An Operator Splitting-Based Interior-Point Method for Conic Linear Programming

Han-Tao Nie[1], Ji-Rui Ma[1*], Zai-Wen Wen[1] and Fan Zhang[2]

[1]Beijing International Center for Mathematical Research, Peking University, Beijing 100871, China.
[2]Theoretical Research Lab, Huawei Technologies, Hong Kong SAR, China.

*Corresponding author. E-mail: majirui@pku.edu.cn;
Contributing authors: nht@pku.edu.cn; wenzw@pku.edu.cn;
zhang.fan2@huawei.com;

Abstract

We introduce the Douglas-Rachford splitting interior point method (DRSIP), a novel algorithm that combines operator splitting with second order approaches to solve conic linear programming problems. Our method originates from the fixed-point mapping derived from the Douglas-Rachford splitting method, which transforms the barrier penalized conic programming problem into a series of nonlinear equations. We apply the Newton-type method with a path-following scheme for acceleration. We prove the global convergence of DRSIP and establish its local quadratic convergence under the strict complementarity condition. Our numerical results showcase the algorithm's robustness, scalability, and adaptability, positioning DRSIP as a versatile and effective solution for large-scale conic programming challenges.

Keywords: conic programming, interior-point methods, Douglas-Rachford splitting, path-following

Mathematics Subject Classification: 49M15 , 65K05 , 90C25 , 90C51

1 Introduction

Given a linear operator $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$ with full row rank, a vector $b \in \mathbb{R}^m$, and a closed convex cone $\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_p$, where $\mathcal{K}_i$ is a irreducible closed convex cone in $\mathbb{R}^{n_i}$, we consider the following conic linear programming problem:

$$\begin{array}{ll} \min\limits_{x} & \langle c, x\rangle \\ \text{s.t.} & \mathcal{A}x = b, \\ & x \in \mathcal{K}, \end{array} \qquad \text{(P)} \qquad\qquad \begin{array}{ll} \max\limits_{y,s} & b^T y \\ \text{s.t.} & \mathcal{A}^* y + s = c, \\ & s \in \mathcal{K}^*. \end{array} \qquad \text{(D)}$$

Here $\mathcal{K}^*$ denotes the dual cone of $\mathcal{K}$, that is, $\mathcal{K}^* = \{s \in \mathbb{R}^n \mid \langle x, s\rangle \geq 0, \forall x \in \mathcal{K}\} = \mathcal{K}_1^* \times \mathcal{K}_2^* \times \cdots \times \mathcal{K}_p^*$. Some common cones include the positive orthant $\mathbb{R}^n_+ = \{x \in \mathbb{R}^n \mid x_i \geq 0, i = 1, 2, \cdots, n\}$, positive semidefinite matrices $\mathbb{S}^r_+ = \{X \in \mathbb{S}^r \mid X \succeq 0\}$, the second order cone $\mathcal{Q} = \{(x_0, \bar{x}) \in \mathbb{R}^r \mid x_0 \geq \|\bar{x}\|_2\}$, the rotated second order cone $\mathcal{Q}^r = \left\{(x_0, x_1, \bar{x}) \in \mathbb{R}^r \mid 2x_0x_1 \geq \|\bar{x}\|_2^2, x_0, x_1 \geq 0\right\}$, etc.

Conic linear programming problems, as formulated in (P) and (D), are pivotal in numerous applications ranging from machine learning and data science to operations research and financial engineering. The classic approach to solving conic programming is interior point methods (IPMs) [11] and its multiple variants. By utilizing the Jacobian of the KKT conditions or its approximation, IPM can take more informed steps towards the optimum. This typically results in superior convergence properties, albeit at a higher computational cost per iteration. When properly implemented, general conic linear programming problems (P) and (D) can be efficiently solved using both open-source and commercial software. Notable open-source mathematical programming software include SeDuMi [22], SDPT3 [24], ECOS [3] and commercial software include Gurobi [7], MOSEK [15], COPT [6], MindOpt [17], etc.

The field of optimization has also witnessed significant advancements in the development of algorithms via operator splitting methods, including alternating direction augmented Lagrangian methods (SDPAD) [25], ADMM-based interior point method (ABIP) [12], operator splitting with homogeneous self-dual embedding (SCS) [18]. They are particularly well-suited for large-scale problems where the computational cost of higher-order methods becomes prohibitive. However, first-order methods often suffer from slow convergence rates, especially when approaching the optimal solution.

On the other hand, recent research has focused on combining the strengths of operator splitting methods and second-order methods. Sopasakis et al. [21] and Themeles et al. [23] consider utilizing Newton-type methods to accelerate fixed point iterations or Krasnosel'skii-Mann-type iterations derived from operator splitting methods. Li et al. [13] apply a semismooth Newton-type method to solve a fixed-point mapping derived from the alternating direction method of multipliers (ADMM). Despite the successes of existing methods and solvers, the quest for more efficient and robust algorithms continues.

1.1 Contributions

This paper contributes to this ongoing effort by proposing a novel algorithm that combines the strengths of Douglas-Rachford Splitting with Newton-type optimization techniques. Our method aims to achieve a balance between the computational efficiency of operator splitting methods and the rapid convergence of second-order methods, thereby expanding the frontier of conic programming solvers. Our contributions are as follows:

1. **An efficient path-following algorithm**
   Starting from the fixed-point mapping associated with the DRS, we transform the original programming problem and the barrier penalized form into a series of nonlinear equations. Instead of solving them individually, we connect different equations by establishing a group of paths, along which the residual is proportional to the barrier parameter. The behavior of the paths when the barrier parameter goes to zero is meticulously analyzed, while the primal-dual optimal solutions of (P) and (D) can then be retrieved from the relationship established by DRS. This novel approach of constructing paths builds a link between different fixed-point mappings from the DRS. Based on these paths, we design a path-following based algorithm called DRS interior point method (DRSIP), which employs a second-order method to update along the direction of the path. Compared to the previous work [14], our method generalizes the path scheme and allows full Newton steps to be taken along the path. This strategic design ensures that the algorithm converges to the solution path with a quadratic rate, thereby enhancing the convergence properties of the DRS method.
2. **Convergence analysis**
   We establish the global convergence of the DRSIP algorithm. We tailor a line search strategy based on the merit function reduction ensure this global convergence property. This strategy guarantees that the algorithm converges from any starting point within the feasible region. Under the strict complementarity condition, we further prove the local quadratic convergence of the DRSIP algorithm by employing techniques in Newton-type methods. The convergence analysis is rigorously underpinned by the local characteristics of the fixed-point mapping and a meticulously designed path-following scheme.
3. **Numerical performance**
   We extensively evaluate the numerical performance of DRSIP through computational experiments involving both synthetic and real-world datasets. Our method demonstrates enhanced computational efficiency, robustness, and adaptability across diverse conic programming settings, including quadratic programming (QP), second-order cone programming (SOCP). Compared to existing methods involving interior-point methods, operator splitting methods, augmented Lagrangian methods, DRSIP shows superior convergence rates, reduced computational times, and improved accuracy. These results establish DRSIP as a scalable and

effective solution for large-scale optimization challenges, highlighting its potential for broad applicability in complex numerical settings.

1.2 Organization

This paper is organized as follows. In Section 2, we present the algorithm framework and introduce several common assumptions. In Section 3 and Section 4, global convergence and local quadratic convergence are investigated. In Section 5, numerical experiments are performed to demonstrate the efficiency of DRSIP. Finally, we conclude this paper in Section 6.

1.3 Notation

For linear operator $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$, $\mathcal{A}^*$ denotes its adjoint operator and $\mathcal{A}^\dagger$ denotes its Moore-Penrose inverse. Under the full row rank assumption, $\mathcal{A}^\dagger = \mathcal{A}^\top(\mathcal{A}\mathcal{A}^\top)^{-1}$. We use $\mathcal{I}$ to denote the identity operator. The L_2-norm of a vector or the spectral norm of a matrix is represented by $\|\cdot\|$. The inner product of two elements x and y in the Hilbert space is denoted by $\langle x, y\rangle$. Given $z \in \mathbb{R}^n, b > 0$, we use $N(z, b)$ and $\overline{N}(z, b)$ to denote the open and closed ball centered at z with radius b, respectively, that is, $N(z, b) = \{x \in \mathbb{R}^n \mid \|x - z\| < b\}$ and $\overline{N}(z, b) = \{x \in \mathbb{R}^n \mid \|x - z\| \le b\}$.

For a set $\Omega \subseteq \mathbb{R}^n$, the relative interior of Ω is denoted by $\mathrm{ri}(\Omega)$. The notation $x \succeq_K y$ means $x - y \in \mathcal{K}$ and $x \succ_K y$ means $x - y \in \mathrm{ri}\,\mathcal{K}$. Notations $\prec_K$ and $\preceq_K$ are defined similarly. The indicator function of a set Ω is denoted by $\delta_\Omega(x) = \begin{cases} 0, & \text{if } x \in \Omega \\ \infty, & \text{otherwise} \end{cases}$ and the projection of a point x onto a set Ω is denoted by $\Pi_\Omega(x) = \arg\min_{y\in\Omega} \|x - y\|$. The Fenchel conjugate of f is represented by f^*, which is defined as $f^*(y) = \sup_{x\in \mathrm{dom} f}(\langle x, y\rangle - f(x))$.

The gradient of a smooth function f is denoted by ∇f and the Hessian of f is denoted by $\nabla^2 f$. For nonsmooth functions, we use ∂f to denote the Clarke's generalized gradient of f. For a multivariable differentiable function $f(x_1, ..., x_p)$, we use $\partial_{x_i} f$ to denote the partial derivatives of f with respect to x_i. If f is not differentiable with respect to some variable, then this partial derivative is replaced by the subdifferential with respect to that variable.

2 A path-following algorithm based on fixed-point mapping

2.1 Operator splitting for conic programming

To ensure the feasibility of the conic programming problem (P), we introduce assumptions on the conic constraints.

Assumption 1 (Slater's condition) *The conic programming problem* (P) *and* (D) *has feasible solution (x, y, s) with* $x \in \mathrm{ri}\mathcal{K}$ *and* $s \in \mathrm{ri}\mathcal{K}^*$.

We consider the penalized form of (P) by replacing the conic constraints $x \in \mathcal{K}$ with a barrier function g as follows:

$$\begin{aligned} \min_{x} \quad & \langle c, x\rangle + \mu g(x) \\ \text{s.t.} \quad & \mathcal{A}x = b, \\ & x \in \mathcal{K}, \end{aligned} \qquad (\mathrm{P}(\mu)) \qquad \begin{aligned} \max_{y,s} \quad & b^T y - \mu g^*(s) \\ \text{s.t.} \quad & \mathcal{A}^* y + s = c, \\ & s \in \mathcal{K}^*. \end{aligned} \qquad (\mathrm{D}(\mu))$$

To guarantee the well-definedness and smoothness of the proximal operator, we assume that the barrier function g satisfies the following properties.

Assumption 2 *(i) The function $g : \mathcal{K} \to \mathbb{R}$ is strictly convex, closed, proper. (ii) $g(x) < \infty$ and is C^3 smooth for $x \in \mathrm{ri}(\mathcal{K})$ and $g(x) = +\infty$ for $x \in \mathbb{R}^n \backslash \mathrm{ri}\mathcal{K}$.*

A widely considered choice of g is the logarithmic barrier function. For common cones, the expression of logarithmic barrier is written in the Table 1.

Table 1: Log-barrier function for common cones

Cone type	Log-barrier function $g(x)$
Nonnegative orthant	$-\sum_{i=1}^{n} \log x_i$
Quadratic cone	$-\frac{1}{2}\log(x_1^2 - \sum_{i=2}^{n} x_i^2)$
Rotated quadratic cone	$-\frac{1}{2}\log(x_1 x_2 - \sum_{i=3}^{n} x_i^2)$
Semidefinite cone	$-\log\det X$

The Karush–Kuhn–Tucker(KKT) conditions of the penalized problems are give by

$$\begin{aligned} \mathcal{A}x - b = 0, \quad & x \in \mathcal{K}, \\ \mathcal{A}^* y + s - c = 0, \quad & s \in \mathcal{K}^*, \\ \mu \nabla g(x) + s = 0 \Leftrightarrow \mu \nabla g^*(s) + x = 0. & \end{aligned} \qquad (1)$$

To apply the operator splitting method, we first divide the objective function of (P(μ)) into two parts as follows:

$$\min_{x \in \mathcal{K}} \quad h(x) + \mu g(x), \qquad (2)$$

where $h(x)$ is defined as

$$h(x) = \langle c, x\rangle + \delta_{\{\mathcal{A}x = b\}}(x). \qquad (3)$$

Define the proximal operator of x over a closed, convex, proper function ϕ as

$$\mathrm{prox}_{\gamma\phi}(x) = \arg\min_{u}(\phi(u) + \frac{1}{2\gamma}\|u - x\|^2), \quad x \in \mathrm{dom}\phi, \gamma > 0.$$

Then applying the DRS to (2) gives the following iterative scheme:

$$x^{k+1} = \text{prox}_{\gamma\mu g}(z^k), \tag{4a}$$
$$z^{k+1} = z^k + \text{prox}_{\gamma h}(2x^{k+1} - z^k) - x^{k+1}, \tag{4b}$$

where the constant $\gamma > 0$. From the definition of proximal operator we can compute expression of $\text{prox}_{\gamma h}$ as

$$\text{prox}_{\gamma h}(z) = (\mathcal{I} - \mathcal{A}^\dagger\mathcal{A})(z - \gamma c) + \mathcal{A}^\dagger b, \quad z \in \mathbb{R}^n,$$

where $\mathcal{A}^\dagger = \mathcal{A}^*(\mathcal{A}\mathcal{A}^*)^{-1}$ denotes the pseudoinverse of $\mathcal{A}$. By eliminating the auxiliary variable x , (4) is equivalent to a fixed-point iteration

$$z^{k+1} = z^k - F(z^k, \mu),$$

where the residual mapping $F(z, \mu)$ is defined as

$$F(z, \mu) = \text{prox}_{\gamma\mu g}(z) - \text{prox}_{\gamma h}(2\text{prox}_{\gamma\mu g}(z) - z), \quad z \in \mathrm{R}^n. \tag{5}$$

Remark 1 *This operator splitting procedure can also be applied to the original conic programming problem* (P). *Let*

$$g_0(x) = \delta_{\mathcal{K}}(x), \tag{6}$$

then $\text{prox}_{\gamma g_0}(x) = \Pi_{\mathcal{K}}(x)$ *and the DRS residual mapping is given by*

$$\bar{F}(z) = \text{prox}_{\gamma g_0}(z) - \text{prox}_{\gamma h}(2\text{prox}_{\gamma g_0}(z) - z). \tag{7}$$

From the fact that $\lim_{\mu\to 0} \text{prox}_{\gamma\mu g}(x) = \Pi_{\mathcal{K}}(x)$, *we know that* $\lim_{\mu\to 0} F(z, \mu) = \bar{F}(z), z \in \mathbb{R}^n$. *For simplicity of notation, we also write* $F(z, 0)$ *as* $\bar{F}(z)$.

From the property of DRS we know solving (2) is equivalent to locating the unique root z^* of F with μ fixed, while the unique optimal solution of (P(μ)) can be obtained by $x = \text{prox}_{\mu g}(z^*)$. Similarly, any root z^* of $\bar{F}(z)$ leads to an optimal solution x^* of (P) under the relationship $x^* = \Pi_{\mathcal{K}}(z^*)$. Therefore, to solve (P), it suffices to compute the root of $F(\cdot, 0)$ with respect to z. The reasons of considering (P(μ)) rather than (P) directly are 1) $F(z, \mu)$ is a smooth mapping as long as $\mu > 0$ and 2) the solution to $F(z, \mu) = 0$ is unique.

Proposition 1 *The residual mapping* $F(z, \mu)$ *has the following properties:*

(i) $F(z, \mu)$ *is a smooth mapping for* $(z, \mu) \in \mathbb{R}^n \times \mathbb{R}_+$.

(ii) $F(z, \mu)$ *has a unique root for any* $\mu > 0$.

Proof The smoothness of $F(z,\mu)$ is guaranteed by the smoothness of the proximal operator. The Assumption 1 and the strong convexity ensure that (P(μ)) and (D(μ)) have optimal solution (x,y,s) which satisfies KKT condition (1). Then $z = x - s$ satifies $x = \text{prox}_{\mu g}(z)$ and hence $F(z,\mu) = 0$. This gives the existence of root. To prove the uniqueness of the root, it suffices to show that $F(z,\mu)$ is monotone with respect to z. Denote the derivative of $F(z,\mu)$ with respect to z by a operator $J(z,\mu) : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$. By some computation, we have

$$J(z,\mu) = \frac{1}{2}\mathcal{I} + \frac{1}{2}(2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I})(2\partial_z \text{prox}_{\gamma\mu g}(z) - \mathcal{I}). \tag{8}$$

The property of proximal operator gives $\partial_z \text{prox}_{\gamma\mu g}(z) = (\mathcal{I} + \gamma\mu\nabla^2 g(x))^{-1}$ with $x = \text{prox}_{\gamma\mu g}(z) \in \mathcal{K}$. Since g is convex, it holds that $0 \prec \partial_z \text{prox}_{\gamma\mu g}(z) \prec \mathcal{I}$. Note that the eigenvalues of $2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I}$ are either 1 or -1. Thus the minimal eigenvalue of $J(z,\mu)$ is greater than 0, which implies that $F(z,\mu)$ is monotone with respect to z.

2.2 Relation between Infeasibility and DRS residual mapping

The DRS iterative process yields a sequence of points that converge to a fixed point, which corresponds to the solution of the penalized conic programming problem. Upon obtaining the fixed point z, we can recover the primal and dual solutions of the log-barrier problems (P(μ)) and (D(μ)). The recovery process is facilitated by the following transformations:

$$\begin{aligned} x &= \text{prox}_{\gamma\mu g}(z), \\ s &= (x - z)/\gamma = -\mu\nabla g(x), \\ y &= -(\mathcal{A}\mathcal{A}^*)^{-1}\left((\mathcal{A}x - b)/\gamma + \mathcal{A}(s - c)\right). \end{aligned} \tag{9}$$

The above relations are instrumental in characterizing the infeasibilities associated with the DRS residual mapping $F(z,\mu)$. The following lemma encapsulates the relationship between the primal and dual infeasibilities and the DRS fixed point iteration:

Lemma 1 *The infeasibilities and complementarity arising from the DRS method satisfy the following equations:*

(i) Primal infeasibility:

$$\mathcal{A}x - b = \mathcal{A}F(z,\mu). \tag{10}$$

(ii) Dual infeasibility:

$$\mathcal{A}^*y + s - c = -F(z,\mu)/\gamma. \tag{11}$$

(iii) Complementarity:

$$\langle c, x\rangle - \langle b, y\rangle = \langle x, s\rangle + \langle F(z,\mu), x/\gamma + \mathcal{A}^*y\rangle. \tag{12}$$

Proof The primal infeasibilty and dual infeasibilty are straightforward. Now we prove the complementarity:

$$\begin{aligned}\langle c, x\rangle - \langle b, y\rangle &= \langle \mathcal{A}^* y + s + F(z,\mu)/\gamma, x\rangle - \langle Ax - AF(z,\mu), y\rangle \\ &= \langle s + F(z,\mu)/\gamma, x\rangle + \langle AF(z,\mu), y\rangle \\ &= \langle x, s\rangle + \langle F(z,\mu), x/\gamma + \mathcal{A}^* y\rangle.\end{aligned}$$

This completes the proof.

Corollary 1 *Suppose z is a root of $F(z,\mu)$, then (x,y,s) derived from (9) is a KKT triplet of (P(μ)) and (D(μ)).*

Proof This is a direct conclusion of Lemma 1 at $F(z,\mu) = 0$.

2.3 A solution path based on fixed point mapping

For $\mu > 0$, the fixed point mapping $F(z,\mu)$ has a unique root $z^*(\mu)$. The curve

$$\Gamma^* = \{z^*(\mu) \mid \mu \geq 0\} \subseteq \mathbb{R}^n$$

is called the solution path of the penalized conic programming problem (P(μ)). For any (z,μ) outside Γ^*, the residual $\|F(z,\mu)\|$ is a natural measure of the discrepancy between (z,μ) and Γ^*. From the implicit function theorem, we know $z^*(\mu)$ is a smooth function of μ in a neighborhood. The solution path Γ is a continuous curve that connects the optimal solution of the original conic programming problem (P) and the solution of the penalized problem (P(μ)) as $\mu \to 0$.

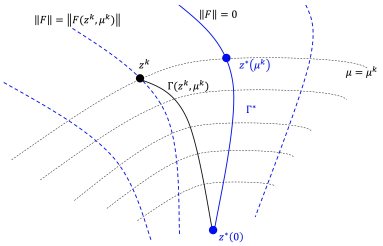
For each iteration, the current point is (z,μ) is outside the solution path Γ^*. The goal is to find a direction Δz that moves the current point towards the solution path and decrease μ at the same time. Now we consider moving to a target point $(\xi, \beta(\nu)\mu)$, where $\beta(\nu)$ is a monotonically increasing smooth function on $[0,1]$ with $\beta(0) = 0, \beta(1) = 1$ and $\tau = 1 - \nu$ is the step size. $\beta(\nu)$ may be dependent on the residual the current μ and $\|F(z,\mu)\|$. We aim to reduce the residual function $F(z,\mu)$ to a factor of $\beta(\nu)$ of its original value. Thus we have the following equation:

$$F(\xi, \beta(\nu)\mu) = \beta(\nu)F(z,\mu), \quad \xi \in \mathbb{R}^n, \nu \in [0,1].$$

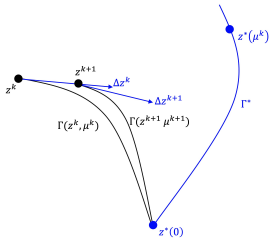
From the implicit function theorem we know this equation define a path

$$\Gamma(z,\mu) = \{\xi(\nu) \mid F(\xi(\nu), \beta(\nu)\mu) = \beta(\nu)F(z,\mu), \nu \in [0,1]\} \subseteq \mathbb{R}^n,$$

which starts from $\xi(1) = z$ and ends at some root of $F(\cdot, 0)$ as $\nu \to 0$. This path is an approximation of the solution path Γ^* as if z^k is no far away from $z^*(\mu^k)$. Figure 1(a) illustrates the establishment of paths.



(a) The black dotted lines represent contour lines of μ and the blue dotted lines represent contour lines of $\|F(z,\mu)\|$(written as $\|F\|$ for simplicity). The contour lines of $\|F\| = 0$ coincide with the solution path Γ^*. The approximation path $\Gamma(z,\mu)$ not only directs to $z^*(0)$ but also has a trending to the solution path Γ^* as μ decreases.



(b) We use a first-order approximation of the $\Gamma(z^k,\mu^k)$ to compute the direction Δz^k at the k-th iteration. Then we update the current point to z^{k+1} along this tangent direction with some proper step size and decrease μ. After that we derive a new path $\Gamma(z^{k+1},\mu^{k+1})$ starting from z^{k+1} and repeat the above procedure.

Fig. 1: Solution path and iteration of the path-following scheme.

2.4 First-order approximation of the solution path

Since it is hard to compute $\xi(\nu)$ for $\nu \in (0,1)$, the path-following procedure is done by walking along the tangent direction of $\Gamma(z,\mu)$. Differentiating (2.3) with respect to ν gives

$$J\left(\xi(\nu),\beta(\nu)\mu\right)\xi'(\nu) + \beta'(\nu)\mu\partial_\mu F\left(\xi(\nu),\beta(\nu)\mu\right) - F(z,\mu) = 0, \tag{13}$$

where $J(z,\mu)$ is the Jacobian of $F(z,\mu)$ with respect to z.

Starting from $\xi(1) = z$, a first-order approximation of $\xi(1-\tau)$ can be computed by the following scheme:

$$\xi(1-\tau) \approx \xi(1) - \tau\xi'(1), \tag{14}$$

where the step size τ lies in a suitable range in $(0,1]$. The differential $\xi'(1)$ is attainable through the solution of the linear system (13) at $\nu = 1$, that is

$$\xi'(1) = J(z,\mu)^{-1}\left(F(z,\mu) - \beta'(1)\mu\partial_\mu F(z,\mu)\right). \tag{15}$$

Note that $\|F(\xi(1-\tau),\beta(1-\tau)\mu)\|_2 = \beta(1-\tau)\|F(z,\mu)\|_2$, hence $\xi(1-\tau)$ is anticipated to be closer to the solution path Γ^*. Therefore we utilize the first-order approximation of $\xi(1-\tau)$, that is (14), as the target point. The update formula of μ is also derived from the first order approximation of the target point as $\beta(1-\tau)\mu \approx (1-\beta'(1)\tau)\mu = \mu + \tau\Delta\mu$ where $\Delta\mu = -\beta'(1)\mu$.

At a new iteration, we construct a new path repeat the above procedure until the residual $\|F(z,\mu)\|$ is sufficiently small. This strategy is similar to the explicit Euler scheme for solving ordinary differential equations (ODE), while the major purpuse of our path-following scheme is to reduce the residual function $F(z,\mu)$ rather than fitting the curve $\Gamma(z,\mu)$ as Euler scheme does.

To summarize, the direction Δz^k and $\Delta\mu^k$ at the k-th iteration are computed by solving the linear system

$$J(z^k,\mu^k)\Delta z^k = -F(z^k,\mu^k) + \eta^k\mu^k\partial_\mu F(z^k,\mu^k), \tag{16}$$

and $\Delta\mu^k = -\eta^k\mu^k$, where $\eta^k = \beta'(1)$ at the k-th iteration. The selection strategy for η^k is defined as follows:

$$\eta^k = \max\{1-\alpha_1\|F(z^k,\mu^k)\|,\bar{\eta}\} \in [\bar{\eta},1], \tag{17}$$

where α_1 and $\bar{\eta} \in (0,1)$ are constants. Figure 1(b) illustrates the iteration scheme.

2.5 A Line search strategy

Define the merit function as

$$\Theta(z,\mu) = \frac{1}{2}\left(\|F(z,\mu)\|^2 + \mu^2\right),$$

and denote $\bar{\Theta}(z) = \Theta(z,0)$. At the iteration k, we aim to find a step size by backtracking line search to ensure the decrease of the merit function, that is, find the smallest non-negative integer m_k such that the step size $\tau^k = \rho^{m_k}$ satisfies

$$\Theta(z^k+\tau^k\Delta z^k,\mu^k+\tau^k\Delta\mu^k) - \Theta(z^k,\mu^k) \le -2\sigma\tau^k\Theta(z^k,\mu^k), \tag{18}$$

where $\sigma \in (0,1)$ is a constant. After that we update the current point by $z^{k+1} = z^k + \tau^k\Delta z^k$, $\mu^{k+1} = (1-\eta^k\tau^k)\mu^k$. The complete procedure of DRSIP is summarized in Algorithm 1.

Algorithm 1 DRSIP

Require: Choose $\sigma \in (0,1)$,$\rho \in (0,1)$, $\bar{\eta} \in (0,1)$, $\mu^0 \ge 0$ and z^0. Set $k = 0$.

1: **while** not converged **do**
2: Compute η^k by (17).
3: Compute Δz^k by solving the linear systems (16).
4: Find the smallest integer $m_k > 0$ satisfying (18) and set $\tau^k = \rho^{m_k}$.
5: Update $z^{k+1} = z^k + \tau^k\Delta z^k$, $\mu^{k+1} = (1-\eta^k\tau^k)\mu^k$.
6: $k \leftarrow k+1$.
7: **end while**

3 Global convergence analysis

First we establish the boundedness properties of $F(z,\mu)$.

Proposition 2 *(i) Assume $\mu \geq 0$ is fixed. Then for any positive t, $\{z \mid \|F(z,\mu)\| \leq t\}$ is a compact set.*

(ii) For any positive t, $\{(z,\mu) \mid \|F(z,\mu)\| \leq t, 0 \leq \mu \leq \bar{\mu}\}$ and $\{(z,\mu) \mid \Theta(z,\mu) \leq t, 0 \leq \mu \leq \bar{\mu}\}$ are both compact sets, where $\bar{\mu} > 0$ is a constant.

Proof The above two properties of $F(z,\mu)$ are direct consequences of its continuity.

Since $\Theta(z^k,\mu^k)$ is monotonically decreasing, a corollary is that the sequence $\{(z^k,\mu^k)\}$ generated by Algorithm 1 is bounded in the domain $\{(z,\mu) \mid \Theta(z,\mu) \leq \Theta(z^0,\mu^0), 0 \leq \mu \leq \mu^0\}$.

Lemma 2 *The line search step size τ^k in Algorithm 1 is well defined, that is, there exists a non-negative integer m_k such that (18) holds.*

Proof By the definition of $\Theta(z,\mu)$ and (16), we have

$$\begin{aligned}
&\Theta(z^k+\tau\Delta z^k,\mu^k+\tau\Delta\mu^k)-\Theta(z^k,\mu^k)\\
=&\tau\left(\langle\partial_z\Theta(z^k,\mu^k),\Delta z^k\rangle+\partial_\mu\Theta(z^k,\mu^k)\Delta\mu^k\right)+o(\tau)\\
=&\tau\left(\langle F(z^k,\mu^k),J(z^k,\mu^k)\Delta z^k\rangle+\langle F(z^k,\mu^k),\partial_\mu F(z^k,\mu^k)\rangle\Delta\mu^k+\mu^k\Delta\mu^k\right)+o(\tau)\\
=&-\tau\left(\langle F(z^k,\mu^k),F(z^k,\mu^k)\rangle+\eta^k(\mu^k)^2\right)+o(\tau)\\
=&-2\tau\Theta(z^k,\mu^k)+\tau(1-\eta^k)(\mu^k)^2+o(\tau)\\
\leq&-2\tau\Theta(z^k,\mu^k)+2\tau(1-\bar{\eta})\Theta(z^k,\mu^k)+o(\tau)\\
=&-2\tau\bar{\eta}\Theta(z^k,\mu^k)+o(\tau).
\end{aligned}$$

Since $\sigma < \bar{\eta}$, (18) holds for τ small enough. This implies that there exists a non-negative integer m_k such that (18) holds. This completes the proof.

Theorem 3 (Global convergence) *The sequence $\{z^k\}$ generated by Algorithm 1 satisfies*

$$\lim_{k\to\infty}\Theta(z^k,\mu^k)=0,\quad \lim_{k\to\infty}\mu^k=0.$$

Furthermore, any accumulation point z^* of $\{z^k\}$ is a root of $\bar{F}(z)$.

Proof The update formula of μ and the bounded of η^k gives $\mu^{k+1} \leq (1-\eta^k\tau^k)\mu^k \leq (1-\bar{\eta})\mu^k$. This implies that μ^k converges to 0 as $k \to \infty$. By Lemma 2, we know that the line search step size τ^k is well defined, hence the merit function $\Theta(z^k, \mu^k)$ is monotonically decreasing. Thus it converges to some nonnegative value Θ^*.

Now we prove that $\Theta^* = 0$ by contradiction. Suppose $\Theta^* > 0$. Taking limit on both sides of the equation $\Theta(z^{k+1}, \mu^{k+1}) - \Theta(z^k, \mu^k) \leq -2\sigma\tau^k\Theta(z^k, \mu^k)$ gives $\lim_{k\to\infty} \tau^k = 0$. We choose two constants Θ_1, Θ_2 satisfying $0 < \Theta_1 < \Theta^* < \Theta_2$ and $\Theta_2/\Theta_1 > \bar{\eta}/\sigma$, then there exists a positive integer K such that $\Theta(z^k, \mu^k) \in (\Theta_1, \Theta_2)$ for all $k \geq K$. From the proof of Lemma 2, we know that

$$\begin{aligned}\Theta(z^k + \tau\Delta z^k, \mu^k + \tau\Delta\mu^k) - \Theta(z^k, \mu^k) &\leq -2\tau\bar{\eta}\Theta(z^k, \mu^k) + o(\tau)\\ &\leq -2\tau\bar{\eta}\Theta_1 + o(\tau),\end{aligned}$$

and $-2\tau\sigma\Theta(z^k, \mu^k) \leq -2\sigma\tau\Theta_2$. Since $D = \{(z, \mu) \mid F(z, \mu) \leq \Theta_2, 0 \leq \mu \leq \mu^K\}$ is a compact set, the $o(\tau)$ term is bounded by $M\tau^2$ with some constant $M = \frac{1}{2}\sup_D \|\partial^2\Theta(z, \mu)\|$. This implies that there exist a positive value $\tau_0 = 2(\bar{\eta}\Theta_1 - \sigma\Theta_2)/M$ such that for all $\tau \in (0, \tau_0)$, we have $-2\tau\bar{\eta}\Theta_1 + o(\tau) \leq -2\sigma\tau\Theta_2$. Therefore (18) for all $\tau \in (0, \tau_0)$. The backtracking line search strategy guarantees we can at least find a step size $\tau^k \geq \tau/\rho$ such that (18) holds. This contradicts to the fact that $\lim_{k\to\infty} \tau^k = 0$. Thus $\Theta^* = 0$. This implies that $\lim_{k\to\infty} \bar{F}(z^k) = 0$ and any accumulation point z^* of $\{z^k\}$ is a root of $F(z, 0)$. The proof is completed.

4 Local convergence analysis

4.1 Strict complementarity condition

First we introduce the concept of prox-regular function.

Definition 1 (Prox-regular function) A function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ is prox-regular at $\bar{x}$ for $\bar{v}$ if f is finite and locally lower semicontinuous at $\bar{x}$ with $\bar{v} \in \partial f(\bar{x})$, and there exist constants $\varepsilon > 0$ and $\rho \geq 0$ such that

$$f(y) \geq f(x) + \langle v, y - x\rangle - \frac{\rho}{2}\|y - x\|^2, \quad \forall y \in B(\bar{x}, \varepsilon),$$

whenever $v \in \partial f(x), \|v - \bar{v}\| < \varepsilon, \|x - \bar{x}\| < \varepsilon, f(x) < f(\bar{x}) + \varepsilon$. We say that f is prox-regular at $\bar{x}$ if f is prox-regular at $\bar{x}$ for every $\bar{v} \in \partial f(\bar{x})$. We say f is prox-regular if it is prox-regular at every point of its domain.

From the definition, we know that every proper convex closed function is prox-regular, hence $h(x)$ and $g(x)$ in (2) are prox-regular.

Then we introduce the concept of strict complementarity for composite optimization problem of two prox-regular functions.

Definition 2 (Strict complementarity) Consider the following composite optimization problem:

$$\min_{x\in\mathbb{R}^n} \quad h(x) + g(x), \tag{19}$$

where $h, g : \mathbb{R}^n \to \overline{\mathbb{R}}$ are proper closed convex and prox-regular functions. Let x^* be a solution of the problem (19) We say that the strict complementarity condition holds at x^* if $0 \in \mathrm{ri}(\partial(h+g)(x^*))$.

Proposition 4 *$\bar{F}(z)$ is Lipschitz continuous on $\mathbb{R}^n$ with modulus $L_0 = 1$.*

Proof For $\forall z_1, z_1 \in \mathbb{R}^n$, we have

$$\bar{F}(z_1) - \bar{F}(z_2) = (2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I})(\Pi_{\mathcal{K}}(z_1) - \Pi_{\mathcal{K}}(z_2)).$$

The property of projection operator gives $\|\Pi_{\mathcal{K}}(z_1) - \Pi_{\mathcal{K}}(z_2)\| \le \|z_1 - z_2\|$, combining with the fact that the eigenvalues of $2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I}$ are either 1 or -1, we have $\|\bar{F}(z_1) - \bar{F}(z_2)\| \le \|z_1 - z_2\|$. This completes the proof.

Denote the zeros of $\bar{F}(z)$ by the set $\Omega_0 = \left\{ z \mid \bar{F}(z) = 0 \right\}$. Assume z^* is some point in Ω_0. We compute $x^* = \Pi(z^*)$ and $s^* = (x^* - z^*)/\gamma$. Then we have the following lemma.

Lemma 3 *The following conditions are equivalent:*

(i) *For any i, either $x_i^* \in \mathrm{ri}\,\mathcal{K}_i, s_i^* = 0$ or $x_i^* \in \mathcal{K}_i \setminus \mathrm{ri}\,\mathcal{K}_i, s_i^* \in \mathrm{ri}\,\mathcal{K}_i$.*

(ii) *The strict complementarity condition holds at (x^*, y^*, s^*) for (P) and (D).*

Proof (P) is equivalent to a composite optimization problem $\min_x h(x) + g_0(x)$, where $h(x), g_0(x)$ are defined in (3) and (6), respectively. From the Remark 4.8 in [10] we know the strict complementarity condition is equivalent to

$$\frac{x^* - z^*}{\gamma} \in \mathrm{ri}\,\partial h\left(x^*\right), \quad \frac{z^* - x^*}{\gamma} \in \mathrm{ri}\,\partial g_0\left(x^*\right). \tag{20}$$

We can compute that

$$\begin{aligned}
&\mathrm{ri}\,\partial h(x^*) = \partial h(x^*) = c + \mathrm{Range}(\mathcal{A}^*),\\
&\mathrm{ri}\,\partial g_0(x^*) = \partial_{x_1} g_0(x^*) \times \cdots \times \partial_{x_p} g_0(x^*),\\
&\partial_{x_i} g_0(x^*) = \begin{cases} \{0\}, & x_i^* \in \mathrm{ri}\,\mathcal{K}_i \\ -\mathrm{ri}\,\mathcal{K}_i, & x^* \in \mathcal{K}_i \setminus \mathrm{ri}\,\mathcal{K}_i \end{cases}, \quad i = 1, \ldots, p.
\end{aligned}$$

Since $\mathcal{A}^* y^* + s^* = c$ and $s^* = (x^* - z^*)/\gamma$, the left statement in (20) always holds. The right statement holds if and only if $-s^* \in \mathrm{ri}\,\mathcal{K}$, which is equivalent to that $-s_i^* \in \mathrm{ri}\,\mathcal{K}_i$ for $i = 1, \ldots, p$. This is equivalent to that for $\forall i$, either $x_i^* \in \mathrm{ri}\,\mathcal{K}_i, s_i^* = 0$ or $x_i^* \in \mathcal{K}_i \setminus \mathrm{ri}\,\mathcal{K}_i, s_i^* \in \mathrm{ri}\,\mathcal{K}_i$. This completes the proof.

4.2 Local quadratic convergence

In this subsection, we present the quadratic local convergence of DRSIP, which updates (z^k, μ^k) with full step size $\tau_k = 1$. We rewrite the update formula of DRSIP as follows:

$$\eta^k = 1 - \alpha_1 \|\bar{F}(z^k)\|, \tag{21a}$$

$$\Delta z^k = -J(z^k, \mu^k)^{-1}\big(F(z^k, \mu^k) - \eta^k \mu^k \partial_\mu F(z^k, \mu^k)\big), \tag{21b}$$

$$z^{k+1} = z^k + \Delta z^k, \tag{21c}$$

$$\mu^{k+1} = \alpha_1 \|\bar{F}(z^k)\| \mu^k. \tag{21d}$$

Remark 2 *Note that* (21b) *is equivalent to*

$$\Delta z^k = -J(z^k, \mu^k)^{-1}\left(\bar{F}(z^k) + r(z^k, \mu^k, \eta^k)\right), \tag{22}$$

where $r(z, \mu, \eta) = F(z, \mu) - \eta\mu\partial_\mu F(z, \mu) - \bar{F}(z)$.

The challenge in analyzing convergence stems from the possible singularity of $\bar{J}(z)$, which could make $J(z, \mu)^{-1}$ unbounded. To address this, we establish local smoothness and derive a local error bound for $\bar{F}(z)$. Define $\bar{J}(z)$ as $\lim_{\mu \to 0} J(z, \mu)$ for $z \in \mathbb{R}^n$. The continuity of $J(z, \mu)$ ensures that $\bar{J}(z)$ is well-defined within the generalized Jacobian of $\bar{F}(z)$.

To prove the local convergence of DRSIP, we need the following assumptions.

Assumption 3 *The strict complementarity condition holds at some optimal solution* (x^*, y^*, s^*) *for* (P) *and* (D).

Remark 3 *Under this assumption,* $z^* = x^* - \gamma s^*$ *does not lies on the relative boundary of* $\mathcal{K}$. *Thus there exist a neighborhood* $N(z^*, \delta)$ *that not intersect with the relative boundary of* $\mathcal{K}$.

Proposition 5 *(i)* (Local C^2 smoothness) *There exist a constant* L_1 *such that*

$$\|\bar{F}(z_1) - \bar{F}(z_2) - \bar{J}(z_2)(z_1 - z_2)\| \le L_1 \|z_1 - z_2\|^2, \quad \bar{z}, z \in N(z^*, \delta). \tag{23}$$

(ii) (Local error bound) $\bar{F}(z)$ *satisfies local error bound condition on* $N(z^*, \delta)$, *that is, there exist constants* $\gamma_0 > 0$ *such that*

$$\big\|\bar{F}(z)\big\| \ge \gamma_0 \mathrm{dist}(z, \Omega_0), \quad \forall z \in N(z^*, \delta). \tag{24}$$

Proof From Remark 3, we know that z^* does not lies on the boundary of $\mathcal{K}$, hence there exists a neighborhood $N(z^*, \delta)$ that not intersect with the boundary of $\mathcal{K}$. In this neighborhood, the projection operator $\Pi_{\mathcal{K}}(z)$ is twice continuously differentiable and satifies local error bound condition. Thus $\bar{F}(z)$ is twice continuously differentiable

in $N(z^*,\delta)$. Since $\text{prox}_{\gamma h}$ is also smooth, the chain rule gives that $\bar{F}(z)$ is locally C^2 smooth and satisfies local error bound condition in $N(z^*,\delta)$. This completes the proof.

After that we construct the boundness of $J(z,\mu), J(z,\mu)^{-1}$ and $r(z,\mu,\eta)$.

Lemma 4 *The following bounds hold for $J(z,\mu), J(z,\mu)^{-1}$ and $r(z,\mu,\eta)$:*

(i) For any $z \in N(z^*,\delta)$, given $\mu \le \bar{\mu}$ for some constants $\bar{\mu} > 0$, then there exists a constant δ_0 such that $\nabla^2 g(\text{prox}_{\gamma\mu g}(z)) \succeq \delta_0 \mathcal{I}$ and $\|J(z,\mu)^{-1}\|$ is bounded by $O(\mu^{-1})$, that is,

$$\left\|J(z,\mu)^{-1}\right\| \le c_1 \mu^{-1}, \tag{25}$$

where $c_1 = \frac{2(\delta_0^{-1}+\gamma\bar{\mu})}{\gamma}$ is a constant.

(ii) $J(z,\mu)$ is Lipschitz continuous with respect to μ at $\mu = 0$, that is,

$$\left\|J(z,\mu) - \bar{J}(z)\right\| \le c_2 \mu, \tag{26}$$

where $c_2 = \sup_{N(z^*,\delta)} \|\partial_\mu J(z,\mu)\|$ is a constant.

(iii) $r(z,\mu,\eta)$ is bounded by $O(\mu^2 + (1-\eta)\mu)$ in the neighborhood $N(z^*,\delta)$, that is,

$$\|r(z,\mu,\eta)\| \le c_3(\mu^2 + (1-\eta)\mu), \quad \forall z \in N(z^*,\delta), \mu \in (0,\bar{\mu}), \eta \in (0,1), \tag{27}$$

where $c_3 = \max\left\{\frac{1}{2}\sup_{N(z^*,\delta)\times[0,\bar{\mu}]} \|\partial^2_{\mu\mu} F(z,\mu)\|, \sup_{N(z^*,\delta)\times[0,\bar{\mu}]} \|\partial_\mu F(z,\mu)\|\right\}$ is a constant.

Proof (i) For given $z \in \mathbb{R}^n, \mu > 0$, $\text{prox}_{\gamma\mu g}(z) \in \text{ri}(\mathcal{K})$. By the continuity of proximal operator, we know that $\{\text{prox}_{\gamma\mu g}(z) \mid z \in N(z^*,\delta)\}$ is compact. Let $\delta_0 = \inf_{z\in N(z^*,\delta)} \nabla^2 g(\text{prox}_{\gamma\mu g}(z))$. Since g is strictly convex, $\delta_0 > 0$.

From the property of proximal operator, we have $\partial_z \text{prox}_{\gamma\mu g}(z) = (\mathcal{I} - \gamma\mu\nabla^2 g(x))^{-1}$, where x denotes $\text{prox}_{\gamma\mu g}(z)$. Utilizing the lower bound of $\nabla^2 g(x)$, we obtain

$$\begin{aligned}(\mathcal{I} + \gamma\mu\nabla^2 g(x))^{-1} &= \mathcal{I} - \gamma\mu((\nabla^2 g(x))^{-1} + \gamma\mu\mathcal{I})^{-1} \\ &\preceq (1 - \gamma\mu(\delta_0^{-1} + \gamma\mu)^{-1})\mathcal{I}.\end{aligned} \tag{28}$$

Note that the eigenvalues of $2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I}$ are either 1 or -1. Thus the minimal eigenvalue of $J(z,\mu)$ is greater than or equal to

$$\frac{1}{2} - \frac{1}{2}(1 - \gamma\mu(\delta_0^{-1} + \gamma\mu)^{-1}) = \frac{\gamma\mu}{2(\delta_0^{-1} + \gamma\mu)}.$$

This implies that $\|J(z,\mu)^{-1}\| \le \frac{2(\delta_0^{-1}+\gamma\mu)}{\gamma\mu} \le c_1\mu^{-1}$, where $c_1 = \frac{2(\delta_0^{-1}+\gamma\bar{\mu})}{\gamma}$. This completes the proof.

(ii) By differential midpoint rule, we have

$$\|J(z,\mu) - \lim_{\mu\to 0} J(z,\mu)\| = \mu\partial_\mu J(z,\mu)\,|_{\mu=\tilde{\mu}} \le \mu \sup_{N(z^*,\delta)\times[0,\bar{\mu}]} \|\partial_\mu J(z,\mu)\|,$$

where $\tilde{\mu} \in [0, \mu]$. $\partial_\mu J(z,\mu)$ can be computed as

$$\begin{aligned}\partial_\mu J(z,\mu) &= (2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I})\partial^2_{\mu,z}\mathrm{prox}_{\gamma\mu g}(z)\\ &= (2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I})\partial_\mu\left((\mathcal{I} + \gamma\mu\nabla^2 g(x))^{-1}\right)\\ &= (2\mathcal{A}^\dagger\mathcal{A} - \mathcal{I})(\mathcal{I} + \gamma\mu\nabla^2 g(x))^{-1}\gamma(\nabla^2 g(x) + \mu\nabla^3 g(x)\partial_\mu x)\cdot\\ &\quad (\mathcal{I} + \gamma\mu\nabla^2 g(x))^{-1}.\end{aligned}$$

Here $\partial_\mu x = \partial_\mu \mathrm{prox}_{\gamma\mu g}(z) = -(I + \gamma\mu\nabla^2 g(x))^{-1}\gamma\nabla g(x)$. Hence $\partial_\mu J(z,\mu)$ is continuous and bounded in the compact set $N(z^*,\delta)\times[0,\bar{\mu}]$, that is, $c_2 = \sup_{N(z^*,\delta)\times[0,\bar{\mu}]}\|\partial_\mu J(z,\mu)\| < +\infty$. This completes the proof.

(*iii*) Since $F(z,\mu)$ is smooth with respect to μ, the differential midpoint rule gives

$$F(z,\mu) - F(z,0) - \mu\partial_\mu F(z,\mu) = \frac{1}{2}\mu^2\partial^2_{\mu\mu}F(z,\mu)\mid_{\mu=\tilde{\mu}},$$

where $\tilde{\mu} \in [0,\mu]$. This gives

$$\begin{aligned}\|r(z,\mu,\eta)\| &= \|\frac{1}{2}\mu^2\partial^2_{\mu\mu}F(z,\mu)\mid_{\mu=\tilde{\mu}} + (1-\eta)\partial_\mu F(z,\mu)\|\\ &\le \frac{1}{2}\mu^2\sup_{N(z^*,\delta)\times[0,\bar{\mu}]}\|\partial^2_{\mu\mu}F(z,\mu)\| + (1-\eta)\sup_{N(z^*,\delta)\times[0,\bar{\mu}]}\|\partial_\mu F(z,\mu)\|\\ &\le c_3(\mu^2 + (1-\eta)\mu), \forall z \in N(z^*,\delta), \mu \in (0,\bar{\mu}),\end{aligned}$$

where $c_3 = \max\left\{\frac{1}{2}\sup_{N(z^*,\delta)\times[0,\bar{\mu}]}\|\partial^2_{\mu\mu}F(z,\mu)\|, \sup_{N(z^*,\delta)\times[0,\bar{\mu}]}\|\partial_\mu F(z,\mu)\|\right\}$. Since $F(z,\mu)$ is smooth with respect to μ, it holds that $c_3 < +\infty$. This completes the proof.

Denote the projection of z^k onto Ω_0 by $\bar{z}^k$, namely $\bar{z}^k \in \arg\min_{z\in\Omega_0}\left\|z - z^k\right\|$. Now we prove the boundness of Δz^k.

Lemma 5 *Under the condition of Lemma 4, then the length of the step is bounded by*

$$\left\|\Delta z^k\right\| \le c_2 L_1(\mu^k)^{-1}\left\|\bar{z}^k - z^k\right\|^2 + (c_1c_2+1)\|\bar{z}^k - z^k\| + c_1c_3(\mu^k + 1 - \eta^k). \quad (29)$$

Proof Let $d^k = -J(z^k,\mu^k)^{-1}\bar{F}(z^k)$, $w^k = d^k - (\bar{z}^k - z^k)$, $v^k = -\bar{F}(z^k) - J(z^k,\mu^k)(\bar{z}^k - z^k)$. From the Newton equation (22), we obtain the relationship between w^k and v^k as follows:

$$J(z^k,\mu^k)w^k = v^k.$$

Utilizing proposition 5 and (26), we deduce:

$$\begin{aligned}\left\|v^k\right\| &\le \left\|\bar{F}(z^k) + \bar{J}(z^k)(\bar{z}^k - z^k)\right\| + \left\|(J(z^k,\mu^k) - \bar{J}(z^k))(\bar{z}^k - z^k)\right\|\\ &= L_1\left\|\bar{z}^k - z^k\right\|^2 + \left\|(J(z^k,\mu^k) - \bar{J}(z^k))(\bar{z}^k - z^k)\right\|\\ &\le L_1\left\|\bar{z}^k - z^k\right\|^2 + c_2\mu^k\left\|\bar{z}^k - z^k\right\|.\end{aligned}$$

Subsequent application of equation (25) leads to:

$$\left\|w^k\right\| \leq \left\|J(z^k,\mu^k)^{-1}\right\| \cdot \left\|v^k\right\| \leq c_2 L_1 (\mu^k)^{-1} \left\|\bar{z}^k - z^k\right\|^2 + c_1 c_2 \|\bar{z}^k - z^k\|.$$

Thus we obtain the boundness of d^k as

$$\left\|d^k\right\| \leq \left\|w^k\right\| + \left\|\bar{z}^k - z^k\right\| \leq c_2 L_1 (\mu^k)^{-1} \left\|\bar{z}^k - z^k\right\|^2 + (c_1 c_2 + 1)\|\bar{z}^k - z^k\|.$$

From (25) and (27), we know the difference between d^k and Δz^k is also bounded, therefore

$$\begin{aligned}\left\|\Delta z^k\right\| &\leq \left\|d^k\right\| + \left\|J(z^k,\mu^k)^{-1} r(z^k,\mu^k)\right\| \\ &\leq c_2 L_1 (\mu^k)^{-1} \left\|\bar{z}^k - z^k\right\|^2 + (c_1 c_2 + 1)\|\bar{z}^k - z^k\| + c_1 c_3 (\mu^k + 1 - \eta^k).\end{aligned}$$

This completes the proof.

We come to prove the quadratic convergence of DRSIP.

Theorem 6 *For any $c_0 > 0$, assume $z^k \in N(z^*, \delta') \cap \{z \mid \bar{F}(z) \leq c_0\}$ where $\delta' = \frac{1}{1+c_4}\delta$, $c_4 = c_2 L_1 + (c_1 c_2 + 1)L_0 + c_1 c_3 L_0^2$. Then it holds that*

$$\operatorname{dist}(z^{k+1}, \Omega_0) + \mu^{k+1} \leq c_5 (\operatorname{dist}(z^k, \Omega_0) + \mu^k)^2, \tag{30}$$

where the constant $c_5 = \max\{(c_3 + c_1^2 c_3^2 L_1 + c_1 c_2 c_3)/\gamma_0, L_1 c_4^2/\gamma_0^3, \frac{1}{2}((c_2 c_4/\gamma_0 + c_3 \alpha_1 L_0 + 2L_1 c_1 c_3 c_4/\gamma_0)/\gamma_0 + \alpha_1 L_1)\}$.

Proof Since $z^k \in N(z^*, \delta/2)$, we have

$$\left\|\bar{z}^k - z^*\right\| \leq \left\|\bar{z}^k - z^k\right\| + \left\|z^k - z^*\right\| \leq \left\|z^k - z^*\right\| + \left\|z^k - z^*\right\| \leq \delta.$$

This gives $z^k \in N(z^*, \delta)$. By (24) and Proposition 4, we have

$$\gamma_0 \left\|\bar{z}^k - z^k\right\| \leq \left\|\bar{F}(z^k)\right\| = \left\|\bar{F}(z^k) - \bar{F}(\bar{z}^k)\right\| \leq L_0 \left\|\bar{z}^k - z^k\right\|.$$

Thus $\left\|\bar{F}(z^k)\right\| \leq c_0$ implies $\left\|\bar{z}^k - z^k\right\| \leq c_0/\gamma_0$. It follows from Proposition 5 and Lemma 5 that

$$\begin{aligned}\left\|\Delta z^k\right\| &\leq c_2 L_1 (\mu^k)^{-1} \|\bar{z}^k - z^k\|^2 + (c_1 c_2 + 1)\|\bar{z}^k - z^k\| + c_1 c_3 (\mu^k + 1 - \eta^k) \\ &\leq c_4/\gamma_0 \|\bar{z}^k - z^k\| + c_1 c_3 \mu^k \\ &= c_4/\gamma_0 \operatorname{dist}(z^k, \Omega_0) + c_1 c_3 \mu^k,\end{aligned}$$

where $c_4 = c_2 L_1 + (c_1 c_2 + 1)L_0 + c_1 c_3 \alpha_1 L_0$, and

$$\begin{aligned}\left\|\bar{F}(z^k + \Delta z^k)\right\| &\leq \left\|\bar{F}(z^k) + \bar{J}(z^k)\Delta z^k\right\| + L_1 \|\Delta z^k\|^2 \\ &= \left\|\left(J(z^k,\mu^k) - \bar{J}(z^k)\right)\Delta z^k\right\| + \|r(z^k,\mu^k,\eta^k)\| + L_1 \|\Delta z^k\|^2 \\ &\leq c_2 \mu^k \|\Delta z^k\| + c_3 \mu^k (\mu^k + \alpha_1 L_0 \|\bar{z}^k - z^k\|) + L_1 \|\Delta z^k\|^2.\end{aligned}$$

By $\left\|z^{k+1}-z^*\right\| \le \left\|z^k-z^*\right\| + \left\|\Delta z^k\right\| \le \delta$ and (24), it yields that

$$\begin{aligned}\operatorname{dist}(z^k+\Delta z^k,\Omega_0)+\mu^{k+1} &\le \frac{1}{\gamma_0}\left\|\bar{F}(z^k+\Delta z^k)\right\| + \alpha_1\left\|\bar{F}(z^k)\right\|\mu^k \\ &\le \frac{1}{\gamma_0}\left\|\bar{F}(z^k+\Delta z^k)\right\| + \alpha_1 L_1\|\bar{z}^k-z^k\|\mu^k \\ &\le c_5(\left\|\bar{z}^k-z^k\right\|+\mu^k)^2,\end{aligned}$$

where c_5 is defined as in the theorem. This completes the proof.

Theorem 7 *Assume the conditions in Theorem 6 holds, then z^k converges to some $\hat{z}\in\Omega_0$ R-quadratically.*

Proof It follows from (30) and (4.2) that

$$\sum_{k=0}^{\infty}\left\|\Delta z^k\right\| < +\infty.$$

As a result, z^k converges to some point $\bar{z}\in\Omega_0$. Note that

$$\begin{aligned}\left\|\Delta z^k\right\|+\mu^k &\ge \operatorname{dist}(z^k,\Omega_0)+\mu^k-\operatorname{dist}(z^{k+1},\Omega_0)-\mu^{k+1} \\ &\ge \operatorname{dist}(z^k,\Omega_0)+\mu^k-c_5(\operatorname{dist}(z^k,\Omega_0)+\mu^k)^2 \\ &\ge (1-c_5(c_0/\gamma_0+\bar{\mu}))\operatorname{dist}(z^k,\Omega_0).\end{aligned}$$

Together with (29), it yields that

$$\begin{aligned}\left\|\Delta z^{k+1}\right\|+\mu^{k+1} &\le \max(c_4,1)(\operatorname{dist}(z^{k+1},\Omega_0)+\mu^{k+1}) \\ &\le \max(c_4,1)c_5(\operatorname{dist}(z^k,\Omega_0)+\mu^k)^2 \\ &\le c_6(\left\|\Delta z^k\right\|+\mu^k)^2,\end{aligned}$$

where $c_6 = \max(c_4,1)c_5(\frac{1}{1-c_5(c_0/\gamma_0+\bar{\mu})}+1)^2$. Therefore $\left\|\Delta z^k\right\|+\mu^k$ converges quadratically. To prove the convergence of z^k, we have

$$\begin{aligned}\lim_{k\to\infty}\frac{\|z^{k+1}-\bar{z}\|+\mu^{k+1}}{(\|z^k-\bar{z}\|+\mu^k)^2} &= \lim_{k\to\infty}\frac{\|\sum_{j=k+1}^{\infty}\Delta z^k\|+\mu^{k+1}}{(\|\sum_{j=k}^{\infty}\Delta z^k\|+\mu^k)^2} \\ &\le \lim_{k\to\infty}\frac{\|\Delta z^{k+1}\|+\mu^{k+1}}{(\|\Delta z^k\|+\mu^k)^2} \le c_6.\end{aligned}$$

Hence z^k converges to $\hat{z}$ R-quadratically. This completes the proof.

Remark 4 *For the case that z^* is not unique, a possiblility is that the root of $\bar{F}(z)$ is not isolated. In this case, we do not have the convergence of z^k to z^*, but we can still prove the sequence $\{z^k\}$ converges to some point $\hat{z}$ in the neighborhood of z^*.*

5 Numerical Experiments

In this section, we evaluate the performance of DRSIP on various type of conic optimization problems using synthetic and real-world datasets. We compare DRSIP against other open-source and commercial solvers. The basic setup of our numerical experiments is described as follows.

Implementation and environment. DRSIP is implemented in C++, with usage of Eigen 3.4 [8] and PARDISO 8.0 [19, 20] for numerical factorization. All the experiments are conducted on a server with 2 x Intel Xeon Gold 6326 CPU @ 2.40GHz and 256GB RAM.

Metrics and stopping criteria. Let (x, y, s) be a primal-dual solution pair of (P) and (D). The relative duality gap (g_{rel}), the relative primal infeasibility (p_{rel}), and the relative dual infeasibility (d_{rel}) are defined as

$$g_{\text{rel}} = \frac{|\langle c, x\rangle - b^\top y|}{1 + |\langle c, x\rangle| + |b^\top y|}, \quad p_{\text{rel}} = \frac{\|\mathcal{A}x - b\|}{1 + \|b\|}, \quad d_{\text{rel}} = \frac{\|\mathcal{A}^* y + s - c\|}{1 + \|c\|}. \tag{31}$$

The tolerance is set as $\text{tol} = 10^{-6}$ in all our experiments, that is, DRSIP terminates as long as

$$\text{error} = \max(g_{\text{rel}}, p_{\text{rel}}, d_{\text{rel}}) \leq \text{tol}. \tag{32}$$

Other solvers may use different metrics and stopping criteria. For example, some solvers compute the relative primal infeasibility and relative dual infeasibility with L_1-norm or $L_\infty-$norm. For sake of fairness, we set $\text{tol} = 10^{-6}$ as their input parameters and accept the solution as long as the error computed by (31) and (32) is smaller than $100 \cdot \text{tol}$.

5.1 Portfolio optimization

Consider a basic long-only portfolio optimization task. We aim to maximize the expected risk-adjusted return of a portfolio by selecting the relative weights of assets. The portfolio optimization problem is given as:

$$\begin{aligned} \max_{x} \quad & \mu^T x - \gamma \left(x^T \Sigma x\right) \\ \text{s.t.} \quad & \mathbf{1}^T x = 1, \quad z \geq 0, \end{aligned}$$

The variable x, μ represents the portfolio and the expected returns of p assets, respectively. $\gamma > 0$ denotes the risk aversion parameter. Σ is the asset return covariance matrix, also known as the risk model matrix. Σ has the form as

$$\Sigma = FF^T + D,$$

where F is the factor loading matrix, and D is a diagonal matrix representing asset-specific risk. Usually, the number of risk factors q is substantially fewer than the number of assets p. Employing the factor model form alleviates both

the statistical burden of estimating entries in Σ and the computational burden of handling a full $p \times p$ matrix.

The problem can be formulated as the following standard second order cone programming (SOCP):

$$\begin{aligned} \max_{x,t,s,u,v} \quad & \mu^T x - \gamma(t+s) \\ \text{s.t.} \quad & \mathbf{1}^T x = 1, \quad x \geq 0, \\ & \left\| D^{1/2} x \right\|_2 \leq u, \quad \left\| F^T x \right\|_2 \leq v, \\ & \|(1-t, 2u)\|_2 \leq 1+t, \\ & \|(1-s, 2v)\|_2 \leq 1+s, \end{aligned}$$

where t, s, u, v are additional variables.

In the experiments, we generate μ satifying $\log(1+\mu_i) \sim \mathcal{N}(0, 0.01)$. The diagonal matrix D is generated by $D_{ii} \sim \mathcal{N}(0, 0.1)$. The factor loading matrix F is generated with the sparsity of 0.1 and reciprocal condition number of 0.5, and its nonzeros are drawn from $\mathcal{N}(0, 0.1)$. γ is set to 1. We generate the data matrix of different sizes as $(p, q) = (10000, 2000), (20000, 4000), (30000, 6000)$.

We compare the performance of DRSIP with other second order algorithms, including ECOS [3], superSCS [21], SDPNAL+ [26], SDPT3 [24]. ECOS and SDPT3 are open-source implementations of interior point methods. SuperSCS is based on an operator splitting method and utilize Newton-type method for acceleration. SDPNAL+ is based on a majorized semi-smooth Newton-CG augmented Lagrangian method. The results are shown in Table 2. Both superSCS and SDPNAL+ have an automatic mechanism that takes

Table 2: Results on portfolio optimization.

	instance1		instance2		instance3	
variables p	10000		20000		30000	
measurements q	2000		4000		6000	
columns	22012		44012		66012	
rows	12009		24009		36009	
nonzeros	1.93×10^6		7.62×10^6		1.72×10^7	
solver	time	iter	time	iter	time	iter
ECOS	74	34	580	33	2547	46
superSCS	81	1616	302(2.0e-04)	1335	304(2.7e-04)	640
SDPNAL+	119	1030	1348	1228	10000(3.6e-03)	8301
SDPT3	119	38	-		-	
DRSIP	57	14	651	22	2024	21

Note: Some solvers may exit due to the time limit (10000 seconds) or some other reasons and output the solution before reaching the target precision. We mark the error of solution computed by (31) and (32) in the parentheses. "-" indicates that the solver do not provide a solution in the limited time.

DRS or ADMM type updating steps to guarantee the stability of the algorithm, hence it may take large number of iterations but do not take much time. From the parentheses in Table 2, we see it may be difficult for this first order step scheme to achieve high accuracy as the problem sizes become large. ECOS is based on embedded system with advantage lying in the fast construction of linear systems. For the problem sizes in Table 2, the time per iteration is less than that of DRSIP. However, compared to open-source software based on the IPM (ECOS, SDPT3), DRSIP takes about $1/3 \sim 1/2$ fewer iterations. This reduction in iteration allows DRSIP to achieve the desired precision fastest for two of the problems.

5.2 QP problems

The DRSIP algorithm can be extended to solve quadratic cone programming, which is in the following form:

$$\begin{aligned} \min_{x} \quad & \frac{1}{2}x^\top Qx + \langle c, x\rangle \\ \text{s.t.} \quad & Ax = b, \\ & x \in \mathcal{K}, \end{aligned} \tag{33}$$

where Q is a positive semidefinite symmetric matrix and $A \in \mathbb{R}^{m\times n}$. The quadratic term transforms $h(x)$ in (2) to $h(x) = \frac{1}{2}x^\top Qx + \langle c, x\rangle + \delta_{\{Ax=b\}}(x)$. Then its proximal operator is computed as

$$\text{prox}_{\gamma h}(x) = \arg\min_{x} \quad \frac{1}{2\gamma}\|x-u\|^2 + \frac{1}{2}\langle u, Qu\rangle + \langle c, u\rangle, \quad \text{s.t. } Au = b.$$

We compute $F(z,\mu) = \text{prox}_{\gamma\mu g}(z) - \text{prox}_{\gamma h}(2\text{prox}_{\gamma\mu g}(z) - z)$ with the above proximal operator $\text{prox}_{\gamma h}$ and substitute in Algorithm 1. The subsequent processing in Algorithm 1 is the same as the original version.

We compare the running time of DRSIP with other QP solvers, including CLP [2], HiGHS [9] and Gurobi [7]. CLP is open-source software based on IPM. HiGHS is an open-source solver implementing an variant of IPM, which utilizes iterative methods to solve the linear system. Gurobi is the most efficient commercial software in the world and also implements IPM as its core solver. All of this software is in C++ version.

The test problems are from QPLIB [5] and Maros-Mészáros QP set [1]. QPLIB is a library containing 319 discrete and 134 continuous instances of different characteristics, including quadratic programming (QP), quadratic constraint linear programming (QCLP) and quadratic constraint quadratic programming (QCQP). We select the subset of QP problems in QPLIB. The Maros-Mészáros QP set is a collection of convex quadratic programming examples from a variety of sources.

The results are shown in Table 3. Here the time limit is set to 3600 seconds. The geomean over all instances is calculated by the following formula:

$$\text{geomean} = \left(\prod_{i=1}^{n}(t_i + \zeta_0)\right)^{1/n} - \zeta_0, \tag{34}$$

where n denotes the number of instances, t_i denotes the time consumed by the i-th instance and $\zeta_0 > 0$ is a small value to avoid the case that $t_i = 0$. We set $\zeta_0 = 1$ second in the computation.

Table 3: Results on QP instances.

instance	CLP	HiGHS	Gurobi	DRSIP	instance	CLP	HiGHS	Gurobi	DRSIP
QPLIB_10034	-	-	2.76	18.01	QPLIB_10038	1684.232	-	3.29	15.97
QPLIB_8495	3.842	-	2.03	7.06	QPLIB_8515	1304.122	-	-	17.05
QPLIB_8559	-	43.24	3.67	4.54	QPLIB_8567	-	109.18	5.3	11.17
QPLIB_8602	4833.062	-	2.8	33.58	QPLIB_8616	-	405.54	0.24	2.75
QPLIB_8785	48.582	-	0.79	10.53	QPLIB_8790	6.812	-	3.48	2.05
QPLIB_8792	2.182	1101.98	1.69	1.92	QPLIB_8991	0.702	-	1.49	0.81
QPLIB_8845	-	-	0.36	1.29	QPLIB_8906	0.202	3.56	0.22	0.69
QPLIB_8938	0.082	-	0.24	1.16	AUG2D	0.142	320.7	1.03	2.28
AUG2DC	0.142	317.13	0.15	2.26	AUG2DCQP	0.222	3743.07	0.38	5.91
AUG2DQP	0.272	2940.76	0.48	6.96	AUG3D	0.022	5.58	0.05	0.79
AUG3DC	0.022	16.32	0.05	1.07	AUG3DCQP	0.042	23.05	0.09	1.14
AUG3DQP	0.042	7.01	0.09	1.56	BOYD1	71.092	7200.6	0.9	13.85
BOYD2	-	-	11.58	190.57	CONT-050	0.782	-	0.23	1.28
CONT-100	18.492	-	0.44	43.18	CONT-101	33.412	-	0.42	2.9
CONT-200	492.422	-	1.12	300.05	CONT-201	707.502	-	1.36	10.6
CONT-300	-	-	2.31	18.83	CVXQP1_S	0.002	0.01	0.06	0.08
CVXQP2_L	82.532	54.74	3.54	4.22	CVXQP2_M	0.112	0.08	0.2	0.25
CVXQP2_S	0.002	0.01	0.06	0.07	CVXQP3_M	1.952	0.63	0.45	0.37
CVXQP3_S	0.002	0.01	0.06	0.06	DPKLO1	0.002	0.02	0.04	0.12
DTOC3	0.062	794.05	0.16	2.12	DUAL1	0.002	0.03	0.06	0.1
DUAL2	0.002	0.05	0.06	0.11	DUAL3	0.002	0.06	0.07	0.12
DUAL4	0.002	0.02	0.06	0.08	DUALC1	0.032	0.01	0.12	0.08
DUALC2	0.042	0.02	0.1	0.09	DUALC5	0.042	0.01	0.14	0.08
DUALC8	0.352	0.02	0.14	0.15	EXDATA	81.402	26.41	2.76	28.01
GENHS28	0.002	0.01	0.04	0.07	GOULDQP2	0.242	-	0.06	0.15
GOULDQP3	0.242	0.08	0.06	0.11	HS118	0.002	0.01	0.07	0.7
HS21	0.002	0.01	0.05	1.1	HS268	0.002	0.01	0.05	0.06
HS35	0.002	0.01	0.05	0.05	HS35MOD	0.002	0.01	0.06	0.06
HS51	0.002	0.01	0.04	0.05	HS52	0.002	0.01	0.04	0.13
HS53	0.002	0.01	0.05	0.06	HS76	0.002	0.01	0.05	0.05
HUES-MOD	0.032	3254.38	0.07	1.48	HUESTIS	0.042	3177.74	0.07	0.24
KSIP	2.072	0.02	0.08	0.18	LASER	4.782	-	0.08	0.2
LISWET1	0.092	0.92	0.37	0.61	LISWET10	0.122	-	0.72	0.45
LISWET11	0.132	-	0.51	0.61	LISWET12	0.142	1.04	23.02	0.56
LISWET2	0.082	-	0.29	6.61	LISWET3	0.132	-	0.27	8.53
LISWET4	0.132	-	0.32	7.19	LISWET5	0.132	-	0.24	13.31
LISWET6	0.122	-	0.25	6.52	LISWET7	0.092	0.95	3.49	0.44
LISWET8	0.152	-	19.28	0.51	LISWET9	0.152	-	20.18	0.5
LOTSCHD	0.002	0.01	0.05	1.06	MOSARQP1	0.142	-	0.07	0.18
MOSARQP2	0.072	0.4	0.07	0.12	PRIMAL1	0.012	0.04	0.07	0.09
PRIMAL2	0.012	0.06	0.07	0.1	PRIMAL3	0.042	0.15	0.08	0.99
PRIMAL4	0.032	0.82	0.07	0.15	PRIMALC1	0.002	0.01	0.05	0.09
PRIMALC2	0.002	1.32	0.06	0.1	PRIMALC5	0.002	0.01	0.05	0.13
PRIMALC8	0.012	0.01	0.05	0.37	Q25FV47	1.512	-	0.38	2.31
QADLITTL	0.002	0.08	0.05	0.06	QAFIRO	0.002	0.01	0.06	0.07
QBANDM	0.032	-	0.07	0.12	QBEACONF	0.012	-	0.06	0.13
QBORE3D	0.012	-	0.06	0.22	QBRANDY	0.012	-	0.06	0.11
QCAPRI	-	-	0.14	0.56	QE226	0.462	-	0.07	0.88
QETAMACR	0.102	-	0.1	0.37	QFORPLAN	0.012	-	0.07	2.2
QGROW15	0.032	-	0.07	0.21	QGROW22	0.042	-	0.08	0.36
QGROW7	0.012	0.03	0.06	0.17	QISRAEL	0.032	-	0.07	0.13
QPCBLEND	0.002	0.02	0.06	1.26	QPCBOEI1	0.012	0.08	0.07	1.73
QPCSTAIR	0.032	0.1	0.08	0.2	QPTEST	0.002	0.01	0.05	0.07
QRECIPE	0.002	-	0.06	0.1	QSC205	0.002	0.01	0.06	0.11
QSCAGR25	0.042	-	0.06	0.15	QSCAGR7	0.002	4.37	0.06	0.07
QSCFXM1	0.022	2.68	0.08	0.17	QSCFXM2	0.042	-	0.1	0.24
QSCFXM3	0.062	-	0.11	0.31	QSCORPIO	0.002	0.02	0.06	0.1
QSCRS8	0.102	-	0.08	0.28	QSCSD1	0.002	0.04	0.06	0.1
QSCSD6	0.012	-	0.06	0.15	QSCSD8	0.022	-	0.07	0.61
QSCTAP1	0.012	0.02	0.07	0.13	QSCTAP2	0.022	0.21	0.08	0.18
QSCTAP3	0.042	0.55	0.1	0.23	QSEBA	0.202	0.04	0.2	0.25
QSHARE1B	0.012	-	0.07	0.06	QSHARE2B	0.002	0.01	0.06	0.07
QSHELL	0.452	-	0.49	0.72	QSHIP04L	0.032	0.26	0.07	0.21
QSHIP04S	0.012	-	0.07	0.15	QSHIP08L	0.162	0.42	0.22	0.66
QSHIP08S	0.072	0.15	0.15	0.31	QSHIP12L	0.322	-	0.26	1.01
QSHIP12S	0.082	-	0.19	0.38	QSIERRA	0.032	-	0.08	0.82
QSTAIR	0.032	0.06	0.08	0.16	QSTANDAT	0.022	-	0.07	0.16
S268	0.002	0.01	0.05	0.06	STADAT2	0.452	-	0.09	0.15
STADAT3	0.072	-	0.23	0.38	STCQP1	20.322	63.82	0.49	1.15
STCQP2	2.432	14.92	0.51	0.83	TAME	0.002	0.02	0.06	0.06
YAO	0.012	0.06	0.88	0.07	ZECEVIC2	0.002	0.01	0.06	1.05
geomean	1.92	79.47	1.00	1.11					

Note: "-" indicates that the precision limit is not reached within the time limit, and it is recorded as 3600 seconds in calculating the geometric mean.

From Table 3 we see all the solvers performs well on most of easy instances in Maros-Mészáros QP. On difficult instances, DRSIP outperforms the academical software (CLP, HiGHS) in efficiency and stability. In view of geomean, DRSIP lags behind Gurobi by about 11%. This is because Gurobi implements more efficient linear algebra operations and linear system solver than Eigen and Pardiso which we use.

5.3 CBLIB second-order conic programming dataset

The CBLIB [4] library is constructed with a primary focus on realistic optimization problems with second-order cones. The problems in CBLIB dataset are formulated as a general form SOCP as follows:

$$\begin{aligned} \min_{x} \quad & \langle c, x\rangle \\ \text{s.t.} \quad & b_l \le \mathcal{A}x \le b_u, \\ & x \in \mathcal{K}, l \le x \le u, \end{aligned} \tag{P}$$

where $b_l, l \in (\mathbb{R} \cup \{-\infty\})^n, b_u, u \in (\mathbb{R} \cup \{+\infty\})^n$ are the lower and upper bounds, respectively. $\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_p$ and each sub-cone $\mathcal{K}_i$ is either positive orthants, a quadratic cone or a rotated quadratic cone. By adding auxiliary variables and applying some linear algebra techniques, we can transform the above form to the standard form (P).

We test on the CBLIB problems listed on Hans Mittelmann's SOCP Benchmark [16]. In Table 4, we compare the running time of DRSIP with open-source IPM based solvers ECOS [3], SDPT3 [24] and Gurobi [7]. DRSIP outperforms both ECOS and SDPT3 on all problems except for the instance "2013_firL2Linfalph". In terms of runtime, as indicated by the geometric mean, DRSIP is approximately 13 times faster than ECOS and about 5 times faster than SDPT3, and about 4 times slower than Gurobi.

ECOS is designed primarily for solving SOCPs on embedded systems, which makes it challenging to handle large-scale problems, especially those with dense coefficient matrices. This limitation results in extremely large and dense linear systems being solved in each iteration. SDPT3, on the other hand, is a standard implementation of the interior point method, but it does not account for the dense columns associated with quadratic cone variables during the implementation. This shortcoming is evident in instances like "beam7" and "beam30", where SDPT3 fails to reach the desired precision. Additionally, SDPT3 struggles with robustness when solving large-scale SOCPs, particularly on ill-conditioned or dense problems, as seen in cases like "2013_wbNRL" and "db-plate-yield-line," where the solver fails to achieve the specified precision. Gurobi implements IPM with numerical strategies such as presolving, handling dense columns, rescaling, reordering and so on. Additionally, highly optimized linear algebra operations are implemented in Gurobi, especially for computations involving sparse matrices. As a result, ill-conditioned instances have little

impact on Gurobi, as it successfully solves all instances. As for the implementation of DRSIP, the dense column strategy can be directly applied to enhance its efficiency in tackling specific challenging problems. The incorporation of line search into DRSIP allows the algorithm to better handle ill-conditioned problems. Consequently, DRSIP successfully solved the most problems in Table 4, outperforming ECOS by 5 problems and SDPT3 by 4 problems.

While DRSIP performs well on most instances, there are two specific problems where it fails to converge. These non-converging instances are significantly ill-conditioned. By inspecting the logs, DRSIP initially performs well, rapidly improving accuracy to around 10^{-5} in the early iterations. However, due to the excessively large condition number of the Newton system, the specified precision is not reached in limited time.

Table 4: Results on CBLIB instances.

instance	ECOS	SDPT3	Gurobi	DRSIP
beam7	206	-	37	110
beam30	2465	-	176	722
chainsing-50000-1	-	36	12	6
chainsing-50000-2	-	63	15	11
chainsing-50000-3	-	-	12	26
db-joint-serensen	-	-	50	-
db-plate-yield-line	-	-	25	57
2013_dsNRL	-	69	58	-
2013_firL1	1305	798	40	110
2013_firL1Linfalph	2847	1433	80	429
2013_firL1Linfeps	2531	168	8	225
2013_firL2a	945	67	2	13
2013_firL2L1alph	202	133	16	92
2013_firL2L1eps	797	805	41	108
2013_firL2Linfalph	-	85	76	194
2013_firL2Linfeps	687	356	18	173
2013_firLinf	3479	62	64	116
2013_wbNRL	1333	-	18	28
geomean	1732	752	31	132

Note: "-" indicates that the precision limit is not reached within the time limit, and it is recorded as 3600 seconds in calculating the geometric mean.

6 Conclusions

In this paper, we present the Douglas-Rachford splitting interior point method (DRSIP), which effectively merges the operational simplicity of Douglas-Rachford splitting with the rapid convergence characteristics of Newton-type optimization. Our method showcases significant improvements in computational efficiency and scalability for large-scale conic linear programming problems, outperforming traditional methods in both synthetic and

real-world datasets. DRSIP's ability to adapt seamlessly to various conic structures without the need for extensive parameter tuning underscores its practical utility in diverse optimization scenarios, making it an invaluable tool in fields requiring robust and precise computational methods.

In the further development of DRSIP, we aim to explore the method's potential for solving more complex optimization problems, such as SDP+ problem and nonconvex cases. We are considering the application of predictor-corrector strategies to enhance the method's convergence rate and robustness. Additionally, refined implementation can further enhance the computational efficiency and scalability. We believe that these enhancements will further improve the method's performance and make it a valuable tool for a wide range of optimization problems.

References

[1] Caron, S., Zaki, A., Otta, P., Arnström, D., Carpentier, J., Yang, F., Leziart, P.-A.: qpbenchmark: Benchmark for quadratic programming solvers available in Python. https://github.com/qpsolvers/qpbenchmark, Version 2.2.2, Accessed 2024

[2] COIN-OR CLP. https://github.com/coin-or/Clp

[3] Domahidi, A., Chu, E., Boyd, S.: ECOS: An SOCP solver for embedded systems. In: Proceedings of the 2013 European Control Conference (ECC), pp. 3071–3076. IEEE (2013)

[4] Friberg, H. A.: CBLIB 2014: A benchmark library for conic mixed-integer and continuous optimization. Math. Program. Comput. **8**, 191–214 (2016)

[5] Furini, F., Traversi, E., Belotti, P., Frangioni, A., Gleixner, A., Gould, N., Liberti, L., Lodi, A., Misener, R., Mittelmann, H.: QPLIB: A library of quadratic programming instances. Math. Program. Comput. **11**, 237–265 (2019)

[6] Ge, D., Huangfu, Q., Wang, Z., Wu, J.: Cardinal Optimizer (COPT) user guide. https://guide.coap.online/copt/en-doc, Accessed 2022

[7] Gurobi Optimization, LLC.: Gurobi Optimizer Reference Manual. https://www.gurobi.com, Accessed 2021

[8] Guennebaud, G., Jacob, B., et al.: Eigen. URL: http://eigen.tuxfamily.org, Accessed 2010

[9] Huangfu, Q., Hall, J. A. J.: Parallelizing the dual revised simplex method. Math. Program. Comput. **10**(1), 119–142 (2018)

[10] Hu, J., Tian, T., Pan, S., Wen, Z.: On the local convergence of the semismooth Newton method for composite optimization. arXiv preprint arXiv:2211.01127 (2022)

[11] Karmarkar, N.: A new polynomial-time algorithm for linear programming. In: Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing, pp. 302–311 (1984)

[12] Lin, T., Ma, S., Ye, Y., Zhang, S.: An ADMM-based interior-point method for large-scale linear programming. Optim. Methods Softw. **36**(2-3), 389–424 (2021)

[13] Li, Y., Wen, Z., Yang, C., Yuan, Y.: A semismooth Newton method for semidefinite programs and its applications in electronic structure calculations. SIAM J. Sci. Comput. **40**(6), A4131–A4157 (2018)

[14] Liu, Y., Liu, H., Nie, H., Wen, Z.: A DRS-based path-following algorithm for linear programming. Preprint

[15] MOSEK ApS: The MOSEK optimization toolbox for MATLAB manual. Version 9.0. http://docs.mosek.com/9.0/toolbox/index.html, Accessed 2019

[16] Mittelmann, H. D.: An independent benchmarking of SDP and SOCP solvers. Math. Program. **95**(2), 407–430 (2003)

[17] MindOpt: MindOpt Solver. https://solver.damo.alibaba.com/, Accessed 2022

[18] O'Donoghue, B.: Operator splitting for a homogeneous embedding of the linear complementarity problem. SIAM J. Optim. **31**(3), 1999–2023 (2021)

[19] Petra, C. G., Schenk, O., Lubin, M., Gärtner, K.: An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization. SIAM J. Sci. Comput. **36**(2), C139–C162 (2014)

[20] Petra, C. G., Schenk, O., Anitescu, M.: Real-time stochastic optimization of complex energy systems on high-performance computers. Comput. Sci. Eng. **16**(5), 32–42 (2014)

[21] Sopasakis, P., Menounou, K., Patrinos, P.: SuperSCS: fast and accurate large-scale conic optimization. In: Proceedings of the 2019 18th European Control Conference (ECC), pp. 1500–1505. IEEE (2019)

[22] Sturm, J. F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optim. Methods Softw. **11**(1-4), 625–653 (1999)

[23] Themelis, A., Stella, L., Patrinos, P.: Douglas–Rachford splitting and ADMM for nonconvex optimization: accelerated and Newton-type linesearch algorithms. Comput. Optim. Appl. **82**(2), 395–440 (2022)

[24] Toh, K.-C., Todd, M. J., Tütüncü, R. H.: SDPT3—a MATLAB software package for semidefinite programming, version 1.3. Optim. Methods Softw. **11**(1-4), 545–581 (1999)

[25] Wen, Z., Goldfarb, D., Yin, W.: Alternating direction augmented Lagrangian methods for semidefinite programming. Math. Program. Comput. **2**(3), 203–230 (2010)

[26] Yang, L., Sun, D., Toh, K.-C.: SDPNAL+: a majorized semismooth Newton-CG augmented Lagrangian method for semidefinite programming with nonnegative constraints. Math. Program. Comput. **7**(3), 331–366 (2015)