Improving the Approximated Projected Perspective Reformulation by Dual Information

Antonio Frangioni

Dipartimento di Informatica, Università di Pisa, frangio@di.unipi.it

Fabio Furini

LAMSADE, Université Paris-Dauphine, fabio.furini@dauphine.fr

Claudio Gentile

Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti", gentile@iasi.cnr.it

Abstract

We propose an improvement of the Approximated Projected Perspective Reformulation (AP^{2}R) for dealing with constraints linking the binary variables. The new approach solves the Perspective Reformulation (PR) once, and then use the corresponding dual information to reformulate the problem prior to applying AP^{2}R, thereby combining the root bound quality of the PR with the reduced relaxation computing time of AP^{2}R. Computational results for the cardinality-constrained Mean-Variance portfolio optimization problem show that the new approach is competitive with state-of-the-art ones.

Keywords: Mixed-Integer Non-Linear Problems, Semi-continuous Variables, Perspective Reformulation, Projection, Lagrangian Relaxation, Portfolio Optimization

1. Introduction

We study solution techniques for convex separable Mixed-Integer Non-Linear Programs (MINLP) with n semi-continuous variables $x_i \in \mathbb{R}$ for $i \in N = \{1, \dots, n\}$ which either assume the value 0 or lie in the interval $\mathcal{X}_i = [\underline{x}_i, \bar{x}_i]$ $(-\infty < \underline{x}_i < \bar{x}_i < \infty)$. This can be expressed, introducing $y_i \in \{0, 1\}$ for $i \in N$, as

$$
\begin{aligned}
(\mathrm{P}) \qquad & \min h(z) + \textstyle\sum_{i \in N} f_i(x_i) + c_i y_i && && (1)\\
& Ax + By + Cz = b && && (2)\\
& (x, z) \in \mathcal{O} && && (3)\\
& \underline{x}_i y_i \le x_i \le \bar{x}_i y_i \quad , \quad y_i \in [0,1]^n \quad , \quad x_i \in \mathbb{R}^n && i \in N && (4)\\
& y_i \in \mathbb{Z} && i \in N. && (5)
\end{aligned}
$$

We assume the functions f_i to be closed convex, one time continuously differentiable and finite in the interval $(\underline{x}_i, \bar{x}_i)$; w.l.o.g. we also assume $f_i(0) = 0$. In (P) we single out the *linking constraints* (2) that contain all the relationships linking the y_i variables among them and with the other variables of the problem, except those (4) that "define" the semi-continuous nature of the x_i. The reformulation technique developed in [7] require (2) to be empty; the extension developed in [1] allows to overcome this limitation, but potentially at the cost of a worse bound quality. The aim of this paper is to deal with constraint (2) in a cost-effective way. For our approach to work, (2) must have a compatible structure with that of (1); we initially assume equality constraints, with extensions discussed in §3. Because our approach hinges on availability of dual information for the continuous relaxation, we assume that the function $h(\cdot)$ in the "other variables z" and the "other constraints (3)" are convex, i.e., (P) is a convex MINLP. Actually, in many applications everything but (1) is linear. It will be sometimes expedient to refer to (3)–(4) as "$(x, y, z) \in \mathcal{P}$", and to $\underline{\mathcal{P}}$ as the set obtained by $\mathcal{P}$ relaxing integrality constraints on z and x, if any.

Often, the most pressing issue in solving (P) is to derive tight lower bounds on its optimal value ν(P), which is typically done by solving its (convex) continuous relaxation ($\underline{\text{P}}$) (we denote by ν(X) and ($\underline{\text{X}}$), respectively, the optimal value and the continuous relaxation of any problem (X)). However, often $\nu(\underline{\text{P}}) \ll \nu$(P), making the solution approaches inefficient. The presence of semi-continuous variables has been exploited to propose *reformulations* (P′) of (P) such that $\nu(\text{P}) = \nu(\text{P}') \geq \nu(\underline{\text{P}'}) \gg \nu(\underline{\text{P}})$. This starts from considering (1) as $h(z) + \sum_{i \in N} f_i(x_i, y_i)$, where $f_i(x_i, y_i) = f_i(x_i) + c_i$ if $y_i = 1$ and $\underline{x}_i \leq x_i \leq \bar{x}_i$, $f_i(0, 0) = 0$, and $f_i(x_i, y_i) = \infty$ otherwise. The *convex envelope* of $f_i(x_i, y_i)$ is known [4] to be $\tilde{f}_i(x_i, y_i) = y_i f_i(x_i/y_i) + c_i y_i$—using the *perspective function* of f_i—which yields the *Perspective Reformulation* of (P)

$$\text{(PR)} \qquad \min \left\{ \, h(z) + \textstyle\sum_{i \in N} \tilde{f}_i(x_i, y_i) \; : \; (2) \; , \; (x, y, z) \in \mathcal{P} \; , \; (5) \, \right\} \; .$$

As f_i is convex, $\tilde{f}_i$ is convex for $y_i \geq 0$; since $x_i = 0$ if $y_i = 0$, $\tilde{f}_i$ can be extended by continuity assuming $0 f_i(0/0) = 0$. Hence, (PR) is a convex MINLP if (P) is. Its continuous relaxation ($\underline{\text{PR}}$)—the *Perspective Relaxation* of (P)—usually has $\nu(\underline{\text{PR}}) \gg \nu(\underline{\text{P}})$, making (PR) a more convenient formulation [8, 9]. If f_i is SOCP-representable then so is $\tilde{f}_i$, hence the PR of a Mixed-Integer Second-Order Cone Program (MI-SOCP) is still a MI-SOCP. Thus, ($\underline{\text{PR}}$) is not necessarily more complex to solve—and, sometimes, even less so [2]—than ($\underline{\text{P}}$). Alternatively, one can consider a Semi-Infinite MINLP reformulation of (PR) where *Perspective Cuts* [4]—linear outer approximations of the epigraph of $\tilde{f}_i$—are dynamically added. This is often the best approach [6], in particular for "general" (P) where no other structure is available. It is appropriate to remark that the (PR) approach also applies if the x_i are *vectors* such that $y_i = 0 \Longrightarrow x_i = 0$ and $y_i = 1 \Longrightarrow x_i \in \mathcal{X}_i$, with $\mathcal{X}_i$ a polytope; yet, here, as in [1, 7], each x_i must be a single variable.

While ($\underline{\text{PR}}$) provides a better bound, it is also usually more time consuming to solve than ($\underline{\text{P}}$) because $\tilde{f}_i$ is "more complex" than f_i. This trade-off is nontrivial, in particular if f_i is "simple". For instance, if f_i is quadratic and everything else is linear, (P) is a Mixed-Integer Quadratic Program (MIQP) whereas (PR) is a MI-SOCP; hence, ($\underline{\text{P}}$)—a QP—can be significantly cheaper to solve than ($\underline{\text{PR}}$)—a SOCP. The *Projected PR* (P^{2}R) idea underpinning the approach studied here was indeed proposed in [7] for the quadratic case, and $\underline{x}_i \geq 0$. It was then extended in [1] to a more general class of functions, and allowing $\underline{x}_i < 0$. However, $\underline{x}_i < 0 < \bar{x}_i$ renders some of the arguments significantly more complex, hence for the sake of simplicity we will only present here the case where $\underline{x}_i \geq 0$; it will be plain to see that the arguments immediately extend to the more general one. The P^{2}R idea is to analyze $\tilde{f}_i$ *as a function of* x_i *only*, i.e., projecting away y_i: under appropriate assumptions, *and if there are no linking constraints* (2), this turns out to be a piecewise-convex functions with a "small" number of pieces, that can be characterized by just looking at the data of (P) (cf. (7)). Hence, ($\underline{\text{PR}}$) can be reformulated in terms of piecewise-convex objective functions, which makes it easier to solve, especially when $\mathcal{O}$ has some valuable structure (e.g., flow or knapsack) [7]. However, in several applications (2) are indeed present [4, 8, 1, 3, 10]. Furthermore, since the binary variables y_i are removed from the formulation, branching has to be done "indirectly", which rules out using off-the-shelf solvers. To overcome these two limitations, in [1] the *Approximated* P^{2}R (AP^{2}R) reformulation has been proposed whereby the y_i, after having been eliminated, are re-introduced in the formulation in order to encode the piecewise nature of $\tilde{f}_i$. This is possible even if (2) are present, and it has the advantage that (AP^{2}R) is still a MIQP if (P) is. However, $\nu(\underline{\text{AP}^2\text{R}}) < \nu(\underline{\text{PR}})$ may, and does, happen when linking constraints (2) are present, whence the "Approximate" moniker. This is still advantageous in some cases, but it may happen that the weaker bounds outweigh the faster solution time, making the approach not competitive with more straightforward implementations of the PR [1].

The aim of this paper is to improve the AP^{2}R by presenting a simple and effective way to ensure that $\nu(\underline{\text{AP}^2\text{R}}) = \nu(\underline{\text{PR}})$ even if (2) are present, while keeping the shape of the formulation—and therefore, hopefully, the cost of ($\underline{\text{AP}^2\text{R}}$)—exactly the same. Since bound equivalence only holds at the root node of the B&C it is not obvious that the approach, despite the quicker solution times of ($\underline{\text{AP}^2\text{R}}$), is competitive. However, this is shown to be true in at least one relevant application, the Mean-Variance problem (with min buy-in and cardinality constraints) in portfolio optimization.

2. A quick overview of AP^{2}R

We now quickly summarize the analysis in [1], albeit limited to the case $\underline{x}_i \geq 0$, in order to prepare the ground for the new extension. We focus on the basic problem corresponding to *one* pair (x_i, y_i)

$$(\mathrm{P}_i) \qquad \min\left\{\, f_i(x_i) + c_i y_i \;:\; \underline{x}_i y_i \leq x_i \leq \bar{x}_i y_i \;,\; y_i \in \{0,1\} \,\right\} \;.$$

The analysis hinges on considering the $(\underline{\mathrm{PR}})$ of (P_i) rewritten as

$$(\underline{\mathrm{PR}}_i) \qquad \min\left\{\, p_i(x_i) = \min\left\{\, \tilde{f}_i(x_i, y_i) \;:\; \underline{x}_i y_i \leq x_i \leq \bar{x}_i y_i \;,\; y_i \in [0,1] \,\right\} \;:\; x_i \in [0, \bar{x}_i] \,\right\} \;,$$

i.e., first minimizing $\tilde{f}_i(x_i, y_i)$ with respect to y_i, and then minimizing the resulting function $p_i(x_i)$ with respect to x_i. The function $p_i(x_i)$ is convex, and can be characterized by studying the optimal solution $y_i^*(x_i)$ of the inner problem in $(\underline{\mathrm{PR}}_i)$. Differentiability of f_i implies that $y_i^*(x_i)$ is strictly related to the solutions (if any) of the first-order optimality conditions

$$c_i + f_i(x_i/y_i) - f_i'(x_i/y_i)x_i/y_i = 0 \;. \tag{6}$$

The approach of [1] requires (6) to only have (at most) one solution, whose dependency on y_i is "easy":

Assumption 1. *(6) has at most one solution for $x_i \geq 0$, which has the form $\tilde{y}_i(x_i) = g_i x_i$ where $g_i \geq 0$ is a constant that can be determined by the data of the problem*

For instance, for the quadratic case $f_i(x_i) = a_i x_i^2 + b_i x_i$ the first-order optimality condition (6) is $c_i - a_i x_i^2/y_i^2 = 0$, whence $\tilde{y}_i(x_i) = |x_i|\sqrt{a_i/c_i}$ if $c_i > 0$, and there is no solution otherwise. Assumption 1 is satisfied by a surprisingly large set of functions, and a more general version can be stated for the case $\underline{x}_i < 0$ [1]. If (6) has a solution then $y_i^*(x_i)$ can be found by projecting $\tilde{y}_i(x_i)$ over the interval $[\, x_i/\bar{x}_i \,,\, \min\{\, 1 \,,\, x_i/\underline{x}_i \,\}\,]$; if no solution exists then $y_i^*(x_i)$ is in one of the two extremes. In all cases one can then write $p_i(x_i) = \tilde{f}_i(x_i, y_i^*(x_i))$. All this gives that there exists some $\underline{x}_i \leq \check{x}_i \leq \bar{x}_i$ such that

$$p_i(x_i) = \left\{\, (\, f_i(\check{x}_i)/\check{x}_i + c_i/\check{x}_i \,)x_i \quad \text{if } 0 \leq x_i \leq \check{x}_i \quad , \quad f_i(x_i) + c_i \quad \text{if } \check{x}_i \leq x_i \leq \bar{x}_i \,\right\} \;. \tag{7}$$

Thus, $p_i(x_i)$ is piecewise-convex with at most two pieces (although these become four if $\underline{x}_i < 0$), one of which is linear and the other is the original objective function. The crucial breakpoint $\check{x}_i$ can be determined a-priori: in particular, $\check{x}_i = 1/g_i$ if (6) has a solution and $1/g_i \in \mathcal{X}_i$, and $\check{x}_i \in \{\, \underline{x}_i \,,\, \bar{x}_i \,\}$ otherwise [1]. For a numerical illustration, the quadratic case

$$(\mathrm{P}_1) \qquad \min\left\{\, 2x_1^2 + 8y_1 \;:\; y_1 \leq x_1 \leq 10y_1 \quad,\quad y_1 \in \{0,1\} \,\right\}$$

has $\tilde{y}_1(x_1) = x_1\sqrt{a_1/c_1} = x_1/2$, thus $g_1 = 1/2$, and therefore $1/g_1 = 2 \in \mathcal{X}_1 = [1, 10]$: hence,

$$p_1(x_1) = \left\{\, 8x_1 \quad \text{if } 0 \leq x_1 \leq 2 \quad , \quad 2x_1^2 + 8 \quad \text{if } 2 \leq x_1 \leq 10 \,\right\} \;. \tag{8}$$

Writing (7) as the objective function is typically done with the "variable splitting" approach [7], whereby two new variables $0 \leq x_i' \leq \check{x}_i$ and $0 \leq x_i'' \leq \bar{x}_i - \check{x}_i$ are introduced such that $x_i = x_i' + x_i''$ (although a different form is sometimes preferable [3]): x_i' gets the linear cost, while x_i'' has cost $f_i(x_i'')$. This yields the Projected Perspective Reformulation (P^{2}R), having the same form as (P)—a MIQP if (P) was one—with at most (and, often, less than) twice as many variables. The corresponding $(\underline{\mathrm{P^2R}})$ might be more efficient to solve, especially if (P) has some structure that allows application of specialized approaches [7]. However, removing the y_i variables from the formulation prevents from using off-the-shelf software to solve (P^{2}R). This is why in [1] it was proposed to "lift back" (7) in the original (x_i, y_i) space by defining the problem

$$(\mathrm{LP}_i(x_i)) \quad \min\left\{\, y_i p_i(\check{x}_i) + f_i(x_i'' + \check{x}_i) + c_i - p_i(\check{x}_i) : (\underline{x}_i - \check{x}_i)y_i \leq x_i'' \leq (\bar{x}_i - \check{x}_i)y_i \;,\; x_i = \check{x}_i y_i + x_i'' \;,\; y_i \in [0,1] \,\right\} .$$

It can be proven [1] that $\nu(\underline{\mathrm{LP}}_i(x_i)) = p_i(x_i)$ for each feasible x_i; therefore,

$$(\mathrm{AP^2R}_i) \quad \min\left\{\, y_i p_i(\check{x}_i) + f_i(x_i'' + \check{x}_i) + c_i - p_i(\check{x}_i) \;:\; (\underline{x}_i - \check{x}_i)y_i \leq x_i'' \leq (\bar{x}_i - \check{x}_i)y_i \;,\; x_i = \check{x}_i y_i + x_i'' \;,\; y_i \in \{0,1\} \,\right\}$$

is a *reformulation* of (P_i) and $\nu(\underline{\mathrm{AP^2R}_i}) = \nu(\underline{\mathrm{PR}}_i)$, which implies that, typically, $\nu(\underline{\mathrm{AP^2R}_i}) \gg \nu(\underline{\mathrm{P}}_i)$. For illustration, consider (P_1): plugging (8) into $(\mathrm{AP^2R}_i)$ yields

$$(\mathrm{AP^2R_1})\quad \min\ \{\ 2(x_1'')^2 + 8x_1'' + 16y_1\ :\ -y_1 \le x_1'' \le 8y_1\ \ ,\ \ x_1 = 2y_1 + x_1''\ \ ,\ \ y_1 \in \{0,1\}\ \}\ .$$

It can be verified that $\nu(\underline{\mathrm{AP^2R_1}}) \ge \nu(\underline{\mathrm{P}}_1)$ for any fixed x_1. For instance, for $x_1 = 2$ the optimal solution to $(\underline{\mathrm{P}}_1)$ is $y_1 = 1/5$, yielding $\nu(\underline{\mathrm{P}}_1) = 9 + 3/5$, while the optimal solution to $(\underline{\mathrm{AP^2R_1}})$ is $(y_1, x_1'') = (1, 0)$, yielding $\nu(\underline{\mathrm{AP^2R_1}}) = 16$, i.e., the same estimate as $(\underline{\mathrm{PR}}_1)$ (for $x_1 = 2$). In fact,

$$\nu(\underline{\mathrm{PR}}_1) = \min\{\ 2x_1^2/y_1 + 8y_1\ :\ y_1 \le x_1 \le 10y_1\ \ ,\ \ y_1 \in [0,1]\ \} = 16$$

since $\min\{\ 8y_1 + 8/y_1\ :\ y_1 \in [1/5, 1]\ \}$ has optimal solution $y_1 = 1$.

We will denote by (AP^{2}R) the reformulation of (P) where (AP^{2}R$_i$) is separately applied to each x_i for $i \in N$. If there are no linking constraints (2), then obviously $\nu(\underline{\mathrm{AP^2R}}) = \nu(\underline{\mathrm{PR}})$. Otherwise the reformulation is still possible, but $(\underline{\mathrm{PR}}_i)$ is a *relaxation* of the true projection problem, which, besides on x_i, also depends on all the other variables that y_i is linked with. Hence, $\nu(\underline{\mathrm{AP^2R}}) < \nu(\underline{\mathrm{PR}})$ can happen, and it does in practice. For illustration consider the problem

$$(\mathrm{P}_{12})\quad \min\ 2x_1^2 + 2x_2^2 + 8y_1 + 8y_2 \tag{9}$$

$$y_1 \le x_1 \le 10y_1 \quad , \quad y_2 \le x_2 \le 10y_2 \tag{10}$$

$$y_1 \in \{0,1\}\ \ ,\ \ y_1 + y_2 = 1\ \ ,\ \ y_2 \in \{0,1\}\ \ ,\ \ x_1 + x_2 = 8 \tag{11}$$

obtained by "duplicating" (P$_1$) and adding the linking constraint $y_1 + y_2 = 1$. The optimal solution of $(\underline{\mathrm{P}}_{12})$ is $x_1 = x_2 = 4$, $y_1 = y_2 = 1/2$, yielding $\nu(\underline{\mathrm{P}}_{12}) = 72 \ll \nu(\mathrm{P}_{12}) = 136$, the latter obtained by setting $x_1 = 8$, $y_1 = 1$, $x_2 = y_2 = 0$ (or the symmetric solution). The $(\underline{\mathrm{PR}})$, obtained by replacing (9) with

$$\min\ 2x_1^2/y_1 + 2x_2^2/y_2 + 8y_1 + 8y_2$$

has the same optimal solution as $(\underline{\mathrm{P}}_{12})$: however, that same solution yields the much stronger (in fact, exact) bound of 136. Instead, for the (AP^{2}R)

$$(\mathrm{AP^2R_{12}})\quad \min\ 2(x_1'')^2 + 2(x_2'')^2 + 8x_1'' + 8x_2'' + 16y_1 + 16y_2$$
$$(11)\ \ ,\ \ -y_1 \le x_1'' \le 8y_1\ \ ,\ \ -y_2 \le x_2'' \le 8y_2\ \ ,\ \ x_1 = 2y_1 + x_1''\ \ ,\ \ x_2 = 2y_2 + x_2''$$

(cf. $(\underline{\mathrm{AP^2R_1}})$) the optimal solution of $(\underline{\mathrm{AP^2R_{12}}})$ is $x_1 = x_2 = 4$, $y_1 = y_2 = 1/2$, $x_1'' = x_2'' = 3$, yielding $\nu(\underline{\mathrm{P}}_{12}) = 72 \ll \nu(\underline{\mathrm{AP^2R_{12}}}) = 100 \ll \nu(\underline{\mathrm{PR}}_{12}) = 136$. In the next section we modify the (AP^{2}R) to increase its lower bound, avoiding the bound disadvantage with the (PR)—at least at the root node—while retaining the simpler (hence, cheaper) model shape.

3. Improving AP^{2}R using dual information

The idea is to reformulate (P) to include information about the linking constraints (2) in the objective function (1), so that it can be "processed" by the AP^{2}R. This hinges on the availability of dual information, and hence mainly concerns the continuous relaxations. The Lagrangian relaxation of $(\underline{\mathrm{P}})$ w.r.t. (2)

$$(\underline{\mathrm{P}}^{\lambda})\quad \min\ \{\ h(z) + \textstyle\sum_{i \in N} f_i(x_i) + c_i y_i + \lambda\big(\ Ax + By + Cz - b\ \big)\ :\ (x,y,z) \in \underline{\mathcal{P}}\ \}$$

has an objective function that is still separable in the x_i

$$h(z) + \lambda Cz + \textstyle\sum_{i \in N} \big(\ f_i(x_i) + \lambda A^i x_i + (c_i + \lambda B^i) y_i\ \big) - \lambda b\ \ . \tag{12}$$

Hence one can apply the PR to $(\underline{\mathrm{P}}^{\lambda})$, which—since the PR does not change linear functions—yields

$$(\underline{\mathrm{PR}}^{\lambda})\quad \phi(\lambda) = \min\ \{\ h(z) + \lambda Cz + \textstyle\sum_{i \in N} \big(\ y_i f_i(x_i/y_i) + \lambda A^i x_i + (c_i + \lambda B^i) y_i\ \big)\ :\ (x,y,z) \in \underline{\mathcal{P}}\ \} - \lambda b\ \ .$$

We will assume that the corresponding Lagrangian dual satisfies $\max_{\lambda}\{\ \phi(\lambda)\ \} = \nu(\underline{\mathrm{PR}})$, and in particular that an optimal dual solution λ^* is available that satisfies $\phi(\lambda^*) = \nu(\underline{\mathrm{PR}})$. This requires some conditions that are typically satisfied in most applications (e.g., all the constraints are linear or some appropriate Slater-type condition holds), so that we can expect that λ^* is generated by any approach used to solve $(\underline{\mathrm{PR}})$. Note that any inexact solution of the Lagrangian dual could be used as well, as long as it (significantly) improves the bound. Also, (12) clearly satisfies Assumption 1 if f_i does: (6) for $f_i(x_i) + \lambda A^i x_i$ only differs from (6) for f_i for the constant λA^i. Hence, one can form $(\underline{\mathrm{AP^2R}}^{\lambda})$, the AP^{2}R of $(\underline{\mathrm{PR}}^{\lambda})$, for any value of λ. Taking $\lambda = \lambda^*$ yields our main result:

Theorem 2. *Assume that f_i satisfies Assumption 1 and that λ^* is available; for the reformulation of* (P)

$$\text{(P+)} \quad \min\left\{\, h(z) + \lambda^* C z + \textstyle\sum_{i \in N} \big(f_i(x_i) + \lambda^* A^i x_i + (c_i + \lambda^* B^i) y_i \big) \;:\; (2) \;,\; (x, y, z) \in \mathcal{P} \;,\; (5) \,\right\} - \lambda^* b$$

denote by (PR+) *its* PR *and by* (AP^{2}R+) *its* AP^{2}R. *Then,* $\nu(\underline{\text{AP}^2\text{R+}}) = \nu(\underline{\text{PR}}) = \nu(\underline{\text{PR}}+)$.

Proof. First of all, (P+) is a *valid reformulation* of (P): since it contains (2), the Lagrangian term $\lambda^*\big(Ax + By + Cz - b \big)$ is always null at feasible points, hence both the feasible regions and the objective functions coincide. Also, as discussed for ($\underline{\text{AP}^2\text{R}}^\lambda$) the objective function of (P+) satisfies Assumption 1 since the f_i do, and therefore (AP^{2}R+) can be formed and $\nu(\text{P}) = \nu(\text{P+}) = \nu(\text{AP}^2\text{R+})$ holds. However, note that (1) and (12) are in general different: in fact, the equivalence between the two optimal values only holds when taking into account the constant $-\lambda^* b$. Now, clearly $\nu(\underline{\text{PR}}) = \nu(\underline{\text{PR}}^{\lambda^*}) = \nu(\underline{\text{PR}}+)$ due to the choice of λ^*. Note that ($\underline{\text{PR}}^{\lambda^*}$) is a relaxation of ($\underline{\text{PR}}$+), which in itself would give only $\nu(\underline{\text{PR}}^{\lambda^*}) \leq \nu(\underline{\text{PR}}+)$; however the relaxed constraints are precisely (2), those of which λ^* are the optimal multipliers, and therefore equality must hold. Now, since ($\underline{\text{PR}}^\lambda$) has no linking constraints (2), $\nu(\underline{\text{AP}^2\text{R}}^\lambda) = \nu(\underline{\text{PR}}^\lambda)$ for each λ, and therefore in particular for $\lambda = \lambda^*$. Hence, $\nu(\underline{\text{PR}}) = \nu(\underline{\text{PR}}^{\lambda^*}) = \nu(\underline{\text{AP}^2\text{R}}^{\lambda^*}) \leq \nu(\underline{\text{AP}^2\text{R+}})$, because, again, ($\underline{\text{AP}^2\text{R}}^{\lambda^*}$) is the relaxation of ($\underline{\text{AP}^2\text{R+}}$) w.r.t. (2). But on the other hand $\nu(\underline{\text{PR}}+) \geq \nu(\underline{\text{AP}^2\text{R+}})$, as the bound of AP^{2}R is always at most as good as that the PR, and therefore the thesis is proven. ■

To illustrate Theorem 2, consider again (P$_{12}$). The optimal dual multiplier of the linking constraint $y_1 + y_2 = 1$ in ($\underline{\text{PR}}_{12}$) can be seen to be $\lambda^* = 120$. Hence,

$$(\text{P}_{12}+) \quad \min\left\{\, 2x_1^2 + 2x_2^2 + 128y_1 + 128y_2 \;:\; (10) \;\;,\;\; (11) \,\right\} - 120 \;\; .$$

In its AP^{2}R, $c_i = 128$ gives $g_i = \sqrt{a_i/c_i} = \sqrt{2/128} = 1/8$, i.e., $\check{x}_i = 8$ for $i = 1, 2$. Hence,

$$p_i(x_i) = y_i p(\check{x}_i) + f_i(x_i'' + \check{x}_i) + c_i - p_i(\check{x}_i) = 256y_i + 2(x_i'')^2 + 32x_i'' \implies$$

$$(\text{AP}^2\text{R}_{12}+) \;\; \min\, 2(x_1'')^2 + 2(x_2'')^2 + 32x_1'' + 32x_2'' + 256y_1 + 256y_2$$
$$(11) \;\;,\;\; -7y_1 \leq x_1'' \leq 2y_1 \;\;,\;\; -7y_2 \leq x_2'' \leq 2y_2 \;\;,\;\; x_1 = 8y_1 + x_1'' \;\;,\;\; x_2 = 8y_2 + x_2'' \;\; .$$

The optimal solution of ($\underline{\text{AP}^2\text{R}_{12}+}$) is $x_1 = x_2 = 4$, $y_1 = y_2 = 1/2$, $x_1'' = x_2'' = 0$, yielding an objective function value of 256: counting the constant $-\lambda^* b = -120$, this finally gives $\nu(\underline{\text{AP}^2\text{R}_{12}+}) = 136$: (much) better than $\nu(\underline{\text{AP}^2\text{R}}_{12}) = 100$, and in fact precisely equal to $\nu(\underline{\text{PR}}_{12})$ as predicted.

We end this section by remarking that the assumption that (2) is an equality constraint is not crucial. Inequality linking constraints $Ax + By + Cz \leq b$ can be transformed into equalities by the addition of slack variables: $Ax + By + Cz + s = b$, $s \geq 0$. Note that this has to be done in (P), so that after the reformulation they will have cost $\lambda^* s$ in the objective function, i.e., they no longer will be slack variables. In principle the constraints could also be nonlinear in x and z (linearity in y can always be assumed without loss of generality), provided that $A(x) = \sum_{i \in N} A^i(x_i)$ with each $A^i(\cdot)$ convex, Assumption 1 holds for $f_i(x_i) + \lambda^* A^i(x_i)$, and $C(z)$ is convex. However, in order for (2) to be convex they necessarily had to be inequalities. This means that the optimal dual multipliers λ^* would be non-negative, retaining convexity of $\lambda^* A^i(x_i)$, but the constraints would have to be turned into nonlinear equalities, that can never be convex.

4. Computational results

In this section we report results of computational tests of the proposed approach for the Mean-Variance cardinality-constrained portfolio optimization problem on n risky assets

$$(\text{MV}) \qquad \min\left\{\, x^T Q x \;:\; \textstyle\sum_{i \in N} x_i = 1 \;\;,\;\; \sum_{i \in N} \mu_i x_i \geq \rho \;\;,\;\; \sum_{i \in N} y_i \leq k \;\;,\;\; (4) \;\;,\;\; (5) \,\right\} \;,$$

where μ is the vector of expected unitary returns, ρ is the prescribed total return, Q is the variance-covariance matrix, and $k \leq n$ is the maximum number of purchasable assets. Without the cardinality constraint ($k = n$), (MV) is well suited for AP^{2}R: the bound is the same as that of the PR, and the computation time per node is greatly reduced with respect to the Perspective Cut (P/C) technique. While AP^{2}R is competitive also for $k \ll n$, it becomes less so as the quality of the bound significantly deteriorates

[1]. Hence, (MV) is a promising application for AP^{2}R+. Since (MV) is a *non-separable* MIQP, a diagonal matrix D has to be determined such that $Q - D$ is positive semidefinite: the PR technique is applied to $\sum_{i \in N} D_{ii} x_i^2$, leaving the remaining part $x^T(Q - D)x$ untouched. Choosing D is nontrivial: one can use e.g., a "small" SDP as advocated in [5], or a "large" SDP as proposed in [10]. We denote these two by D_s and D_l. Although D_l provides a better root node bound, it is not necessarily the best choice throughout the enumeration tree: sometimes a convex combination between the two, denoted by D_c, works better [10].

For our tests we used the 90 randomly-generated instances, 30 for each value of $n \in \{200, 300, 400\}$, already employed in [1, 4, 5, 6, 10] to which the interested reader is referred for details. Here we only remark that "$^+$" instances are strongly diagonally dominant, "^{0}" ones are weakly diagonally dominant, and "$^-$" ones are not diagonally dominant; the less diagonally dominant, the harder an instance is. We have set $k = 10$, as in [1, 10]; this is a "tight" value, since the maximum number of assets that the model can choose, due to the lower limits $\underline{x}_i > 0$, without the cardinality constraint is ≈ 20 for all n. The (MV) instances and the diagonals used in the experiments are available at `http://www.di.unipi.it/optimize/Data/MV.html`.

The experiments have been performed on a computer with a 3.40 Ghz 8-core Intel Core i7-3770 processor and 16Gb RAM, running a 64 bits Linux operating system. All the codes were compiled with `g++` (version 4.8.4) using `-O3` as optimization option. We have tested AP^{2}R+ vs. AP^{2}R using `Cplex 12.7`, single-threaded, with all default parameters (save for one explicitly described below, and only for the tests with the D_l diagonal). All the formulations have been given in the natural form, i.e., as a MIQP with linear constraints and convex quadratic nonseparable objective function. We have obtained the ($\underline{\text{PR}}$) root node bound with the P/C technique, implemented through callbacks, which is typically the best choice when AP^{2}R is not available [5]; hence, for completeness we also report results for the full B&C using P/C. Since we are not interested in comparing the cost/effectiveness of the different diagonal choices, this having been done in [10], we don't report detailed SDP times beyond saying that the "small" SDP requires on average about 0.2, 0.7, and 1.6 seconds while the "large" SDP requires 9, 21 and 47 seconds, respectively for $n = 200$, 300 and 400 (with little variance for the same n). We also don't report comparisons with either not using PR techniques at all, or using SOCP formulations, because they are well-known to be from dramatically [4, 5] to significantly [6] less effective than the ones employed here.

The results are reported in Table 1, 2 and 3 for the three diagonals D_s, D_c, and D_l, respectively. In the tables we report the (average) total B&C time and root time when using P/C. For AP^{2}R and AP^{2}R+ we report the (average) total number of B&C nodes, total B&C time, root node time and root gap (in percentage). As predicted by Theorem 2 the root node gap of AP^{2}R+ and P/C was identical, which is why we do not report it for P/C. The total time of AP^{2}R+ already includes the P/C root time, since it is needed to compute λ^* prior to performing the reformulation, and therefore starting the AP^{2}R+ B&C.

	P/C		AP^{2}R				AP^{2}R+			
	time		nodes	time		root	nodes	time		root
	tot	root		tot	root	gap		tot	root	gap
200^+	1.17	0.19	205	0.50	0.12	0.77	118	0.55	0.13	0.51
200^0	1256.99	0.18	23354	27.91	0.12	3.14	9325	11.09	0.13	2.75
200^-	21941.53	0.20	170682	194.31	0.13	4.75	40193	43.02	0.12	4.17
300^+	4.60	0.49	1288	4.20	0.35	1.08	320	1.87	0.37	0.49
300^0	2299.75	0.49	75456	188.53	0.34	2.91	20381	44.85	0.36	2.34
300^-	68043.87	0.52	424166	1017.28	0.35	3.92	86100	196.04	0.38	3.57
400^+	4.98	1.10	990	6.07	0.79	0.85	184	2.91	0.77	0.41
400^0	37836.53	1.10	334730	1439.63	0.71	3.00	49841	200.71	0.72	2.34
400^-	175843.65	1.10	1784051	7279.23	0.72	4.53	328800	982.38	0.79	3.80

Table 1: Results with diagonal D_s

Tables 1 and 2 show that AP^{2}R+ is highly competitive with AP^{2}R, and *a fortiori* with P/C, reducing total time by up to an order of magnitude. While the exact ratio depends on n, the type of instance and

	P/C		AP^{2}R				AP^{2}R+			
	time		nodes	time		root	nodes	time		root
	tot	root		tot	root	gap		tot	root	gap
200$^+$	0.93	0.20	129	0.42	0.12	0.59	81	0.52	0.12	0.27
200^0	47.54	0.20	6318	8.07	0.12	1.91	1881	2.72	0.12	1.44
200$^-$	622.25	0.22	45267	49.80	0.13	3.01	6210	7.48	0.14	2.33
300$^+$	2.36	0.51	558	2.49	0.35	1.02	121	1.44	0.33	0.23
300^0	71.40	0.52	20698	52.60	0.35	1.86	3216	8.20	0.34	1.14
300$^-$	1649.26	0.54	122138	292.30	0.37	2.49	15475	37.30	0.38	2.00
400$^+$	3.17	1.12	524	4.30	0.79	0.88	53	2.50	0.77	0.21
400^0	613.01	1.13	94154	410.07	0.73	2.01	7134	25.02	0.76	1.19
400$^-$	8095.26	1.15	345940	1432.37	0.73	2.97	37705	117.63	0.85	2.00

Table 2: Results with diagonal D_c

the diagonal, the trend is clear: the improvement is due to the much reduced number of nodes, itself a consequence of the much improved bound, while the computing time per node remains the same. The results are somewhat different for D_l, which therefore requires separate discussion. In Table 3, although the nodes count does decrease, the running time does not nearly as much, and can actually increase. While the average time per node of AP^{2}R and AP^{2}R+ is almost identical for D_s and D_c, for D_l that of AP^{2}R+ is roughly an order of magnitude larger. Investigating the issue showed that D_l causes `Cplex` to change the (automatic) selection of the relaxation algorithm at the nodes, settling to one that turns out to be much less efficient. Tests determined that setting `CPX_PARAM_SUBALG = 5`, i.e., using the "sifting" approach, restored an average time per node similar to that of AP^{2}R. This is why Table 3 reports, besides AP^{2}R+, also "AP^{2}R++" which is just obtained changing that parameter. To save on space, root times for AP^{2}R$*$ are not reported; they are, however, similar, even between AP^{2}R+ and AP^{2}R++, since it was subproblem time at the inner nodes that made a difference. The table shows that AP^{2}R++ is competitive w.r.t. AP^{2}R, albeit with a somewhat smallest ratio. This is due to another frankly unfathomable—but not unheard-of with today's complex MIP solvers—phenomenon: just by changing the relaxation solver, the number of nodes significantly increases w.r.t. AP^{2}R+. Yet, on the largest and hardest instances AP^{2}R++ enumerates a third of the nodes of AP^{2}R, with the corresponding time advantage. Remarkably, for the D_l diagonal alone, P/C can sometimes be competitive with AP^{2}R+.

	P/C		AP^{2}R			AP^{2}R+		AP^{2}R++		
	time		nodes	time	root	nodes	time	nodes	time	root
	tot	root		tot	gap		tot		tot	gap
200$^+$	1.41	0.25	131	0.78	0.93	32	1.28	366	1.48	0.09
200^0	4.46	0.36	887	2.56	1.52	147	3.76	859	3.17	0.38
200$^-$	14.99	0.40	5893	9.81	2.12	480	5.55	2782	6.50	0.77
300$^+$	3.18	0.78	784	4.82	1.89	17	2.92	174	2.23	0.03
300^0	8.81	0.96	2211	10.12	2.00	178	15.04	1473	8.67	0.18
300$^-$	26.86	0.99	13032	39.86	2.04	1307	55.35	9593	36.68	0.64
400$^+$	5.33	1.59	906	10.35	1.94	11	4.96	182	4.44	0.06
400^0	50.60	1.91	9195	63.93	2.22	366	64.75	3098	26.47	0.26
400$^-$	52.91	2.15	18096	132.09	2.65	1867	272.79	7928	57.85	0.48

Table 3: Results with diagonal D_l, default and with option `CPX_PARAM_SUBALG = 5`

5. Conclusions

The main advantage of the proposed AP^{2}R+ technique is its simplicity: just solving (PR)—possibly even approximately with a dual approach—produces the dual solution λ^* which can be used to first construct (P+) and then its (AP^{2}R). Yet, this improves many-fold the performances over plain AP^{2}R, and even more so over P/C. Notably, AP^{2}R+ is quite general and applies to a much larger class than MIQP. It may be worth contrasting 175843 seconds (P/C in Table 1) with 58 seconds (AP^{2}R++ in Table 3) for 400^- instances: this is well over over three orders of magnitude difference for solving the same instances with the same underlying solver, and the gap with the standard MIQP formulation would be even more humungous. This nicely illustrates the power of reformulation techniques like AP^{2}R+.

Acknowledgements

The first and third authors acknowledge the contribution of the Italian Ministry for University and Research under the PRIN 2012 Project 2012JXB3YF "Mixed-Integer Nonlinear Optimization: Approaches and Applications". All the authors acknowledge networking support by the COST Action TD1207. We are grateful to the anonymous Referee and the Associate Editor of the Journal for constructive comments that helped us in significantly improve the manuscript.

References

[1] A. Frangioni, F. Furini, and C. Gentile. Approximated Perspective Relaxations: a Project&Lift Approach. *Computational Optimization and Applications*, 63(3):705–735, 2016.

[2] A. Frangioni, L. Galli, and M.G. Scutellà. Delay-Constrained Shortest Paths: Approximation Algorithms and Second-Order Cone Models. *Journal of Optimization Theory and Applications*, 164(3):1051–1077, 2015.

[3] A. Frangioni, L. Galli, and G. Stea. Delay-constrained routing problems: Accurate scheduling models and admission control. Technical report, Dipartimento di Informatica, Università di Pisa, 2015.

[4] A. Frangioni and C. Gentile. Perspective Cuts for 0-1 Mixed Integer Programs. *Mathematical Programming*, 106(2):225–236, 2006.

[5] A. Frangioni and C. Gentile. SDP Diagonalizations and Perspective Cuts for a Class of Nonseparable MIQP. *Operations Research Letters*, 35(2):181 – 185, 2007.

[6] A. Frangioni and C. Gentile. A Computational Comparison of Reformulations of the Perspective Relaxation: SOCP vs. Cutting Planes. *Operations Research Letters*, 37(3):206 – 210, 2009.

[7] A. Frangioni, C. Gentile, E. Grande, and A. Pacifici. Projected Perspective Reformulations with Applications in Design Problems. *Operations Research*, 59(5):1225–1232, 2011.

[8] A. Frangioni, C. Gentile, and F. Lacalandra. Tighter Approximated MILP Formulations for Unit Commitment Problems. *IEEE Transactions on Power Systems*, 24(1):105–113, 2009.

[9] O. Günlük and J. Linderoth. Perspective Relaxation of MINLPs with Indicator Variables. In A. Lodi, A. Panconesi, and G. Rinaldi, editors, *Proceedings 13^{th} IPCO*, volume 5035 of *Lecture Notes in Computer Science*, pages 1–16, 2008.

[10] X. Zheng, X. Sun, and D. Li. Improving the Performance of MIQP Solvers for Quadratic Programs with Cardinality and Minimum Threshold Constraints: A Semidefinite Program Approach. *INFORMS Journal on Computing*, 26(4):690–703, 2014.