Stronger and faster semidefinite programming bounds for the p-median Quadratic Facility Location Problem

Dilson Lucas Pereira
Universidade Federal de Lavras
Departamento de Computação Aplicada
Lavras, Brazil
dilson.pereira@ufla.br

Alexandre Salles da Cunha*
acunha@dcc.ufmg.br
Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil

Abstract

The p-Median Quadratic Facility Location Problem (pMQFLP) is a binary quadratic optimization problem that seeks to select p facilities from a set of candidate locations in order to serve a set of clients at minimum total cost. The objective function includes fixed facility installation costs, distribution costs for serving client demands, and interaction costs between facilities. In this paper, we introduce a stronger semidefinite programming (SDP) relaxation for the pMQFLP, together with a Lagrangian relaxation (LR) algorithm to accurately approximate the resulting SDP bounds. From a theoretical standpoint, our SDP relaxation dominates the SDP relaxation proposed in the literature. From a practical perspective, numerical experiments conducted on 600 benchmark pMQFLP instances show that our exact SDP bounds, computed using a convex optimization solver, are on average 1.54% stronger than SDP bounds from the literature, obtained in a similar manner. In addition, the proposed LR algorithm provides highly accurate approximations of these SDP bounds at a substantially reduced computational cost. On average, the LR bounds deviate by only 0.054% from the best attainable exact SDP bounds, and the maximum deviation never exceeds 0.096%. Overall, the LR approach requires approximately 16% of the CPU time needed to compute our exact SDP bounds. Despite the inexact nature of the Subgradient Method used to solve the Lagrangian Dual problem, the LR bounds outperformed the SDP bounds from the literature for all 600 instances tested, being on average 1.48% stronger.

Keywords

p-Median Quadratic Facility Location, Semidefinite Programming, Lagrangian Relaxation

1 Introduction

Given a set $N = \{1, \ldots, n\}$ of candidate facility locations and a set $M = \{1, \ldots, m\}$ of clients, the $p-$Median Facility Location Problem (pMFLP) seeks to determine the locations of p facilities so as to serve all client demands at minimum total cost. Two types of costs are considered: fixed opening costs $\mathbf{c} = (c_i)_{i \in N}$ associated with installing facilities, and service costs $A = (a_{ik})_{i \in N, k \in M}$ incurred when client k is served by a facility located at i.

*This research is funded by CNPq grant 305253/2024-7 and FAPEMIG grants APQ-02932-24 and RED-00119-21.

INOC '26, Liège (Belgium)

Let $\mathbf{x} \in \{0, 1\}^n$ be a vector of binary decision variables used to indicate if a facility is installed at location $i \in N$ ($x_i = 1$) or not ($x_i = 0$), and let $Y \in \{0, 1\}^{n \times m}$ be a binary matrix of variables used to indicate if the demand of client k is entirely served by a facility placed at i ($y_{ik} = 1$) or not ($y_{ik} = 0$). The pMLFP can be formulated as the Integer Program (IP) $\min\{\sum_{i \in N} c_i x_i + \sum_{i \in N} \sum_{k \in M} a_{ik} y_{ik} : (\mathbf{x}, Y) \in \mathcal{X}\}$, where $\mathcal{X}$ is the discrete feasible set defined by constraints (1)-(5).

$$\sum_{i \in N} x_i = p, \tag{1}$$

$$\sum_{i \in N} y_{ik} = 1, \qquad k \in M, \tag{2}$$

$$y_{ik} \leq x_i, \qquad i \in N, k \in M, \tag{3}$$

$$x_i \in \{0, 1\}, \qquad i \in N, \tag{4}$$

$$y_{ik} \in \{0, 1\}, \qquad i \in N, k \in M. \tag{5}$$

The problem investigated in this paper, the $p-$Median Quadratic Facility Location Problem (pMQFLP), is defined in a similar fashion, but extends the pMFLP by incorporating a quadratic term in the objective function, involving the facility location variables $\mathbf{x}$. More precisely, let $D = (d_{ij})_{i,j \in N}$ be a symmetric $n \times n$ matrix with $d_{ii} = 0$ for all $i \in N$, representing the cost of jointly selecting locations i and j. The pMQFLP can then be formulated as the following Binary Quadratic Problem

$$\min \left\{ \frac{1}{2} \sum_{i \in N} \sum_{j \in N} d_{ij} x_i x_j + \sum_{i \in N} c_i x_i + \sum_{i \in N} \sum_{k \in M} a_{ik} y_{ik} : (\mathbf{x}, Y) \in \mathcal{X} \right\}. \tag{6}$$

The pMQFLP was investigated by [1, 13, 16]. A distinctive feature of the problem is that the quadratic term $\frac{1}{2} \sum_{i,j \in N} d_{ij} x_i x_j$ does not depend on flows between the facilities. This characteristic makes the pMQFLP particularly suitable for modeling applications arising in electrical networks or cable routing problems [16]. In this respect, the pMQFLP contrasts with other location problems, such as the p-Hub Median Problem [3], in which the connection costs depend on the flows. We refer the reader to [16] for a comprehensive and up-to-date review of the literature on the pMQFLP, its applications, and related location problems.

Yang et al. [16] introduced exact solution approaches for the pMQFLP that exploit the fact that, once a partial solution $\widetilde{\mathbf{x}} \in \{0, 1\}^n$ satisfying $\sum_{i \in N} \widetilde{x}_i = p$ is fixed, the optimization over the assignment variables Y reduces to a Linear Program. Indeed, it suffices to assign every client $k \in M$ to a location $\widetilde{i}(k) \in \arg\min_{i \in N, \widetilde{x}_i = 1}\{a_{ik}\}$. Hence, the integrality on Y needs not to be explicitly enforced

10.48786/inoc.2026.03

and these variables can be projected out of the pMQFLP formulation, at the expense of enlarging the master program with exponentially many Benders cuts. Since the pMQFLP is uncapacitated, Benders feasibility cuts are not required, as it is always possible to identify a facility capable of serving each client. The approaches proposed in [16] share this common decomposition principle but differ in the specific linearization strategies that accompany the projection step.

SDP+BD, the best performing pMQFLP algorithm from [16], relies on a matrix of decision variables Z that relates to $\mathbf{x}$ according to the rank-one (and thus non-convex) constraint $Z = \mathbf{x}\mathbf{x}^T$, where each entry z_{ij} represents the product $x_i x_j$. The rank one constraint is then replaced by the linearized SDP constraint

$$Z' := \begin{pmatrix} 1 & \mathbf{x}^T \\ \mathbf{x} & Z \end{pmatrix} \in \mathcal{S}_+^{n+1}, \tag{7}$$

where $\mathcal{S}_+^{n+1}$ denotes the cone of real symmetric positive semidefinite matrices of order $n+1$. The other component of the method is the projection step itself. In SDP+BD, the integrality requirements on $\mathbf{x}$, together with the exponentially many Benders optimality cuts, are relaxed in order to construct an SDP based cutting plane algorithm that generates tight dual bounds for the pMQFLP.

SDP+BD is a Branch-and-bound (BB) algorithm that relies on solving SDPs to generate valid dual bounds and on adding optimality cuts at every node of the search tree. Assume that $(\mathbf{x}', \widetilde{Z}')$ is an optimal solution to a relaxed Benders master program. The p largest entries in the diagonal of $\widetilde{Z}'$ are used to devise a point $\widetilde{x}^* \in \{0,1\}^n$, with exactly p non-zero entries, that is used to separate Benders optimality cuts. Any violated cuts are added to a strengthened SDP relaxation, which is subsequently reoptimized. At each iteration, the optimal solution of the SDP provides a valid lower bound for the pMQFLP while, after separating optimality cuts, valid pMQFLP upper bounds are obtained, by Linear Programming duality. The process of alternating between the resolution of SDPs and generating Benders cuts continues until the improvement in the lower bound falls below a prescribed threshold. If, at termination, the solution $\widetilde{Z}'$ satisfies the integrality conditions and the point $\widetilde{\mathbf{x}}^*$ violates no optimality cuts, the algorithm terminates with an optimal solution. Otherwise, the lower bounding procedure is embedded within a BB framework to solve the pMQFLP to proven optimality.

Our contribution. Yang et al. [16] provided sound theoretical reasoning for the tightness of their SDP bounds, particularly when D is a matrix distance. In this paper, we introduce a stronger SDP relaxation for the pMQFLP and a Lagrangian relaxation (LR) algorithm to accurately approximate these bounds with limited computational effort.

From a theoretical standpoint, our SDP bounds are at least as strong as those proposed in [16], since we apply a partial first level Reformulation Linearization Technique (RLT) [14] to the feasible set $\mathcal{X}$, a strengthening step that is not considered in [16]. From a practical perspective, numerical experiments conducted on 600 benchmark instances from the literature show that our exact SDP bounds are significantly stronger. On average, when computed using an exact convex optimization solver, our SDP bounds are 1.54% stronger than those obtained from the SDP relaxation of [16], computed in a similar manner.

In addition, our LR algorithm provides highly accurate approximations of the best attainable bounds delivered by our SDP relaxation. More precisely, over the full set of 600 instances, the LR bounds are on average only 0.054% weaker than the corresponding exact SDP bounds, and the deviation never exceeds 0.096%. Moreover, to achieve this level of accuracy, the LR algorithm requires, on average, only about 16% of the CPU time needed by the convex optimization solver to compute the exact SDP bounds of our model.

Finally, the SDP bounds provided by our LR approach consistently dominate the exact SDP bounds provided by the relaxation of [16]. On average, our LR bounds are 1.48% stronger than those bounds. Furthermore, for none of the 600 tested instances, are the LR bounds less than 1.10% stronger than the corresponding bounds computed from the SDP relaxation of [16] using an exact convex solver.

The paper is organized as follows. Section 2 presents our SDP relaxation, the proposed LR strategy, and the associated Lagrangian dual problem (LDP). Section 3 describes the main ideas underlying the Subgradient Method (SM) used to approximate our SDP bounds. Our preliminary numerical experiments are reported in Section 4. Finally, Section 5 concludes the paper and outlines directions for future research.

2 SDP Relaxation and Lagrangian Relaxation

In this paper, matrices are indicated by capital letters while vectors are denoted in boldface. Given a matrix A, its transpose is A^T. Given two conformable matrices, say A and Y, $\langle A, Y\rangle = \sum_{i,k} a_{ik} y_{ik}$ defines the Frobenius inner product and $\|A\|_F = \sqrt{\langle A, A\rangle}$ is the Frobenius norm of A. Whenever we state $\Lambda \succeq 0$, we mean that Λ is positive semidefinite, without necessarily being symmetric. If, in addition to that, Λ is of order n and symmetric, we state $\Lambda \in \mathcal{S}_+^n$. Given any square matrix, say Z, diag(Z) gives the vector corresponding to the diagonal of Z. The inverse operator, Diag$(\mathbf{x})$, produces a diagonal matrix from a vector $\mathbf{x} \in \mathbb{R}^n$. $\mathbb{J}_{n\times m}$ denotes the $n \times m$ matrix of ones and $\mathbf{e}_n$ denotes the vector of n ones.

In order to devise a stronger SDP relaxation for the pMQFLP, we implement the following first steps:

(1) Apply a partial RLT [14] to the set $\mathcal{X}$, namely by multiplying constraint (1) and the bounds $\{0 \le x_j \le 1 : j \in N\}$ by every nonnegative factor x_i with $i \in N$.
(2) Linearize the quadratic terms in the resulting constraints and in the objective function, by means of a linearization matrix $Z = (z_{ij})_{i,j\in N}$, defined as $Z = \mathbf{x}\mathbf{x}^T$.
(3) Relax the rank one constraint $Z = \mathbf{x}\mathbf{x}^T$ to $Z \in \mathcal{S}_+^n$.
(4) Relax the integrality constraints on $\mathbf{x}$ and $\mathbf{y}$.

The resulting SDP relaxation is

$$lb_{sdp} = \min\left\{\frac{1}{2}\langle D, Z\rangle + \mathbf{c}^T\mathbf{x} + \langle A, Y\rangle : (\mathbf{x}, Y, Z) \in F_{sdp}\right\}, \tag{8}$$

where F_{sdp} is the convex set defined by (1)-(3) and

$$\sum_{j \in N} z_{ij} = p x_i, \qquad i \in N, \tag{9}$$
$$z_{ii} = x_i, \qquad i \in N, \tag{10}$$
$$z_{ij} \leq x_i, \qquad i, j \in N, \tag{11}$$
$$z_{ij} = z_{ji}, \qquad 1 \leq i < j \leq n, \tag{12}$$
$$Z \in \mathcal{S}_+^n, \tag{13}$$
$$x_i \in [0,1], \qquad i \in N, \tag{14}$$
$$y_{ik} \in [0,1], \qquad i \in N, k \in M, \tag{15}$$
$$z_{ij} \geq 0, \qquad i, j \in N. \tag{16}$$

The SDP above differs from that in [16] (before projecting out Y) in the following aspects. First, constraints (9) and (11), which result from the application of the first level partial RLT, are not present in [16]. Second, we do not impose positive semidefiniteness of the Schur complement (7). This choice is motivated by a result of [4], brought to our attention by [15], which shows that imposing constraint (7) does not improve bounds when $\text{diag}(Z)u = Z\mathbf{e}_n$ for some $u \in \mathbb{R}$. This condition holds in our setting due to (9)–(10), with $u = p$. Finally, note that the SDP relaxation above enforces the symmetry on Z twice, namely in (12) and in (13). The reason for this is discussed next.

We now apply LR to the SDP program (8). We relax constraints (3), (12) and (13) and dualize them in the objective function. To that aim, we attach Lagrangian multipliers $\{\phi_{ik} \geq 0 : i \in N, k \in M\}$, $\{\beta_{ij} : i, j \in N, i < j\}$ and $\Lambda \in \mathcal{S}_+^n$, respectively to constraints (3), (12) and (13). For simplicity, we collect all multipliers β_{ij} in a matrix $\Pi = (\pi_{ij})_{i,j \in N}$ defined as: (a) $\pi_{ii} = 0$ for all $i \in N$ and (b) $\beta_{ij} = \pi_{ij} = -\pi_{ji}$ for $1 \leq i < j \leq n$. Note that $\Pi = -\Pi^T$ and, thus, Π is skew-symmetric. Likewise, $\Phi = (\phi_{ij})_{i \in N, j \in M} \in \mathbb{R}^{n \times m}$ collects all multipliers assigned to (3).

Notice that, for any $Z \in \mathcal{S}_+^n$, the inner product $\langle \Lambda, Z \rangle$ is nonnegative whenever $\Lambda \succeq 0$. Hence, imposing $\Lambda \succeq 0$ alone allows the inner product $\langle \Lambda, Z \rangle$ to be subtracted in the LR objective function while still providing valid dual bounds for a minimization problem such as the pMQFLP. For this reason, in our initial applications of LR for approximating SDP bounds for other combinatorial optimization problems (see [6, 10]), the matrix Λ attached to an SDP constraint analogous to (13) was not required to be symmetric; only positive semidefiniteness was imposed. However, after a multiplier update in the SM, the matrix Λ had to be projected onto the space of positive semidefinite matrices, an expensive operation, in order to keep dual feasibility.

In the standard conic formulation of an SDP, the primal variable belongs to the space of symmetric matrices and the data matrices are symmetric. Consequently, symmetry of the dual slack matrix follows structurally, while positive semidefiniteness is required to prevent unboundedness of the dual problem. Our LR, however, is derived from an equivalent reformulation of the primal problem in $\mathbb{R}^{n \times n}$, where symmetry is imposed explicitly through constraints (12) and subsequently dualized in order to enable decomposability of the subproblem. In this setting, matrix Π of multipliers arises from the dualization of (12). As shown in [12], Λ can also be enforced to be symmetric without loss of generality. Indeed, any square matrix, Λ in particular, can be decomposed as the sum of a symmetric part, Λ_S, and a skew-symmetric part, Λ_A. More formally, one can write $\Lambda = \Lambda_A + \Lambda_S$, where $\Lambda_A = \frac{1}{2}(\Lambda - \Lambda^T)$. Therefore, the effect of the dualization of constraints (12) and (13) in the objective function of our Lagrangian function is $\langle \Pi - \Lambda, Z \rangle = \langle \Pi - \Lambda_A - \Lambda_S, Z \rangle = \langle \Pi - \Lambda_A, Z \rangle - \langle \Lambda_S, Z \rangle$. Now, notice that $\Pi - \Lambda_A$ is skew-symmetric. Thus, the skew symmetric part of Λ can be incorporated in Π and we can enforce $\Lambda \in \mathcal{S}_+^n$.

As any $\Lambda \in \mathcal{S}_+^n$ can be factored as $\Lambda = \Gamma\Gamma^T$ for $\Gamma \in \mathbb{R}^{n \times n}$, the LDP, obtained with the dualization of (13), can be reformulated in terms of the factor Γ instead of Λ. This allows the SM to update Γ directly, rather than updating Λ and subsequently projecting it onto the positive semidefinite cone. In this factorized reformulation, positive semidefiniteness is guaranteed by construction, since the product $\Gamma\Gamma^T$ is always positive semidefinite. Consequently, dual feasibility with respect to the semidefinite constraint is preserved without requiring spectral projections. Avoiding repeated eigendecompositions of Λ significantly reduces the computational cost per iteration. Moreover the term $\langle \Lambda, Z \rangle$ reduces to bilinear forms $\mathbf{\Gamma}_i^T \mathbf{\Gamma}_j$, which preserves the separable structure of the Lagrangian subproblem and enables its efficient solution via the Gilmore-Lawler (GL) procedure [5, 8]. Substantial computational savings are therefore obtained (see [11, 12]).

We are now ready to state our LR subproblem as

$$L(\Pi, \Phi, \Gamma) = \min \left\{ \left\langle \frac{1}{2} D + \Pi - \Gamma\Gamma^T + V, Z \right\rangle + \langle A + \Phi, Y \rangle : (\mathbf{x}, Y, Z) \in F_{lag} \right\}, \tag{17}$$

where $V \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $v_{ii} = c_i - \sum_{k \in M} \phi_{ik}$ with $i \in N$ and F_{lag} is the convex set defined by constraints (1)-(2), (9)-(11) and (14)-(16). Finally, the associated LDP is

$$ld = \max L(\Pi, \Phi, \Gamma), \tag{18}$$
$$s.t.\ \Pi \text{ skew symmetric}, \Phi \geq 0, \Gamma \in \mathbb{R}^{n \times n}.$$

3 Algorithms

In this section, we present an SM to approximate ld, the dual bound provided by the LDP (18). To this end, we first describe how to solve the LR subproblem (17) given a feasible dual solution (Π, Φ, Γ) to (18).

3.1 Solving the LR (17)

The problem (17) decomposes in two independent subproblems, one defined in terms of variables Y and the other in terms of variables $(\mathbf{x}, Z)$. The problem in Y, constrained by (2) and (15), can be solved by inspection for every $k \in M$. It suffices to pick the location $i(k)$ with the least Lagrangian modified cost $a_{ik} + \phi_{ik}$ and set $\overline{y}_{i(k),k} = 1$ and $\overline{y}_{ik} = 0$ for all $i \in N$ such that $i \neq i(k)$. Denote by $\overline{Y} \in \{0,1\}^{n \times m}$ the optimal solution to the problem in Y, obtained as outlined above.

The problem in $(\mathbf{x}, Z)$ decomposes into $n+1$ subproblems and can be solved via the GL procedure. First, for each $i \in N$, consider the subproblem of determining $\tilde{q}_i = \min\{\sum_{j=1}^n \overline{q}_{ij} z_{ij} : \sum_{j : j \neq i} z_{ij} = p-1, z_{ii} = 1, z_{ij} \in \{0,1\}, j \in N \setminus \{i\}\}$, where $\overline{q}_{ij} = \frac{1}{2} d_{ij} + \pi_{ij} + v_{ij} - \mathbf{\Gamma}_i^T \mathbf{\Gamma}_j$ and $\mathbf{\Gamma}_i \in \mathbb{R}^n$ denotes the $i-$th row of Γ. Let $\widetilde{\mathbf{Z}}_i \in \{0,1\}^n$ be an optimal solution to the $i-$th subproblem and let $\widetilde{Z} \in \{0,1\}^{n\times n}$ collect the solutions of all n subproblems, with $\widetilde{\mathbf{Z}}_i$ as its $i-$th row. Observe that $\tilde{q}_i$ can be determined by setting $\tilde{z}_{ij} = 1$ for the $p-1$ facilities with the smallest values of $\overline{q}_{ij}$ for $j \neq i$, together with $\tilde{z}_{ii} = 1$, and $\tilde{z}_{ij} = 0$ otherwise.

Once $\{(\tilde{q}_i, \widetilde{\mathbf{Z}}_i) : i \in N\}$ are computed, solve $L(\Pi, \Phi, \Gamma) = \min\left\{\sum_{i=1}^n \tilde{q}_i x_i : \sum_{i=1}^n x_i = p, \mathbf{x} \in \{0,1\}^n\right\}$, the problem in $\mathbf{x}$, and let $\overline{\mathbf{x}} \in \{0,1\}^n$ be an optimal solution to it. This problem can also be solved in polynomial time, by selecting the p facilities with the smallest values of $\{\tilde{q}_i : i \in N\}$. Given $\overline{\mathbf{x}}$, a companion optimal solution $\overline{Z} \in \{0,1\}^{n\times n}$ to (17) is defined row-wise as follows: $\overline{\mathbf{Z}}_i = \widetilde{\mathbf{Z}}_i$ if $\overline{x}_i = 1$ holds, or $\overline{\mathbf{Z}}_i = \mathbf{0}_n$ (the n dimensional vector of zeros), otherwise. In doing so, an optimal solution $(\overline{\mathbf{x}}, \overline{Y}, \overline{Z})$ to (17) is obtained.

3.2 Subgradient Optimization

The LDP (18) we aim to solve is not a piecewise linear function of (Π, Φ, Γ), as it depends quadratically on the entries of Γ. Therefore, strictly speaking, the method implemented here to solve (18) is not a pure SM, since the maximization problem (18) is not necessarily concave. Nevertheless, in a mild abuse of terminology, we refer to the multiplier adjustment method introduced here as an SM.

Let $t \geq 0$ denote the iteration index of the method. The algorithm is initialized with with $\Pi^0 = 0_{n\times n}$, $\Phi^0 = 0_{n\times m}$, and $\Gamma^0 = I_n$, that is, the $n \times n$ and $n \times m$ matrices of zeros and the identity matrix of order n, respectively. For any $t \geq 0$, the multipliers are updated according to

$$(\Pi^{t+1}, \Phi^{t+1}, \Gamma^{t+1}) = (\Pi^t, \Phi^t, \Gamma^t) + s^t \widehat{g}^t, \tag{19}$$

where s^t is a positive step size along the normalized search direction $\widehat{g}^t := \frac{g^t}{\|g^t\|_F^2}$ and g^t is defined as

$$g^t = (\overline{Z}^t - (\overline{Z}^t)^T, \overline{Y}^t - \text{Diag}(\overline{\mathbf{x}}^t)\mathbb{J}_{n\times m}, -(\overline{Z}^t + (\overline{Z}^t)^T)\Gamma^t). \tag{20}$$

After the update (19), the multipliers ϕ_{ik}^{t+1} are projected according to $\phi_{ik}^{t+1} = \max\{0, \phi_{ik}^{t+1}\}$, in order to preserve dual feasibility.

Observe that g^t, as defined in (20), is obtained by differentiating the Lagrangian function $L(\Pi, \Phi, \Gamma)$ with respect to (Π, Φ, Γ) at $(\Pi^t, \Phi^t, \Gamma^t)$. If the Lagrangian function were concave, the vector g^t would correspond to a supergradient of $L(\Pi, \Phi, \Gamma)$ in a standard implementation of the SM.

Given the search direction $\widehat{g}^t$, the SM computes the step size s^t as $s^t = s\kappa^t$, where s and κ are implementation parameters. These parameters are chosen so that, at the final iteration of the method, the desired terminal step size s' (another implementation parameter) coincides with s^t. Assuming that the SM performs `SMMAXITER` iterations, this yields $\kappa = \left(\frac{s'}{s}\right)^{\frac{1}{\texttt{SMMAXITER}}}$. In our implementation, we set $s = 1$, $s' = 0.001$, and `SMMAXITER` $= 50000$ iterations, after which the best LR bound encountered is recorded. Upon termination, the SM returns the bound lb_{sm}, which provides a lower approximation of ld, as defined by (18).

4 Numerical experiments

The first objective of our numerical experiments is to assess, in practical terms, the strength of our SDP relaxation (8) in comparison with the SDP relaxation proposed in [16]. The second objective is to evaluate how accurately and efficient the SM approximates the bounds lb_{sdp} provided by relaxation (8). Finally, we compare the quality of our LR SDP bounds lb_{sm} with the best exact dual bounds attainable through the SDP relaxation from the literature [16].

Our numerical experiments are conducted on the Euclidean benchmark instances for uncapacitated facility location problems introduced by Kochetov and Ivanenko [7]. This dataset corresponds to one of the three pMQFLP instance sets considered in [16]. The remaining two datasets were unavailable to us and, consequently, are not included in this preliminary study.

Kochetov and Ivanenko [7] generated 30 sets of points in the two-dimensional Euclidean plane, with integer coordinates chosen from a square of side length 7000. Based on these point sets, Yang et al. [16] constructed pMQFLP instances as follows. First, every point serves both as a client and as a potential facility location, so that $n = m$ holds.

Five different problem sizes are obtained by selecting the first $n \in \{60, 70, 80, 90, 100\}$ points from each of the 30 point sets from [7]. Fixed installation costs $c_i = 3000$ are set for all $i \in N$. Connection costs d_{ij} and a_{ik} are defined as the Euclidean distance between the points, rounded to the nearest integer, so as to satisfy the triangle inequalities. For each point set, four different values of $p \in \{15, 20, 25, 30\}$ are tested. In total, our numerical experiments comprise 600 pMQFLP instances.

Table 1 reports five groups of numerical results, organized in blocks of rows. Each block corresponds to one specific performance metric investigated in this study. For each block, the table presents average (avg), minimum (min) and maximum (max) values for the corresponding metric, reported for every (p, n) combination.

The first block of results reports the gap between the best dual bound lb_{sm} obtained by the SM after `SMMAXITER` iterations and the exact bound lb_{sdp}, computed by solving (8) using a cone optimization algorithm [9] through the `CVXPY` convex optimization suite [2]. The gap is defined as $100\left(\frac{lb_{sdp} - lb_{sm}}{lb_{sdp}}\right)$. The second block of results reports the total CPU time, in seconds, required by the SM to compute bounds lb_{sm}. The third block reports the ratio between the CPU time taken by the SM to compute lb_{sm} and the time required to solve the convex problem (8) using `CVXPY`, that is, to compute the exact bound lb_{sdp}. The fourth block of results reports the gap $100\left(\frac{lb_{sdp} - lb_{wsdp}}{lb_{wsdp}}\right)$, between the exact bounds lb_{sdp} provided by our SDP relaxation (8) and the exact SDP bounds lb_{wsdp} obtained from the weaker SDP relaxation proposed in [16], defined as

$$lb_{wsdp} = \min\left\{\frac{1}{2}\langle D, Z\rangle + \mathbf{c}^T\mathbf{x} + \langle A, Y\rangle : (\mathbf{x}, Y, Z') \in F_{wsdp}\right\}, \tag{21}$$

where F_{wsdp} is the convex set defined as the intersection of $\{z_{ij} \in [0,1] : i, j \in N\}$ and constraints (1), (2), (7), (10), and (14)-(15). The bound lb_{wsdp} is also computed by `CVXPY`. The SDP relaxation (21) coincides with the SDP model introduced in [16], on which the `SDP+BD` approach relies, prior to projecting out the variables Y (see model (19) in [16], which corresponds exactly to (21), with Y variables and without the inclusion of Benders optimality cuts). Finally, the fifth block of results reports the gap $100\left(\frac{lb_{sm}-lb_{wsdp}}{lb_{wsdp}}\right)$, which indicates how much stronger the SDP bounds computed by our SM algorithm are compared to the bounds lb_{wsdp} obtained from (21). For 12 instances, CVXPY encountered numerical difficulties when solving the SDP relaxation (21). Thus, the statistics reported in the fourth and fifth blocks of Table 1 exclude these 12 instances.

The SM was implemented in Python 3 and our numerical experiments were conducted on a machine equipped with an AMD Ryzen 7900X CPU, with 256 GB RAM, running at 5.3Ghz, under Linux. Only a single core was used.

Our numerical results show that the bounds lb_{sdp} are indeed stronger than the exact SDP bounds lb_{wsdp} from [16]. On average, considering the 588 instances (600 minus 12) for which the bounds lb_{wsdp} could be computed exactly, the bounds lb_{sdp} are 1.54% stronger than their lb_{wsdp} counterparts.

Another noteworthy finding is that, despite the inexact nature of its search procedure, the SM provides quite accurate approximations lb_{sm} of best attainable bounds lb_{sdp}. On average, over the full set of 600 instances, the bounds lb_{sm} are only 0.054% weaker than the corresponding lb_{sdp} values. Moreover, the deviation never exceeds 0.096%. Among the 600 instances tested, only one case (with $n = 60$ and $p = 30$) exhibits a higher CPU time for the LR algorithm (5.45 seconds) than for the corresponding CVXPY computation (4.25 seconds). Overall, the SM requires, on average, only 16.2% of the CPU time needed by the convex optimization algorithm invoked by CVXPY to solve the relaxation (8). Results reported in Table 1 further show that the CPU time advantage of the LR scheme over the convex optimization solver increases as the problem size (n) grows, without compromising the accuracy of the SDP bound approximation.

Complementing these results, we observe that the bounds lb_{sm} produced by the SM are always stronger than the exact bounds lb_{wsdp} defined by (21). As shown in the fifth block of Table 1, the bounds lb_{sm} are, on average, 1.48% stronger than their lb_{wsdp} counterparts. In fact, for the 588 instances for which a direct comparison is possible, the bounds lb_{sm} are always at least 0.51% stronger than lb_{wsdp}.

In summary, the LR bounds lb_{sm} computed by SM are very close to the best bounds that can be obtained by exactly solving (8). When compared to the CPU times needed to compute SDP relaxations such as (21) and (8), the computational effort needed by the SM to produce these bounds is small. These results suggest that an exact algorithm, of the Branch-and-bound type, using the SM approach outlined here to provide dual bounds, may be a promising alternative for solving the pMQFLP to proven optimality.

5 Conclusions

In this paper, we investigated the p-Median Quadratic Facility Location Problem and proposed a stronger SDP relaxation together with a Lagrangian Relaxation (LR) algorithm to accurately approximate the resulting SDP bounds. Numerical experiments conducted on 600 benchmark pMQFLP instances show that our exact SDP bounds, computed using a convex optimization solver, are on average 1.54% stronger than SDP bounds from the literature, obtained in a similar manner. In addition, the proposed LR algorithm provides highly accurate approximations of these bounds at a substantially reduced computational cost. On average, the LR bounds deviate by only 0.054% from the best attainable exact SDP bounds, with a maximum deviation of 0.096%, while requiring approximately 16% of the CPU time needed to compute the exact SDP bounds. Despite the inexact nature of the Subgradient Method used to solve the LDP, the LR bounds consistently outperform the SDP bounds from the literature for all 600 tested instances, being on average 1.48% stronger.

Since the best performing algorithm from the literature computes weaker SDP bounds by repeatedly solving more expensive SDP relaxations within a cutting plane framework, these results suggest that developing an exact Branch-and-bound algorithm based on the proposed LR scheme is a promising direction for future research.

References

[1] P. Adasme, A. Viveros, and A. Dehghan Firoozabadi. 2024. Quadratic p-Median Problem: A Bender's Decomposition and a Meta-Heuristic Local-Based Approach. *Symmetry* 16, 9 (2024).

[2] S. Diamond and S. Boyd. 2016. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research* 17, 83 (2016), 1–5.

[3] A. T. Ernst and M. Krishnamoorthy. 1996. Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science* 4, 3 (1996), 139–154.

[4] D. Gijswijt. 2006. *Matrix algebras and semidefinite programming techniques for codes.* Ph. D. Dissertation. University of Amsterdam.

[5] P. C. Gilmore. 1962. Optimal and Suboptimal Algorithms for the Quadratic Assignment Problem. *J. Soc. Indust. Appl. Math.* 10, 2 (jun 1962), 305–313.

[6] D. Guimarães, A. Salles da Cunha, and D. L. Pereira. 2020. Semidefinite programming lower bounds and branch-and-bound algorithms for the Quadratic Minimum Spanning Tree Problem. *European Journal of Operations Research* 280, 1 (2020), 46–58.

[7] Y. Kochetov and D. Ivanenko. 2005. *Computationally Difficult Instances for the Uncapacitated Facility Location Problem.* 351–367.

[8] E. L. Lawler. 1963. The Quadratic Assignment Problem. *Management Science* 9, 4 (jul 1963), 586–599.

[9] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. 2016. Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding. *Journal of Optimization Theory and Applications* 169 (2016), 1042–1068. Issue 3.

[10] D. L. Pereira, D. A. Guimarães, A. Salles da Cunha, and A. Lucena. 2024. Quadratically Constrained Reformulation, Strong Semidefinite Programming Bounds, and Algorithms for the Chordless Cycle Problem. In *Combinatorial Optimization - 8th International Symposium, ISCO 2024, La Laguna, Tenerife, Spain, May 22-24, 2024, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 14594)*, A. Basu, A. R. Mahjoub, and J. J. S. González (Eds.). Springer, 30–42.

[11] D. L. Pereira, D. A. Guimarães, A. Salles da Cunha, and A. Lucena. 2025. Semidefinite Programming bounds and a Branch-and-bound algorithm for the Chordless Cycle Problem. *Discrete Applied Mathematics* (2025). Submitted for publication.

[12] D. L. Pereira and A. Salles da Cunha. 2026. Branch-and-bound algorithms for the K-cluster problem based on SDP bounds evaluated by Lagrangian relaxation. *Networks* 87, 1 (2026), 23–38.

[13] C. Sandoval, P. Adasme, and A. D. Firoozabadi. 2021. Quadratic p-Median Formulations with Connectivity Costs Between Facilities. In *Mobile Web and Intelligent Information Systems*, J. Bentahar, I. Awan, M. Younas, and T. Grønli (Eds.). Springer International Publishing, 99–107.

[14] H. D. Sherali and W. P. Adams. 1994. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero—one programming problems. *Discrete Applied Mathematics* 52, 1 (1994), 83 – 106.

[15] R. Sotirov. 2018. Graph bisection revisited. *Ann. Opers. Res.* 265 (2018), 143–154.

Table 1: Summary of numerical results.

Gap between the LR SDP dual bound lb_{sm} and the exact SDP bound lb_{sdp}, in %

p	$n = 60$			$n = 70$			$n = 80$			$n = 90$			$n = 100$		
	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max
15	0.066	0.045	0.076	0.074	0.066	0.086	0.079	0.072	0.089	0.082	0.071	0.096	0.086	0.079	0.094
20	0.047	0.026	0.058	0.054	0.040	0.062	0.061	0.046	0.074	0.065	0.055	0.074	0.068	0.054	0.079
25	0.033	0.026	0.049	0.039	0.027	0.070	0.047	0.039	0.062	0.052	0.034	0.066	0.057	0.035	0.067
30	0.023	0.000	0.039	0.030	0.023	0.052	0.035	0.010	0.050	0.042	0.031	0.074	0.047	0.034	0.056

CPU time taken by SM to evaluate the bounds lb_{sm}, in seconds

p	$n = 60$			$n = 70$			$n = 80$			$n = 90$			$n = 100$		
	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max
15	5.15	5.09	5.29	6.93	6.88	7.10	7.99	7.94	8.08	9.34	9.25	9.55	10.92	10.82	11.05
20	5.24	5.19	5.46	7.04	6.99	7.26	8.12	8.05	8.20	9.53	9.38	9.71	11.02	10.91	11.14
25	5.38	5.32	5.57	7.18	7.12	7.41	8.24	8.20	8.30	9.83	9.76	10.07	11.24	11.10	11.54
30	5.55	5.45	5.72	7.34	7.27	7.57	8.42	8.37	8.75	9.97	9.93	10.02	11.63	11.55	11.72

Ratio between the SM CPU time and the CPU time taken by `CVXPY` to solve SDP relaxation (8)

p	$n = 60$			$n = 70$			$n = 80$			$n = 90$			$n = 100$		
	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max
15	0.262	0.149	0.524	0.182	0.090	0.342	0.135	0.101	0.197	0.119	0.068	0.275	0.101	0.060	0.159
20	0.270	0.120	0.601	0.198	0.093	0.390	0.133	0.082	0.224	0.098	0.073	0.200	0.079	0.059	0.131
25	0.265	0.113	0.527	0.199	0.089	0.321	0.140	0.076	0.254	0.103	0.066	0.183	0.079	0.059	0.122
30	0.351	0.135	1.282	0.197	0.090	0.346	0.150	0.076	0.322	0.103	0.066	0.157	0.080	0.060	0.113

Gap between the exact SDP bound lb_{sdp} and the exact SDP bound lb_{wsdp} (21) from [16], in %

p	$n = 60$			$n = 70$			$n = 80$			$n = 90$			$n = 100$		
	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max
15	1.26	0.63	2.12	1.24	0.61	2.72	1.21	0.59	2.60	1.20	0.60	2.24	1.19	0.59	1.78
20	1.63	0.80	3.00	1.47	0.67	2.30	1.38	0.84	2.24	1.39	0.87	2.88	1.27	0.54	2.41
25	1.93	0.79	2.71	1.79	0.85	2.79	1.61	0.77	2.55	1.53	0.82	2.52	1.48	0.42	2.10
30	1.97	0.41	3.51	1.91	0.58	2.76	1.88	0.28	2.61	1.76	0.81	2.84	1.71	0.85	2.74

Gap between the LR SDP bound lb_{sm} and the exact SDP bound lb_{wsdp} (21) from [16], in %

p	$n = 60$			$n = 70$			$n = 80$			$n = 90$			$n = 100$		
	avg	min	max	avg	min	max	avg	min	max	avg	min	max	avg	min	max
15	1.19	0.57	2.05	1.16	0.54	2.64	1.13	0.50	2.52	1.12	0.52	2.16	1.11	0.51	1.69
20	1.58	0.75	2.95	1.42	0.62	2.24	1.32	0.78	2.17	1.32	0.81	2.81	1.20	0.49	2.34
25	1.90	0.75	2.68	1.75	0.82	2.74	1.57	0.71	2.50	1.48	0.79	2.47	1.42	0.39	2.04
30	1.95	0.41	3.49	1.88	0.54	2.73	1.84	0.27	2.58	1.72	0.78	2.79	1.66	0.81	2.69

[16] Y. Yang, H. T. Bui, and R. Loxton. 2025. SDP-based Benders decomposition for solving p-median quadratic facility location problems. *Computers & Operations Research* 182 (2025), 107119.