Improving ADMM-based Optimization of Mixed Integer Objectives

(51st Annual Conference on Information Sciences and Systems, March 2017)

Alborz Alavian
Department of Electrical and Computer Engineering
University of Maryland
College Park, Maryland 20742, USA
alavian@umd.edu

Michael C. Rotkowitz
Instutite for Systems Research and
Department of Electrical and Computer Engineering
University of Maryland
College Park, Maryland 20742, USA
mcrotk@umd.edu

***Abstract*—We consider a class of mixed integer programs where the problem is convex except for a vector of discrete variables. Two methods based on the Alternating Direction Method of Multipliers (ADMM) are presented. The first, which has appeared in the recent literature, duplicates the discrete variable, with one copy allowed to vary continuously. This results in a simple projection, or rounding, to determine the discrete variable at each iteration. We introduce an alternate method, whereby part of the objective is replaced by a new variable instead. When the objective satisfies a certain condition, this allows the update of the discrete variables to be handled separately for each one, thus maintaining linear complexity of this update, while incorporating some of the objective into the update. Initial comparisons on examples for which both methods are applicable show that the latter exhibits clear improvements in both performance and runtime.**

I. Introduction

Mixed Integer Programs (MIP) are hard problems in general with much interest in finding bounds or approximate solutions for them. These include linear program (LP) and Semidefinite relaxations (SDP). The LP methods consider a linear relaxation of the integer variable to obtain a lower bound, and its projection to the discrete space for an upper-bound, whereas SDP relaxations consider the trace of a rank 1 matrix instead of the terms that involve products of the integer variables [1], [2]. Tighter relaxations can be obtained by lift-and-project methods, that introduce new auxiliary variables to transform the nonlinear integer constraints into a form with linear constraints in a higher dimension, and then solve the convex problem in the higher dimension to obtain a lower-bound [3], [4]. When there would also be higher degree non-convex objectives functions or constraints other than the integer constraint, one can also consider using polynomial optimization methods to obtain such lower-bounds [5]–[7]. We are interested in upper-bounds for MIP problems in this paper and refer the reader to the above works for certificates of the optimality.

The so called relax-and-round algorithms replace the discrete variable with its continuous counterpart and solve the obtained convex program to obtain a lower-bound on the exact optimal value. Projection of this optimal solution onto the discrete set will then give an upper-bound. This projection would simply be a rounding step toward the closest discrete value in each dimension. It has been suggested in [8] that one can use the information from the dual problem through Alternating Direction Method of Multipliers (ADMM) to do several passes of such steps, which will result in a better upper-bound. Although ADMM has been originally developed for convex problems [9], there has been much recent interest in applying it to the non-convex problems, with some analysis of convergence available in some cases [10]. This has led to a broad class of heuristics with great flexibility for problems that were originally very hard to solve. In particular, the binary quadratic problems (BQP) has attracted much attention and authors of [8] have demonstrated their algorithm for this class of problems, which can simply be extended to a more generalized setting as in [11]. Also, [12] has considered using ADMM for the binary constraints, and has compared it to another novel heuristic that encourages finding binary values by considering piecewise-linear functions that penalizes non-binary values. As this piecewise-linear function would be locally convex, better and more flexible sequential algorithms can then be developed based on this heuristic.

We will consider a variant of such ADMM based algorithms in this paper. Instead of only replacing the discrete variable by a relaxed continuous one as in the relax-and-round algorithms, we will try to capture as a generalized part of the objective as possible through an auxiliary equality constraint. This will create the base formulation for the ADMM algorithm. The algorithm then finds the best discrete variable in each step by checking the captured part values at the discrete values, rather than rounding. We will demonstrate when this last part is possible through a linear number of function evaluations in the dimension of the discrete variable, versus an exponential number that corresponds to the exhaustive search. This has shown significant improvements when the objective is not necessarily symmetrical around its optimal point.

One main motivation behind this work was approximating the decentralized assignability measure of [13] that quantifies the robustness of a linear time invariant dynamical systems in the decentralized settings. This metric involves both continuous and binary variables that appear affinely in a non-convex

This material is based upon work supported by the National Science Foundation under Grant No. 1351674.

objective function (a particular singular value of a matrix). We noticed that a preliminary version of our algorithm in this paper could be applied to the aforementioned optimization problem [14]. This method was tracking the actual metric very closely even when we had to use it in conjunction with other convex heuristics for the objective function itself. This led to the interesting question of whether this behavior is what one could expect more generally, and is what we study in this paper.

The organization of this paper is as follows. We will construct the problem setup in Section II and review a very closely related framework in Section III. We will then demonstrate our algorithm in Section IV, discuss when and how it could capture the effect of the discrete variables better in Section IV-A. We will finally investigate this method through a set of numerical examples in Section IV-B.

II. Problem Formulation

We formulate the problem of interest in this section. We consider objectives that have mixed discrete and continuous parts in a form that we will focus on in this paper here and then discuss the generalizations and constraints wherever applicable.

To this end we will first illustrate the space in which the variables would lie and then state the optimization problem of interest. Denote the $\{0,1\}$-valued indicator function by $\mathbf{1}(\cdot)$, the set of real numbers by $\mathbb{R}$, the extended real numbers by $\bar{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$, and the set of binary numbers by $\mathbb{B} = \{0,1\}$. Also denote any finite subset of the reals by $\mathcal{Z}$, i.e.:

$$\mathcal{Z} \triangleq \{\alpha_1, \cdots, \alpha_{|\mathcal{Z}|}\},$$

where $|\mathcal{Z}| < \infty$, and $\alpha_i \in \mathbb{R}$, for $i \in \{1, \cdots, |\mathcal{Z}|\}$. Furthermore, denote the Cartesian product of m possibly different instances of such sets by:

$$\mathcal{Z}^{(m)} \triangleq \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_m.$$

We will denote the projection of a real variable $y \in \mathbb{R}^m$ onto the set $\mathcal{Z}^{(m)}$ by:

$$\Pi_{\mathcal{Z}^{(m)}}(y) = \left[\Pi_{\mathcal{Z}_1}(y_1) \quad \cdots \quad \Pi_{\mathcal{Z}_m}(y_m)\right]^T,$$

where

$$\Pi_{\mathcal{Z}_i}(y_i) = \arg\min_{z \in \mathcal{Z}_i} \|y_i - z\|_2,$$

for $i \in \{1, \cdots, m\}$.

Remark 1: In its simplest form, all such $\mathcal{Z}_i$ could be taken the same as the binary set, i.e., $\mathcal{Z}_i = \mathbb{B}$ for every $i \in \{1, \cdots, m\}$, which would result in $\mathcal{Z}^{(m)} = \mathbb{B}^m$, for which the projection would be simply picking the closest of either 0 or 1 to each element of the vector y, i.e., $\Pi_{\mathbb{B}}(y_i) = \mathbf{1}(y_i > 0.5)$.

Consider the following optimization problem:

$$\text{minimize} \quad f(g(x,z)), \tag{1}$$

with variables $x \in \mathbb{R}^n$ and $z \in \mathcal{Z}^{(m)}$. Throughout the rest of this paper, the inner function $g(\cdot,\cdot)$ is from $\mathbb{R}^n \times \mathcal{Z}^{(m)}$ to $\mathbb{R}^p$. The extended function $f(\cdot)$ is from $\mathbb{R}^p$ to $\bar{\mathbb{R}}$ and is assumed to be convex in its variable.

We will discuss what assumptions we need on the function g. We will clearly state these assumptions and their implications wherever they are imposed in the following sections.

Even without any further constraints, and even if $f \circ g$ is convex, this would typically be a hard problem due to the presence of the discrete variable z.

III. A Round-off Based Algorithm

We will reformulate a class of relax-and-round heuristics for mixed integer programs in this section. We will then discuss how a simple modification to this algorithm would most likely enhance it for the considered class of functions.

We can rewrite the optimization problem (1) with an extra constraint such that instead of having the discrete variable in the objective, we will have it in the constraints:

$$\begin{array}{ll} \text{minimize} & f(g(x,y)) \\ \text{subject to} & y = z, \end{array} \tag{2}$$

with variables $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ and $z \in \mathcal{Z}^{(m)}$.

The augmented Lagrangian for this problem, for any parameter $\rho > 0$, can be written as:

$$L_\rho(x,y,z,\nu) = f(g(x,y)) + \nu^T(y-z) + \frac{\rho}{2}\|y-z\|_2^2,$$

which with some basic rearrangement of the term and a change of variable $\mu = (1/\rho)\nu$ can be equivalently written as:

$$L_\rho(x,y,z,\mu) = f(g(x,y)) + \frac{\rho}{2}\|y-z+\mu\|_2^2 - \frac{\rho}{2}\|\mu\|_2^2.$$

Then, the ADMM algorithm would consist of optimization jointly over the variables (x,y), projecting onto the discrete set $\mathcal{Z}^{(m)}$, and the dual update:

$$\begin{array}{rcl} (x^{(k)}, y^{(k)}) & = & \arg\min\limits_{\substack{x \in \mathbb{R}^n \\ y \in \mathbb{R}^m}} \; L_\rho(x, y, z^{(k-1)}, \mu^{(k-1)}) \\ z^{(k)} & = & \Pi_{\mathcal{Z}^{(m)}}\left(y^{(k)} + \mu^{(k-1)}\right) \\ \mu^{(k)} & = & \mu^{(k-1)} + y^{(k)} - z^{(k)}. \end{array} \tag{3}$$

The second and third steps of the ADMM are straightforward, however the first step requires convexity of $f(g(x,y))$ in (x,y), which in turn could be guaranteed by the following assumption:

Assumption 2: Throughout the rest of this section we will assume that $g(\cdot,\cdot)$ is jointly affine in its variables.

This heuristic was considered for the binary quadratic problems in [8], and has been generalized through the same way to allow for some other kinds of mixed integer programs in [11]. The objective of [8] has the form:

$$f(v) = 1/2 \; v^T P v + q^T v, \tag{4}$$

with positive semidefinite P, which can be matched to (2) by choosing:

$$g(x,z) = z, \tag{5}$$

which indicates that there is no continuous variable. Also it is noteworthy that considering another affine function for $g(x,z)$ other than the one mentioned above, such as $g(x,z) = Az + b$, would be in effect the same as a new quadratic function with P being changed to $A^T P A$ and q to $A^T q + A^T P b$, while still having $g(x,z) = z$. Although this suggests that considering the composition of functions in (2) might not be fundamentally different in the quadratic case as it would yield another similar quadratic function, this is not what one would generally observe in non-quadratic cases. We will describe this aspect in the next section.

IV. Main Idea

We will describe the main idea in this section. We will first lay out the modification to the ADMM algorithm in the previous section and then describe its advantages. We will begin by putting a different assumption on the function $g(\cdot,\cdot)$. This assumption is more restrictive than Assumption 2 in the sense that it requires partial separability over the discrete variable, but is also more general in allowing non-linear dependencies on the discrete variables. Particularly we will require that each element of g would depend on at most one discrete variable, but not necessarily in an affine manner:

Assumption 3: Throughout the rest of this section we will assume that $g : \mathbb{R}^n \times \mathcal{Z}^{(m)} \mapsto \mathbb{R}^p$ would be affine in its continuous variable, and further is such that for all $i \in \{1, \cdots, p\}$ we have that:

$$g_i(x,z) \;=\; \tilde{g}_i(x, z_{l_i}),$$

for some $l_i \in \{1, \cdots, m\}$ and $\tilde{g}_i : \mathbb{R}^n \times \mathcal{Z}_{l_i} \mapsto \mathbb{R}$, and for all $x \in \mathbb{R}^n$ and $z \in \mathcal{Z}^{(m)}$.

This assumption can be equivalently expressed in a more explicit way as in the following remark:

Remark 4: Assumption 3 is equivalent to the following form for each element of the function g, i.e., for all $i \in \{1, \cdots, p\}$, we have that:

$$g_i(x,z) \;=\; \left(a_i(z_{l_i})\right)^T x + b_i(z_{l_i}),$$

for some $l_i \in \{1, \cdots, m\}$, and where $a_i(\cdot)$ and $b_i(\cdot)$ are possibly nonlinear functions of the following form:

$$\begin{array}{rcl} a_i : \mathcal{Z}_{l_i} & \mapsto & \mathbb{R}^n \\ b_i : \mathcal{Z}_{l_i} & \mapsto & \mathbb{R}. \end{array}$$

With this assumption in place, we will consider the ADMM algorithm that will be based on the following equivalent form of (1):

$$\begin{array}{ll} \text{minimize} & f(v) \\ \text{subject to} & v = g(x,z), \end{array} \tag{6}$$

with variables $x \in \mathbb{R}^n$, $v \in \mathbb{R}^p$ and $z \in \mathcal{Z}^{(m)}$.

The augmented Lagrangian for this problem can be written as:

$$\tilde{L}_\rho(x,v,z,\mu) = f(v) + \frac{\rho}{2}\|v - g(x,z) + \mu\|_2^2 - \frac{\rho}{2}\|\mu\|_2^2,$$

for which the ADMM algorithm would be:

$$\begin{array}{rcl} (x^{(k)}, v^{(k)}) & = & \arg\min\limits_{\substack{x\in\mathbb{R}^n \\ v\in\mathbb{R}^p}} \; \tilde{L}_\rho(x, v, z^{(k-1)}, \mu^{(k-1)}) \\ z^{(k)} & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \|v^{(k)} - g(x^{(k)}, z) + \mu^{(k-1)}\|_2^2 \\ \mu^{(k)} & = & \mu^{(k-1)} + v^{(k)} - g(x^{(k)}, z^{(k)}). \end{array} \tag{7}$$

The first step of this algorithm would be a convex minimization step due to the convexity of $f(\cdot)$ and Assumption 3. The second step of this algorithm is what makes this algorithm different from (3). In particular, this step would not correspond to a similar projection as in (3), though it would be computationally tractable due to the assumption that is put on dependency on the discrete variable in $g(\cdot,\cdot)$ (Assumption 3). This is explicitly stated in the following theorem:

Theorem 5: Given a function g that satisfies Assumption 3, the optimal z in (7) can be equivalently obtained by independently solving for each of the discrete variables, i.e., for all $j \in \{1, \cdots, m\}$ we have that:

$$z_j{}^{(k)} \;=\; \arg\min_{z_j\in\mathcal{Z}_j} \sum_{\{i \,|\, l_i = j\}} \left(v_i{}^{(k)} - \tilde{g}_i(x^{(k)}, z_j) + \mu_i{}^{(k-1)}\right)^2. \tag{8}$$

Proof: We have that:

$$\begin{array}{rcl} z^{(k)} & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \tilde{L}_\rho(x^{(k)}, v^{(k)}, z, \mu^{(k-1)}) \\ & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \|v^{(k)} - g(x^{(k)}, z) + \mu^{(k-1)}\|_2^2 \\ & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \sum\limits_{i=1}^{p} \left(v_i{}^{(k)} - \tilde{g}_i(x^{(k)}, z) + \mu_i{}^{(k-1)}\right)^2 \\ & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \sum\limits_{i=1}^{p} \left(v_i{}^{(k)} - \tilde{g}_i(x^{(k)}, z_{l_i}) + \mu_i{}^{(k-1)}\right)^2 \\ & = & \arg\min\limits_{z\in\mathcal{Z}^{(m)}} \; \sum\limits_{j=1}^{m} \sum\limits_{\{i \,|\, l_i=j\}} \Big(v_i{}^{(k)} - \tilde{g}_i(x^{(k)}, z_{l_i}) \\ & & \qquad\qquad + \; \mu_i{}^{(k-1)}\Big)^2 \end{array}$$

$$\Longrightarrow$$

$$z_j{}^{(k)} = \arg\min_{z_j\in\mathcal{Z}_j} \sum_{\{i \,|\, l_i=j\}} \left(v_i{}^{(k)} - \tilde{g}_i(x^{(k)}, z_j) + \mu_i{}^{(k-1)}\right)^2,$$

where the second equality follows because that is the only term involving the discrete variable z, the third is due to the fact that $\|\cdot\|_2^2$ also separates in its elements, the forth is due to Assumption 3, and the fifth is an equivalent representation of the sum from $i = 1$ to p. ■

This alternative approach makes the computation of z-update minimization a tractable one whenever Assumption 3 is in place. This is described in more details in the following remark:

Remark 6 (Per-iteration complexity): The z-update in (7) requires $|\mathcal{Z}^{(m)}| = |\mathcal{Z}_1| \times \cdots \times |\mathcal{Z}_m|$ function evaluations (for instance, 2^m in the binary case), whereas when Assumption 3 is satisfied, solving for z by (8) only requires $\sum_{j=1}^{m} |\mathcal{Z}_j|$ function evaluations (for instance, $2m$ in the binary case). This alternative step has linear complexity, and is comparable in complexity to the projection step in (3).

Remark 7 (Matrix variables): This can be easily generalized to handle matrix variables by replacing the 2-norm with the Frobenius norm, which corresponds to the standard inner product in the matrix spaces. This was what we indeed first considered in [14].

A. Discussions

We will provide more intuitions on the suggested modifications to the ADMM-based algorithm described above in this section.

In (3), the discrete variable is replaced by a continuous one, and the solution of the primal optimization step that is solved with these continuous variables is then projected onto the discrete set in the hope that this projection would still minimize $f(g(x^{(k)}, z))$ for the discrete variable z, which might be not the case in general. The quadratic objective (4) is symmetrical around its optimal point in each of the directions, and thus the projection would be a best choice when one requires separability in the z-update. In other words, as illustrated in Figure 1, when we keep all the variables fixed except for a single one-dimensional discrete variable, the discrete value (0 or 1 in here) that minimizes a quadratic function is indeed the one closest to its critical point (0.4 here).

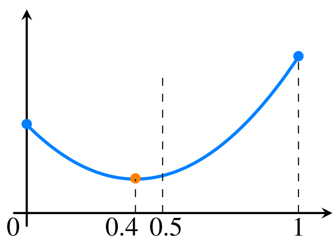


Fig. 1: A quadratic function in which the projection would be a best choice in a single dimension

However, as illustrated in Figure 2, this special property might not be in place for a wide variety of convex functions such as piecewise linear, sum of logarithmics or sum of exponential functions. The proposed ADMM-based method also separates in the z-update, and compared to rounding the solution of the relaxed problem, it will actually plug in the binary values and picks the best among them, making it more likely that it would be a better choice for non-quadratic functions. This would be further investigated through numerical examples in the next section.

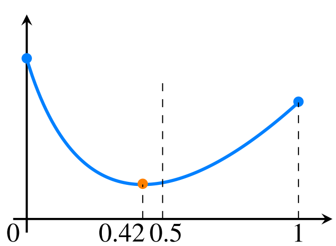


Fig. 2: A logarithmic function for which the projection would not be a best choice even in a single dimension

B. Numerical Examples

We will investigate the performance of the suggested algorithm versus the one based on a relax-and-round method here in this section.

We will compare the algorithms in (3) and (7) for random instances of a problem with fixed dimensions, and compare the run-time and the value that each algorithm obtains in our first example:

Example 8: Consider the optimization problem (1) with a single continuous variable ($n = 1$) and where the discrete variables are all in the binary space ($\mathcal{Z}_i = \mathbb{B}$, for $i = 1. \cdots, m$), i.e., $x \in \mathbb{R}$ and $z \in \mathbb{B}^m$. Let $g(\cdot, \cdot)$ be given as:

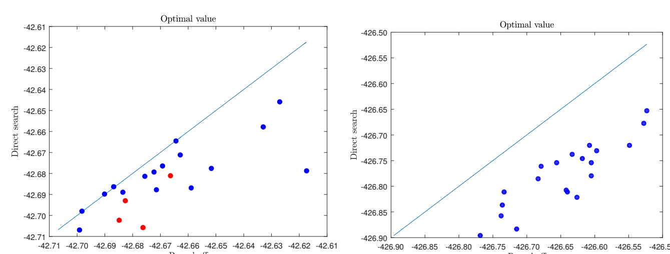
$$g(x, z) \;=\; Dz + b + \mathbf{1}\, x, \tag{9}$$

where D is a diagonal matrix in $\mathbb{R}^{m\times m}$, $b \in \mathbb{R}^m$ and $\mathbf{1}$ is a vector of all ones of compatible dimension. Let $p = m$ and also take $f(\cdot)$ as:

$$f(v) \;=\; \sum_{i=1}^{m} -a_1 \log(a_0 v_i + c_1) - a_2 \log(-a_0 v_i + c_2), \tag{10}$$

where $a_0, a_1, a_2, c_1,$ and c_2 are all positive real numbers, and c_1 and c_2 are such that $[0, 1]$ is in the domain. This function is convex in its domain and resembles the one illustrated in Figure 2. In this example we consider two cases of $m = 10$ and $m = 100$, for each we generate 20 instances of (9) with random b and diagonal D of compatible dimension, and solve the optimization problem (1) by methods (3) and (7), with ρ being fixed to 0.5.

Figure 3 shows the optimal value obtained from the rounding-off method of (3) versus the direct search in (7) for $m = 10$ and 100. The x-axis corresponds to the round-off method of (3) and the y-axis is for the direct search of (7). The blue dots indicate when direct search was faster and the red dots indicate when the rounding-off was faster. Each dot below the $y = x$ solid line means that the direct search has obtained a lesser value, as desired. This means that the direct search method of (7) has shown better performance in the 20 considered samples when $m = 100$, and mostly when $m = 10$.

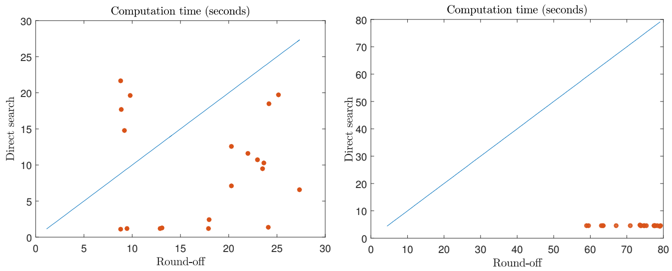


(a) For $m = 10$ (b) For $m = 100$

Fig. 3: Comparison of the optimal values

Next, we plot the computation time required to get to these values in Figure 4. Similar to the previous figure, every point below the $y = x$ line indicates that the direct search has taken less time, as desired. Iteration counts that each of the methods take to get to these points are also illustrated in Figure 5.

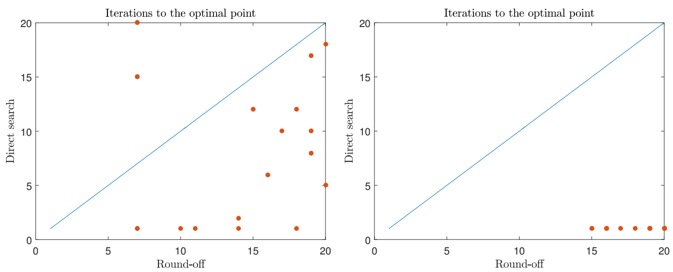
Next, we inspect how these two methods compare to the exact solution. When $m = 10$, the problem is small enough that we can find the exact solution by exhaustive search



(a) For $m = 10$

(b) For $m = 100$

Fig. 4: Comparison of the computational time (in seconds)



(a) For $m = 10$

(b) For $m = 100$

Fig. 5: Iterations to the optimal point

over 2^{10} instances of (1) with fixed z in each instance. Name the binary solution that corresponds to the exact exhaustive search by $z^{(\mathbf{ex})}$, the one that corresponds to the rounding method of (3) by $z^{(\mathbf{round})}$, and the one that corresponds to the direct search method of (7) by $z^{(\mathbf{direct})}$. How much these binary values differ is illustrated in Figure 6, where the x-axis denotes the sample index, the blue dots show how many elements $z^{(\mathbf{ex})}$ and $z^{(\mathbf{direct})}$ are far apart ($\|z^{(\mathbf{direct})} - z^{(\mathbf{ex})}\|_1$), and the red dots show the same for $z^{(\mathbf{round})}$, i.e., $\|z^{(\mathbf{round})} - z^{(\mathbf{ex})}\|_1$. As illustrated in Figure 6, method of (7) has recovered closer discrete variables to the exact solution in most cases, although it might happen that in a few cases rounding off would be better (as in sample 2).

In the next example we will vary the problem size and inspect how the two methods compare.

Example 9: We consider the optimization problem (1) again, and take $f(\cdot)$ and $g(\cdot, \cdot)$ as (10) and (9). In this example, we will vary m from 5 to 100. For each m, we will generate 20 instances of (9) with random b and diagonal D of compatible dimension, and then plot the optimal value, computational time, and the iterations to the optimal point.

Figure 7 plots the the difference of the optimal value obtained by rounding-off method of (3) from the direct search (7). The minimum of this difference (purple line) is almost always positive, except for 39 times out of all 1920 simulations (2%). This indicates that the method of (7) has mostly performed better for the considered functions, which satisfy Assumption 3. The black line denotes the average, the solid green denotes the median, whereas the dashed greens denote the 5% and 95% percentiles for this difference.

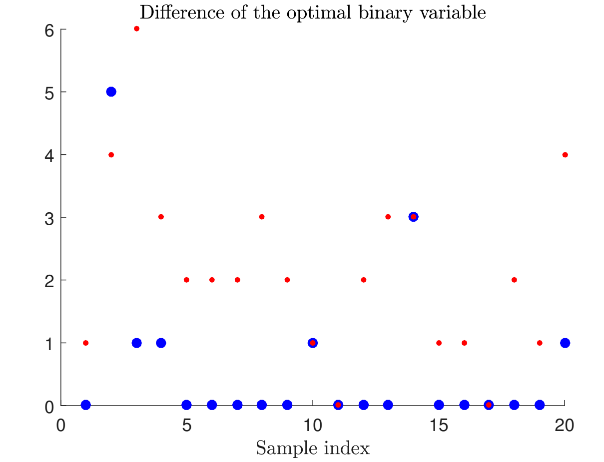


Fig. 6: Comparison to the exact solution when $m = 10$

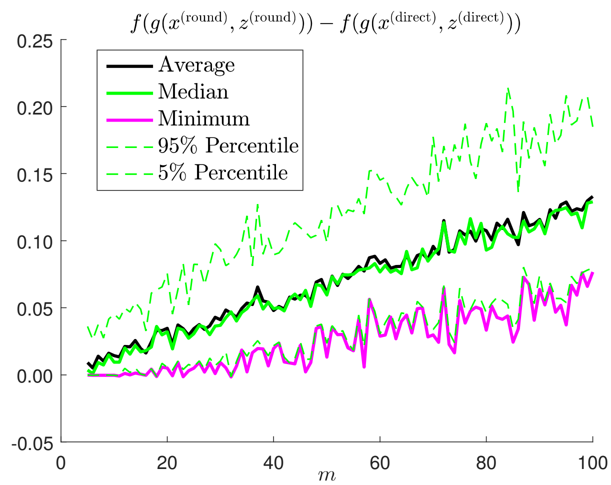


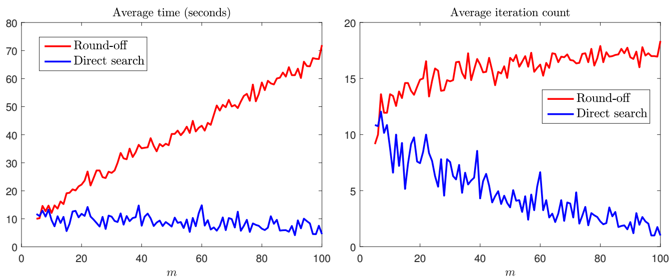
Fig. 7: Various statistics for the difference of the optimal values of the two methods

Figure 8a compares the time required for each of these methods, and Figure 8b illustrates the iterations required to reach the optimal point. As shown in these figures, the direct search exhibits better performance on average. Also, the number of iterations required to get to a local solution decreases for the direct search as the problem size gets bigger. This could be the case as the effect of the single continuous variable x decreases as the dimension of the problem increases, and hence the initial iterations that directly solve for the binary variables would be more crucial as m increases.

Finally we compare the exact solution to the these two methods. This was only an option when the problem size was small enough ($m < 15$). We see that, as illustrated in Figure 9, direct search solutions are closer to the exact value.

V. Conclusion

We have considered mixed integer programs where the problem is convex except for the integer constraint. We discussed a new heuristic that will use an auxiliary equality constraint to capture composite part of the objective. We



(a) Time to the optimal point (b) Required iterations

Fig. 8: Time and iterations required to obtain a local solution

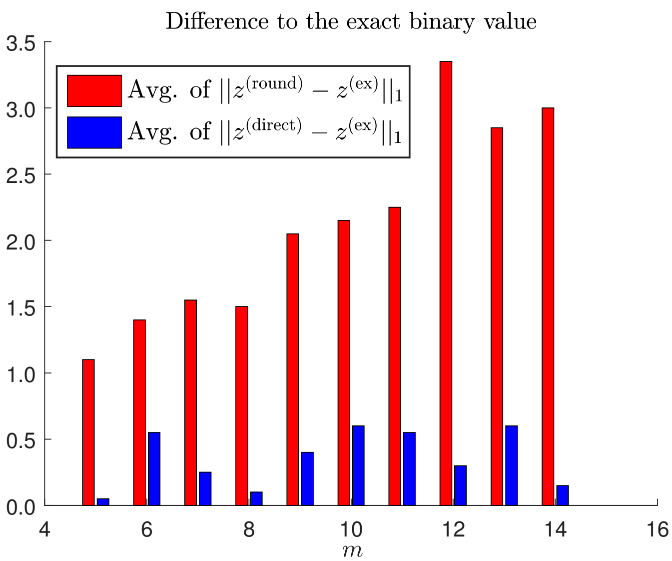


Fig. 9: Comparison to the exact solution for $5 \leq m \leq 14$

have then used ADMM to derive a new algorithm based on this formulation that will be better suited for asymmetrical objectives, and have shown that we can find the best discrete variable corresponding to the replaced part in linear time when that part is separable in the discrete variable. We have investigated this method numerically by comparing it to a relax-and-round algorithm, and have seen improvements in terms of the optimal value, optimal discrete point, and the run-time and have seen improvements in

REFERENCES

[1] S. Boyd and L. Vandenberghe, “Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization,” in *Communications, Computation, Control, and Signal Processing*. Springer, 1997, pp. 279–287.

[2] F. Alizadeh, “Interior point methods in semidefinite programming with applications to combinatorial optimization,” *SIAM Journal on Optimization*, vol. 5, no. 1, pp. 13–51, 1995.

[3] L. Lovász and A. Schrijver, “Cones of matrices and set-functions and 0-1 optimization,” *SIAM Journal on Optimization*, vol. 1, no. 2, pp. 166–190, 1991.

[4] S. Burer and D. Vandenbussche, “Solving lift-and-project relaxations of binary integer programs,” *SIAM Journal on Optimization*, vol. 16, no. 3, pp. 726–750, 2006.

[5] J. B. Lasserre, “Global optimization with polynomials and the problem of moments,” *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 796–817, 2001.

[6] ——, “Semidefinite programming vs. LP relaxations for polynomial programming,” *Mathematics of operations research*, vol. 27, no. 2, pp. 347–360, 2002.

[7] H. D. Sherali and W. P. Adams, “A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems,” *SIAM Journal on Discrete Mathematics*, vol. 3, no. 3, pp. 411–430, 1990.

[8] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, “A simple effective heuristic for embedded mixed-integer quadratic programming,” in *Proc. American Control Conference*, 2016, pp. 5619–5625.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[10] Y. Wang, W. Yin, and J. Zeng, “Global convergence of ADMM in nonconvex nonsmooth optimization,” *preprint arXiv:1511.06324*, 2016.

[11] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, “A general system for heuristic solution of convex problems over nonconvex sets,” *arXiv preprint:1601.07277*, 2016.

[12] A. Yadav, R. Ranjan, U. Mahbub, and M. Rotkowitz, “New methods for handling binary constraints,” in *Proceedings of the 54th Annual Allerton Conference on Communication Control and Computing*, 2016, pp. 1074–1080.

[13] A. Vaz and E. Davison, “A measure for the decentralized assignability of eigenvalues,” *Systems and Control Letters*, vol. 10, no. 3, pp. 191 – 199, 1988.

[14] A. Alavian and M. C. Rotkowitz, “An optimization-based approach to decentralized assignability,” in *Proc. American Control Conference*, 2016, pp. 5199–5204.