2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

# Reinforcement Learning-Guided Quadratically Constrained Quadratic Programming for Enhanced Convergence and Optimality

Chaoying Pei, Zhi Xu, Sixiong You, Jeffrey Sun, and Ran Dai

*Abstract*— **In the context of Quadratically Constrained Quadratic Programming (QCQP) with dynamic parameters, the effectiveness of various optimization approaches is heavily influenced by the quality of the initial guess. To address this challenge, this paper proposes a novel approach that leverages reinforcement learning (RL) to generate high-performing initial guesses for iterative algorithms, with the dynamic parameters serving as inputs. Our approach aims to accelerate convergence and improve the objective value, thereby enabling efficient and effective solutions to the QCQP problem under variability. In this study, we evaluate the proposed approach by applying it to an iterative algorithm, specifically the Iterative Rank Minimization (IRM) algorithm. Our empirical evaluations demonstrate the efficacy of the proposed approach in solving QCQP problems with dynamic parameters. The RL-guided IRM algorithm yields high-quality solutions, as evidenced by significantly improved optimality and faster convergence when compared to the original IRM algorithm.**

## I. INTRODUCTION

Quadratically constrained quadratic programming (QCQP) is a class of nonlinear optimization problems aimed at minimizing a quadratic objective function under quadratic constraints. In recent years, QCQP has garnered considerable attention due to its extensive range of applications in fields such as engineering [1], economics [2], and finance [3]. Despite its numerous applications, the real-world instances of QCQP often feature variability in the problem settings. Such instances are referred to as QCQP under variability, and their popularity has grown exponentially in recent years due to the prevalence of dynamic parameters in real-world optimization problems.

There has been considerable progress in the development of numerical methods for solving general and nonconvex QCQP problems. Two main categories of methods have emerged: relaxation [4] and successive convex approximation (SCA) [5], [6]. While relaxation methods involve relaxing the constraints of the problem to obtain a tractable optimization problem that can be solved using standard techniques, SCA approximates the nonconvex problem with a sequence of convex subproblems. Both approaches have been extensively studied and have shown promising results in solving QCQPs. However, while relaxation methods can obtain a lower bound on the cost function, they do not ensure finding the optimal or feasible solutions for most nonconvex QCQPs. On the other hand, SCA approaches tackle QCQPs by using sequential convex optimization, where each iteration is solved by a convex solver like CVX [7]. Nevertheless, for large-scale QCQPs, SCA may not be computationally efficient if another iterative approach is needed to solve each sequential problem formulated in SCA, such as the interior point method [8].

Many real-world optimization problems can be approximated by a polynomial optimization problem and then equivalently transformed into a QCQP problem. However, real-world optimization problems are often characterized by dynamic natures, with changing parameters and constraints over time, under varying conditions or different scenarios. For example, real-time optimal control is a typical dynamic optimization problem where the objective function and constraints change depending on the environment and system dynamics. Another example is the power system scheduling problem [9], which involves scheduling power generation units to minimize operation costs while taking into account the time-varying power demand and availability of the units. In these cases, the formulated QCQP problem has dynamic parameters in the objective or constraints or both, leading to varying solution space. Then the optimization problem needs to be solved repeatedly, which can be computationally expensive and require significant computing resources to ensure efficient solving.

To enhance the efficiency of the optimization process in each step, researchers have developed various techniques, such as approximation methods [10], decomposition methods [11], and heuristic methods [12]. These methods can be effective in reducing the computational burden of solving dynamic optimization problems. However, they also have limitations in terms of optimality and accuracy. To improve computational efficiency, generating a well-informed initial guess for successive algorithms can be a game-changer. A well-informed initial guess can significantly improve the efficiency and optimality of the optimization algorithm. Therefore, to address the challenges posed by the variability in QCQP problems, this work proposes a reinforcement learning (RL)-based method for generating high-quality initial guesses, which can lead to significantly improved convergence performance and optimality.

In recent years, machine learning, particularly RL, has gained significant traction as a powerful tool for optimization [13], [14]. An illustrative application of this trend lies in solving quadratic programming problems [13], where RL techniques have been harnessed to fine-tune parameters and expedite convergence. While existing research predominantly focuses on augmenting convergence speed, there remains a relative dearth of exploration concerning the optimization of objective values. Notably, some alternative methods have leveraged RL for solving QCQP problems. For instance, [14] showcases RL's effectiveness in tackling constrained 0-1 quadratic programming problems. However, few approaches address the broader domain of utilizing RL techniques for tackling general QCQP problems with dynamic settings.

In the context of QCQP with dynamic settings, where the problem parameters exhibit variability, a good initial guess is crucial to ensure convergence to a desirable solution. However, the current literature lacks a comprehensive investigation into the generation of high-quality initial guesses through the use of RL techniques. To address this gap, we propose to apply RL to generate a good initial guess for the employed optimization algorithm, thus improving both convergence performance and optimality of the solution. Unlike using RL alone to search for an optimal solution, which can suffer from low precision, the combined approach of RL and QCQP

This research is supported in part by NSF grant CPS-2201568.
Chaoying Pei, Zhi Xu, Sixiong You and Ran Dai are with the School of Aeronautic and Astronautics, Purdue University, West Lafayette, IN, 47907, USA. Jeffrey Sun is with West Lafayette High School. Emails: `peic@purdue.edu`, `xu1710@purdue.edu`, `you84@purdue.edu`, and `randai@purdue.edu`

algorithm provides an initial guess that is less sensitive to precision while accelerating the convergence of the iterative algorithm.

In conclusion, this paper presents two significant contributions. Firstly, it proposes a novel framework for enhancing the performance of iterative algorithms for QCQP by generating high-quality initial guesses. Secondly, it demonstrates the effectiveness of integrating RL with QCQP to efficiently produce initial guesses for enhanced optimality and convergence speed. Empirical evaluations of the proposed approach show substantial improvements in objective values and faster convergence when compared to the existing iterative algorithms.

The structure of this paper is organized as follows. In Section II, we provide the problem formulation for QCQP with dynamic parameters and its equivalent reformulation. Section III elaboratefs on the proposed RL-based method for generating high-quality initial guesses. In Section IV, we present a simulation example to demonstrate the effectiveness of our proposed approach, and analyze the results. Finally, we summarize and conclude the paper in Section V.

## II. Problem Formulation

In this section, we discuss the problem formulation of a general inhomogeneous QCQP and its transformation into a homogeneous QCQP and rank-1 constrained semidefinite programming (SDP) problem. The QCQP problem involves minimizing a quadratic objective function subject to a set of quadratic constraints. A general inhomogeneous QCQP problem can be defined as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T\mathbf{A}_0\mathbf{x} + \mathbf{d}_0^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{d}_i^T\mathbf{x} \le b_i, \quad i = 1,\dots,m \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the unknown variable vector, $\mathbf{A}_i \in \mathbb{R}^{d\times d}$, $i = 0,\dots,m$, are symmetric matrices, $d_i \in \mathbb{R}^d$, $i = 0,\dots,m$, are parameter vectors and $b_i \in \mathbb{R}$, $i = 1,\dots,m$, are scalars.

An inhomogeneous QCQP can be equivalently converted into a homogeneous one by introducing an auxiliary variable $\mathbf{z} = \begin{bmatrix}\mathbf{x}^T & 1\end{bmatrix}^T \in \mathbb{R}^{d+1}$ and incorporating it into the problem formulation in (1), leading to a homogeneous QCQP, written as

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{z}^T\bar{\mathbf{A}}_0\mathbf{z} \\ \text{s.t.} \quad & \mathbf{z}^T\bar{\mathbf{A}}_i\mathbf{z} \le 0, \quad i = 1,\dots,m \end{aligned} \tag{2}$$

where $\bar{\mathbf{A}}_0 = \begin{bmatrix}\mathbf{A}_0 & \frac{1}{2}\mathbf{d}_0 \\ \frac{1}{2}\mathbf{d}_0^T & 0\end{bmatrix}$, and $\bar{\mathbf{A}}_i = \begin{bmatrix}\mathbf{A}_i & \frac{1}{2}\mathbf{d}_i \\ \frac{1}{2}\mathbf{d}_i^T & -b_i\end{bmatrix}$, $i = 1,\cdots,m$.

Considering variability in the problem objective and constraints, we introduce an variability matrix $\mathbf{E_i}$, $i = 0,\dots,m$, into the QCQP formulation, resulting in the following optimization problem:

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{z}^T(\bar{\mathbf{A}}_0 + \mathbf{E}_0)\mathbf{z} \\ \text{s.t.} \quad & \mathbf{z}^T(\bar{\mathbf{A}}_i + \mathbf{E}_i)\mathbf{z} \le 0, \quad i = 1,\dots,m \end{aligned} \tag{3}$$

where the matrix $\mathbf{E}_0$ models the dynamic parameters in the objective function, and $\mathbf{E}_i$, $i = 1,\dots,m$ represents variability in the constraints.

The next step in our approach is to transform the QCQP problem with dynamic parameters into a rank-1 constrained SDP problem by introducing an unknown matrix $\mathbf{Y} = \mathbf{z}\mathbf{z}^T \in \mathbb{R}^{n\times n}$ with $n = d+1$. To simplify the notation, we introduce $\mathbf{Q}_0 = \bar{\mathbf{A}} + \mathbf{E}_0$ and $\mathbf{Q}_i = \bar{\mathbf{A}}_i + \mathbf{E}_i$. Then the homogeneous QCQP in (3) is then rewritten as

$$\begin{aligned} \min_{\mathbf{Y}} \quad & \operatorname{tr}(\mathbf{Q}_0\mathbf{Y}) \\ \text{s.t.} \quad & \operatorname{tr}(\mathbf{Q}_i\mathbf{Y}) \le 0, \quad i = 1,\dots,m \\ & \mathbf{Y} \succeq 0 \quad \operatorname{rank}(\mathbf{Y}) = 1 \end{aligned} \tag{4}$$

where tr$(\cdot)$ denotes the trace of a matrix, and $\mathbf{Y} \succeq 0$ indicates that $\mathbf{Y}$ is a positive semidefinite matrix. The rank-1 constraint on $\mathbf{Y}$ implies that there exists a vector $\mathbf{z} \in \mathbb{R}^n$ such that $\mathbf{Y} = \mathbf{z}\mathbf{z}^T$. This rank-1 constrained SDP problem allows for the application of various relaxation techniques to find a relaxed solution of the original QCQP, such as SDP relaxation or spectral relaxation.

## III. RL-Guidded QCQP Algorithm

This section presents a framework that utilizes RL to enhance the computational performance of iterative algorithms for solving QCQP problems with dynamic parameters. The primary solver used in this framework is the Iterative Rank Minimization (IRM) algorithm developed in [15], [16]. We introduce RL to generate high-quality initial guesses for IRM, and analyze the guaranteed improvements in terms of convergence and objective value resulting from this approach. Finally, we present the framework of our proposed algorithm, which combines the IRM solver with the RL-guided initialization strategy.

### *A. Iterative Rank Minimization Algorithm*

IRM was developed to solve general QCQPs with guaranteed local convergence [15], [16]. Specifically, for a symmetric positive semi-definite matrix $\mathbf{Y} \in \mathbb{S}^n$, if its rank is one, it has at least $n-1$ zero eigenvalues. By sorting the eigenvalues of $\mathbf{Y}$, we obtain the matrix $\mathbf{V} \in \mathbb{R}^{n\times(n-1)}$ consisting of the eigenvectors of the $n-1$ smallest eigenvalues of $\mathbf{Y}$. The IRM algorithm aims to gradually reduce the $n-1$ smallest eigenvalues of $\mathbf{Y}$ to zero through iterations while minimizing the cost function. More details about the IRM algorithm can be found in [15], [16].

When applying the IRM algorithm to solve the QCQP in (4), at the $l$th step, the iterative optimization problem is formulated as

$$\min_{\mathbf{Y}^l, e^l} \quad J = \operatorname{tr}(\mathbf{Q}_0\mathbf{Y}) + \omega^l e^l \tag{5a}$$

$$\text{s.t.} \quad \operatorname{tr}(\mathbf{Q}_i\mathbf{Y}) \le 0, \quad i = 1,\dots,m \tag{5b}$$

$$e^l\mathbf{I}_{n-1} - (\mathbf{V}^{l-1})^T\mathbf{Y}^l\mathbf{V}^{l-1} \succeq 0, \tag{5c}$$

$$e^l \le e^{l-1}, \quad \mathbf{Y}^l \in \mathbb{S}^n+, \tag{5d}$$

where $\mathbf{Y}^l$ is the matrix to be optimized at the $l$th iteration of the IRM method and $\mathbf{V}^{l-1}$ are the eigenvectors corresponding to the $n-1$ smallest eigenvalues of $\mathbf{Y}^{l-1}$, which is solved at iteration $l-1$. Constraint (5c) provides an upper bound for $\mathbf{V}^T\mathbf{Y}\mathbf{V}$, which is $e^l\mathbf{I}_{n-1}$. Moreover, $\omega^l$ is a weighting factor used to balance the trade-off between minimizing the cost function and satisfying the rank-1 constraint in the IRM algorithm. Typically, $\omega^l$ is chosen to escalate exponentially across iterations. This strategic choice of $\omega^l$ allows us to strike a balance between enhancing convergence speed and optimizing the objective value. By iteratively solving the convex optimization problem (5) and minimizing $e^l$ as a penalty term of the objective function, we approach the rank constraint gradually. We assume the iteration converges when $e^l \le \epsilon$, where $\epsilon$ is a sufficiently small value used for the convergence criterion.

In the first iteration, the initial guess of $\mathbf{Y}^0$ is usually generated by ignoring the rank-1 constraint in (4). Then a relaxed solution of (4) by solving an SDP problem is obtained and used as the initial guess $\mathbf{Y}^0$. Its corresponding eigenvectors $\mathbf{V}^0$ are then used for the first iteration in (5). The convergence rate of the IRM algorithm can be improved significantly if the algorithm starts with a good initial guess of the unknown $\mathbf{Y}$. Specifically, if the initial guess satisfies certain conditions, the algorithm can converge within a single step. In this paper, we present the conditions under which the algorithm can achieve this rapid convergence and provide rigorous proof of this theorem.

***Definition** 3.1:* Let $[\mathbf{v}_1, ..., \mathbf{v}_n] \in \mathbb{R}^{n\times n}$ be the eigenvectors of an initial guess $\mathbf{Y}^0 \in \mathbb{R}^{n\times n}$, corresponding to its eigenvalues $[\lambda_1, ..., \lambda_n]$, where $\lambda_1 \le \lambda_2 \le ... \le \lambda_n$. Similarly, let $[\mathbf{u}_1, ..., \mathbf{u}_n] \in \mathbb{R}^{n\times n}$ be the eigenvectors of a local optimal solution $\mathbf{Y}^* \in \mathbb{R}^{n\times n}$ of problem (4), corresponding to its eigenvalues $[\beta_1, ..., \beta_n]$, where $\beta_1 \le \beta_2 \le ... \le \beta_n$.

***Proposition** 3.2:* (Proposition 10 in [15]) When $\lim_{l\to\infty} \mathbf{e}^l = \mathbf{0}$, and each sequential problem (5) has only one optimum, the corresponding solution $\mathbf{Y}^l = \mathbf{Y}^*$ is the local optimum of (4).

***Proposition** 3.3:* If an initial guess $\mathbf{Y}^0$ leads to a solution of (5) that satisfies the convergence criterion $e^l \le \epsilon$ at $l = 1$, there exists a local optimal solution $\mathbf{Y}^*$ that shares the same eigenvectors corresponding to the $n-1$ smallest eigenvalues of $\mathbf{Y}^0$, i.e., $\mathbf{v}_1 = \mathbf{u}_1, \mathbf{v}_2 = \mathbf{u}_2, ..., \mathbf{v}_{n-1} = \mathbf{u}_{n-1}$.

*Proof:* Assume that the initial guess $\mathbf{Y}^0$ leads to a solution of (5) satisfying $e^l \le \epsilon$ at $l = 1$, then in the first iteration of the IRM algorithm, we have

$$\min_{\mathbf{Y}^1, e^1} \quad J = \text{tr}\left(\mathbf{Q}_0 \mathbf{Y}^1\right) + \omega^1 e^1 \tag{6a}$$

$$\text{s.t.} \quad \text{tr}\left(\mathbf{Q}_i \mathbf{Y}^1\right) \le 0, \quad i = 1, \dots, m \tag{6b}$$

$$e^1 \mathbf{I}_{n-1} - (\mathbf{V}^0)^T \mathbf{Y}^1 \mathbf{V}^0 \succeq 0, \tag{6c}$$

$$e^1 \le e^{1-1}, \quad \mathbf{Y}^1 \in \mathbb{S}^n +, \tag{6d}$$

where $\mathbf{Y}^1$ is the optimized matrix at $l = 1$, and $\mathbf{V}^0$ are the eigenvector matrix corresponding to the $n-1$ smallest eigenvalues of $\mathbf{Y}^0$. By Proposition 3.2, the IRM algorithm guarantees convergence to a local optimum of the original problem (4). Therefore, the optimal solution $\mathbf{Y}^*$ obtained by solving problem (6) is a local optimum of (4).

Since the convergence criterion for the IRM algorithm is $e^l < \epsilon$, the rank constraint in (6) is satisfied, i.e., $(\mathbf{V}^0)^T \mathbf{Y}^* \mathbf{V}^0 \le \epsilon \mathbf{I}_{n-1}$. Because $\epsilon$ is sufficiently small and $\mathbf{Y}^*$ is a positive semidefinite matrix, we have $(\mathbf{V}^0)^T \mathbf{Y}^* \mathbf{V}^0 \to (\mathbf{V}^*)^T \mathbf{Y}^* \mathbf{V}^* = \mathbf{0}$. Hence, $\mathbf{Y}^*$ shares the same eigenvectors as $\mathbf{Y}^0$, up to a permutation. Since the eigenvectors are orthonormal, the permutation does not affect the orthogonality of the eigenvectors. Thus, we have shown that the existence of a local optimum $\mathbf{Y}^*$ that satisfies $\mathbf{v}_1 = \mathbf{u}_1, \mathbf{v}_2 = \mathbf{u}_2, ..., \mathbf{v}_{n-1} = \mathbf{u}_{n-1}$ is a necessary consequence of $\mathbf{Y}^0$ leading the IRM algorithm to converge within one step. ∎

To validate Proposition 3.3, Section IV presents simulation results of a QCQP example. Particularly, we compare the eigenvectors of the initial guess $Y^0$ obtained from RL and the corresponding eigenvectors from its global optimal solution $Y^*$. The comparative results demonstrate the practical relevance of this proposition in solving QCQP problems.

## *B. Reinforcement Learning*

A good initial guess is crucial for enhancing the convergence and optimality of the IRM algorithm when dealing with QCQP problems with dynamic parameters. However, there is currently no universal method for finding a good initial guess. To address this issue, we propose using model-free deep reinforcement learning to automatically fine-tune a neural network for generating an optimized initial guess for the IRM algorithm. While RL employing solvers like Markov decision processes adds computational complexity, this complexity is managed offline during training. Once trained, the model offers efficient online deployment, optimizing memory and time usage.

The basic framework of RL includes two major components, agent and environment. The agent will determine the policy of taking actions $\mathbf{a}_t$ for different input states $\mathbf{s}_t$, where $*_t$ means $*$ at stage $t$. The environment will give feedback for different input actions $\mathbf{a}_t$. Generally, the feedback includes the corresponding rewards $\mathbf{R}_t$ and next states $\mathbf{s}_{t+1}$. For the studied QCQP, we define state, action, and rewards as

$$\mathbf{a}_t = \mathbf{z}^0 \quad \mathbf{s}_t = \delta \quad \mathbf{R}_t = h(\mathbf{a}_t, \mathbf{s}_t) \tag{7}$$

where $\mathbf{Y}^0 = \mathbf{z}^0(\mathbf{z}^0)^T$ is the initial guess for IRM. $\delta$ represents the variability of parameters. In addition, as only one step is required to find the initial guess for IRM, the next stage will be the terminal stage, the corresponding $\mathbf{R}_t$ can be accurately obtained without estimating $\mathbf{R}_{t+1}$, which makes the learning process more effective.

According to the type of action space, the existing RL algorithms can be divided into two categories. For the discrete action space, examples of developed methods include Q-learning [17], State-action-reward-state-action [18], and deep Q network (DQN) [19]. For continuous action space, examples include asynchronous advantage actor-critic algorithm, deep deterministic policy gradient (DDPG), proximal policy optimization, and twin-delayed deep deterministic (TD3) policy gradient. Since we focus on general QCQP problems with continuous action spaces, TD3 is introduced in this context.

*1) Introduction of TD3 Algorithm:* TD3 is an extension of DDPG, which can be regarded as the combination of actor-critic algorithm and DQN, where the neural networks can be divided into two levels, local network and target network. In each episode, the target network is softly updated, which means it will be updated slowly, and the local network will copy the target network after a fixed number of episodes. The actor network takes in observation states and outputs actions, while the critic network takes in both the state and action and outputs a reward value. The local and target networks each contain an actor and two critic networks. Then, the target critic and actor network will update "softly" by minimizing the TD error between local-network and target-network. Note that, updating "softly" means it will update these networks via discounting between the old and new target network. After that, it will transfer its network to the local network. In the following subsection, more details of implementing RL in finding initial guess are described below.

*2) Initial Guess Generation via Reinforcement Learning:* In this section, we will introduce the implementation details of using RL to generate a high-performing initial guess for the IRM algorithm. In order to simultaneously reduce the number of IRM iterations and improve the objective value, we formulate the initial guess generation problem as a suitable RL problem. This involves identifying the environment, choosing the appropriate state space, action space, and reward function to maximize the cumulative reward collected by the RL agent.

**Environment**: In our RL-based approach, the process of solving QCQP problem is modeled as an environment. The dynamic parameters in the problem setting are observed by the RL agent in each episode, and the agent selects an initial guess as the action while also receiving a reward based on the convergence iteration and the objective value. The interaction between the RL agent and the IRM solver is illustrated in Fig. 1.

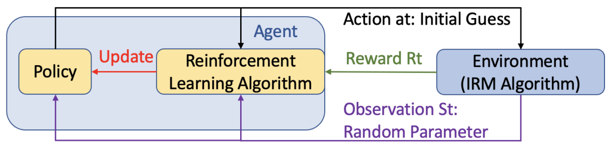

Action at: Initial Guess
Agent
Policy
Update
Reinforcement Learning Algorithm
Reward Rt
Environment (IRM Algorithm)
Observation St: Random Parameter

Fig. 1: IRM and RL Agent Interaction

**State**: In the proposed RL process, each episode involves a single step, and the state at each step is represented by the dynamic parameters in the QCQP problem, which can be expressed as $\delta = [\delta_{k1}, \delta_{k2}, ..., \delta_{kt}]$, where $\delta_{kj}$, $j = 1, ..., t$, are the non-zero entries in the dynamic matrices $E_j, j = 0, ..., m$ in (3).

**Action**: To comply with the requirements of the IRM algorithm, an $n \times n$ initial guess matrix $\mathbf{Y}^0$ is needed, as shown in (4). However, in order to reduce the dimension of actions and improve efficiency, we represent the action space as a vector $\mathbf{z}^0 \in \mathbb{R}^n$, and obtain $\mathbf{Y}^0$ by computing the outer product of $\mathbf{z}^0$, i.e., $\mathbf{Y}^0 = \mathbf{z}^0(\mathbf{z}^0)^T$. This approach allows us to significantly reduce the dimension of the action space without compromising the quality of the generated initial guess. Furthermore, the rank-1 constraint is naturally satisfied by this initial guess construction. The action space provides a wide range of possible initial guesses which can be used by the IRM algorithm to solve the QCQP problem. However, to increase the efficiency of the RL-IRM algorithm, we restrict the action space to initial guesses that are close to the feasible region of the problem. This ensures that the agent selects initial guesses that are likely to converge to the optimal solution with fewer IRM iterations.

**Reward Function**: Our reward function is designed to encourage the IRM to converge quickly to the optimal solution. Specifically,

$$\text{Reward} = \begin{cases} \gamma - \text{iter} & \text{if obj} > \sigma \\ 2\gamma - \text{iter} & \text{if obj} = \sigma \\ 3\gamma - \text{iter} & \text{if obj} < \sigma \end{cases} \tag{8}$$

where obj is the objective value, iter is the number of iterations taken to converge, $\sigma$ is the estimated optimal solution obtained by solving the dynamic QCQP problem using SDP initial guess. The weight $\gamma$ is a positive scalar that can be adjusted to balance the importance of convergence speed and objective value reduction in the reward function. A larger $\gamma$ prioritizes objective value, while a smaller value emphasizes convergence speed. When tuning $\gamma$, estimating the iterations helps ensure that $\gamma$ and the iteration count (iter) are within the same order of magnitude. By providing different reward values based on the objective value, our reward function incentivizes the IRM to converge quickly to the optimal solution. Then, we can have the following proposition.

***Proposition** 3.4:* For the studied QCQP problem (4), there exist some feasible initial guesses $\mathbf{z}^0$, which can achieve Reward $= 2\gamma - 1$ or $= 3\gamma - 1$.

*Proof:* Assume the global optimal solution of the studied problem (4) is $\mathbf{Y}^{opt}$, and the solution from IRM with the SDP initial guess to be $\mathbf{Y}^{sdp}$. Then, we have

$$\text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{opt}\right) \leq \text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{sdp}\right) = \sigma. \tag{9}$$

Let $\mathbf{Y}^{opt} = \mathbf{z}^0{\mathbf{z}^0}^T$, and the solution for the first iteration of the IRM algorithm (6) be $(\mathbf{Y}^{1*}, e^{1*})$. For the convex subproblem (6), the pair $(\mathbf{Y}^{opt}, 0)$ satisfying all constraints in (6) is its feasible solution. Then, the optimal solution pair $(\mathbf{Y}^{1*}, e^{1*})$ and feasible solution pair $(\mathbf{Y}^{opt}, 0)$ for the convex problem (6) satisfy

$$\text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{1*}\right) + \omega^1 e^{1*} \leq \text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{opt}\right), \tag{10}$$

where $\omega^1 e^{1*} \geq 0$. Assume IRM converges at $l$th iteration, then the optimal solution of (4), $\mathbf{Y}^{opt}$, should also be the optimal solution of (5) at the $l$th iteration, with $\mathbf{V}^l$ being its eigenvectors corresponding to its zero eigenvalues, which means

$$\text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{opt}\right) \leq \text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{1*}\right) + \omega^l e^{1*}. \tag{11}$$

Let $\omega^l = \omega^1$, combining the above (10) and (11), we can have

$$\text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{opt}\right) = \text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{1*}\right) + \omega^l e^{1*}. \tag{12}$$

According to Proposition 3.2, as there is only one optimum for problem (5), we can claim $\mathbf{Y}^{opt} = \mathbf{Y}^{1*}$, which means IRM algorithm can coverge to $\mathbf{Y}^{opt}$ within one step with initial guess $\mathbf{z}^0$ obtained from $\mathbf{Y}^{opt} = \mathbf{z}^0{\mathbf{z}^0}^T$. Next, according to (11), the objective value satisfies obj $= \text{tr}\left(\mathbf{Q}_0\mathbf{Y}^{opt}\right) \leq \sigma$. And according to (8), when $\mathbf{Y}^{opt} = \mathbf{z}^0{\mathbf{z}^0}^T$, the corresponding reward function can be

$$\text{Reward} = \begin{cases} 2\gamma - 1 & \text{if obj} = \sigma \\ 3\gamma - 1 & \text{if obj} < \sigma \end{cases} \tag{13}$$

Then we show the initial guesses $\mathbf{z}^0$ exists, which can achieve Reward $= 2\gamma - 1$ or $= 3\gamma - 1$. It completes the proof. ■

***Corollary** 3.5:* With sufficient exploration space for the designed RL algorithm to learn and generate an appropriate initial guess $\mathbf{z}^0$, it can lead to accelerated convergence and improved objective value for the IRM algorithm, compared to using an initial guess obtained from the SDP method.

*Proof:* Assume the designed RL algorithm has sufficient exploration space to learn and generate an appropriate initial guess. Then, proposition 3.4 suggests that the RL can identify some feasible initial guesses resulting in a Reward of either $2\gamma - 1$ or $3\gamma - 1$. This implies that using the RL-generated initial guess, $\mathbf{z}^0$ can lead to faster convergence and better objective value for the IRM algorithm in just one iteration, compared to using the SDP initial guess. In other words, IRM with the RL-generated initial guess can accelerate convergence and improve the objective value for the IRM algorithm, compared to using an initial guess obtained from the SDP method. Consequently, we have completed the proof. ■

### C. Iterative Algorithm with Reinforcement Learning

Via integrating the RL with IRM, the iterative algorithm based on RL is obtained. The flowchart of the proposed algorithm is shown in Table. I. The proposed algorithm consists of two main phases: offline training and online optimization. During the offline training phase, the algorithm employs RL to establish the relationship between dynamic parameters and the initial guess of IRM. This phase involves five steps. First, a suitable reward function is defined. Second, actor and critic neural networks are designed and initialized with dynamic weights. Third, RL is performed with carefully selected hyperparameters, such as learning rate and exploration noise. Fourth, the TD3 agent is trained offline using the learning results. Finally, the algorithm is ready for the online optimization phase.

For the online optimization phase, the algorithm takes inputs from values of the dynamic parameters $\delta$. The goal is to determine the initial matrix $\mathbf{Y}^0$. By leveraging the well-trained TD3 agent and the given variability of parameters $\sigma$, an initial guess $\mathbf{y}^0$ can be obtained. Subsequently, problem (5) is solved iteratively using a convex solver. If the error $e^l \leq \epsilon$, $\mathbf{Y}^l$ is the final output. Otherwise, the algorithm updates $\mathbf{V}^l$ based on $\mathbf{Y}^l$ and repeats the loop until it converges or reaches the maximum iteration.

## IV. Simulation Results

In this section, we present the simulation results of our proposed RL-guided initial guess generator for solving a small-scale QCQP problem with dynamic parameters. With the analytical global optimum solution available, this problem provides a comprehensive evaluation of claimed advantages of the proposed method. The example QCQP problem is formulated as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T Q_0 \mathbf{x} \\ \text{s.t.} \quad & 0.5x_2 = x_1^2 \quad x_1 + x_2 \leq 2.5 \quad -x_1 + x_2 \leq 2 \end{aligned} \tag{14}$$

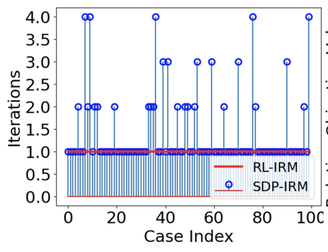
TABLE I: Flowchart of RL-guided iterative algorithm

```
Algorithm: Iterative Algorithm based on RL
Offline Training Phase:
1) Develop a proper reward function.
2) Design and initialize actor and critic neural networks with
dynamic weights.
3) Set hyperparameters such as learning rate and exploration noise.
4) Train the TD3 agent using the reinforcement learning results.
Online Optimization Phase:
Input: problem variability $\delta$
Output: Unknown matrix $\mathbf{Y}$
Initialization: Generate the initial guess $z^0$ via the well-trained
TD3 agent, and calculate the eigenvector $\mathbf{V}^0$ based on $z^0 {z^0}^T$.
begin
1) Set $l = 1$,
2) for $l = 1, 2, ..., \hat{l}_{\max}$
3)     Solve (5) to obtain solution $\mathbf{Y}^l, e^l$,
4)     If $e^l \leq \epsilon$, break; else, Update $\mathbf{V}^l$ from eigenvectors of $\mathbf{Y}^l$
5)     $l = l + 1$
end
```

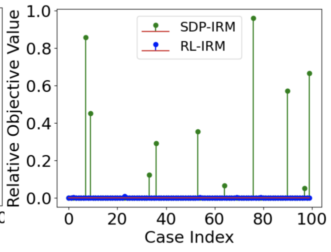
where $\mathbf{x} = [x_1, x_2]^T$ is the unknown vector, and $Q_0 = \begin{bmatrix} 0 & r_1 \\ r_1 & r_2 \end{bmatrix}$ is the objective matrix with dynamic parameters $r_1, r_2$ drawn independently from the uniform distribution on the interval $[-1, 1]$.

Traditionally, an initial guess for the IRM algorithm is generated using SDP relaxation by ignoring the rank-1 constraint on $\mathbf{X}$. Here we use SDP-IRM with SDP initial guess as a benchmark to compare the performance of the proposed RL-IRM approach with RL-guided initial guess. Due to the simplicity of problem (14), the number of iterations is anticipated to be around 3. Therefore, we select $\gamma$ in the reward function (8) as 1. Additionally, we set the weighing factor $\omega^l$ in (5) as $\omega^l = \alpha\omega^{l-1}$, where $\omega^0 = 1$ and $\alpha = 2$.

We present the results of 100 test cases solved using both algorithms, as depicted in Fig. 2. We observe that the RL-IRM algorithm consistently converges within one single iteration for all 100 cases, whereas SDP-IRM requires multiple iterations to converge for 26 of the 100 cases. In terms of the objective value, RL-IRM is able to converge to the global optimum in all cases with an accuracy of 0.01, as demonstrated in Fig. 2b. Furthermore, RL-IRM achieves a significantly lower objective value than SDP-IRM in 10 out of the 100 cases, with a reduction of more than 0.05. In conclusion, RL-IRM has either enhanced or achieved the same convergence rate and optimality compared to the SDP-IRM approach.



(a) Number of convergence iterations



(b) Objective Deviation from Global Minima

Fig. 2: Comparison of Performance between RL and SDP

To provide a more comprehensive comparison between RL-IRM and SDP-IRM, we analyzed the convergence iterations and objective values of 10,000 cases. RL-IRM achieves an average of 1.0106 convergence iterations, outperforming SDP-IRM's 1.3370 iterations, while also exhibiting an average optimization time of 0.0471 compared to SDP-IRM's 0.0327. Moreover, in 9970 cases, RL-IRM converged to the global optimum solution, while in 9260 cases, SDP-IRM converged to the global optimum. The mean objective value of RL-IRM was -0.7389, compared to -0.7093 for SDP-IRM. Overall, these results suggest that RL-IRM outperforms SDP-IRM in terms of both convergence speed and objective value. In addition, RL-IRM's memory-efficient nature sets it apart from the memory-intensive SDP-IRM. By employing a pre-optimized model for initial guess generation, RL-IRM effectively addresses memory constraints, making it particularly well-suited for on-board optimization in scenarios where memory limitations are pivotal.

Next, we will delve into a detailed analysis of specific cases and elaborate on why RL-IRM outperforms SDP-IRM in these instances. Since RL-IRM is able to reach the optimal solution within a single iteration in all cases, our analysis will mainly focus on cases where the SDP initial guess requires multiple iterations to converge. We discovered that in these cases, the objective matrices $Q_0$ are all in a certain range, resulting in the SDP relaxation problem consistently yielding the same initial guess. This initial guess, along with the feasible region of the SDP relaxation problem, are shown in Fig. 3, providing an intuitive representation of the constraints that need to be satisfied.

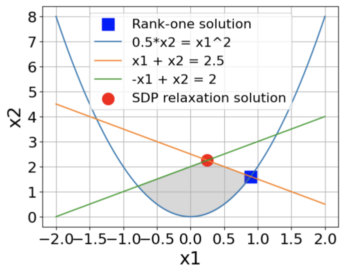


Fig. 3: Feasible region and SDP initial guess

According to Proposition 3.3, if the initial guess $\mathbf{X}^0$ leads the IRM to converge within one step, there exists a local optimal solution $\mathbf{X}^*$ that shares the same eigenvectors corresponding to the $n-1$ smallest eigenvalues of $\mathbf{X}^0$. For this particular problem, we have $n = 2$, thus we can assume that $\mathbf{v} = [v_1, v_2]$ satisfies this condition and serves as the eigenvector of both the optimal solution $X^*$ and the initial matrix $X^0 = \begin{bmatrix} x_1^2 & x_1x_2 \\ x_1x_2 & x_2^2 \end{bmatrix}$. Consequently, we can express $Xv = \lambda v$ as $\begin{bmatrix} x_1^2 & x_1x_2 \\ x_1x_2 & x_2^2 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \lambda v_1 \\ \lambda v_2 \end{bmatrix}$ Expanding this expression yields the following equations

$$x_1^2 v_1 + x_1x_2v_2 = \lambda v_1 \quad x_1x_2v_1 + x_2^2 v_2 = \lambda v_2 \tag{15}$$

Rearranging (15) gives

$$x_1^2 + (v_2/v_1 - v_1/v_2)x_1x_2 - x_2^2 = 0. \tag{16}$$

Substituting $a = v_2/v_1 - v_1/v_2$, we can solve for $x_2$ in (16) as follows

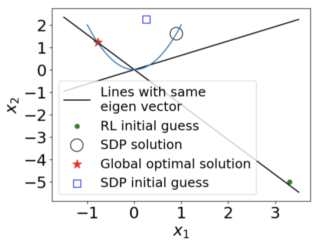
$$x_2^2 = x_1^2 + ax_1x_2 \Rightarrow x_2^2 - ax_1x_2 - x_1^2 = 0 \tag{17}$$

$$\Rightarrow x_2 = \frac{ax_1 \pm \sqrt{(ax_1)^2 + 4x_1^2}}{2}. \tag{18}$$

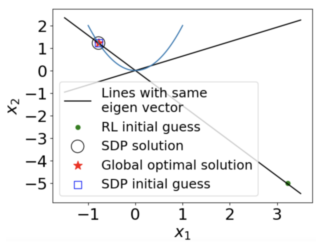
After solving the equation, it becomes evident that the plot of $x_2 \sim x_1$ results in two intersecting lines. To illustrate this concept, we consider a specific case and plot the two lines on a graph in Fig. 4a. In this case, RL-IRM starts at the initial guess represented by the green dot and converges within a single iteration to the global optimal point marked by the red star. On the other hand, SDP-IRM starts at the initial guess in the blue dot and converges to a local optimum marked by the black circle.

Upon examining Fig. 4a, we observe that while the initial guess from RL-IRM may appear further from the global optimal solution than the initial guess from SDP relaxation, it lies on the same eigenvector lines that we previously discussed in (18). This implies that the initial guess from RL-IRM and the optimal solution share the same eigenvector associated with the smaller eigenvalue. Conversely, the SDP-IRM initial guess appears closer to the global optimal solution, yet it is not on the same eigenvector line. This observation provides an explanation for why the RL-IRM initial guess converges within a single iteration to the global optimum, while the SDP-IRM initial guess requires more iterations and converges to a local optimum.

To provide a comparison, in Fig. 4b, we have presented a case where RL-IRM and SDP-IRM perform equally well, as both converge to the global optimum within a single iteration. In this specific case, SDP-IRM obtained an initial guess that was very close to the global optimal solution, and therefore, it also lay on the eigenvector lines.



(a) SDP/RL initial guesses with different solutions



(b) SDP/RL initial guesses with a same solution

Fig. 4: Initial guesses from SDP/RL and eigenvector lines that leads to samd/different solution

Through our analysis of these cases, we can deduce that RL has learned a policy that can produce initial guesses with suitable eigenvectors, derived from the dynamic settings in the problem. This information cannot be obtained by solving the SDP relaxation problem, highlighting the advantage of using RL over SDP for initial guesses. Hence, we conclude that RL is a more effective approach in generating initial guesses for IRM, which can converge to a better optimum with an enhanced convergence rate.

## V. Conclusion

This paper proposes a novel approach that leverages reinforcement learning (RL) to generate high-performing initial guesses for iterative algorithms to solve general quadratically constrained quadratic programming (QCQP) problems with dynamic parameters. The proposed approach is evaluated by applying it to the iterative rank minimization (IRM) algorithm, and the results demonstrate its effectiveness in enhancing convergence and optimality when solving QCQP problems. The RL-guided IRM algorithm yields high-quality solutions, as evidenced by significantly reduced objective values and faster convergence compared to the original IRM algorithm. Additionally, the approach shows promise in reducing the dependence on SDP relaxation, which is widely used in solving QCQP problems. Overall, the proposed approach provides a promising solution for solving QCQP problems with dynamic parameters, with potential applications in various fields such as finance, engineering, and machine learning.

## References

[1] S. Bose, D. F. Gayme, K. M. Chandy, and S. H. Low, "Quadratically constrained quadratic programs on acyclic graphs with application to power flow," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 3, pp. 278–287, 2015.

[2] T. Ding, R. Bo, F. Li, and H. Sun, "A bi-level branch and bound method for economic dispatch with disjoint prohibited zones considering network losses," *IEEE Transactions on Power Systems*, vol. 30, no. 6, pp. 2841–2855, 2014.

[3] A. Pichler, S. Poledna, and S. Thurner, "Systemic risk-efficient asset allocations: Minimization of systemic risk as a network optimization problem," *Journal of Financial Stability*, vol. 52, p. 100809, 2021.

[4] X. Bao, N. V. Sahinidis, and M. Tawarmalani, "Semidefinite relaxations for quadratically constrained quadratic programming: A review and comparisons," *Mathematical programming*, vol. 129, pp. 129–157, 2011.

[5] A. S. Bedi, K. Rajawat, V. Aggarwal, and A. Koppel, "Escaping saddle points for successive convex approximation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 307–321, 2021.

[6] A. Mokhtari and A. Koppel, "High-dimensional nonconvex stochastic optimization by doubly stochastic successive convex approximation," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6287–6302, 2020.

[7] M. Grant and S. Boyd, "Cvx: Matlab software for disciplined convex programming, version 2.1," 2014.

[8] M. Andersen, J. Dahl, Z. Liu, L. Vandenberghe, S. Sra, S. Nowozin, and S. Wright, "Interior-point methods for large-scale cone programming," *Optimization for machine learning*, vol. 5583, 2011.

[9] N. K. Jha, "Low power system scheduling and synthesis," in *IEEE/ACM International Conference on Computer Aided Design. ICCAD 2001. IEEE/ACM Digest of Technical Papers (Cat. No. 01CH37281)*. IEEE, 2001, pp. 259–263.

[10] H. Li and C. L. Swartz, "Approximation techniques for dynamic real-time optimization (drto) of distributed mpc systems," *Computers & Chemical Engineering*, vol. 118, pp. 195–209, 2018.

[11] J.-H. Hours and C. N. Jones, "A parametric nonconvex decomposition algorithm for real-time and distributed nmpc," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 287–302, 2015.

[12] G. You, "Sustainable vehicle routing problem on real-time roads: The restrictive inheritance-based heuristic algorithm," *Sustainable Cities and Society*, vol. 79, p. 103682, 2022.

[13] J. Ichnowski, P. Jain, B. Stellato, G. Banjac, M. Luo, F. Borrelli, J. E. Gonzalez, I. Stoica, and K. Goldberg, "Accelerating quadratic optimization with reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 043–21 055, 2021.

[14] S. Gu and Y. Zhuang, "Method for solving constrained 0-1 quadratic programming problems based on pointer network and reinforcement learning," *Neural Computing and Applications*, vol. 35, no. 14, pp. 9973–9993, 2023.

[15] C. Sun and R. Dai, "Rank-constrained optimization and its applications," *Automatica*, vol. 82, pp. 128–136, 2017.

[16] ——, "An iterative rank penalty method for nonconvex quadratically constrained quadratic programs," *SIAM Journal on Control and Optimization*, vol. 57, no. 6, pp. 3749–3766, 2019.

[17] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992698

[18] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering Cambridge, UK, 1994, vol. 37.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.