Feasible Point Pursuit and Successive Approximation of Non-Convex QCQPs

Omar Mehanna, Student Member, IEEE, Kejun Huang, Student Member, IEEE, Balasubramanian Gopalakrishnan, Student Member, IEEE, Aritra Konar, Student Member, IEEE, and Nicholas D. Sidiropoulos, Fellow, IEEE

Abstract—Quadratically constrained quadratic programs (QCQPs) have a wide range of applications in signal processing and wireless communications. Non-convex QCQPs are NP-hard in general. Existing approaches relax the non-convexity using semi-definite relaxation (SDR) or linearize the non-convex part and solve the resulting convex problem. However, these techniques are seldom successful in even obtaining a feasible solution when the QCQP matrices are indefinite. In this letter, a new feasible point pursuit successive convex approximation (FPP-SCA) algorithm is proposed for non-convex QCQPs. FPP-SCA linearizes the non-convex parts of the problem as conventional SCA does, but adds slack variables to sustain feasibility, and a penalty to ensure slacks are sparingly used. When FPP-SCA is successful in identifying a feasible point of the non-convex QCQP, convergence to a Karush-Kuhn-Tucker (KKT) point is thereafter ensured. Simulations show the effectiveness of our proposed algorithm in obtaining feasible and near-optimal solutions, significantly outperforming existing approaches.

Index Terms—Feasible point pursuit, linearization, multicast beamforming, non-convex QCQP, semi-definite relaxation, successive convex approximation

I. INTRODUCTION

QUADRATICALLY constrained quadratic programs (QCQPs) are an important class of optimization problems that have a wide spectrum of applications ranging from transmit beamforming in wireless networks, to portfolio risk management in financial engineering [1], [2]. A QCQP can be expressed as

$$\Pi_1: \min_{x \in \mathbb{C}^n} \quad x^H A_0 x$$

s.t. \quad $x^H A_m x \leq c_m, \quad m = 1, \ldots, M$ \hspace{1cm} (1)

where $A_0 \succeq 0$, i.e., positive semi-definite, and $A_m \in \mathbb{C}^{n \times n}$ are Hermitian matrices for all $m \in \{1, \ldots, M\}$. In the special case when $A_m \succeq 0$ for all $m \in \{1, \ldots, M\}$, the QCQP $\Pi_1$ becomes a convex optimization problem which can be efficiently solved to optimality using interior point methods [3]. For general indefinite $A_m$, this problem is non-convex and NP-hard [4], except for special cases, such as when $M \leq 3$ [5]–[9].

Several methods have been proposed to approximate non-convex QCQPs, including (a) the (prevailing) semi-definite relaxation (SDR) approach [3]; (b) the reformulation linearization technique (RLT) [10], [11]; and (c) successive convex approximation (SCA) [12]–[15]. RLT consists of a reformulation step and a linearization step. The reformulation step creates redundant nonlinear constraints involving pairwise product combinations of the individual scalar variables, by multiplying different constraint pairs. The linearization step then substitutes a continuous variable for each distinct product of variables. The resulting convex optimization problem is solved to obtain an approximate solution of the non-convex problem. The main issue with RLT is that the solution of the linear program is seldom feasible for the non-convex problem. Furthermore the size of the linear program approximation is much larger than the original problem, thereby making it computationally involved.

The SCA approach is a more general scheme to deal with non-convex problems, and its application to non-convex QCQPs is sometimes called the convex-concave approach [4]: each quadratic term is separated into convex and concave parts, and the latter is replaced by a convex (usually linear) approximation around a feasible point. The resulting convex problem is solved to obtain the next iterate, which also serves as the approximation point for the next iteration. Scutari et al. recently proposed a parallel and distributed SCA framework to obtain stationary points for non-convex optimization problems [15]. The algorithm starts from an initial feasible point, the non-convex constraints are approximated by a strictly convex function around this point, and the resulting convex problem is solved to obtain the next iterate. This procedure is repeated until convergence to a stationary point.

The drawback with approaches (b) and (c) is that they need a feasible point as initialization, which is difficult to obtain in general. Constraint approximation about a feasible point yields a nonempty set that contains at least the given point, whereas constraint approximation about an infeasible point tends to yield an empty set, even if the original problem is feasible. Existing convergence results for SCA depend on a feasible initialization, e.g., [12].

The most popular among the above is the SDR approach [1], [3], where the original problem is reformulated by introducing...
\( \mathbf{X} := \mathbf{x}\mathbf{x}^H \) and solving the semi-definite program (SDP) \( \Pi_{1\tau} \) obtained after relaxing the rank-1 constraint.

\[
\Pi_{1\tau} \quad \min_{\mathbf{X} \in \mathbb{C}^{n\times n}} \quad \text{Trace}(\mathbf{A}_2 \mathbf{X}) \\
\text{s.t.} \quad \text{Trace}(\mathbf{A}_m \mathbf{X}) \leq c_m, \quad m = 1, \ldots, M
\]

Because of the rank relaxation, the solution to \( \Pi_{1\tau} \) gives a lower bound on the optimal value of the cost function of \( \Pi_1 \). Note that the SDR \( \Pi_{1\tau} \) is the Lagrange bi-dual of \( \Pi_1 \). If the solution \( \mathbf{X}^* \) to problem \( \Pi_{1\tau} \) is rank-1, the optimal solution \( \mathbf{x}^* \) to \( \Pi_1 \) is the principal eigenvector of \( \mathbf{X}^* \), scaled by the square root of the maximum eigenvalue of \( \mathbf{X}^* \); otherwise, randomization techniques are used [1]. If the matrices \( \{\mathbf{A}_m\}_{m=1}^M \) are all negative semidefinite, then any randomly generated point can be scaled up to satisfy the constraints of the QCQP \( \Pi_1 \); finding a feasible solution using randomization is easy in this case, and the challenge is to find one that is close to optimum, see [1], [16].

In the general setting where \( \{\mathbf{A}_m\}_{m=1}^M \) are all indefinite, or when one deals with two-sided positive semidefinite constraints such as in [17], SDR with randomization often fails to find a point that satisfies the constraints in \( \Pi_1 \). That is why it is important to develop an alternate approach (instead of SDR followed by randomization) that has a high probability of finding a feasible solution to the NP-hard QCQP \( \Pi_1 \), when it exists.

In this letter we propose an iterative algorithm for obtaining good feasible solutions to general QCQPs, where we approximate the feasible region through a linear restriction of the non-convex parts of the constraints. In order to guarantee feasibility of the modified problem, slack variables are added, and a penalty is used to ensure that slacks are sparingly used. The solution of the resulting optimization problem is then used to compute a new linearization, and the procedure is repeated until convergence. The proposed Feasible Point Pursuit - Successive Convex Approximation (FPP-SCA) algorithm differs from the conventional SCA approach [12]–[14] in that the latter requires the starting point to be in the feasible region of the original problem. Finding a feasible point is easy in some cases, such as when all the constraints \( x^H A_m x \leq c_m \) involve negative semi-definite \( A_m \), as considered in [14]. For general QCQPs, however, finding an initial feasible point is hard. The performance of the proposed algorithm is compared with the conventional SDR followed by randomization, and simulations show that the proposed algorithm attains a feasible solution for a much larger percentage of problem instances. Furthermore, the feasible solution obtained using the proposed algorithm is much closer to the SDR lower bound than SDR followed by randomization.

II. THE FPP-SCA APPROACH

Problem \( \Pi_1 \) may or may not be feasible, and establishing (in)feasibility is generally NP-hard. When infeasible, one may instead seek a compromise that minimizes constraint violations in some sense - this is common in engineering applications. In order to account for potential infeasibility, consider adding slack variables \( \mathbf{s} \in \mathbb{R}^M \) and a slack penalty to \( \Pi_1 \)

\[
\Pi_2 \quad \min_{\mathbf{X} \in \mathbb{C}^{n\times n}, \mathbf{s} \in \mathbb{R}^M} \quad \mathbf{x}^H \mathbf{A}_0 \mathbf{x} + \lambda \| \mathbf{s} \|_1 \\
\text{s.t.} \quad \mathbf{x}^H \mathbf{A}_m \mathbf{x} \leq c_m + s_m, \quad m = 1, \ldots, M
\]

where \( \lambda \) trades off the original objective function and the slack penalty term, and \( \| \cdot \|_1 \) can be any vector norm. Problem \( \Pi_2 \) is always feasible, and if \( (\mathbf{x}_0, \mathbf{s}_0) \) is an optimal solution of \( \Pi_2 \) and it so happens that \( s_0 = 0 \), then \( \mathbf{x}_0 \) is an optimal solution of \( \Pi_1 \); else using the \( l_1 \) norm of \( \mathbf{s} \) in \( \Pi_1 \) (which reduces to the sum of the slacks, due to the non-negativity constraints) promotes sparsity in terms of constraint violations. The difficulty though is that problem \( \Pi_1 \) remains non-convex and NP-hard in general.

Successive convex approximation (SCA). Using eigen-decomposition, the matrix \( \mathbf{A}_m \) can be expressed as \( \mathbf{A}_m = \mathbf{A}_m^{(+)} + \mathbf{A}_m^{(-)} \), where \( \mathbf{A}_m^{(+)} \geq 0 \) and \( \mathbf{A}_m^{(-)} \leq 0 \) (negative semi-definite).

Algorithm 1 Feasible Point Pursuit Successive Convex Approximation (FPP-SCA) Algorithm

Initialization: Set \( k = 0 \) and randomly generate an initial point \( \mathbf{z}_0 \).

Repeat

1) Solve

\[
\Pi_3 \quad \min_{\mathbf{x}, \mathbf{s}} \quad \mathbf{x}^H \mathbf{A}_0 \mathbf{x} + \lambda \sum_{m=1}^M s_m \\
\text{s.t.} \quad \mathbf{x}^H \mathbf{A}_m \mathbf{x} + 2 \text{Re} \left\{ \mathbf{z}_k^H \mathbf{A}_m^{(-)} \mathbf{x} \right\} \leq c_m + \mathbf{z}_k^H \mathbf{A}_m^{(-)} \mathbf{z}_k + s_m, \quad \begin{array}{c} s_m \geq 0, \quad m = 1, \ldots, M \end{array}
\]

2) Let \( \mathbf{x}_k \) denote the optimal \( \mathbf{x} \) obtained by solving \( \Pi_3 \) at the \( k \)-th iteration, and set \( \mathbf{z}_{k+1} = \mathbf{x}_k \).

3) Set \( k = k + 1 \).

until convergence.

Some important remarks and claims are in order.

- We first relaxed the constraints in \( \Pi_1 \) by adding slacks, then tightened the relaxed constraints via partial linear restriction of their non-convex parts. We could instead first tighten the original constraints (risking turning a feasible original problem into an infeasible one) then relax by adding slacks to make the restriction feasible–the net result turns out being \( \Pi_3 \) in both cases, and it is always feasible.
- FPP-SCA yields a non-increasing cost sequence, i.e., the optimal cost of \( \Pi_3 \) is non-increasing in \( k \). This is because the cost function is independent of \( k \), and the solution of the \( k \)-th iteration is also feasible for the \((k + 1)\)-th iteration. To see this, note that \( \mathbf{z}_{k+1} = \mathbf{x}_k \) is the optimal solution of \( \Pi_3 \) at the \( k \)-th
iteration, so it satisfies the restriction \( y_{k+1}^H A_m x_k + 2 \Re \{ y_{k+1}^H A_m z_{k+1} \} \leq c_m^* + \sigma_{z_{k+1}}^a \).

Looking at the corresponding constraint at the next iteration, \( x^H A_m x + 2 \Re \{ z_{k+1}^H A_m x \} \leq c_m^* + s_m \).

- Problem \( \Pi_3 \) is convex and can be easily formulated as a second-order cone program (SOCP). Worst-case complexity of solving \( \Pi_3 \) is \( O(n + M)^3 \) \( \ll O(n^3) \) for SDR. FPP-SCA usually takes just a few iterations to converge.

- FPP-SCA can be run using different starting points \( \mathbf{z}_0 \) and the best solution can be taken. Simulations suggest that SDR can provide a good initialization for FPP-SCA, if the extra complexity of SDR is acceptable. Otherwise random initialization(s) can be used.

- We propose using \( \lambda \gg 1 \) to force the slack variables toward zero, thereby pushing the iterates towards the feasible region of \( \Pi_1 \) when this is non-empty. Higher \( \lambda \) also helps ensure that if a feasible point of \( \Pi_1 \) is found, subsequent iterates will remain in the feasible region of \( \Pi_1 \), although there are no analytical guarantees for this. Note that the main advantage of FPP-SCA over conventional SCA is the ability to find a feasible point with high probability. Once the slacks are all 0, one can simply switch to conventional SCA without the slacks. Our simulations show that the subsequent iterates are almost identical between these two schemes (with or without the slack variables).

- If FPP-SCA converges, it converges to a KKT point for problem \( \Pi_2 \), according to Beck et al. [12]. If the converged slack variables turn out being all zero, then it is easy to show that the remaining variables satisfy the KKT conditions for the original problem \( \Pi_1 \).

**Illustrative example.** To get a better understanding of the approximations used in the FPP-SCA algorithm and how the solution evolves after each iteration, Fig. 1 considers a setup in \( \mathbb{R}^2 \) with \( M = 3 \), where \( \mathbf{A}_0 = \mathbf{I} \), \( \mathbf{A}_1 = \begin{bmatrix} -1.48 & 0.68 \\ -0.93 & -0.17 \end{bmatrix}, \mathbf{A}_2 = \begin{bmatrix} -1.07 & 0.68 \\ -0.35 & -0.17 \end{bmatrix}, \mathbf{A}_3 = \begin{bmatrix} -0.17 & 0.41 \\ -1.59 & -0.17 \end{bmatrix} \), \( c_1 = c_2 = -1 \), and \( c_3 = 1 \). Note that \( \mathbf{A}_1 \) and \( \mathbf{A}_2 \) are negative semidefinite, whereas \( \mathbf{A}_3 \) is positive semidefinite. The ellipsoids that correspond to \( x^H \mathbf{A}_m x = c_m^* \) for \( m = 1, 2 \) are plotted in red in Fig. 1, while the ellipsoid for \( m = 3 \) is plotted in green. Two different initializations for \( \mathbf{z}_0 \) are considered, both are chosen with relatively large scale so that the first two constraints of the QCQP are satisfied, but not the third one. Therefore, only the slack variables that correspond to the third constraint can be nonzero. The FPP-SCA algorithm is run for 3 iterations using \( \lambda = 10 \), and the figure shows the solution after each iteration.

### III. NUMERICAL RESULTS

To test the performance of the FPP-SCA algorithm, a problem with \( n = 8 \) complex dimensions is considered, with \( M \in \{16, 24, 32\} \). The entries of the matrices \( \{\mathbf{A}_m\}_{M=1}^M \) are randomly and independently generated from a complex Gaussian distribution (with zero-mean and variance 2), then symmetrized. An initial point \( \mathbf{x}_{\text{init}} \) is randomly generated, and the values of \( \{c_m\}_{M=1}^M \) are randomly generated from a Gaussian distribution \( c_m \sim \mathcal{N}(\mathbf{x}_{\text{init}}^H \mathbf{A}_m \mathbf{x}_{\text{init}}, 1) \). If \( \mathbf{x}_{\text{init}}^H \mathbf{A}_m \mathbf{x}_{\text{init}} > c_m \), the matrix \( \mathbf{A}_m^c \) and \( c_m^* \) are multiplied by \(-1\) to get \( \leq \) inequalities. The matrix \( \mathbf{A}_0 \) is set to the identity. To solve the SDR \( \Pi_1 \), and the SCA \( \Pi_3 \), the modeling language YALMIP [19] is used and the generic conic programming solver SeDuMi [20] is chosen as the
TABLE I
RESULTS USING THE SDR APPROACH FOR n = 5

<table>
<thead>
<tr>
<th>M</th>
<th>16</th>
<th>24</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank-1 solution</td>
<td>45%</td>
<td>16%</td>
<td>5.5%</td>
</tr>
<tr>
<td>No feasible sol. after randomization</td>
<td>42%</td>
<td>80%</td>
<td>94.2%</td>
</tr>
<tr>
<td>Feasible sol. after randomization</td>
<td>13%</td>
<td>4%</td>
<td>0.3%</td>
</tr>
<tr>
<td>Avg. loss (dB)</td>
<td>1.3</td>
<td>1.5</td>
<td>-</td>
</tr>
</tbody>
</table>

TABLE II
RESULTS USING THE FPP-SCA APPROACH FOR n = 5

<table>
<thead>
<tr>
<th>M</th>
<th>16</th>
<th>24</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feasible solution</td>
<td>100%</td>
<td>99.5%</td>
<td>92.8%</td>
</tr>
<tr>
<td>Avg. itrs. for feasibility</td>
<td>3.207</td>
<td>8.471</td>
<td>7.893</td>
</tr>
<tr>
<td>Avg. itrs. for convergence</td>
<td>10.97</td>
<td>11.6371</td>
<td>12.1352</td>
</tr>
<tr>
<td>Avg. loss (dB)</td>
<td>0.942</td>
<td>1.5684</td>
<td>1.9256</td>
</tr>
</tbody>
</table>

TABLE III
RESULTS USING THE SDR APPROACH FOR n = 20

<table>
<thead>
<tr>
<th>M</th>
<th>32</th>
<th>40</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank-1 solution</td>
<td>11.5%</td>
<td>2.9%</td>
<td>0.4%</td>
</tr>
<tr>
<td>No feasible sol. after randomization</td>
<td>84.7%</td>
<td>96.4%</td>
<td>99.3%</td>
</tr>
<tr>
<td>Feasible sol. after randomization</td>
<td>3.60%</td>
<td>8.7%</td>
<td>1.3%</td>
</tr>
</tbody>
</table>

TABLE IV
RESULTS USING THE FPP-SCA APPROACH FOR n = 20

<table>
<thead>
<tr>
<th>M</th>
<th>32</th>
<th>40</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feasible solution</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Avg. itrs. for feasibility</td>
<td>4.3640</td>
<td>5.0927</td>
<td>5.9840</td>
</tr>
<tr>
<td>Avg. itrs. for convergence</td>
<td>16.2130</td>
<td>16.3568</td>
<td>17.2805</td>
</tr>
<tr>
<td>Avg. loss (dB)</td>
<td>0.4570</td>
<td>0.4881</td>
<td>0.5618</td>
</tr>
</tbody>
</table>

This paper previously published in IEEE Signal Processing Letters
REFERENCES


