 1$. If we apply ADMM to it, as for Theorem 1, we can obtain convergence for sufficiently large ϱ starting from any initial v_0, w_0, λ_0. For example, we can start with $v_0 = 1$, $w_0 = 1$, $\lambda_0 = 0$ with $\varrho = 100$. Then, we can see that the ADMM algorithm converges to the solution $v = w = 0 = \lambda = 0$, which is a stationary point of the augmented Lagrangian $\mathcal{L}_\varrho(v, w, \lambda)$. If we start with a different starting point $v_0 = 0$, $w_0 = 0.5, \lambda_0 = 0$ with the same ϱ, then convergence is attained to the point $v = w = 1, \lambda = 2$, which is the optimal solution of the original problem, and another stationary point of the augmented Lagrangian.

From the abovementioned example, one can understand the implications of convergence of ADMM in the nonconvex setting, where one may converge to a feasible point, but not necessarily optimal for the problem. This is in general not a very unsatisfactory behaviour, especially in nonconvex setting, where one is often concerned about finding "good" feasible points.

C. MIXED-BINARY THREE-BLOCK ADMM

We move now to generalize the mixed-binary ADMM to three-block implementation. The reason behind the three blocks is that the assumptions in Theorem 1 are restrictive for MBO problems and they would not be satisfied in general (as we see later).

Consider the prototypical (mixed-binary) problem

$$\underset{x\in\mathcal{X},\bar{x}\in\mathbb{R}^l,y\in\mathbb{R}^m}{\text{minimize}} \quad f_0(x) + f_1(\bar{x}) + h(y) \quad (6a)$$

$$\text{subject to: } A_0 x + A_1 \bar{x} + By = 0 \quad (6b)$$

where we have introduced the functions $f_0 : \mathbb{R}^n \to \mathbb{R}$, $f_1 : \mathbb{R}^l \to \mathbb{R}$, the matrices $A_0 \in \mathbb{R}^{n\times p}$, $A_1 \in \mathbb{R}^{l\times p}$ and we have put ourselves already in the mixed-binary case. (For completeness, recall the definition of function $h : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ as closed convex proper function, and matrix $B \in \mathbb{R}^{m\times p}$.)

Then, the three-block ADMM is the following algorithm.

1) Initialize the sequences $(x_k)_{k\in\mathbb{N}}$, $(\bar{x}_k)_{k\in\mathbb{N}}$, $(y_k)_{k\in\mathbb{N}}$, $(\lambda_k)_{k\in\mathbb{N}}$ as $x_0 \in \mathbb{R}^n$, $\bar{x}_0 \in \mathbb{R}^l$ $y_0 \in \mathbb{R}^m$, $\lambda_0 \in \mathbb{R}^p$. Choose a penalty parameter $\varrho > 0$.
2) For $k = 1, 2, \dots$ do:
   1) First block update

$$x_k = \arg\min_{x\in\mathbb{R}^n} f_0(x) + \iota_\mathcal{X}(x)$$

$$+ \lambda_{k-1}^\mathsf{T} (A_0 x + A_1 \bar{x}_{k-1} + By_{k-1})$$

$$+ \frac{\varrho}{2} \|A_0 x + A_1 \bar{x}_{k-1} + By_{k-1}\|_2^2. \quad (7a)$$

2) Second block update

$$\bar{x}_k = \arg\min_{\bar{x}\in\mathbb{R}^l} f_1(\bar{x}) + \lambda_{k-1}^\mathsf{T} (A_0 x_k + A_1 \bar{x} + By_{k-1})$$

$$+ \frac{\varrho}{2} \|A_0 x_k + A_1 \bar{x} + By_{k-1}\|_2^2. \quad (7b)$$

3) Third block update

$$y_k = \arg\min_{y\in\mathbb{R}^m} h(y) + \lambda_{k-1}^\mathsf{T} (A_0 x_k + A_1 \bar{x}_k + By)$$

$$+ \frac{\varrho}{2} \|A_0 x_k + A_1 \bar{x}_k + By\|_2^2. \quad (7c)$$

4) Dual variable update

$$\lambda_k = \lambda_{k-1} + \varrho(A_0 x_k + A_1 \bar{x}_k + By_k). \quad (7d)$$

This is in general a heuristic. However, under some more restricting conditions the sequence generated by ADMM converges also in this case as follows.

Theorem 2 (Convergence of Mixed-Binary Three-Block ADMM [37]): Consider the following assumptions.

A1) (Coercivity). The objective function $f_0^{\text{NC}}(x) + f_1(\bar{x}) + h(y)$ is coercive over the set $A_0 x + A_1 \bar{x} + By = 0$, where we have defined $f_0^{\text{NC}}(x) := f_0(x) + \iota_\mathcal{X}(x)$.
A2) (Feasibility). $\text{Im}(A) \subseteq \text{Im}(B)$, where $A = [A_0, A_1]$.
A3) (Lipschitz subminimization paths). There exists a positive constant $\bar{M}$, such that for any iterate counters k_1 and k_2, we have

$$\|x_{k_1} - x_{k_2}\| \le \bar{M}\|A_0 x_{k_1} - A_0 x_{k_2}\|$$

$$\|\bar{x}_{k_1} - \bar{x}_{k_2}\| \le \bar{M}\|A_1 \bar{x}_{k_1} - A_1 \bar{x}_{k_2}\|$$

$$\|y_{k_1} - y_{k_2}\| \le \bar{M}\|By_{k_1} - By_{k_2}\|. \quad (8)$$

A4) (Objective f-regularity). Function $f_0^{\text{NC}}(x)$ is lower semicontinuous and $f_1(\bar{x})$ is restricted prox-regular.
A5) (Objective h-regularity). Function $h(y)$ is Lipschitz differentiable with constant L_h.

Define the augmented Lagrangian

$$\mathcal{L}_\varrho(x, \bar{x}, y, \lambda) = f_0^{\text{NC}}(x) + f_1(\bar{x}) + h(y)$$

$$+ \lambda^\mathsf{T}(A_0 x + A_1 \bar{x} + By)$$

$$+ \frac{\varrho}{2} \|A_0 x + A_1 \bar{x} + By\|_2^2. \quad (9)$$

Then, mixed-binary three-block ADMM converges subsequently for any sufficiently large ϱ, that is, starting from any $x_0, \bar{x}_0, y_0, \lambda_0$, it generates a sequence that is bounded, has at least one limit point, and that each limit point $(x^*, \bar{x}^*, y^*, \lambda^*)$ is a stationary point of $\mathcal{L}_\varrho$, namely, $0 \in \partial\mathcal{L}_\varrho(x^*, \bar{x}^*, y^*, \lambda^*)$.

In addition, if $\mathcal{L}_\varrho$ is a KŁfunction [63]–[65], then $(x_k, \bar{x}_k, y_k, \lambda_k)$ converges globally to the unique limit point $(x^*, \bar{x}^*, y^*, \lambda^*)$.

Theorem 2 is a special case of the more general [37, Th. 1] adapted to our problem setting (and where we have chosen to use a stronger version of A3) for sake of clarity and ease

of implementation). Functions satisfying the restricted prox-regularity assumptions are, for example, convex functions, including indicator functions of convex sets (which will be the ones that we will use in the sequel).

What is now fundamental in the three-block ADMM is that we can restrict variable x to be binary, and shift all the other constraints on $\bar{x}$ (any restricted prox-regular constraints, e.g., linear inequalities). This without affecting the variable y, which stays unconstrained, and whose function $h(y)$ needs to be smooth (so one cannot add an indicator function to represent additional constraint there). This "trick" was first explored in [38] in the context of distributed computations and discussed in the following example.

Example 2: Consider the problem

$$\min_{v \in \{0,1\}, w \geq 1/2} -2v + w^2, \text{ subject to: } v = w. \quad (10)$$

This problem does not satisfy the assumptions of Theorem 1, since y is now constrained (although ADMM here is nonetheless converging in practice). But a possible way to overcome this (without adding constraints on the binary variable v), is to use the relaxed problem

$$\min_{v \in \{0,1\}, \bar{v} \geq 1/2, w \in \mathbb{R}} -2v + \bar{v}^2 + \frac{\beta}{2} w^2, \text{ subject to: } v = \bar{v} + w$$

$$(11)$$

for a large $\beta < \varrho$.

Starting $v_0 = \bar{v}_0 = w_0 = \lambda_0 = 0$ with $\beta = 1000$ and $\rho = 1001$, we obtain a sequence converging to $v = 1, \bar{v} = 0.998, w = 0.002, \lambda = 1.996$, which is close to the optimal solution of the original problem.

In [38], a proper dualization of the constraint $w = 0$ is imposed, but the convergence of the then two-level approach has more restricting assumptions that the ones that we consider here, in particular $\bar{v}$ needs to be constrained in an hypercube [38, As. 4.2].

IV. TWO AND THREE-BLOCK ADMM ALGORITHMS FOR MBO

A. FROM MBOS TO TWO-BLOCK ADMM

We are now ready to tackle MBOs. In this article, we will consider the following reference problem (P):

$$\underset{x \in \mathcal{X}, u \in \mathcal{U} \subseteq \mathbb{R}^l}{\text{minimize}} \quad q(x) + \varphi(u) \quad (12a)$$

$$\text{subject to: } Gx = b, \quad g(x) \leq 0 \quad (12b)$$

$$\ell(x, u) \leq 0 \quad (12c)$$

with the corresponding functional assumptions.

Assumption 1 (Functional Assumptions): The following assumptions hold.

1) Function $q : \mathbb{R}^n \to \mathbb{R}$ is quadratic, i.e., $q(x) = x^\mathsf{T} Q x + a^\mathsf{T} x$ for a given symmetric squared matrix $Q \in \mathbb{R}^n \times \mathbb{R}^n, Q = Q^\mathsf{T}$, and vector $a \in \mathbb{R}^n$.

2) The set $\mathcal{X} = \{0, 1\}^n = \{x_{(i)}(1 - x_{(i)}) = 0, \forall i\}$ enforces the binary constraints.
3) Matrix $G \in \mathbb{R}^n \times \mathbb{R}^{n'}$, vector $b \in \mathbb{R}^{n'}$, and function $g : \mathbb{R}^n \to \mathbb{R}$ is convex.
4) Function $\varphi : \mathbb{R}^l \to \mathbb{R}$ is convex and $\mathcal{U}$ is a convex set.
5) Function $\ell : \mathbb{R}^n \times \mathbb{R}^l \to \mathbb{R}$ is *jointly* convex in x, u.

Problem (P) with the required functional assumptions can still capture many relevant problems in mathematical programming, such as vehicle routing [1], [67], [68] and facility location [69]. Formulations for bin packing (BP) and knapsack problems will be discussed in Section VI.

In order to put problem (P) in the ADMM standard form, we need to write problem (P) as problem (3). First, in this article, following mainstream quantum practice (see [18] and [21]) and because we need to retrieve a QUBO, we soft-constrain the equality constraint (whenever present) as an augmented term in the cost function. Then, we introduce the new variable $z \in \mathbb{R}^n$ and problem (P) can be written as the soft-constrained problem (P')

$$\underset{x \in \mathcal{X}, z \in \mathbb{R}^n, u \in \mathcal{U} \subseteq \mathbb{R}^l}{\text{minimize}} \quad q(x) + \frac{c}{2} \|Gx - b\|_2^2 + \varphi(u) \quad (13a)$$

$$\text{subject to: } g(z) \leq 0, \quad \ell(z, u) \leq 0 \quad (13b)$$

$$x = z \quad (13c)$$

for a large positive constant $c > 0$. Problem (P') is a soft-constrained version of problem (P) (it would be equivalent if $G = 0$ and $b = 0$): it is, however, a convenient splitting of binary and continuous variables.

Now, call $\bar{x} = [z^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}$, $m = n + l$, define $f_0(x) := q(x) + \frac{c}{2} \|Gx - b\|_2^2$, $f_1(\bar{x}) = \varphi(u) + \iota_{\bar{\mathcal{X}}}(\bar{x})$, where the set $\bar{\mathcal{X}} := \{(z \in \mathbb{R}^n, u \in \mathcal{U}) | g(z) \leq 0, \quad \ell(z, u) \leq 0\}$. Then, (P') reads as problem (P'')

$$\underset{x \in \mathcal{X}, \bar{x} \in \mathbb{R}^m}{\text{minimize}} \quad f_0(x) + f_1(\bar{x}) \quad (14a)$$

$$\text{subject to: } A_0 x + A_1 \bar{x} = 0 \quad (14b)$$

where $A_0 = I_n$ and $A_1 = [-I_n, 0_{l \times l}]$.

A first possible strategy to use ADMM on (P'') is summarized in Algorithm 1, in a two-block implementation (2-ADMM-H). As we discussed in Section III and Example 2, this strategy is in general a heuristic, since the variable $\bar{x}$ is constrained, however, in some cases Algorithm 1 can deliver good solutions (as we will explore). In order to keep track of the solution quality during the iterations, we compute a merit value associated with each iterate x_k. Let $\zeta_k = \max(g(x_k), 0) + \max(l(x_k, \bar{x}_k), 0)$ be the violation of the constraints on decision variable x in problem (P) at iteration k, and μ be a penalization for ζ_k. Then, the merit value η_k of x_k is a linear combination $q(x_k) + \phi(\bar{x}_k) + \mu \zeta_k$ of the constraint violation and solution cost in problem (P). Iterates with high merit value are both not likely to be of optimal value and close to feasibility for problem (P), hence, the minimum merit value solution is returned by Algorithm 1.

Algorithm 1: 2-ADMM-H Mixed-Binary Heuristic.

Require: Initial choice of x_0, y_0, λ_0. Choice of $\varrho, c, \mu > 0$, tolerance $\epsilon > 0$, and maximum number of iterations $K_{\max}$.

1: **while** $k < K_{\max}$ **and** $\|A_0 x^k + A_1 \bar{x}^k\| < \epsilon$, **do**
2: First block update (QUBO):

$$x_k = \arg\min_{x \in \{0,1\}^n} q(x) + \frac{c}{2}\|Gx - b\|_2^2 + \lambda_{k-1}^\mathsf{T} A_0 x$$
$$+ \frac{\varrho}{2}\|A_0 x + A_1 \bar{x}_{k-1}\|^2$$

3: Second block update (Convex):

$$\bar{x}_k = \arg\min_{\bar{x} \in \mathbb{R}^m} f_1(\bar{x}) + \lambda_{k-1}^\mathsf{T} A_1 \bar{x} + \frac{\varrho}{2}\|A_0 x_k + A_1 \bar{x}\|^2$$

4: Dual variable update:

$$\lambda_k = \lambda_{k-1} + \varrho(A_0 x_k + A_1 \bar{x}_k)$$

5: Compute merit value:

$$\eta_k = q(x_k) + \phi(\bar{x}_k)$$
$$+ \mu(\max(g(x_k), 0) + \max(l(x_k, \bar{x}_k), 0))$$

6: **end while**
7: **return** $x_{k^*}, \bar{x}_{k^*}, y_{k^*}$, with $k^* = \min_k \eta_k$.

The strength of Algorithm 1 is that the original MBO is now split into a QUBO (that can be solved on the QUBO oracle, or on quantum devices) and a convex problem that can be solved with off-the-shelf solvers, such as SPDT3 [70] and MOSEK [71].

Remark 1: In [36], the authors explore a slightly different decomposition of the same MBO problem (12). In particular, the authors let $f_1(\bar{x}) = \varphi(u) + \iota_{\bar{\mathcal{X}} \cup \{G\bar{x}=b\}}(\bar{x}) + q(\bar{x})$, while $f_0(x) = 0$. In this way, the QUBO problem (first block update) becomes a projection problem of dimension n onto the one dimensional constraint $\{0, 1\}$, which is easily solvable, while the convex problem (second block update) becomes the convex relaxation of the MBO problem (with an additional penalization term). This nonconvex ADMM heuristic is effective in finding approximate solutions to a wide variety of problems in classical computation, depending on an appropriate setting of the initial parameters. However, it is not readily applicable on quantum devices, as it does not involve QUBOs.

B. FROM TWO-BLOCK TO THREE-BLOCK ADMM FOR MBOS

To overcome the limitation imposed by the convergence theorems (see Theorems 1 and 2) on the smoothness of function $f_1(\bar{x})$, we use the same approach explored in Example 2, as well as in [46]. We exploit a three-block implementation of ADMM (3-ADMM-H) onto the

---

Algorithm 2: 3-ADMM-H Mixed-Binary Heuristic.

Require: Initial choice of $x_0, \bar{x}_0, y_0, \lambda_0$. Choice of $\varrho, \beta, c > 0$, tolerance $\epsilon > 0$, and maximum number of iterations $K_{\max}$.

1: **while** $k < K_{\max}$ **and** $\|A_0 x^k + A_1 \bar{x}^k - y_k\| < \epsilon$, **do**
2: First block update (QUBO):

$$x_k = \arg\min_{x \in \{0,1\}^n} q(x) + \frac{c}{2}\|Gx - b\|_2^2$$
$$+ \lambda_{k-1}^\mathsf{T} A_0 x + \frac{\varrho}{2}\|A_0 x + A_1 \bar{x}_{k-1} - y_{k-1}\|^2 \quad (15)$$

3: Second block update (Convex):

$$\bar{x}_k = \arg\min_{\bar{x} \in \mathbb{R}^m} f_1(\bar{x}) + \lambda_{k-1}^\mathsf{T} A_1 \bar{x}$$
$$+ \frac{\varrho}{2}\|A_0 x_k + A_1 \bar{x} - y_{k-1}\|^2 \quad (16)$$

4: Third block update (Convex+quadratic):

$$y_k = \arg\min_{y \in \mathbb{R}^n} \frac{\beta}{2}\|y\|_2^2 - \lambda_{k-1}^\mathsf{T} y$$
$$+ \frac{\varrho}{2}\|A_0 x_k + A_1 \bar{x}_k - y\|^2$$

5: Dual variable update:

$$\lambda_k = \lambda_{k-1} + \varrho(A_0 x_k + A_1 \bar{x}_k - y_k)$$

6: Compute merit value:

$$\eta_k = q(x_k) + \phi(\bar{x}_k) + \mu(\max(g(x_k), 0)$$
$$+ \max(l(x_k, \bar{x}_k), 0)) \quad (17)$$

7: **end while**
8: **return** $x_{k^*}, \bar{x}_{k^*}, y_{k^*}$, with $k^* = \min_k \eta_k$.

---

soft-constrained problem (P''')

$$\min_{x \in \mathcal{X}, \bar{x} \in \mathbb{R}^m, y \in \mathbb{R}^n} f_0(x) + f_1(\bar{x}) + \frac{\beta}{2}\|y\|_2^2 \quad (18a)$$

$$\text{subject to: } A_0 x + A_1 \bar{x} = y \quad (18b)$$

where the only difference with (14) is the introduction of variable y, which penalizes constraint violations.

Algorithm 2 reports the 3-ADMM-H algorithm, along with stopping criteria and evaluation metrics. As we can see, once again, problem (18) is split into a QUBO, that can be solved by a QUBO oracle, and convex optimization problems. We note that the two-block implementation is a particular case of the three-block algorithm, with $y_0 = 0 \in \mathbb{R}^n$, and skipping third block update (i.e., step 4 of 2).

We are now ready for the convergence results for Algorithm 2 (3-ADMM-H). First, we present the results when continuous variables are not present, and then extend it to continuous variables.

Theorem 3 (Convergence of Algorithm 2): Consider Problem (12) with no continuous variable u and let Assumption 1

hold. Define the augmented Lagrangian

$$\mathcal{L}_{\varrho}(x, \bar{x}, y, \lambda) = f_0(x) + \iota_{\mathcal{X}}(x) + \frac{c}{2}\|Gx - b\|_2^2$$

$$+ f_1(\bar{x}) + \frac{\beta}{2}\|y\|_2^2 + \lambda^{\mathsf{T}}(A_0 x + A_1 \bar{x} - y)$$

$$+ \frac{\varrho}{2}\|A_0 x + A_1 \bar{x} - y\|_2^2. \tag{19}$$

Then, Algorithm 2 converges subsequently for any sufficiently large $\varrho > \max\{\beta, c\}$, that is, starting from any $x_0, \bar{x}_0, y_0, \lambda_0$, it generates a sequence that is bounded, has at least one limit point, and that each limit point $(x^*, \bar{x}^*, y^*, \lambda^*)$ is a stationary point of $\mathcal{L}_{\varrho}$, namely, $0 \in \partial\mathcal{L}_{\varrho}(x^*, \bar{x}^*, y^*, \lambda^*)$.

In addition, if $f_1(\bar{x})$ is a KŁ function [63]–[65], then $(x_k, \bar{x}_k, y_k, \lambda_k)$ converges globally to the unique limit point $(x^*, \bar{x}^*, y^*, \lambda^*)$.

Proof: We are going to leverage the results of Theorem 2 to prove Theorem 3. In particular, we are going to check that all the assumptions in Theorem 2 are satisfied and determine a necessary condition on how large ϱ must be for the algorithm to converge.

A1) (Coercivity). Coercivity holds since x lies in a bounded set, $h(y) = \frac{\beta}{2}\|y\|_2^2$ is quadratic, therefore coercive, and the same holds for $\bar{x}$.

A2) (Feasibility). $Im(A) \subseteq Im(B)$ holds by direct computation, since $A = [A_0, A_1] = [I_n, -I_n, 0_{l \times l}]$ and $B = -I_n$.

A3) (Lipschitz subminimization paths). $A_0 = -B = I_n$, so trivially $\bar{M} = 1$ for x and y. Consider now $\bar{x}$, since no continuous variables are present $A_1 = -I_n$ and $\bar{M} = 1$ trivially.

A4) (Objective f-regularity). $f_0 + \iota_{\mathcal{X}}(x)$ is lower semicontinuous, and f_1 is restricted prox-regular since the sum of a convex function and the indicator function of a convex set.

A5) (objective h-regularity). $h(y) = \frac{\beta}{2}\|y\|_2^2$ is Lipschitz differentiable with constant β, so A5 holds.

As for ϱ, from the conditions in [38, Lemma 9], then $\varrho > \max\{\beta, c\}$.

And to finish the proof: $f_0(x) + \iota_{\mathcal{X}}(x) + \frac{c}{2}\|Gx - b\|_2^2 + \frac{\beta}{2}\|y\|_2^2 + \lambda^{\mathsf{T}}(A_0 x + A_1 \bar{x} - y) + \frac{\varrho}{2}\|A_0 x + A_1 \bar{x} - y\|_2^2$ is a KŁ function, since it is semialgebraic, and $\mathcal{L}_{\varrho}(x, \bar{x}, y, \lambda)$ is KŁ if $f_1(\bar{x})$ is KŁ. □

Theorem 3 describes a set of assumptions for which Algorithm 2 is proven to converge to a stationary point of the augmented Lagrangian $\mathcal{L}_{\varrho}$, which is a soft-constrained version of the original MBO problem (12). We now expand on Theorem 3 by considering continuous variables.

Theorem 4 (Convergence of Algorithm 2 with Continuous Variables): The same results of Theorem 3 hold if the following conditions hold.

1) The function $\varphi(u)$ is strictly convex and the inequality constraint $\ell(z, u) \leq 0$ is never active, i.e., for each z_k and u_k generated by the algorithm we have $\ell(z_k, u_k) < 0$.
2) The inequality constraint $\ell(z, u) \leq 0$ is always active, i.e., for each z_k and u_k generated by the algorithm we

have $\ell(z_k, u_k) = 0$, and for any fixed z, the inverse mapping $u(z) = \{u | \ell(z, u) = 0\}$ is unique and Lipschitz, i.e., $\|u(z) - u(z')\| \leq C\|z - z'\|$, for $C < \infty$, and $\varrho > \max\{C^2\beta, c + C^2\}$.

Proof: We have only to show that A3 holds in these cases. For the first case, the inequality constraint is redundant and u_k is only determined from $\varphi(u)$. Since $\varphi(u)$ is strictly convex, u_k is unique and the same for all ks, so $\|u_{k_1} - u_{k_2}\| = 0$ and A3 holds. This is the case, for example, when inequality constraints are absent.

For the second case, since $\ell(z_k, u_k) = 0$ and $\|u(z) - u(z')\| \leq C\|z - z'\|$, then A3 holds with $\bar{M} = C$. And the conditions on ϱ are derived from [38, Lemma 9]. This is the case, e.g., when the inequalities are linear equality constraints as $Fz + Hu = g$, and H is full rank. □

The conditions of Theorems 3 and 4 are quite mild in many practical relevant MBO problems. In full generality however, Algorithm 2 is a heuristic algorithm, especially as we remark next.

1) *Equality constraints:* When equality constraints are presents, they are softened with the augmented term $\frac{c}{2}\|Gx - b\|_2^2$ in the cost function. This induces a tradeoff: from the conditions in [38, Lemma 9], then at least $\varrho > \max\{\beta, c\}$; however, to enforce the equality constraints, these have to be at least as important as the enforcing of zero residuals, i.e., $c \geq \varrho$. This introduces the tradeoff of either terminating with a solution with zero residuals (meaning the convergence has been reached, but equality constraints are not necessarily satisfied), or with equality constraints satisfied (without bounds on the magnitude of the residual).
   Note that off-loading the equality constraints to variable $\bar{x}$ and imposing them exactly, only mildly solves the issues, since residual convergence would be achieved with $y \neq 0$ (in general), and therefore, the equality constraints will not be satisfied exactly.
2) *Continuous variables:* When continuous variables are present, which do not satisfy either of the conditions of Theorem 4, then assumption A3 is not satisfied, making Theorem 3 not hold in this situation and Algorithm 2 is still a heuristic for this case.

C. INEXACT OPTIMIZATION AND NOISE

We briefly remark here the effect of inexact optimization of the binary subproblems, as well as noise in real quantum devices. The theoretical analysis assumes that the QUBOs need to be solved exactly to guarantee the validity of the presented theorems. In practical situations, however, this is hard to achieve and this requirement needs to be weakened. For instance, in current noisy quantum computers, noise is inherent in the computations and an exact optimization is rather far-fetched. Even in the case of noise-free quantum computers, solving QUBOs at optimality may be unrealistic, especially in large-scale instances. Within the ADMM framework, it then makes sense to ask i) whether one can

tolerate inexact optimization; ii) whether noise has to be always detrimental for convergence.

On the first point, we do not have an answer yet in the general case (even though this has been studied abundantly and with positive answers in the convex case). In [37], the authors show that one can tolerate inexact computations, which are asymptotically vanishing and summable. In particular, if one can solve the QUBOs with increasing degree of accuracy while the algorithm progresses, then convergence can be still established. This is a promising first result.

The second point is even more interesting and open. It is fairly understood and it has been experimentally observed that a small amount of noise can help to "guide" convergence of first-order algorithms to global optimizers in nonconvex problems (e.g., by escaping local minima, or saddle-points). So, it is possible that a small amount of noise, inexactness, or both, could help convergence instead of jeopardizing it. We will explore this aspect in the simulation results.

V. SIMPLE EXAMPLES

We discuss here some interesting examples to showcase the performance of 2-ADMM-H and 3-ADMM-H for MBOs problems in simple settings, and gain some insights on the solutions obtained.

A. INEQUALTITY CONSTRAINTS

Example 3: Consider the problem

$$\min_{x \in \{0,1\}^2} \quad v + w \tag{20}$$

$$\text{subject to:} \quad 2v + w \leq 2 \tag{21}$$

$$v + w \geq 1 \tag{22}$$

where $x = [v, w]^\mathsf{T}$. We consider the following two cases: Case 1: $1001 = \varrho > \beta = 1000$ (verifying the necessary conditions for Algorithm 2 to converge, but Algorithm 1 is a heuristic), and Case 2: $\varrho = \beta = 1000$, for which both algorithms are heuristics. Fig. 1 shows convergence of the residual of both Algorithms 1 and 2, where we defined the three-block residual as $r_3 = \|A_0 x + A_1 \bar{x} - y\|$, while the restricted three-block residual as $rr_3 = \|A_0 x + A_1 \bar{x}\|$ (that is how far we are from the solution of the nonrelaxed problem), as well as the two-block residual as $r_2 = \|A_0 x + A_1 \bar{x}\|$.

As we can see, in Case 1, Algorithm 2 converges in the residual sense while Algorithm 1 does not. In particular, the results (at three significative digits) yield: $x = [0, 0]$, $\bar{x} = [0.499, 0.500]$, $y = [-0.499, -0.499]$ for the three blocks, while $x = [0, 1]$, $\bar{x} = [0.499, 0.999]$, for the two blocks. We can also see that, despite Algorithm 2 convergence, the result $x = [0, 0]$ is not optimal (not even feasible for the original nonrelaxed problem), while Algorithm 1 delivers one of the two optimal results $x = [0, 1]$, while not converging.

In Case 2, Algorithm 2 converges in the residual sense (even though it is not guaranteed to do so) while
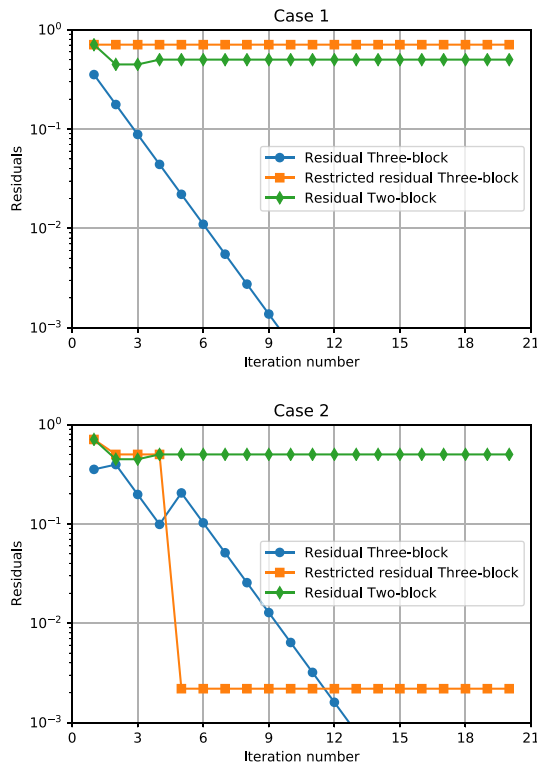

FIGURE 1. Convergence of the residuals for Example 3.

Algorithm 1 does not. In particular, the results (at three significative digits) yield: $x = [0, 1]$, $\bar{x} = [0.002, 0.999]$, $y = [-0.002, -0.000]$ for the three-blocks, while $x = [0, 1]$, $\bar{x} = [0.499, 0.999]$, for the two blocks. In this case, both Algorithms 2 and 1 deliver one of the two optimal results $x = [0, 1]$.

Example 4: Consider the problem

$$\min_{x \in \{0,1\}^3} \quad v + w + t \tag{23}$$

$$\text{subject to:} \quad 2v + 10\,w + t \leq 3 \tag{24}$$

$$v + w + t \geq b \tag{25}$$

where $x = [v, w, t]^\mathsf{T}$ and b is either 1 (Case 1) or 2 (Case 2). We fix $\varrho = 1001$, $\beta = 1000$.

In Case 1, as we can see in Fig. 2, both algorithms converge. Algorithm 2 delivering $x = [0., 0., 0.]$, $\bar{x} = [0.397, 0.178, 0.424]$, $y = [-0.397, -0.178, -0.424]$; Algorithm 1 delivering a feasible solution $x = [1., 0., 1.]$, $\bar{x} = [1., 0., 1.]$.

In Case 2, both algorithms converge and deliver the optimal solution.

B. EQUALITIES AND INEQUALTITY CONSTRAINTS

Example 5: Consider the problem

$$\min_{x \in \{0,1\}^3} \quad v + w + t \tag{26}$$

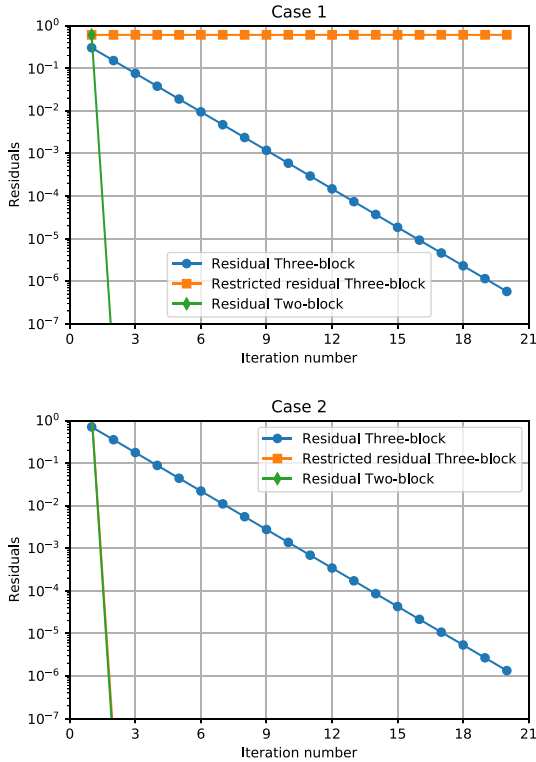$$\text{subject to:} \quad 2v + 2\,w + t \leq 3 \tag{27}$$

FIGURE 2. Convergence of the residuals for Example 4.

FIGURE 3. Convergence of the residuals for Example 5.

$$v + w + t \geq 1 \tag{28}$$

$$v + w = 1 \tag{29}$$

where $x = [v, w, t]^{\mathsf{T}}$. We fix $\varrho = 1001$, $\beta = 1000$, and the penalization parameter for the equality constraint to be $c = 900$ (Case 1), $c = 1100$ (Case 2).

In Case 1, Algorithm 2 is supposed to converge. Both algorithms converge in practice, Algorithm 2 to one optimal solution $x = [1., 0., 0.]$, $\bar{x} = [1., 0.002, 0.002]$, $y = [0., -0.002, -0.002]$; Algorithm 1 to a feasible solution $x = [1., 0., 1.]$, $\bar{x} = [0.999, 0., 0.999]$.

In Case 2, Algorithm 2 is not guaranteed to converge. However, both algorithms seem to converge. Both deliver an optimal solution: $x = [0., 1., 0.]$.

C. CONTINUOUS VARIABLES
Example 6: Consider the problem

$$\min_{x \in \{0,1\}^3, u \in \mathbb{R}} \quad v + w + t + 5(u - 2)^2 \tag{30}$$

$$\text{subject to:} \quad v + 2w + t + u \leq 3 \tag{31}$$

$$v + w + t \geq 1 \tag{32}$$

$$v + w = 1 \tag{33}$$

where $x = [v, w, t]^{\mathsf{T}}$. We fix $\varrho = 1001$, $\beta = 1000$, and the penalization parameter for the equality constraint to be $c = 900$. The inequality constraint with the continuous variable
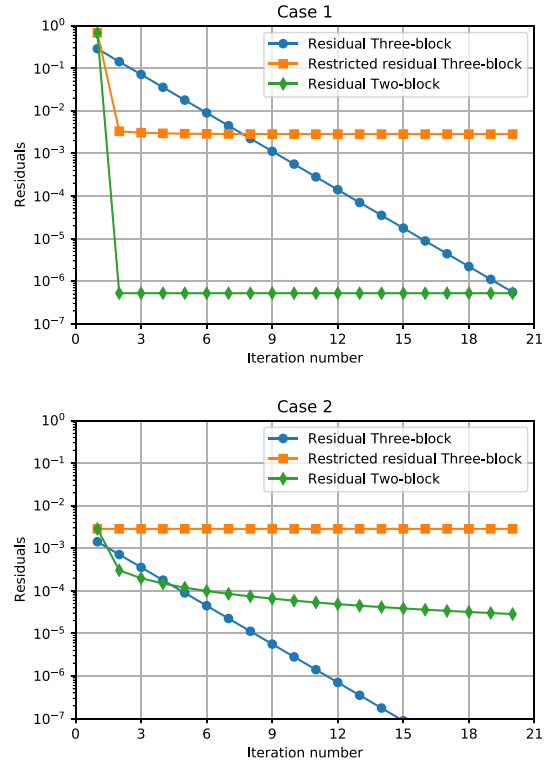
is always active, so Algorithm 2 is supposed to converge (as for Theorem 4).

In Fig. 4, we see how both algorithms converge, but only Algorithm 2 yields the optimal solution (incurring zero optimality gap). In particular, Algorithm 2 delivers the optimal solution $x = [1., 0., 0.]$, $\bar{x} = [1., 0., 0., 2.]$, $y = [0., 0., 0.]$, whereas Algorithm 1 delivers the feasible solution $x = [1., 0., 1.]$, $\bar{x} = [1., 0., 1., 1.]$.

D. INEXACT UPDATES
Example 7: We reconsider now Example 4, in Case 1. There, as we could see in Fig. 2, both algorithms converged, but Algorithm 2 was delivering an infeasible solution. We consider here the case in which the QUBO is solved with some errors, and we model these errors as probability of a bit-flip of the QUBO optimal solution. In particular, for each component $x_{k(i)}$ of the QUBO solution, with value either 0 or 1, we consider that there is a certain probability for it to flip to 1 or 0, respectively. We also consider that this probability decreases as $50/k\%$ as the number of iterations k increases, to model the fact that we are solving the QUBO subproblems more and more accurately. This allows M-ADMM-H to escape bad regions of the solution space at first, and to intensify the search for higher quality solutions in the consequent iterations.

In Fig. 5, we see how both algorithms converge, but now Algorithm 2 yields an optimal solution $x = [1., 0., 0.]$, $\bar{x} = [1., 0.002, 0.002]$, $y = [0.0, -0.002, -0.002]$; Algorithm 1 delivers the same feasible solution as before.
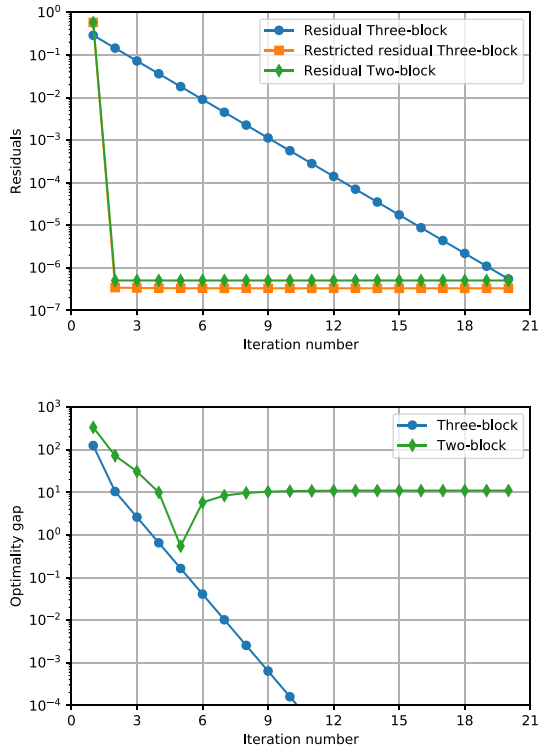
FIGURE 4. Convergence of the residuals and optimality gap for Example 6.
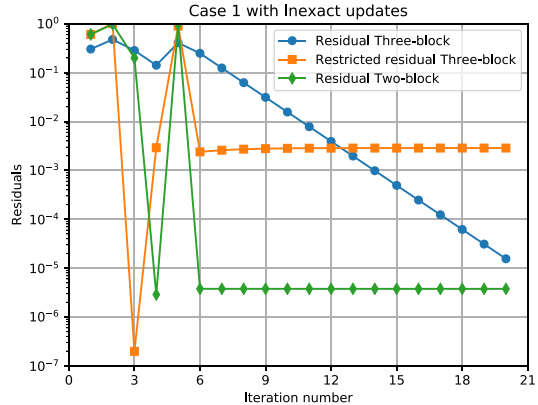

FIGURE 5. Convergence of the residuals for Example 7.

This example showcases how noise in real setting can help the algorithms to converge to optimal solutions. Note that in Case 2, the same optimal solution is achieved also with noise in this setting.

VI. MBO PROBLEMS
As shown in the simple examples presented in Section V, 3-ADMM-H (see Algorithm 2) and 2-ADMM-H (see Algorithm 1) are heuristics, in the general case. In the best case, 3-ADMM-H is guaranteed to converge and it delivers an optimal solution for the original MBO. In the worst case, both algorithms fail to deliver feasible solutions. In the middle,

3-ADMM-H may converge, but the soft-constrained solution is not feasible with respect to the hard-constrained formulation, or both algorithms could converge to a feasible but not optimal solution. With this in mind, we are now ready to apply the algorithms to two well-known MBO problems: BP problem and mixed integer setup knapsack (MISK) problem. The computational results will be discussed in Section VII, where we will show that despite the heuristic nature of M-ADMM-H, we can still obtain feasible solutions in many cases. This is not trivial in general for combinatorial optimization problems [72], [73].

The BP is arguably one of the most studied combinatorial problems [74]. Being strongly NP-hard, it stimulated the study of heuristics, metaheuristics, and worst-case approximation bounds. Given n items, each having an integer weight $w_j, j = 1, \ldots, n$, and m identical bins of integer capacity Q, the aim of BP is to pack all the items into the minimum number of bins so that the total weight packed in any bin does not exceed the capacity. Applications of BP in logistics and scheduling are numerous, and include cutting stock problems, containers loading, data storage, job scheduling, and resource allocation.

The MISK belongs to the class of knapsack problems [75], [76]. The classical knapsack problem is that of deciding which items to pack in a capacitated knapsack, so as to maximize the profit of the items in the knapsack. In the setup knapsack problem (SKP), each item belongs to a family, and an item can be assigned to the knapsack only if a setup charge for the correspondent family is paid [77]. SKP can model capacitated scheduling problems. In the MISK, items can be fractionally assigned to the knapsack. MISK appears as a subproblem of the capacitated coordinated replenishment problem.

A. BINARY LINEAR PROGRAMMING FORMULATION FOR BP
Let $\xi_{ij} \in \{0, 1\}$ be the binary decision variable (BinVar) which, if 1, indicates that item j is assigned to bin i. Let $\chi_i \in \{0, 1\}$ be the BinVar which, if 1, indicates that bin i is containing items. A natural mathematical formulation for BP problem is then given by the binary linear program

$$\underset{\chi,\xi}{\text{minimize}} \quad \sum_{i=1}^{m} \chi_i \tag{34a}$$

$$\text{subject to:} \quad \sum_{i=1}^{m} \xi_{ij} = 1 \quad j = 1, \ldots, n \tag{34b}$$

$$\sum_{j=1}^{n} w_j \xi_{ij} \leq Q\chi_i \quad i = 1, \ldots, m \tag{34c}$$

$$\xi_{ij} \in \{0, 1\} \quad i = 1, \ldots, m$$
$$j = 1, \ldots, n \tag{34d}$$

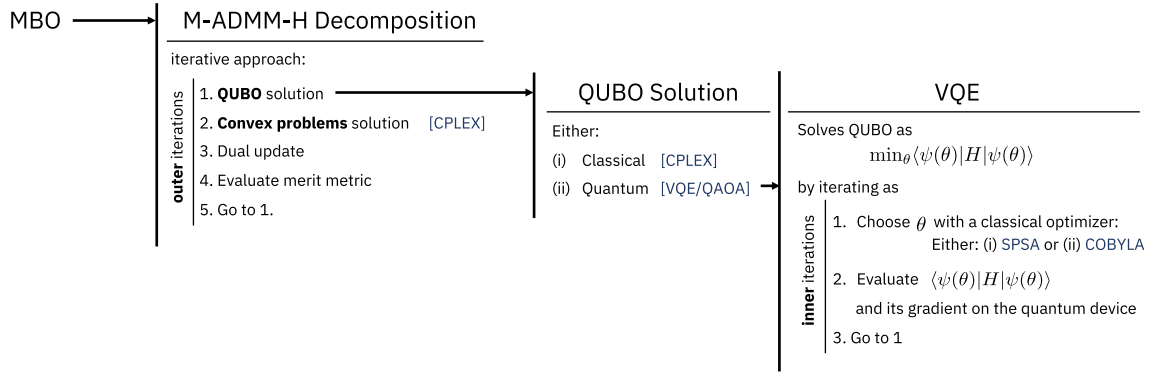$$\chi_i \in \{0, 1\} \quad i = 1, \ldots, m. \tag{34e}$$

FIGURE 6. Illustration diagram of M-ADMM-H with VQE as quantum solver. The optimization solvers adopted for the numerical results are specified (i.e., CPLEX, VQE, SPSA, and COBYLA). There are two nested loops for the selected implementation, specifically the outer ADMM loop with outer iterations, and the inner VQE loop, with inner iterations.

In particular:

1) The objective function (34a) is the number of bins in solution.
2) Constraints (34b) enforce the assignment of each item into a bin.
3) Constraints (34c) ensure the packed items do not exceed the bin capacity.
4) Constraints (34d) and (34e) express the bounds on the decision variables.

The presence of inequalities to express the capacity constraints (34c) forbids the straightforward mapping to an Ising Hamiltonian model, and the direct application of quantum optimization algorithms, such as VQE [6] and QAOA [8].

B. MIXED-BINARY FORMULATION FOR THE MISK

The MISK has received limited attention in the literature. The mixed-integer formulation proposed in [78] is presented in this section. The items belongs to K nonoverlapping families. Each family k has T items, and a setup cost $S_k \geq 0$, when included in the knapsack. Each item t of family k has a value $C_{kt} < 0$, and a resource consumption $D_{kt} \leq 0$, if assigned to a knapsack with capacity P. The decision variables are the fraction ξ_{kt} of item t that is included in the knapsack, and is the binary decision χ_k to setup family k in the knapsack. MISK can then be formulated as

$$\underset{\chi,\xi}{\text{minimize}} \quad \sum_{k=1}^{K} S_k \chi_k + \sum_{k=1}^{K} \sum_{t=1}^{T} C_{kt} \xi_{kt} \tag{35a}$$

$$\text{subject to:} \quad \sum_{k=1}^{K} \sum_{t=1}^{T} D_{kt} \xi_{kt} \leq P \tag{35b}$$

$$\xi_{kt} \leq \chi_k \quad k = 1, \dots, K, t = 1, \dots, T \tag{35c}$$

$$\xi_{kt} \geq 0 \quad k = 1, \dots, K, t = 1, \dots, T \tag{35d}$$

$$\chi_k \in \{0, 1\} \quad k = 1, \dots, K. \tag{35e}$$

The aim is to minimize the setup costs and maximize the value of the assigned items via the objective function (35a). Constraints (35b) ensures that the capacity of the knapsack is not violated: this is a fixed charge capacity constraints, because setup capacity consumption is not considered. Constraints (35c) impose that if item t of family k is assigned to the knapsack, then the setup cost of family k is paid accordingly.

VII. COMPUTATIONAL RESULTS

We discuss here the multiblock (M-ADMM-H) results on BP and MISK. The algorithm has been implemented in Python on a machine with 2.2 GHz, Intel Core i7 processor, and a RAM of 16 GB; the simulations on quantum devices to solve the QUBOs have been conducted by using the Qiskit framework [79] (specifically, *qiskit* version 0.15.0, *qiskit-aqua* version 0.6.1, *qiskit-terra* version 0.10.0, *qiskit-aer* version 0.3.2), while IBM ILOG CPLEX 12.8 has been chosen as classical optimization solver.[1]

In Fig. 6, a summary of the proposed approach and implementation choices are presented with VQE and QAOA as quantum QUBO solvers. It is important to note the presence of two nested iterations: the outer one due to ADMM and the inner one due to the classical solver for VQE/QAOA. In the following, we consistently call the ADMM iterations "outer" iterations, while the classical solver ones are "inner" iterations. The choice of simultaneous perturbation stochastic approximation (SPSA) or constrained optimization by linear approximation (COBYLA) affects the choice of number of inner iterations, therefore, we often say: SPSA/COBYLA inner iterations. Furthermore, for all simulations reported in the following subsections.

1) *The M-ADMM-H algorithm* has been run with a time limit of 1 h, a limit of 500 outer iterations, and with merit parameter $\mu = 1e + 3$.
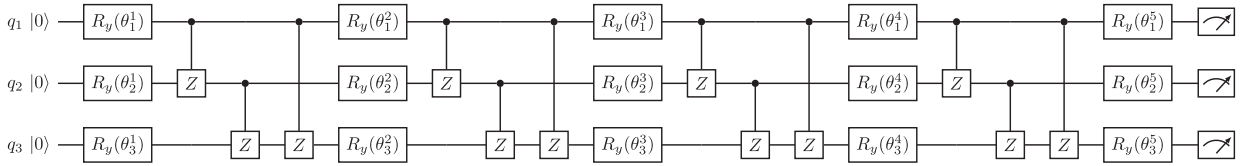
FIGURE 7. Prototype circuit used in the simulation results to evaluate $|\psi(\theta)\rangle = U(\theta)|0\rangle$, here exemplified for three qubits ($q = 3$) and a depth $d = 4$, consisting of $d + 1$ layers. The first operations consists in single-qubit Y rotations, with one variational parameter θ_i^j per qubit to determine the rotation angle. Each additional layer after the first contains entangling gates, more specifically controlled-Z gates applied to all qubit pairs, followed by another set of single-qubit Y rotations with one variational parameter each to represent the angle. The variational form is then parametrized over $q(d + 1)$ angles, arranged in a vector $\theta = [\theta_i^j]_{i=1,\ldots,q; j=1,\ldots,d+1}$.

In addition, to avoid large penalization factors ϱ from the first outer iteration, we start with the value $\varrho = 1e + 4$, which is then increased by 10% at each iteration, until it exceeds the value of $1e + 7$. The penalization c of equality constraints has been set to $1e + 5$. The penalization β of residual $\|y\|$ is initially set to $1e + 3$ and then updated according to scheme described in [38], specifically $\beta^{k+1} = \gamma \beta^k$, if $\|\bar{x}^k\| \leq \omega \|\bar{x}^{k-1}\|$, with $\omega = 0.5$ and $\gamma = 2$, so to foster exact penalization.

2) *The QUBO subproblems* are solved either classically with CPLEX, or on the simulated quantum devices via the Qiskit APIs. A common random seed has been fixed for all simulations. No limitations on the running time of the quantum solver have been imposed but only a maximum number of inner iterations. The variational quantum eigensolver (VQE) has been invoked with the RY variational form in a circuit of depth 5 and full entanglement, and the QASM simulator as Qiskit Aer backend. Fig. 7 represents the circuit that was used in the case of three qubits and depth 4. The quantum approximate optimization algorithm (QAOA) has been tested with circuit depth 3 and the same backend.

3) *VQE* is itself an iterative quantum algorithm that involves defining a parametrized variational form and optimizing classically on the rotation parameter vector θ, while evaluating the variational form and its gradients on the quantum device. In our simulations, the classical solvers used by VQE are the model-based local optimizers SPSA [80], and COBYLA [81]. For both solvers, the Qiskit implementation has been used.

4) *QAOA* generalizes VQE because the variational form is added with parameter vector β of length equal to θ. As for VQE, the classical optimization is performed via SPSA and COBYLA.

5) For the sake of clarity, we indicate the quantum QUBO solvers with name *quantum algorithm-internal classical solver*. For instance, VQE-SPSA solves QUBOs with VQE and SPSA as internal classical optimizer. For the classical optimizers SPSA and COBYLA on the rotation parameters, sensitivity results are reported for 10, 20, 50 maximum inner iterations.

The gap of the minimum-merit-value solution with value v with respect to known optimal value v^* is computed as $\frac{|v-v^*|}{1e-10+|v^*|}$. In order to report the computational results, we have included the number of BinVars, the number of outer iterations (IT) of M-ADMM-H, the gap (Gap) to optimality, and percentage of M-ADMM-H solution that are feasible (Feas) or optimal (Opt) with respect to the constraints and objective of the original constrained problem. For the simulations with VQE and QAOA, the number of BinVars corresponds to the number of qubits.

Remark 2: We notice here that VQE and QAOA do not solve (in general) a QUBO at optimality (and in this sense, they are not a perfect oracle), while CPLEX does (for the considered small instances). In addition, even in cases in which the quantum algorithm solves the QUBO at optimality, the optimizer may be different from CPLEX, since multiple equivalent solutions could exist. In general, then the solution of the quantum algorithm and CPLEX will be different when solving the same QUBO and the outer ADMM loop will be affected by it. In practice, using VQE or QAOA could either worsen or boost convergence since M-ADMM-H is in general a heuristic, small errors can be beneficial in some cases, while worsening performance in others.

We notice that the choice of VQE and QAOA in this article is due to the current technical status of quantum computing. In the future, better QUBO solvers may be available, e.g., based on (iterative) phase estimation, which might deliver optimal solutions at scale.

A. BIN PACKING
We first discuss two implementation improvements to reduce computational complexity and foster convergence in the heuristic case for BP.

1) Removing Unnecessary Decision Variables: Let l be a lower bound on the number of bins required to pack all items (for example, the continuous relaxation bound $\lceil \frac{\sum_{i,j} w_{ij}}{Q} \rceil$). Then, it is possible to discard variables $\chi_1, \ldots, \chi_l$ from the mathematical formulation. In addition, it is not restrictive to assume $\xi_{1,1} = 1$. With these observations, the number of decision variables required is $(mn - n) + (m - l)$. Typically, $n = m$, hence, this boils down to $n^2 - l$. The stronger the bound l is, the fewer binary variables are introduced. In the current implementation, the continuous relaxation bound has been adopted.

2) LS Operator: To improve the convergence of M-ADMM-H to solutions that are feasible for the equality

TABLE 1. Feasibility and Optimality Results of M-ADMM-H on 60 Instances With $n = 2, 3, 4$ and $Q = 40$

Items	3-ADMM-H			2-ADMM-H		
	Gap	Feas	Opt	Gap	Feas	Opt
2	0.00%	100.00%	100.00%	25.00%	50.00%	50.00%
3	15.83%	90.00%	65.00%	9.17%	90.00%	75.00%
4	68.33%	100.00%	5.00%	20.42%	50.00%	35.00%
2, 3, 4	28.06%	95.08%	55.74%	18.19%	63.33%	53.33%

constraints (34c), we have implemented an LS operator [82] to be applied to the solutions of the QUBO in the first block update of (2) and (1). This operator is based on the Karmarkar–Karp differencing method [83], and it shuffles the assignment of items to pairs of bins in such a way to minimize the difference of the weights of the bin.

BP has been tested on M-ADMM-H on the following two groups of instances.

1) Small-sized: $n = 2, 3, 4$. Weights w_j have been randomly picked in $[1, Q]$. The QUBO has been solved via VQE and CPLEX.
2) Scholl dataset [84], with $n = 50$. We have considered 20 instances of the dataset. For 10 of the instances the bin capacity Q is 100, and the weights w_j are sampled either from the interval $[1, 100]$, or the interval $[20, 100]$. For the remaining half of the instances, the weights are determined analogously, and the bin capacity is 120.

On the Scholl dataset instances, the QUBO subproblem has been solved via CPLEX only, to evaluate the quality of M-ADMM-H solutions. The simulations on quantum devices are not of practical implementation at the moment, since the number of qubits in QUBO are $O(n^2)$ and would exceed the capabilities of current quantum technology.

1) SMALL-SIZED DATASET

Simulations on classical devices: For the simulations on CPLEX, Table 1 reports the percentage of instances for which M-ADMM-H finds feasible or optimal solutions, grouped by the number of items of the instance. The 3-block 3-ADMM-H implementation is able to find feasible solutions for over 90% of the instances. The search for optimal solutions becomes more difficult as the number of items increases, and for only 5% of the 4-items instances optimal solutions are found, and the gap to optimality is close to 70% on the 4-items instances. For the two-block implementation 2-ADMM-H, the increase of gap is less, however, the search for feasible solutions is more difficult, as for 63.33% of the instances feasible solutions are found.

Simulations on quantum devices: For the simulations in which QUBO is solved via VQE and QAOA, the classical solvers SPSA and COBYLA have been set with 10, 20, 50 maximum inner iterations on BP instances with $N = 2, 3$ and $Q = 40$. SPSA is known to be more computationally demanding than COBYLA, because it requires two function evaluations per iteration. For each combination of values of

N and Q, 20 instances have been generated with weights in $[1, Q]$, and average results for each group are reported for VQE in Table 2. While the choice of ten maximum inner iterations for the classical optimizer lowers the computational time each ADMM outer iteration, the convergence of ADMM is slowed down and the quality of the solution is also impacted negatively. VQE-COBYLA makes ADMM converge in one outer iteration to the optimal solution for instances with two items with 20, 50 maximum inner iterations. Increasing the number of SPSA iterations is detrimental for the gap, feasibility and optimality of the instances: this is because SPSA runs for as many inner iterations as the limit set in Qiskit. Invoking VQE with 50 maximum inner iterations in COBYLA, enables to increase by 40%, the number of instances with feasible solutions with $N = 3$. Overall, the choice of SPSA as classical solver for VQE with 20 inner iterations is the best one in terms of solutions quality for these instances with two and three items, and outperforms the results obtained with CPLEX displayed in Table 1. This can be explained by the percentage of QUBO suproblems solved to optimality by VQE (column QUBO): while VQE-COBYLA with 20 or 50 inner iterations solves all QUBOs to optimality when $N = 2$, VQE-SPSA reports a nonoptimal QUBO solution in a considerable percentage of the instances when $N = 2$. It seems, therefore, beneficial for ADMM to solve a part of the QUBO suproblems in an inexact fashion. For instances with $N = 3$, the number of qubits increases and VQE hardly ever solves the QUBOs to optimality. Nevertheless, 3-ADMM-H converges to feasible and optimal solutions in all instances, with SPSA chosen as classical solver (cf. Remark 2, and Section IV-C). It is also interesting to note that VQE-SPSA with ten SPSA inner iterations solves the QUBO to optimality in 35% more of the cases w.r.t. 20–50 max inner iterations; however, this is detrimental to the gap, and optimality of the solutions. The residuals are not guaranteed to decrease in each ADMM outer iteration, as reported by Fig. 8 on instance N3C40I8. In this case, 3-ADMM-H explores solutions with three bins for about 70 outer iterations, and then converges to a nonoptimal solution of lower value, which makes the residual equal to 0.

The results of 3-ADMM-H with QAOA as quantum solver are reported in Table 9 in the Appendix. While the convergence of 3-ADMM-H is overall slower w.r.t. VQE, the quality of the solution obtained is similar when COBYLA performs the classical subroutines.

On the same groups of BP instances, 2-ADMM-H has been tested, and average results are reported in Table 3 for VQE. The convergence is overall slower than the three-block implementation in terms of number of outer iterations, regardless of the classical solver called by VQE. SPSA makes M-ADMM-H obtain solutions with higher quality, when its maximum number of inner iterations is set to 50. COBYLA yields solutions with lower quality, in the case of $N = 3$ and 50 maximum inner iterations. As observed for 3-ADMM-H, the choice of ten maximum inner iterations for SPSA and COBYLA delivers ADMM solutions with sensibly lower

TABLE 2. Average Results of 3-ADMM-H on 40 BP Instances With $N = 2, 3$ and $Q = 40$. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

Instance	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
N2C40IT10	2	1	50.00%	0.00%	0.00%	0.00%	9	0.00%	100.00%	100.00%	26.98%
N2Q40IT20	2	8	0.00%	100.00%	100.00%	41.81%	1	0.00%	100.00%	100.00%	100.00%
N2Q40IT50	2	13	35.00%	80.00%	65.00%	64.87%	1	0.00%	100.00%	100.00%	100.00%
N3C40IT10	7	115	21.67%	80.00%	70.00%	35.52%	69	30.00%	20.00%	10.00%	15.86%
N3Q40IT20	7	8	0.00%	100.00%	100.00%	0.00%	16	72.50%	80.00%	15.00%	0.26%
N3Q40IT50	7	6	14.17%	80.00%	75.00%	0.87%	12	52.50%	80.00%	55.00%	0.00%

TABLE 3. Average Results of 2-ADMM-H on 40 BP Instances With $N = 2, 3$ and $Q = 40$. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

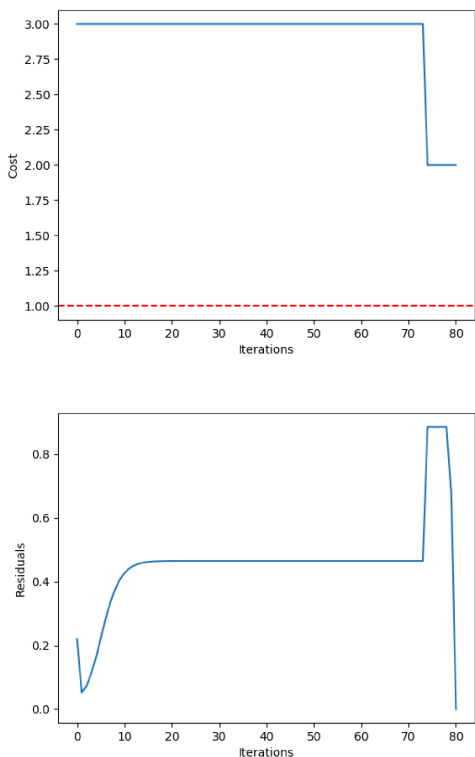| Instance	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
N2C40IT10	2	1	50.00%	0.00%	0.00%	0.00%	6	5.00%	90.00%	90.00%	90.10%
N2Q40IT20	2	21	0.00%	100.00%	100.00%	86.15%	1	0.00%	100.00%	100.00%	100.00%
N2Q40IT50	2	15	0.00%	100.00%	100.00%	1.85%	1	0.00%	100.00%	100.00%	46.40%
N3C40IT10	7	87	42.11%	63.16%	42.11%	11.99%	155	36.67%	30.00%	10.00%	10.00%
N3Q40IT20	7	14	29.17%	95.00%	65.00%	88.75%	49	72.50%	80.00%	15.00%	100.00%
N3Q40IT50	7	5	7.50%	100.00%	90.00%	27.18%	11	41.67%	50.00%	50.00%	11.43%

FIGURE 8. Plots of solution cost and residuals for instance N3C40I8, solved by 3-ADMM-H with COBYLA with 20 inner iterations. The optimal solution value is reported in red dashes.

TABLE 4. Computational Results of M-ADMM-H on 20 BP Instances of the Scholl Dataset [84]. The QUBO subproblems are solved via CPLEX on the classical machine.

Blocks	No LS				With LS			
	IT	Gap	Feas	Opt	IT	Gap	Feas	Opt
3	54	86.08%	100.00%	0.00%	1	86.08%	100.00%	0.00%
2	471	30.17%	10.00%	0.00%	21	33.76%	100.00%	0.00%

in terms of gap, feasibility, and optimality are obtained for 2-ADMM-H.

Finally, the VQE simulations where conducted on three BP instances with $N = 4, Q = 4$. In this case, M-ADMM-H cannot perform more than two outer iterations within the time limit of 1 h. We have also observed that, due to the size of the search state, VQE is not always able to find a solution where the equality constraints (34b) are satisfied. The number of inner iterations of the classical solver invoked by VQE has to be set to a sufficiently large value that ensures to explore solutions without augmented Lagrangian penalty terms. As a representative example, Fig. 9 displays the allocation of items to the bins on a BP instance (referred to as instance N4Q4) with weights [2, 3, 2, 2], obtained from QUBO at outer iteration 1 of 3-ADMM-H. Since one of the items with weight 2 is assigned twice in the solution obtained by VQE-SPSA with 50 inner iterations, it is necessary to increase the inner iterations to 100 to obtain a solution where all items are assigned to one bin. In this case, the solution is feasible and optimal. The time required to perform this outer 3-ADMM-H iteration goes from 2749.87 s in the 50-inner-iteration simulation case to 5380.40 s in the 100-inner-iterations case. This shows that the BP instances with four items are computationally very demanding for the 3-ADMM-H algorithm.

2) SCHOLL DATASET
Table 4 reports the results obtained with M-ADMM-H, both without and with the LS operator described in Section VII-A, with CPLEX as solver for QUBO. We have included

quality. As observed for 3-ADMM-H, a certain degree of inexactness in solving QUBOs is beneficial for the quality of the solutions delivered. In particular, for the instances with 3 items, 2-ADMM-H delivers the best results when VQE solves 27% of the QUBOs to optimality. The results of 2-ADMM-H with QAOA are shown in Table 10 in the Appendix. The choice of QAOA as quantum solver is beneficial in the simulations with COBYLA with 50 inner iterations on instances with 3 items: in this case, the best results
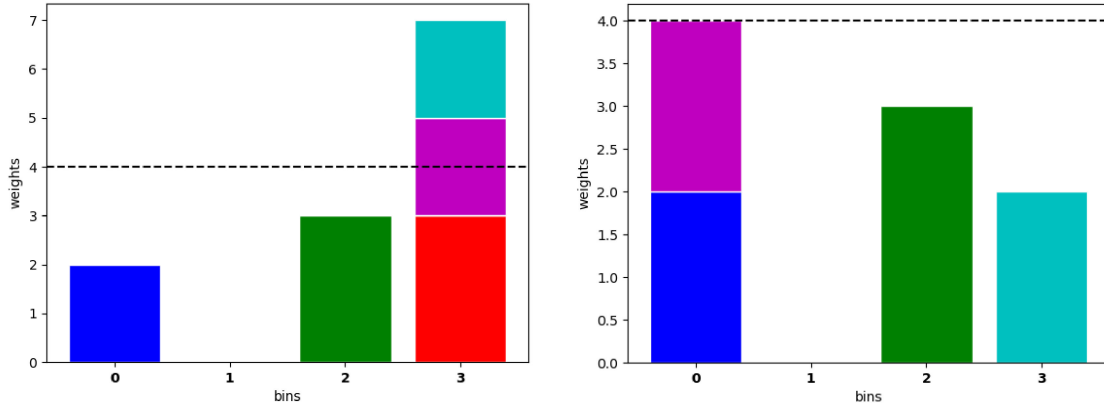
FIGURE 9. Solution found by VQE for the BP QUBO at outer iteration 1 of 3-ADMM-H, on instance N4Q4. The maximum number of SPSA inner iterations is set to 50 in the picture on the left, and to 100 in the one on the right.

TABLE 5. Computational Results of 3-ADMM-H on 12 MISK Instances in Group 1. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

Instances	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	19	62.13%	100.00%	33.33%	41.56%	6	100.00%	100.00%	0.00%	100.00%
K5IT20	5	11	76.20%	100.00%	0.00%	25.87%	6	100.00%	100.00%	0.00%	100.00%
K5IT50	5	6	150.29%	100.00%	0.00%	52.80%	6	100.00%	100.00%	0.00%	100.00%
K8IT10	8	22	82.41%	100.00%	0.00%	7.69%	6	546.00%	100.00%	0.00%	83.33%
K8IT20	8	27	38.74%	100.00%	0.00%	8.03%	6	100.00%	100.00%	0.00%	100.00%
K8IT50	8	12	114.17%	100.00%	0.00%	16.75%	6	100.00%	100.00%	0.00%	100.00%
K11IT10	11	63	42.80%	100.00%	0.00%	0.00%	87	96.39%	100.00%	0.00%	0.00%
K11IT20	11	12	98.57%	100.00%	0.00%	0.00%	6	100.00%	100.00%	0.00%	45.24%
K11IT50	11	5	94.66%	0.00%	0.00%	2.90%	6	93.31%	100.00%	0.00%	18.18%
K14IT10	14	31	39.55%	100.00%	0.00%	0.00%	8	66.83%	100.00%	0.00%	60.71%
K14IT20	14	6	80.91%	33.33%	0.00%	0.00%	6	100.00%	100.00%	0.00%	15.38%
K14IT50	14	3	118.08%	0.00%	0.00%	0.00%	6	170.25%	33.33%	0.00%	0.00%

the percentage of time spent in solving the QUBO (column Block1), the convex subproblem (Block2), the convex and quadratic subproblem (Block3), and boolean indications for the feasibility (Feas) and optimality (Opt) of the M-ADMM-H solution with respect to the constraints and objective of the original constrained problem. Without LS, 3-ADMM-H takes on average 54 outer iterations to converge, and a feasible not optimal solution is found for all instances. The gap to optimality is 86.08% and it is heavily depended on the capacity of the bins: the gap increases from 69.49% on the instances with $Q = 100$ to 102.68% on the instances with $Q = 120$. Applying LS on the QUBO solutions does not help to increase the solution quality, and in fact the same feasible solutions are obtained in the LS simulations. The advantage of LS in this case is to let 3-ADMM-H converge in one outer iteration. The LS is instead extremely beneficial to find feasible solutions in the 2-ADMM-H, and it enables to reach convergence within 21 outer iterations, on average. It is worthy to note that the 2-block implementation enables to find solution with average gaps to optimality less than 50% on those found by the three-block implementation, even if the convergence is often not reached in 500 outer iterations. In the two-block implementation, the gap is less dependent on the capacity of the bins.

B. MIXED-INTEGER SETUP KNAPSACK
MISK problem has been tested on M-ADMM-H on two groups of instances. The first group of instances, Group 1, has been generated by following the guidelines of [78]. To generate challenging MISK instances, the capacity utilization $\frac{\sum_{k=1}^{K} \sum_{t=1}^{T} D_{kt}}{P}$ is set to 2.5, data correlation is medium (i.e., $D_{kt} \in [1, 10]$, $C_{kt} \in -[D_{kt} - 2, D_{kt} + 2]$), and setup costs S_k are randomly sampled in [40, 60]. A second group of instances, Group 2, has been generated with the aim to test M-ADMM-H in cases where the continuous decisions have an impact larger than the binary decisions on the solutions. To this end, the S_k and values C_{kt} have been lowered, specifically $S_k \in [0, 1]$, and $C_{kt} \in [-60, -40]$. In both groups of instances, T has been set to 10, and the number of families K, corresponding to the number of qubits in the QUBO, ranges in the set $\{5, 8, 11, 14\}$. Both groups have been initially tested on M-ADMM-H with QUBO solved via CPLEX on a classical device. In this case, the feasible solution in which no item is assigned to the knapsack is very often the only feasible solution found, which can be arbitrarily far from the optimal value. For the simulations with VQE, Table 5 reports the average results obtained on three instances for fixed K in Group 1, with 3-ADMM-H. While 3-ADMM-H with VQE-SPSA fails to converge within

TABLE 6. Computational Results of 3-ADMM-H on 12 MISK Instances in Group 2. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

Instances	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	19	6.74%	100.00%	0.00%	23.70%	6	100.00%	100.00%	0.00%	100.00%
K5IT20	5	10	12.88%	100.00%	33.33%	14.78%	6	100.00%	100.00%	0.00%	100.00%
K5IT50	5	7	47.30%	100.00%	0.00%	61.11%	6	100.00%	100.00%	0.00%	100.00%
K8IT10	8	24	9.82%	100.00%	0.00%	12.06%	6	3.94%	100.00%	0.00%	83.33%
K8IT20	8	14	7.56%	100.00%	33.33%	4.29%	6	100.00%	100.00%	0.00%	100.00%
K8IT50	8	11	20.54%	100.00%	0.00%	14.88%	6	100.00%	100.00%	0.00%	100.00%
K11IT10	11	30	14.36%	66.67%	0.00%	0.00%	107	21.46%	100.00%	0.00%	0.00%
K11IT20	11	12	12.74%	100.00%	0.00%	0.00%	6	100.00%	100.00%	0.00%	47.62%
K11IT50	11	5	21.56%	0.00%	0.00%	1.52%	8	22.42%	100.00%	0.00%	20.37%
K14IT10	14	19	18.19%	66.67%	0.00%	0.00%	7	28.72%	100.00%	0.00%	57.14%
K14IT20	14	6	21.57%	0.00%	0.00%	0.00%	6	100.00%	100.00%	0.00%	0.00%
K14IT50	14	3	24.11%	0.00%	0.00%	0.00%	6	12.09%	0.00%	0.00%	0.00%

TABLE 7. Computational Results of 2-ADMM-H on 12 MISK Instances in Group 1. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

Instances	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	5	66.67%	100.00%	33.33%	70.00%	2	100.00%	100.00%	0.00%	100.00%
K5IT20	5	26	41.72%	100.00%	0.00%	24.79%	22	100.00%	100.00%	0.00%	100.00%
K5IT50	5	1	205.00%	0.00%	0.00%	72.22%	23	100.00%	100.00%	0.00%	100.00%
K8IT10	8	9	104.15%	100.00%	0.00%	11.67%	2	546.00%	100.00%	0.00%	50.00%
K8IT20	8	63	0.01%	100.00%	0.00%	11.82%	17	111.05%	100.00%	0.00%	100.00%
K8IT50	8	25	28.66%	100.00%	0.00%	0.00%	13	48.43%	0.00%	0.00%	100.00%
K11IT10	11	37	75.31%	100.00%	0.00%	0.00%	93	96.39%	100.00%	0.00%	0.00%
K11IT20	11	12	22.94%	100.00%	0.00%	0.00%	22	100.00%	100.00%	0.00%	88.56%
K11IT50	11	8	77.39%	66.67%	0.00%	0.00%	23	100.00%	100.00%	0.00%	75.00%
K14IT10	14	25	46.42%	100.00%	0.00%	0.00%	4	66.83%	100.00%	0.00%	25.00%
K14IT20	14	4	135.40%	33.33%	0.00%	0.00%	12	93.28%	0.00%	0.00%	0.00%
K14IT50	14	3	132.06%	33.33%	0.00%	0.00%	6	93.77%	100.00%	0.00%	0.00%

TABLE 8. Computational Results of 2-ADMM-H on 12 MISK Instances in Group 2. The QUBO subproblems have been solved via VQE with SPSA and COBYLA solvers with 10, 20, and 50 maximum inner iterations.

Instances	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	2	100.00%	100.00%	0.00%	100.00%	2	100.00%	100.00%	0.00%	100.00%
K5IT20	5	84	0.03%	100.00%	0.00%	13.24%	27	100.00%	100.00%	0.00%	100.00%
K5IT50	5	86	0.00%	100.00%	0.00%	64.60%	19	98.85%	66.67%	0.00%	100.00%
K8IT10	8	6	28.74%	100.00%	0.00%	2.78%	2	3.94%	100.00%	0.00%	50.00%
K8IT20	8	89	0.01%	100.00%	0.00%	0.00%	2	22.32%	0.00%	0.00%	100.00%
K8IT50	8	29	10.53%	100.00%	0.00%	8.33%	1	16.86%	0.00%	0.00%	100.00%
K11IT10	11	8	31.70%	66.67%	0.00%	0.00%	113	13.54%	100.00%	0.00%	0.00%
K11IT20	11	36	6.38%	100.00%	0.00%	0.00%	27	100.00%	100.00%	0.00%	91.83%
K11IT50	11	12	4.78%	100.00%	0.00%	0.00%	27	100.00%	100.00%	0.00%	75.00%
K14IT10	14	20	11.63%	66.67%	0.00%	0.00%	4	28.72%	100.00%	0.00%	25.00%
K14IT20	14	5	14.13%	66.67%	0.00%	0.00%	9	37.42%	0.00%	0.00%	0.00%
K14IT50	14	3	26.63%	0.00%	0.00%	0.00%	6	27.52%	33.33%	0.00%	0.00%

1 h for instances with $K \geq 8$, and 20 SPSA inner iterations, it converges with VQE-COBYLA in a few outer iterations, and produces more feasible solutions. However, VQE-SPSA yields better results in terms of optimality gap, especially when the maximum number of SPSA inner iterations is set to 20. Feasible solutions are found for all instances with VQE-COBYLA with 20 COBYLA inner iterations. The number of ADMM outer iterations generally decreases with the increase of the inner iterations of the classical optimizer.

The Group 2 instances are solved with average optimality gap of 18.11% with VQE-SPSA, as shown in Table 6, reporting a 65% decrease of this metric with respect to Group 1. Hence, 3-ADMM-H finds solutions of higher quality in case the continuous decision variables play an important role in the MBO model. Using VQE-COBYLA to solve the QUBO is beneficial for the larger-sized instances with $K = 14$, since the average optimality gap drops to 12.09% with 50 maximum inner iterations, in 1 h of computation.
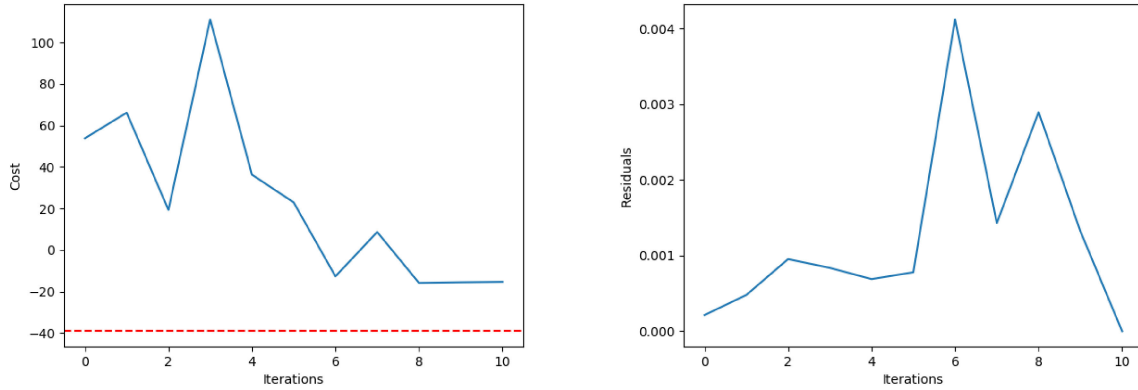
FIGURE 10. Plots of solution cost and residuals for instance K11T10I1, solved by 3-ADMM-H with COBYLA with 50 inner iterations. The optimal solution value is reported in red dashes.

TABLE 9. Average Results of 3-ADMM-H on 40 BP Instances With $N = 2, 3$ and $Q = 40$. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

		SPSA					COBYLA				
Instance	BinVars	IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
N2C40IT10	2	1	50.00%	0.00%	0.00%	0.00%	10	0.00%	100.00%	100.00%	93.15%
N2C40IT20	2	273	0.00%	100.00%	100.00%	48.26%	1	0.00%	100.00%	100.00%	100.00%
N2C40IT50	2	137	0.00%	100.00%	100.00%	70.05%	1	0.00%	100.00%	100.00%	100.00%
N3C40IT10	7	110	40.35%	63.16%	52.63%	3.06%	101	36.67%	20.00%	10.00%	0.00%
N3C40IT20	7	55	17.54%	89.47%	78.95%	0.00%	51	67.50%	60.00%	15.00%	0.26%
N3C40IT50	7	20	24.56%	57.89%	57.89%	1.03%	57	21.67%	80.00%	60.00%	0.14%

Similarly to what observed for the BP problem, the percentage of QUBOs solved to optimality by VQE tends to decrease with the increase of the number of qubits. With VQE-COBYLA, almost all QUBOs are solved to optimality for instances with up to eight qubits. The exception is given by the simulation with ten inner iterations for COBYLA and $K = 8$. We observe that, for the Group 2 instances, a lower percentage of QUBOs solved to optimality corresponds to M-ADMM-H solutions with value closer to the optimal (cf. Remark 2, and Section IV-C). Almost half of the QUBOs are solved to optimality by VQE-COBYLA on the instances with 11 qubits, while VQE-SPSA solves exactly less than 3% of the QUBOs. Fig. 10 shows solution costs and value of the residuals reported in the 3-ADMM-H outer iterations on instance K11T10I1 with 11 qubits. The solution cost changes at each outer iteration in a nonmonotonic way, and 3-ADMM-H converges to a feasible solution in ten outer iterations.

For the 2-ADMM-H implementation, the results are reported in Tables 7 and 8. As observed for 3-ADMM-H, 2-ADMM-H with VQE-SPSA delivers solutions with a lower average gap to optimality for the Group 2 instances. The convergence of 2-ADMM-H is slower than 3-ADMM-H. Instances with 5 and 11 qubits and 20 maximum COBYLA inner iterations are solved by 2-ADMM-H within 1 h of computation. Setting IT = 10 ensures faster simulations, at the price of solution quality. Regarding the feasibility, on the one hand 2-ADMM-H with VQE-SPSA and IT $\geq$ 20 finds feasible solutions for 83.33% of the Group 1 instances, and 91.67% of the Group 2 instances, and on the other hand 2-ADMM-H with VQE-COBYLA yields feasible solutions in, respectively, 75% and 50% of the cases. As in

the 3-ADMM-H case, VQE-COBYLA solves all QUBOs to optimality on instances with up to 8 qubits, except for the case of 20 maximum inner iterations. The percentage of QUBOs solved on the 11-qubits instances increases substantially with respect to the 3-ADMM-H implementation.

The simulations of M-ADMM-H with QAOA as quantum solver are reported in Tables 9–14 in the Appendix. The results further corroborate the claim that for Group 2 instances, M-ADMM-H find solutions of higher quality. This is observed in the drop of the solution gap in the QAOA-SPSA simulations, between Groups 1 and 2 instances. The gap drop corresponds to 76% for 3-ADMM-H, and 51% for 2-ADMM-H. Hence, the impact of continuous decision variables in the convergence of M-ADMM-H could deserve more future studies.

VIII. CONCLUSION
In this article, we have proposed an iterative heuristic method M-ADMM-H, based on ADMM, to solve MBOs on current noisy quantum devices, as well as on classical computers whenever a QUBO solver is available. The method relies on a decomposition of MBO into a QUBO subproblem, which can be tackled via quantum optimization solvers such as VQE and QAOA, and convex subproblems. This enables to extend the range of mathematical optimization problems that can be solved on quantum devices. The method has been tested via the Qiskit framework with VQE as quantum QUBO solver on two representative MBO problems, namely BP Problem, and MISK problem. The simulations indicated the effectiveness of M-ADMM-H in finding solutions feasible for the MBO formulations. In particular, for BP instances

TABLE 10. Average Results of 2-ADMM-H on 40 BP Instances With $N = 2, 3$ and $Q = 40$. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

		SPSA					COBYLA			
Instance	BinVars	IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
N2C40IT10	2	1	50.00%	0.00%	0.00%	0.00%	7	0.00%	100.00%	100.00%	88.39%
N2C40IT20	2	500	0.00%	100.00%	100.00%	62.99%	1	0.00%	100.00%	100.00%	100.00%
N2C40IT50	2	52	0.00%	100.00%	100.00%	69.67%	1	0.00%	100.00%	100.00%	100.00%
N3C40IT10	7	86	47.37%	63.16%	31.58%	6.87%	106	41.67%	20.00%	10.00%	10.00%
N3C40IT20	7	59	12.28%	84.21%	73.68%	24.42%	75	62.50%	50.00%	15.00%	10.46%
N3C40IT50	7	21	25.44%	68.42%	68.42%	17.56%	83	26.67%	90.00%	65.00%	20.69%

TABLE 11. Computational Results of 3-ADMM-H on 12 MISK Instances in Group 1. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

		SPSA					COBYLA			
Instances	BinVars	IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	19	62.13%	100.00%	33.33%	41.56%	6	100.00%	100.00%	0.00%	100.00%
K5IT20	5	11	4.91%	100.00%	33.33%	25.87%	6	100.00%	100.00%	0.00%	100.00%
K5IT50	5	7	198.19%	100.00%	0.00%	52.80%	6	100.00%	100.00%	0.00%	100.00%
K8IT10	8	14	103.15%	100.00%	0.00%	7.69%	6	546.00%	100.00%	0.00%	100.00%
K8IT20	8	26	55.69%	100.00%	0.00%	7.55%	6	100.00%	100.00%	0.00%	100.00%
K8IT50	8	11	136.61%	66.67%	0.00%	45.42%	6	100.00%	100.00%	0.00%	100.00%
K11IT10	11	58	33.30%	100.00%	0.00%	0.00%	7	118.73%	100.00%	0.00%	100.00%
K11IT20	11	34	116.66%	100.00%	0.00%	0.00%	8	117.00%	100.00%	0.00%	34.52%
K11IT50	11	14	80.41%	66.67%	0.00%	10.19%	11	81.00%	100.00%	0.00%	18.18%
K14IT10	14	30	48.90%	100.00%	0.00%	0.00%	8	66.83%	100.00%	0.00%	100.00%
K14IT20	14	12	73.31%	66.67%	0.00%	0.00%	41	54.23%	100.00%	0.00%	0.00%
K14IT50	14	8	116.72%	66.67%	0.00%	0.00%	19	51.46%	100.00%	0.00%	0.00%

TABLE 12. Computational Results of 3-ADMM-H on 12 MISK Instances in Group 2. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

		SPSA					COBYLA			
Instances	BinVars	IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	19	6.74%	100.00%	0.00%	23.70%	6	100.00%	100.00%	0.00%	100.00%
K5IT20	5	24	9.80%	100.00%	0.00%	14.78%	6	100.00%	100.00%	0.00%	100.00%
K5IT50	5	7	30.65%	100.00%	0.00%	61.11%	6	100.00%	100.00%	0.00%	100.00%
K8IT10	8	24	9.82%	100.00%	0.00%	12.06%	6	3.94%	100.00%	0.00%	100.00%
K8IT20	8	49	2.07%	100.00%	0.00%	3.41%	6	100.00%	100.00%	0.00%	100.00%
K8IT50	8	21	12.53%	100.00%	0.00%	14.88%	6	100.00%	100.00%	0.00%	100.00%
K11IT10	11	47	8.59%	100.00%	0.00%	0.00%	10	13.54%	33.33%	0.00%	0.00%
K11IT20	11	45	3.22%	100.00%	0.00%	0.00%	7	29.84%	100.00%	0.00%	19.84%
K11IT50	11	13	14.58%	100.00%	0.00%	0.00%	10	16.11%	100.00%	0.00%	20.37%
K14IT10	14	30	10.47%	100.00%	0.00%	0.00%	5	28.72%	100.00%	0.00%	35.00%
K14IT20	14	18	12.92%	100.00%	0.00%	0.00%	41	9.56%	100.00%	0.00%	0.00%
K14IT50	14	6	21.88%	33.33%	0.00%	0.00%	20	9.63%	100.00%	0.00%	0.00%

TABLE 13. Computational Results of 2-ADMM-H on 12 MISK Instances in Group 1. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

		SPSA					COBYLA			
Instances	BinVars	IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	5	66.67%	100.00%	33.33%	70.00%	2	100.00%	100.00%	0.00%	100.00%
K5IT20	5	2	59.92%	100.00%	0.00%	61.11%	2	100.00%	100.00%	0.00%	100.00%
K5IT50	5	3	40.25%	100.00%	33.33%	41.67%	2	100.00%	100.00%	0.00%	100.00%
K8IT10	8	8	123.85%	100.00%	0.00%	13.06%	2	546.00%	100.00%	0.00%	100.00%
K8IT20	8	29	80.86%	100.00%	0.00%	16.67%	2	100.00%	100.00%	0.00%	100.00%
K8IT50	8	2	78.82%	100.00%	0.00%	16.67%	2	100.00%	100.00%	0.00%	100.00%
K11IT10	11	49	68.36%	100.00%	0.00%	0.00%	66	121.62%	100.00%	0.00%	100.00%
K11IT20	11	12	64.91%	100.00%	0.00%	0.00%	5	121.62%	100.00%	0.00%	24.44%
K11IT50	11	17	101.24%	100.00%	0.00%	1.96%	35	110.19%	100.00%	0.00%	0.00%
K14IT10	14	11	110.14%	33.33%	0.00%	0.00%	4	66.83%	100.00%	0.00%	100.00%
K14IT20	14	6	122.53%	33.33%	0.00%	0.00%	1	71.15%	0.00%	0.00%	0.00%
K14IT50	14	5	44.43%	66.67%	0.00%	0.00%	19	46.81%	100.00%	0.00%	0.00%

with two and three items, feasible solutions are found with an average optimality gap of at most 7.50%. In this case, setting SPSA in 2-ADMM-H as the VQE solver with 50 iterations delivers the best results. On MISK instances, VQE is beneficial to explore feasible solutions different to a trivial one found via the classical computation with CPLEX. It has also been highlighted that 3-ADMM-H finds solutions of higher quality in case the continuous decision variables play an important role in the MBO model, and this could deserve future investigation.

TABLE 14. Computational Results of 2-ADMM-H on 12 MISK Instances in Group 2. The QUBO subproblems have been solved via QAOA with SPSA and COBYLA solvers with 10, 20, and 50 maximum iterations.

Instances	BinVars	SPSA					COBYLA				
		IT	Gap	Feas	Opt	QUBO	IT	Gap	Feas	Opt	QUBO
K5IT10	5	2	100.00%	100.00%	0.00%	100.00%	2	100.00%	100.00%	0.00%	100.00%
K5IT20	5	4	57.44%	100.00%	0.00%	31.94%	2	100.00%	100.00%	0.00%	100.00%
K5IT50	5	2	67.05%	100.00%	0.00%	50.00%	2	100.00%	100.00%	0.00%	100.00%
K8IT10	8	6	28.74%	100.00%	0.00%	2.78%	2	3.94%	100.00%	0.00%	100.00%
K8IT20	8	6	38.58%	100.00%	0.00%	16.67%	2	100.00%	100.00%	0.00%	100.00%
K8IT50	8	2	49.56%	100.00%	0.00%	33.33%	2	100.00%	100.00%	0.00%	100.00%
K11IT10	11	47	8.59%	100.00%	0.00%	0.00%	10	13.54%	33.33%	0.00%	0.00%
K11IT20	11	26	9.60%	100.00%	0.00%	3.04%	17	14.23%	33.33%	0.00%	0.00%
K11IT50	11	14	15.56%	100.00%	0.00%	6.67%	23	13.90%	100.00%	0.00%	6.06%
K14IT10	14	30	10.47%	100.00%	0.00%	0.00%	5	28.72%	100.00%	0.00%	35.00%
K14IT20	14	16	18.54%	100.00%	0.00%	0.00%	2	52.21%	100.00%	0.00%	0.00%
K14IT50	14	8	24.77%	100.00%	0.00%	0.00%	17	9.45%	100.00%	0.00%	0.00%

It is important to observe that M-ADMM-H is a heuristic optimization algorithm for a class of MBO formulations, and it is not tailored to the two applications addressed in this article, namely BP and MISK problems; therefore, the results in terms of feasibility are not trivial on these combinatorial problems.

In theory, we have presented formal requirements under which 3-ADMM-H is guaranteed to converge to a stationary point of a pertinent augmented Lagrangian, which applies on quantum and classical computers alike. In practice, we have offer a glimpse on current research in combinatorial optimization in quantum computing, along with assumptions, challenges, and open problems.

Future works can include the investigation of the impact of continuous decision variables in the M-ADMM-H convergence, the integration of techniques to enforce the feasibility of equality constraints of MBO in the QUBO subproblems [85], different decomposition approaches to devise QUBO subproblems, the combination of ADMM with slack variable approaches [21], and alternative decomposition approaches to devise QUBO subproblems for MBO.

APPENDIX
QAOA SIMULATIONS
Simulation results are provided in Tables 9–14.

REFERENCES
[1] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Oper. Res.*, vol. 54, no. 3, pp. 573–586, 2006, doi: 10.1287/opre.1060.0283.
[2] J. Riera-Ledesma and J.-J. Salazar-González, "Solving the asymmetric traveling purchaser problem," *Ann. Oper. Res.*, vol. 144, no. 1, pp. 83–97, 2006, doi: 10.1007/s10479-006-0014-y.
[3] C. Gambella, F. Maggioni, and D. Vigo, "A stochastic programming model for a tactical solid waste management problem," *Euro. J. Oper. Res.*, vol. 273, no. 2, pp. 684–694, 2019, doi: 10.1016/j.ejor.2018.08.005.
[4] C. A. Floudas, *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. London, U.K.: Oxford Univ. Press, 1995, doi: 10.1023/A:1008256302713.
[5] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131, 2013, doi: 10.1017/S0962492913000032.
[6] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," *New J. Phys.*, vol. 18, no. 2, 2016, Art. no. 023023, doi: 10.1088/1367-2630/18/2/023023.
[7] N. Moll *et al.*, "Quantum optimization using variational algorithms on near-term quantum devices," *Quantum Sci. Technol.*, vol. 3, no. 3, 2018, Art. no. 030503, doi: 10.1088/2058-9565/aab822.
[8] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*.
[9] G. G. Guerreschi and A. Matsuura, "QAOA for max-cut requires hundreds of qubits for quantum speed-up," *Sci. Rep.*, vol. 9, no. 1, 2019, Art. no. 6903, doi: 10.1038/s41598-019-43176-9.
[10] E. Zahedinejad and A. Zaribafiyan, "Combinatorial optimization on gate model quantum computers: A survey," 2017, *arXiv:1708.05294*.
[11] A. Peruzzo *et al.*, "A variational eigenvalue solver on a photonic quantum processor," *Nat. Commun.*, vol. 5, 2014, Art. no. 4213, doi: 10.1038/ncomms5213.
[12] A. Kandala *et al.*, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, pp. 242–246, 2017, doi: 10.1038/nature23879.
[13] J. Romero, J. P. Olson, and A. Aspuru-Guzik, "Quantum autoencoders for efficient compression of quantum data," *Quantum Sci. Technol.*, vol. 2, no. 4, 2017, Art. no. 045001, doi: 1088/2058-9565/aa8072.
[14] E. Farhi and H. Neven, "Classification with quantum neural networks on near term processors," 2018, *arXiv:1802.06002*.
[15] E. S. Fried, N. P. Sawaya, Y. Cao, I. D. Kivlichan, J. Romero, and A. Aspuru-Guzik, "qTorch: The quantum tensor contraction handler," *PLoS One*, vol. 13, no. 12, 2018, Art. no. e0208510, doi: 10.1371/journal.pone.0208510.
[16] E. Farhi, J. Goldstone, S. Gutmann, and H. Neven, "Quantum algorithms for fixed qubit architectures," 2017, *arXiv:1703.06199*.
[17] P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, "Improving variational quantum optimization using CVaR," *Quantum*, vol. 4, p. 256, 2020, doi: 10.22331/q-2020-04-20-256.
[18] G. Nannicini, "Performance of hybrid quantum-classical variational heuristics for combinatorial optimization," *Phys. Rev. E*, vol. 99, no. 1, 2019, Art. no. 013304, doi: 10.1103/PhysRevE.99.013304.
[19] A. V. Fiacco and G. P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, vol. 4. Philadelphia, PA, USA: SIAM, 1990, doi: 10.1002/zamm.19720520716.
[20] C.-Y. Wang and D. Li, "Unified theory of augmented lagrangian methods for constrained global optimization," *J. Glob. Optim.*, vol. 44, no. 3, 2009, Art. no. 433, doi: 10.1007/s10898-008-9347-1.
[21] L. Braine, D. J. Egger, J. Glick, and S. Woerner, "Quantum algorithms for mixed binary optimization applied to transaction settlement," 2019, *arXiv:1910.05788*.
[22] V. Protopopescu and J. Barhen, "Solving a class of continuous global optimization problems using quantum algorithms," *Phys. Lett. A*, vol. 296, no. 1, pp. 9–14, 2002, doi: 10.1016/S0375-9601(02)00187-1.

[23] T. Vyskočil, S. Pakin, and H. N. Djidjev, "Embedding inequality constraints for quantum annealing optimization," in *Proc. Int. Workshop Quantum Technol. Optim. Problems* 2019, pp. 11–22, doi: 10.1007/978-3-030-14082-3_2.

[24] S. Boyd *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011, doi: 10.1561/2200000016.

[25] J. Eckstein, "Splitting methods for monotone operators with applications to parallel optimization," Ph.D. dissertation, Dept. Civil Eng., MIT, Cambridge, MA, USA, Jun. 1989.

[26] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods.* Belmont, MA, USA: Athena Sci., 1997.

[27] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in *ad hoc* WSNs with noisy links— Part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process.*, vol. 56, no. 1, pp. 350–364, Jan. 2008, doi: 10.1109/TSP.2007.906734.

[28] R. Glowinski and A. Marrocco, "Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualité, d'une classe de problems de Dirichlet non lineares," *Revue Francaise d'Automatique, Informatique, et Recherche Opérationelle*, vol. 9, pp. 41–76, 1975.

[29] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximations," *Comput. Math. Appl.*, vol. 2, pp. 17–40, 1976, doi: 10.1016/0898-1221(76)90003-1.

[30] B. He and X. Yuan, "On the $O(1/t)$ convergence rate of alternating direction method," *SIAM J. Optim.*, vol. 22, pp. 1431–1448, 2011, doi: 10.1137/110847639.

[31] D. Davis and W. Yin, "Convergence rate analysis of several splitting schemes," in *Splitting Methods in Communication and Imaging, Science and Engineering*, R. Glowinski, S. Osher, and W. Yin, Eds. Cham, Switzerland: Springer, 2017, doi: 10.1007/978-3-319-41589-5_4.

[32] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. I. Jordan, "A general analysis of the convergence of ADMM," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 343–352, doi: 10.5555/3045118.3045156.

[33] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): Quadratic problems," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 644–658, Mar. 2015, doi: 10.1109/TAC.2014.2354892.

[34] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," *J. Sci. Comput.*, vol. 66, no. 3, pp. 889–916, 2016, doi: 10.1007/s10915-015-0048-x.

[35] P. Giselsson and S. Boyd, "Linear convergence and metric selection for Douglas-Rachford splitting and ADMM," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 532–544, Feb. 2017, doi: 10.1109/TAC.2016.2564160.

[36] S. Diamond, R. Takapoui, and S. Boyd, "A general system for heuristic minimization of convex functions over non-convex sets," *Optim. Methods Softw.*, vol. 33, no. 1, pp. 165–193, 2018, doi: 10.1080/10556788.2017.1304548.

[37] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *J. Sci. Comput.*, vol. 78, no. 1, pp. 29–63, 2019, doi: 10.1007/s10915-018-0757-z.

[38] K. Sun and X. A. Sun, "A two-level distributed algorithm for general constrained non-convex optimization with global convergence," 2019, *arXiv:1902.07654*.

[39] B. Wu and B. Ghanem, "ℓ_p-box ADMM: A versatile framework for integer programming," *IEEE Trans. Pattern Anal Mach. Intell.*, vol. 41, no. 7, pp. 1695–1708, Jul. 2019, doi: 10.1109/TPAMI.2018.2845842.

[40] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM J. Optim.*, vol. 26, no. 1, pp. 337–364, 2016, doi: 10.1109/ICASSP.2015.7178689.

[41] A. Themelis and P. Patrinos, "Douglas-Rachford splitting and ADMM for nonconvex optimization: Tight convergence results," *SIAM J. Optim.*, vol. 30, no. 1, pp. 149–181, 2020, doi: 10.1137/18M1163993.

[42] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, "A simple effective heuristic for embedded mixed-integer quadratic programming," *Int. J. Control*, vol. 93, pp. 1–11, 2017, doi: 10.1080/00207179.2017.1316016.

[43] A. Gilliam, S. Woerner, and C. Gonciulea, "Grover adaptive search for constrained polynomial binary optimization," 2019, *arXiv:1912.04088*.

[44] F. L. B. ao, R. Kueng, and D. Stilck França, "Faster quantum and classical SDP approximations for quadratic binary optimization," 2019, *arXiv:1909.04613*.

[45] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004, doi: 10.1017/CBO9780511804441.

[46] B. Jiang, T. Lin, S. Ma, and S. Zhang, "Structured nonconvex and nonsmooth optimization: Algorithms and iteration complexity analysis," *Comput. Optim. Appl.*, vol. 72, pp. 115–157, 2019, doi: 10.1007/s10589-018-0034-y.

[47] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods," *Math. Program.*, vol. 137, nos. 1/2, pp. 91–129, 2013, doi: 10.1007/s10107-011-0484-9.

[48] J. G. Melo and R. D. C. Monteiro, "Iteration-complexity of a Jacobi-type non-Euclidean ADMM for multi-block linearly constrained nonconvex programs," 2017, *arXiv:1705.07229*.

[49] J. Wang, L. Zhao, and L. Wu, "Multi-convex inequality-constrained alternating direction method of multipliers," 2019, *arXiv:1902.10882*.

[50] R. Rockafellar, *Convex Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 1970, doi: 10.1017/S0013091500010142.

[51] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, "Discrete optimization using quantum annealing on sparse Ising models," *Front. Phys.*, vol. 2, 2014, doi: 10.3389/fphy.2014.00056.

[52] G. Rosenberg, M. Vazifeh, B. Woods, and E. Haber, "Building an iterative heuristic solver for a quantum annealer," *Comput. Optim. Appl.*, vol. 65, pp. 845–869, 2016, doi: 10.1007/s10589-016-9844-y.

[53] H. Karimi, G. Rosenberg, and H. G. Katzgraber, "Effective optimization using sample persistence: A case study on quantum annealers and various Monte Carlo optimization methods," *Phys. Rev. W*, vol. 96, 2017, Art. no. 043312, doi: 10.1103/PhysRevE.96.043312.

[54] R. Shaydulin, H. Ushijima-Mwesigwa, C. F. A. Negre, I. Safro, S. M. Mniszewski, and Y. Alexeev, "A hybrid approach for solving optimization problems on small quantum computers," *IEEE Comput.*, vol. 52, no. 6, pp. 18–26, Jun. 2019, doi: 10.1109/MC.2019.2908942.

[55] F. G. S. L. Brandão, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, "For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances," 2018, *arXiv:1812.04170*.

[56] C. Huang, M. Szegedy, F. Zhang, X. Gao, J. Chen, and Y. Shi, "Alibaba cloud quantum development platform: Applications to quantum algorithm design," 2019, *arXiv:1909.02559*.

[57] R. Shaydulin and Y. Alexeev, "Evaluating quantum approximate optimization algorithm: A case study," in *Proc. 10th Int. Green Sustain. Comput. Conf.*, 2019, pp. 1–6, doi: 10.1109/IGSC48788.2019.8957201.

[58] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Phys. Rev. X*, vol. 10, no. 2, Jun. 2020, Art. no. 021067, doi: 10.1103/PhysRevX.10.021067.

[59] M. B. Hastings, "Classical and quantum bounded depth approximation algorithms," 2019, *arXiv:1905.07047*.

[60] G. E. Crooks, "Performance of the quantum approximate optimization algorithm on the maximum cut problem," 2018, *arXiv:1811.08419*.

[61] R. Shaydulin, I. Safro, and J. Larson, "Multistart methods for quantum approximate optimization," in *Proc. IEEE High Perform. Extreme Comput. Conf.*, 2019, pp. 1–8, doi: 10.1109/HPEC.2019.8916288.

[62] L. Chen, D. Sun, and K.-C. Toh, "A note on the convergence of ADMM for linearly constrained convex optimization problems," *Comput. Optim. Appl.*, vol. 66, no. 2, pp. 327–343, 2017, doi: 10.1007/s10589-016-9864-7.

[63] S. Lojasiewicz, "Sur la géométrie semi- et sous-analytique," *Ann. Inst. Fourier (Grenoble)*, vol. 43, no. 5, pp. 1575–1595, 1993, doi: 10.5802/aif.1384.

[64] J. Bolte, A. Daniilidis, and A. Lewis, "The Lojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems," *SIAM J. Optim.*, vol. 17, no. 4, pp. 1205–1223, 2007, doi: 10.1137/050644641.

[65] H. Attouch, J. Bolte, and B. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods," *Math. Program.*, vol. 137, nos. 1/2, pp. 91–129, 2013, doi: 10.1007/s10107-011-0484-9.

[66] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran, "Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Lojasiewicz inequality," *Math. Oper. Res.*, vol. 35, no. 2, pp. 438–457, 2010, doi: 10.1287/moor.1100.0449.

[67] P. Toth and D. Vigo, *Vehicle Routing: Problems, Methods, and Applications*. Philadelphia, PA, USA: SIAM, 2014, doi: 10.1137/1.9781611973594.

[68] N. Bianchessi, R. Mansini, and M. G. Speranza, "The distance constrained multiple vehicle traveling purchaser problem," *Eur. J. Oper. Res.*, vol. 235, no. 1, pp. 73–87, 2014, doi: 10.1016/j.ejor.2013.10.018.

[69] J. F. Campbell, "Integer programming formulations of discrete hub location problems," *Eur. J. Oper. Res.*, vol. 72, no. 2, pp. 387–405, 1994, doi: 10.1016/0377-2217(94)90318-2.

[70] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, "SDPT3—a MATLAB software package for semidefinite programming, version 1.3," *Optim. Methods Softw.*, vol. 11, nos. 1/4, pp. 545–581, 1999, doi:10.1080/10556789908805762.

[71] E. D. Andersen and K. D. Andersen, "The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm," in *High Performance Optimization*. Berlin, Germany: Springer, 2000, pp. 197–232, doi: 10.1007/978-1-4757-3216-0_8.

[72] M. Fischetti, F. Glover, and A. Lodi, "The feasibility pump," *Math. Program.*, vol. 104, no. 1, pp. 91–104, 2005, doi: 10.1007/s10107-004-0570-3.

[73] L. Bertacco, M. Fischetti, and A. Lodi, "A feasibility pump heuristic for general mixed-integer problems," *Discr. Optim.*, vol. 4, no. 1, pp. 63–76, 2007, doi: 10.1016/j.disopt.2006.10.001.

[74] M. Delorme, M. Iori, and S. Martello, "Bin packing and cutting stock problems: Mathematical models and exact algorithms," *Eur. J. Oper. Res.*, vol. 255, no. 1, pp. 1–20, 2016, doi: 10.1016/j.ejor.2016.04.030.

[75] S. Martello, *Knapsack Problems: Algorithms and Computer Implementations* (Discrete Mathematics, Optimization). Hoboken, NJ, USA: Wiley, 1990, doi: 10.5555/98124.

[76] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Berlin, Germany: Springer, 2013, doi: 10.1007/978-3-540-24777-7.

[77] E. Y.-H. Lin, "A bibliographical survey on some well-known nonstandard knapsack problems," *INFOR, Inf. Syst. Oper. Res.*, vol. 36, no. 4, pp. 274–317, 1998, doi: 10.1080/03155986.1998.11732364.

[78] N. Altay, P. E. Robinson Jr., and K. M. Bretthauer, "Exact and heuristic solution approaches for the mixed integer setup knapsack problem," *Eur. J. Oper. Res.*, vol. 190, no. 3, pp. 598–609, 2008, doi: 10.1016/j.ejor.2007.07.003.

[79] "Qiskit: An open-source quantum computing software development framework," 2019. Accessed: Nov. 2019. [Online]. Available: https://qiskit.org/

[80] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Trans. Autom. Control*, vol. 37, no. 3, pp. 332–341, Mar. 1992, doi: 10.1109/9.119632.

[81] S. Gomez and J. P. Hennart, *Advances in Optimization and Numerical Analysis*, vol. 275. Berlin, Geramny: Springer, 2013, doi: 10.1007/978-94-015-8330-5.

[82] A. Alvim, F. S. Glover, C. C. Ribeiro, and D. J. Aloise, "Local search for the bin packing problem," 1999, doi: 10.1023/A:1021837611236.

[83] W. Michiels *et al.*, "Performance ratios for the Karmarkar-Karp differencing method," *Electron. Notes Discr. Math.*, vol. 13, pp. 71–75, 2003, doi: 10.1016/S1571-0653(04)00442-1.

[84] A. Scholl, R. Klein, and C. Jürgens, "Bison: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem," *Comput. Oper. Res.*, vol. 24, no. 7, pp. 627–645, 1997, doi: 10.1016/S0305-0548(96)00082-2.

[85] Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, "XY-mixers: Analytical and numerical results for the quantum alternating operator ansatz," *Phys. Rev. A*, vol. 101, no. 1, Jan. 2020, Art. no. 012320, doi: 10.1103/PhysRevA.101.012320.