An ADMM-Based Interior-Point Method for Large-Scale Linear Programming

Tianyi Lin * Shiqian Ma † Yinyu Ye ‡ Shuzhong Zhang §

May 29, 2018

Abstract

In this paper, we propose a new framework to implement interior point method (IPM) in order to solve some very large scale linear programs (LP). Traditional IPMs typically use Newton's method to approximately solve a subproblem that aims to minimize a log-barrier penalty function at each iteration. Due its connection to Newton's method, IPM is often classified as *second-order method* – a genre that is attached with stability and accuracy at the expense of scalability. Indeed, computing a Newton step amounts to solving a large system of linear equations, which can be efficiently implemented if the input data are reasonably-sized and/or sparse and/or well-structured. However, in case the above premises fail, then the challenge still stands on the way for a traditional IPM. To deal with this challenge, one approach is to apply the iterative procedure, such as preconditioned conjugate gradient method, to solve the system of linear equations. Since the linear system is different each iteration, it is difficult to find good pre-conditioner to achieve the overall solution efficiency. In this paper, an alternative approach is proposed. Instead of applying Newton's method, we resort to the alternating direction method of multipliers (ADMM) to approximately minimize the log-barrier penalty function at each iteration, under the framework of primal-dual path-following for a homogeneous self-dual embedded LP model. The resulting algorithm is an ADMM-Based Interior Point Method, abbreviated as ABIP in this paper. The new method inherits stability from IPM, and scalability from ADMM. Because of its self-dual embedding structure, ABIP is set to solve any LP without requiring prior knowledge about its feasibility. We conduct extensive numerical experiments testing ABIP with large-scale LPs from NETLIB and machine learning applications. The results demonstrate that ABIP compares favorably with existing LP solvers including SDPT3, MOSEK, DSDP-CG and SCS.

Keywords: Linear Programming, Homogeneous Self-Dual Embedding, Interior-Point Method, Central Path Following, ADMM, Iteration Complexity.

1 Introduction

By and large, traditional interior point method (IPM) for linear program (LP) is based on solving a sequence of log-barrier penalty subproblems using Newton's method [36, 47, 32]. It turns out that with a suitable penalty-parameter choice scheme, one step of Newton's method usually yields a very good initial solution for the next log-barrier penalty subproblem. As a result, the crux of IPMs

*Department of Industrial Engineering and Operations Research, UC Berkeley, Berkeley, CA 94720, USA.
†Department of Mathematics, University of California, Davis, CA 95616, USA.
‡Department of Management Science and Engineering, Stanford University, Stanford, CA 94305, USA
§Department of Industrial and Systems Engineering, University of Minnesota, Minneapolis, MN 55455, USA

boils down to the computation of Newton steps, which requires to solve system of linear equations. On the surface of it, computing Newton's direction then amounts to the Cholesky decomposition of a large (often ill-conditioned) matrix or even computing its inverse, which can be done when the dimensions are not exceedingly high, or the input data are sparse and/or well-structured. In general however, computing Newton's direction is computationally expensive. Because of this connection to Newton's method, IPM is often classified as a second-order approach. Typical to the second-order methods, IPM is known to be stable and accurate, but computing Newton's direction remains a challenge when the problem is dense and large scale. On the other hand, recently there has been considerable research attention paid on the so-called first-order methods, in that no Newton direction is needed. One very successful first-order method is called alternating direction method of multipliers (ADMM), which is essentially a gradient-based algorithm for the dual of a linearly constrained optimization model; see, for example, [20, 28, 18, 19, 15] and the recent survey papers [7, 14]. It turns out that the ADMM is highly scalable; however, it may suffer from numerical instability and it may take overly many iterations to compute an accurate solution. A natural question thus arises: Can we combine the benefits from both campuses? This paper aims to provide an affirmative answer to the afore question.

Before moving further to that question, let us first mention a beautiful technique which resolved a difficulty that baffled researchers in the early days of the IPM. The difficulty is that an IPM requires an interior feasible solution to begin with, but an LP may not even be feasible let alone availability of an interior feasible solution. To tackle this, Ye, Todd and Mizuno [48] proposed a homogeneous self-dual (HSD) reformulation of the original LP, which contains all the information that one may possibly care to obtain: an optimal solution if it exists; or, if the problem is infeasible or unbounded, a certificate that proves the case. Interestingly, the HSD has a ready interior feasible solution, while an optimal solution is guaranteed to exist. A central path following algorithm was subsequently proposed in [48] to solve the HSD. Later, Xu, Hung and Ye [42] proposed a simplified homogeneous self-dual model. However, in either cases computing the Newton directions remains to be the chore. In this paper, we propose to apply the ADMM to approximately solve the log-barrier penalty subproblems from the path-following scheme for the HSD. The resulting algorithm, ADMM-based IPM (abbreviated as ABIP henceforth), inherits advantages of both IPM and ADMM. It turns out that ABIP is robust, highly scalable, and achieves high accuracy. Moreover, as a benefit inherited from HSD, ABIP finds both primal and dual optimal solutions if they exist; otherwise it finds a certificate proving primal or dual infeasibility.

There have been lots of efforts to efficiently compute Newton's step in IPMs. Existing approaches adopted in IPM for computing Newton's step include sparse matrix factorization, Krylov subspace method and preconditioned conjugate gradient method (cf. [12, 23, 24, 17, 6]), but the performance of these methods highly depends on the structure of the problem and the input data. For example, the system of linear equation data of the IPM is changing every iteration and becomes more and more ill-conditioned, so that it is typically hard to find a good pre-conditioner for the CG method. Moreover, often we have to solve more than one system of linear equations with different right-hand-side vectors. By investigating the structure of the problem in HSD, we discover that ADMM can be used to approximately solve the log-barrier penalty problems very efficiently. Though connected with the classical operator splitting methods in [18, 16, 15], renaissance of ADMM in recent years was due to its success in signal processing [9, 44], image processing [21, 45], and machine learning problems; see a recent survey paper [7] and the references therein. An interesting and positive discovery through our study reported in this paper is twofold: 1) we find that the overall IPM strategy such as central-path following greatly improve the stability and robustness of the variable-splitting approach; 2) it turns out that the log-barrier penalty subproblem under the central path following scheme for the HSD is well-structured and can be efficiently solved by ADMM.

Recently, there are several attempts on designing first-order methods for solving LPs (and conic programs); see [41, 46, 49, 40]. Without an HSD framework, such methods are not suited for primal or dual infeasible problems. More recently, O'Donoghue *et al.* [33] proposed a split conic solver (SCS) for solving LPs, which applies ADMM to solve the homogeneous self-dual embedding of the original LP. Therefore, ABIP is similar to SCS in that they are both based on ADMM and HSD. However, conceptually ABIP is built within the IPM framework, so that it can be used as an optional solver when the data is large and dense for any existing IPM solver. In this sense, ABIP has an additional machinery to improve its solution robustness. For example, the scheme of ABIP can easily incorporate the notion of step-sizes and wide neighborhoods of the central path (cf. [38]). The efficacy of ABIP is confirmed by our extensive numerical tests on large-scale LPs from NETLIB and machine learning applications. Finally, we remark that extending ABIP to a more general convex conic optimization setting is straightforward.

1.1 Notation and Organization

Throughout this paper, we denote vectors by bold lower case letters, e.g., $\mathbf{x}$, and matrices by regular upper case letters, e.g., X. The transpose of a real vector $\mathbf{x}$ is denoted as $\mathbf{x}^\top$. For a vector $\mathbf{x}$, and a matrix X, $\|\mathbf{x}\|$ and $\|X\|$ denote the ℓ_2 norm and the matrix spectral norm, respectively. For two symmetric matrices A and B, $A \succeq B$ indicates that $A - B$ is symmetric positive semi-definite. The subscript, e.g., $\mathbf{x}_t$, denotes iteration counter. $\log(\mathbf{x})$ denotes the natural logarithm of $\mathbf{x}$. $\mathbf{e}$ denotes the vector of all ones. e_j denotes the coordinate vector with j-th entry being 1. I_d is an identity matrix with the dimension d. For two vectors $\mathbf{x}$ and $\mathbf{y}$, the Hadamard product is denoted as $\mathbf{x} \circ \mathbf{y} = (x_1 y_1, \ldots, x_n y_n)$.

The rest of the paper is organized as follows. In Section 2, we discuss some background of homogeneous self-dual embedding. In Section 3, we propose our ABIP method for solving the homogeneous self-dual embedding with log barrier functions. We also discuss how ABIP can be simplified and reduced to a matrix-free algorithm. The iteration complexity of ABIP is also analyzed. In Section 4, we propose several techniques that help improve the performance of ABIP in practice. In Section 5, we present extensive numerical results on large-scale LPs from NETLIB and machine learning applications and compare with several existing LP solvers. We make some concluding remarks in Section 6.

2 Homogeneous and Self-Dual Linear Programming

We are interested in solving the following *primal-dual* pair of linear programs (LP):

$$(P) \quad \begin{array}{ll} \min & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq 0, \end{array} \qquad (D) \quad \begin{array}{ll} \max & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} & A^\top \mathbf{y} + \mathbf{s} = \mathbf{c}, \\ & \mathbf{s} \geq 0, \end{array} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the primal variable, $\mathbf{y} \in \mathbb{R}^m$ and $\mathbf{s} \in \mathbb{R}^n$ are the dual variables, the problem data are $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{c} \in \mathbb{R}^n$ with $m \leq n$, and without loss of generality, we assume that A is of full row rank. The primal and dual optimal objective values are denoted as p^* and d^* respectively.

In addition to the celebrated simplex method of Dantzig [10] in 1940's, the interior point method (IPM), which was pioneered by Karmarkar [27] and intensively developed by many researchers in the 1980's and 1990's, has been a standard approach to solve linear program (1). In the early years of IPM, initial feasible interior solutions were assumed to be available at hand. Clearly, this assumption can be restrictive. To address this issue specifically, Ye *et al.* [48] proposed to solve the following homogeneous and self-dual linear programming with arbitrary initial points $\mathbf{x}^0 > 0$, $\mathbf{s}^0 > 0$ and $\mathbf{y}^0$:

$$
(HSD) \quad \begin{array}{lrrrrrrcl}
\min & & & & & & ((\mathbf{x}^0)^\top \mathbf{s}^0 + 1)\theta & & \\
\text{s.t.} & & A\mathbf{x} & -\mathbf{b}\tau & & & +\bar{\mathbf{b}}\theta & = & 0, \\
& -A^\top \mathbf{y} & & +\mathbf{c}\tau & -\mathbf{s} & & -\bar{\mathbf{c}}\theta & = & 0, \\
& \mathbf{b}^\top \mathbf{y} & -\mathbf{c}^\top \mathbf{x} & & & -\kappa & +\bar{\mathbf{z}}\theta & = & 0, \\
& -\bar{\mathbf{b}}^\top \mathbf{y} & +\bar{\mathbf{c}}^\top \mathbf{x} & -\bar{\mathbf{z}}\tau & & & & = & -(\mathbf{x}^0)^\top \mathbf{s}^0 - 1, \\
& \mathbf{y} \text{ free}, & \mathbf{x} \geq 0, & \tau \geq 0, & \mathbf{s} \geq 0, & \kappa \geq 0, & \theta \text{ free}, & &
\end{array} \tag{2}
$$

where

$$
\bar{\mathbf{b}} = \mathbf{b} - A\mathbf{x}^0, \quad \bar{\mathbf{c}} = \mathbf{c} - A^\top \mathbf{y}^0 - \mathbf{s}^0, \quad \bar{\mathbf{z}} = \mathbf{c}^\top \mathbf{x}^0 + 1 - \mathbf{b}^\top \mathbf{y}^0. \tag{3}
$$

The HSD (2) has many nice properties. In the following we give a partial list (cf. [48]).

Theorem 2.1 (Theorem 2 in [48]) *The following holds for (2):*

(i) The optimal value of (2) is zero, and for any feasible point $(\mathbf{y}, \mathbf{x}, \tau, \theta, \mathbf{s}, \kappa)$, it holds

$$
((\mathbf{x}^0)^\top \mathbf{s}^0 + 1) \cdot \theta = \mathbf{x}^\top \mathbf{s} + \tau\kappa.
$$

(ii) There is a feasible solution $(\mathbf{y}, \mathbf{x}, \tau, \theta, \mathbf{s}, \kappa)$ to (2) such that

$$
\mathbf{y} = \mathbf{y}^0, \ \mathbf{x} = \mathbf{x}^0, \ \tau = 1, \ \theta = 1, \ \mathbf{s} = \mathbf{s}^0, \ \kappa = 1.
$$

(iii) There is an optimal solution $(\mathbf{y}^*, \mathbf{x}^*, \tau^*, \theta^* = 0, \mathbf{s}^*, \kappa^*)$ such that $\mathbf{x}^* + \mathbf{s}^* > 0$ and $\tau^* + \kappa^* > 0$, which is called a strictly complementary solution.

Theorem 2.2 (Theorem 3 in [48]) *Let $(\mathbf{y}^*, \mathbf{x}^*, \tau^*, \theta^* = 0, \mathbf{s}^*, \kappa^*)$ be a strictly complementary solution for (2). Then:*

(i) (P) has a solution (feasible and bounded) if and only if $\tau^* > 0$. In this case, $\mathbf{x}^*/\tau^*$ is an optimal solution for (P) and $(\mathbf{y}^*/\tau^*, \mathbf{s}^*/\tau^*)$ is an optimal solution for (D).

(ii) If $\tau^* = 0$, then $\kappa^* > 0$, which implies that $\mathbf{c}^\top \mathbf{x}^* - \mathbf{b}^\top \mathbf{y}^* < 0$, i.e., at least one of $\mathbf{c}^\top \mathbf{x}^*$ and $-\mathbf{b}^\top \mathbf{y}^*$ is strictly less than zero. If $\mathbf{c}^\top \mathbf{x}^* < 0$ then (D) is infeasible; if $-\mathbf{b}^\top \mathbf{y}^* < 0$ then (P) is infeasible; and if both $\mathbf{c}^\top \mathbf{x}^* < 0$ and $-\mathbf{b}^\top \mathbf{y}^*$ then both (P) and (D) are infeasible.

Theorem 2.3 (Corollary 4 in [48]) *Let $(\bar{\mathbf{y}}, \bar{\mathbf{x}}, \bar{\tau}, \bar{\theta} = 0, \bar{\mathbf{s}}, \bar{\kappa})$ be any optimal solution for (2). If $\bar{\kappa} > 0$, then either (P) or (D) is infeasible.*

3 An ADMM-based Interior-Point Method

In [48], Ye *et al.* proposed an $O(\sqrt{n}L)$-iteration interior point algorithm to solve (2). However, like all interior-point methods, it requires solving a linear system at each iteration and therefore does not scale well for dense data. In this section, we propose our ABIP method which uses ADMM to solve the log-barrier penalty subproblems for HSD, and we show that the procedures of ABIP can be simplified. In this section, we also provide an iteration complexity analysis for ABIP.

3.1 The ABIP Method

For ease of presentation, we choose $\mathbf{y}^0 = 0$, $\mathbf{x}^0 = \mathbf{e}$, and $\mathbf{s}^0 = \mathbf{e}$, where $\mathbf{e}$ denotes the vector of all ones. By introducing a constant parameter $\beta > 0$ and constant variables $\mathbf{r} = 0$ and $\xi = -(\mathbf{x}^0)^\top \mathbf{s}^0 - 1 = -n - 1$, (2) can be rewritten as

$$\begin{array}{ll} \min & \beta(n+1)\theta + \mathbb{1}(\mathbf{r} = 0) + \mathbb{1}(\xi = -n-1) \\ \text{s.t.} & Q\mathbf{u} = \mathbf{v}, \\ & \mathbf{y} \text{ free}, \ \mathbf{x} \geq 0, \ \tau \geq 0, \ \theta \text{ free}, \ \mathbf{s} \geq 0, \ \kappa \geq 0, \end{array} \tag{4}$$

where

$$Q = \begin{bmatrix} 0 & A & -\mathbf{b} & \bar{\mathbf{b}} \\ -A^\top & 0 & \mathbf{c} & -\bar{\mathbf{c}} \\ \mathbf{b}^\top & -\mathbf{c}^\top & 0 & \bar{\mathbf{z}} \\ -\bar{\mathbf{b}}^\top & \bar{\mathbf{c}}^\top & -\bar{\mathbf{z}} & 0 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{y} \\ \mathbf{x} \\ \tau \\ \theta \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{r} \\ \mathbf{s} \\ \kappa \\ \xi \end{bmatrix},$$

$$\bar{\mathbf{b}} = \mathbf{b} - A\mathbf{e}, \quad \bar{\mathbf{c}} = \mathbf{c} - \mathbf{e}, \quad \bar{\mathbf{z}} = \mathbf{c}^\top \mathbf{e} + 1, \tag{5}$$

and the indicator function $\mathbb{1}(\mathcal{C})$ equals zero if the constraint $\mathcal{C}$ is satisfied, and equals $+\infty$ otherwise. The reason that we introduce a parameter β in the objective is completely for ease of presentation. It does not change the solution of the problem.

One classical way to solve (4) is to use log-barrier penalty to penalize the variables with non-negativity constraints, which results in a primal-dual interior-point method. The new formulation with log-barrier penalty is:

$$\begin{array}{ll} \min & B(\mathbf{u}, \mathbf{v}, \mu), \\ \text{s.t.} & Q\mathbf{u} = \mathbf{v}, \end{array} \tag{6}$$

where $B(\mathbf{u}, \mathbf{v}, \mu)$ is a barrier function defined as follows:

$$B(\mathbf{u}, \mathbf{v}, \mu) = \beta(n+1)\theta + \mathbb{1}(\mathbf{r} = 0) + \mathbb{1}(\xi = -n-1) - \mu \log(\mathbf{x}) - \mu \log(\mathbf{s}) - \mu \log(\tau) - \mu \log(\kappa), \tag{7}$$

and $\mu > 0$ is the penalty parameter. In the k-th iteration of IPM, one uses Newton's method to solve the KKT system of (6) with $\mu = \mu^k$. One then reduces μ^k to μ^{k+1} for the next iteration. When $\mu^k \to 0$, the solution of (6) approaches that of (4). The computational bottleneck of IPM is that one has to assemble a Newton's direction, which can be expensive when the problem is large and data are dense.

Observing the structure of (6), we propose to use the Alternating Direction Method of Multipliers (ADMM) to solve it inexactly. To do so, we first rewrite (6) as the following problem by introducing auxiliary variables $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$:

$$\begin{array}{ll} \min & \mathbb{1}(Q\tilde{\mathbf{u}} = \tilde{\mathbf{v}}) + B(\mathbf{u}, \mathbf{v}, \mu^k), \\ \text{s.t.} & (\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) = (\mathbf{u}, \mathbf{v}). \end{array} \tag{8}$$

By associating (scaled) Lagrange multipliers $\mathbf{p}$ to constraint $\tilde{\mathbf{u}} = \mathbf{u}$ and $\mathbf{q}$ to constraint $\tilde{\mathbf{v}} = \mathbf{v}$, the augmented Lagrangian function for (8) can be written as

$$\mathcal{L}_\beta(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, \mathbf{u}, \mathbf{v}, \mu^k, \mathbf{p}, \mathbf{q}) := \mathbb{1}(Q\tilde{\mathbf{u}} = \tilde{\mathbf{v}}) + B(\mathbf{u}, \mathbf{v}, \mu^k) - \langle \beta(\mathbf{p}, \mathbf{q}), (\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) - (\mathbf{u}, \mathbf{v}) \rangle + \frac{\beta}{2} \|(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}) - (\mathbf{u}, \mathbf{v})\|^2,$$

where $\beta > 0$ is the same parameter as in (4). The i-th iteration of ADMM for solving (8) is as follows:

$$(\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k) = \operatorname*{argmin}_{\tilde{\mathbf{u}}, \tilde{\mathbf{v}}} \ \mathcal{L}_\beta(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, \mathbf{u}_i^k, \mathbf{v}_i^k, \mu^k, \mathbf{p}_i^k, \mathbf{q}_i^k) = \prod_{Q\mathbf{u}=\mathbf{v}} \left(\mathbf{u}_i^k + \mathbf{p}_i^k, \mathbf{v}_i^k + \mathbf{q}_i^k\right), \tag{9}$$

$$(\mathbf{u}_{i+1}^k, \mathbf{v}_{i+1}^k) = \operatorname*{argmin}_{\mathbf{u}, \mathbf{v}} \ \mathcal{L}_\beta(\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k, \mathbf{u}, \mathbf{v}, \mu^k, \mathbf{p}_i^k, \mathbf{q}_i^k), \tag{10}$$

$$(\mathbf{p}_{i+1}^k, \mathbf{q}_{i+1}^k) = (\mathbf{p}_i^k, \mathbf{q}_i^k) - (\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k) + (\mathbf{u}_{i+1}^k, \mathbf{v}_{i+1}^k), \tag{11}$$

where $\prod_{\mathcal{S}}(\mathbf{x})$ denotes the Euclidean projection of $\mathbf{x}$ onto the set $\mathcal{S}$. A complete description of ABIP is in Algorithm 1.

Algorithm 1 The Basic ABIP for Linear Programming

```
Given parameters β > 0 and γ ∈ (0, 1). Set initial points (u_0^0, v_0^0), (p_0^0, q_0^0) and μ^0 > 0.
for k = 0, 1, 2, ... do
  for i = 0, 1, 2, ... do
    if the termination criterion is satisfied then
      break.
    end if
    Update (ũ_{i+1}^k, ṽ_{i+1}^k) by (9);
    Update (u_{i+1}^k, v_{i+1}^k) by (10);
    Update (p_{i+1}^k, q_{i+1}^k) by (11).
  end for
  Set (u_0^{k+1}, v_0^{k+1}) = (u_{i+1}^k, v_{i+1}^k) and (p_0^{k+1}, q_0^{k+1}) = (p_{i+1}^k, q_{i+1}^k);
  Set μ^{k+1} = γμ^k.
end for
```

3.2 Implementing ABIP

In this subsection we discuss a simplified implementation of ABIP. In particular, we show that the dual variables $\mathbf{p}$ and $\mathbf{q}$ in (9), (10) and (11) can be eliminated using a proper initialization. The framework of our analysis is similar to the one in [33], but the techniques we use are quite different because the *Moreau decomposition* cannot be directly applied to $(\mathbf{u}, \mathbf{v})$ when the log-barrier penalty function is used. The main technical result is summarized in the following theorem.

Theorem 3.1 *For the k-th outer iteration of Algorithm 1, we initialize $\mathbf{p}_0^k = \mathbf{v}_0^k$ and $\mathbf{q}_0^k = \mathbf{u}_0^k$ with*

$$\mathbf{x}_0^k \circ \mathbf{s}_0^k = \frac{\mu^k}{\beta}\mathbf{e}, \quad \tau_0^k \kappa_0^k = \frac{\mu^k}{\beta}, \quad \mathbf{r}_0^k = 0, \quad \xi_0^k = -n-1.$$

It then holds, for all iterations $i \geq 0$, that

$$\mathbf{p}_i^k = \mathbf{v}_i^k, \quad \mathbf{q}_i^k = \mathbf{u}_i^k, \quad \mathbf{x}_i^k \circ \mathbf{s}_i^k = \frac{\mu^k}{\beta}\mathbf{e}, \quad \tau_i^k \kappa_i^k = \frac{\mu^k}{\beta}, \quad \mathbf{r}_i^k = 0, \quad \xi_i^k = -n-1. \tag{12}$$

Proof. We shall prove the result by induction. Indeed, the proof is based on the following steps: (i) Iteration $j = 0$: the result holds true since we can initialize the variables accordingly. (ii) The result holds true for iteration $j = i + 1$ given that it holds true for iteration $j = i$. We prove the desired result in two steps:

Step 1: We claim that

$$\mathbf{u}_i^k + \mathbf{v}_i^k = \tilde{\mathbf{u}}_{i+1}^k + \tilde{\mathbf{v}}_{i+1}^k. \tag{13}$$

Indeed, we rewrite (9) as

$$\left(\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k\right) = \prod_{\mathcal{Q}} \left(\mathbf{u}_i^k + \mathbf{v}_i^k, \mathbf{u}_i^k + \mathbf{v}_i^k\right), \tag{14}$$

where $\mathcal{Q} = \{(\mathbf{u}, \mathbf{v}) : Q\mathbf{u} = \mathbf{v}\}$. Moreover, it follows from Q being skew-symmetric that the orthogonal complement of $\mathcal{Q}$ is $\mathcal{Q}^{\perp} = \{(\mathbf{v}, \mathbf{u}) : Q\mathbf{u} = \mathbf{v}\}$. Therefore, we conclude that,

$$(\mathbf{v}, \mathbf{u}) = \prod_{\mathcal{Q}^{\perp}} (\mathbf{z}, \mathbf{z}), \quad \text{if } (\mathbf{u}, \mathbf{v}) = \prod_{\mathcal{Q}} (\mathbf{z}, \mathbf{z}),$$

because the two projections are identical for reversed output arguments. This implies that

$$\left(\tilde{\mathbf{v}}_{i+1}^k, \tilde{\mathbf{u}}_{i+1}^k\right) = \prod_{\mathcal{Q}^{\perp}} \left(\mathbf{u}_i^k + \mathbf{v}_i^k, \mathbf{u}_i^k + \mathbf{v}_i^k\right). \tag{15}$$

Therefore, combining (14) and (15) yields the desired result.

Step 2: We proceed to proving that

$$\mathbf{p}_{i+1}^k = \mathbf{v}_{i+1}^k, \quad \mathbf{q}_{i+1}^k = \mathbf{u}_{i+1}^k, \quad \mathbf{x}_{i+1}^k \circ \mathbf{s}_{i+1}^k = \frac{\mu^k}{\beta}\mathbf{e}, \quad \tau_{i+1}^k \kappa_{i+1}^k = \frac{\mu^k}{\beta}, \quad \xi_{i+1}^k = -n-1,$$

given

$$\mathbf{p}_i^k = \mathbf{v}_i^k, \quad \mathbf{q}_i^k = \mathbf{u}_i^k \tag{16}$$

and

$$\mathbf{x}_i^k \circ \mathbf{s}_i^k = \frac{\mu^k}{\beta}\mathbf{e}, \quad \tau_i^k \kappa_i^k = \frac{\mu^k}{\beta}, \quad \xi_i^k = -n-1. \tag{17}$$

Indeed, we partition $\mathbf{p}$ and $\mathbf{q}$ as

$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_{\mathbf{y}} \\ \mathbf{p}_{\mathbf{x}} \\ \mathbf{p}_{\tau} \\ \mathbf{p}_{\theta} \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} \mathbf{q}_{\mathbf{r}} \\ \mathbf{q}_{\mathbf{s}} \\ \mathbf{q}_{\kappa} \\ \mathbf{q}_{\xi} \end{bmatrix},$$

and the optimality conditions of (10) are given by

$$0 = \mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k + (\mathbf{p}_{\mathbf{y}})_i^k, \tag{18}$$

$$0 = -\frac{\mu^k}{\beta} \cdot \frac{1}{\mathbf{x}_{i+1}^k} + \mathbf{x}_{i+1}^k - \tilde{\mathbf{x}}_{i+1}^k + (\mathbf{p}_{\mathbf{x}})_i^k, \tag{19}$$

$$0 = -\frac{\mu^k}{\beta} \cdot \frac{1}{\tau_{i+1}^k} + \tau_{i+1}^k - \tilde{\tau}_{i+1}^k + (\mathbf{p}_{\tau})_i^k, \tag{20}$$

$$0 = (n+1) + \theta_{i+1}^k - \tilde{\theta}_{i+1}^k + (\mathbf{p}_{\theta})_i^k, \tag{21}$$

$$0 = \mathbf{r}_{i+1}^k, \tag{22}$$

$$0 = -\frac{\mu^k}{\beta} \cdot \frac{1}{\mathbf{s}_{i+1}^k} + \mathbf{s}_{i+1}^k - \tilde{\mathbf{s}}_{i+1}^k + (\mathbf{q}_{\mathbf{s}})_i^k, \tag{23}$$

$$0 = -\frac{\mu^k}{\beta} \cdot \frac{1}{\kappa_{i+1}^k} + \kappa_{i+1}^k - \tilde{\kappa}_{i+1}^k + (\mathbf{q}_{\kappa})_i^k, \tag{24}$$

$$0 = \xi_{i+1}^k + n + 1. \tag{25}$$

First, we show that

$$\mathbf{r}_{i+1}^k = (\mathbf{p}_\mathbf{y})_{i+1}^k = 0, \qquad \mathbf{y}_{i+1}^k = (\mathbf{q}_\mathbf{r})_{i+1}^k.$$

Indeed, from (11), (18) and (22) we have that

$$(\mathbf{p}_\mathbf{y})_{i+1}^k = (\mathbf{p}_\mathbf{y})_i^k - \tilde{\mathbf{y}}_{i+1}^k + \mathbf{y}_{i+1}^k = 0 = \mathbf{r}_{i+1}^k.$$

Furthermore, we have

$$\begin{aligned}(\mathbf{q}_\mathbf{r})_{i+1}^k &\overset{(11)}{=} (\mathbf{q}_\mathbf{r})_i^k - \tilde{\mathbf{r}}_{i+1}^k + \mathbf{r}_{i+1}^k \overset{(22)}{=} (\mathbf{q}_\mathbf{r})_i^k - \tilde{\mathbf{r}}_{i+1}^k \overset{(13)}{=} (\mathbf{q}_\mathbf{r})_i^k - \left(\mathbf{r}_i^k + \mathbf{y}_i^k - \tilde{\mathbf{y}}_{i+1}^k\right) \\ &\overset{(18)}{=} (\mathbf{q}_\mathbf{r})_i^k - \left(\mathbf{r}_i^k + \mathbf{y}_i^k - \mathbf{y}_{i+1}^k - (\mathbf{p}_\mathbf{y})_i^k\right) = \mathbf{y}_i^k - \left(\mathbf{r}_i^k + \mathbf{y}_i^k - \mathbf{y}_{i+1}^k - \mathbf{r}_i^k\right) \overset{(16)}{=} \mathbf{y}_{i+1}^k.\end{aligned}$$

Second, we shall prove

$$\mathbf{x}_{i+1}^k = (\mathbf{q}_\mathbf{s})_{i+1}^k, \qquad \mathbf{s}_{i+1}^k = (\mathbf{p}_\mathbf{x})_{i+1}^k, \qquad \mathbf{x}_{i+1}^k \mathbf{s}_{i+1}^k = \frac{\mu^k}{\beta} \cdot \mathbf{e}.$$

Indeed, from (11) and (16) we have

$$(\mathbf{p}_\mathbf{x})_{i+1}^k = (\mathbf{p}_\mathbf{x})_i^k - \tilde{\mathbf{x}}_{i+1}^k + \mathbf{x}_{i+1}^k = \mathbf{s}_i^k - \tilde{\mathbf{x}}_{i+1}^k + \mathbf{x}_{i+1}^k,$$

and

$$(\mathbf{q}_\mathbf{s})_{i+1}^k = (\mathbf{q}_\mathbf{s})_i^k - \tilde{\mathbf{s}}_{i+1}^k + \mathbf{s}_{i+1}^k = \mathbf{x}_i^k - \tilde{\mathbf{s}}_{i+1}^k + \mathbf{s}_{i+1}^k.$$

Combining the above two equations with (13) yields

$$(\mathbf{p}_\mathbf{x})_{i+1}^k + (\mathbf{q}_\mathbf{s})_{i+1}^k = \mathbf{x}_{i+1}^k + \mathbf{s}_{i+1}^k. \tag{26}$$

Besides, from (19) and (11) we have

$$\frac{\mu^k}{\beta} \cdot \mathbf{e} = \mathbf{x}_{i+1}^k \circ \left(\mathbf{x}_{i+1}^k - \tilde{\mathbf{x}}_{i+1}^k + (\mathbf{p}_\mathbf{x})_i^k\right) = \mathbf{x}_{i+1}^k \circ (\mathbf{p}_\mathbf{x})_{i+1}^k, \tag{27}$$

and from (23) and (11) we have

$$\frac{\mu^k}{\beta} \cdot \mathbf{e} = \mathbf{s}_{i+1}^k \circ \left(\mathbf{s}_{i+1}^k - \tilde{\mathbf{s}}_{i+1}^k + (\mathbf{q}_\mathbf{s})_i^k\right) = \mathbf{s}_{i+1}^k \circ (\mathbf{q}_\mathbf{s})_{i+1}^k. \tag{28}$$

Therefore, we obtain

$$\begin{aligned} 0 &\overset{(27),(28)}{=} \mathbf{x}_{i+1}^k \circ (\mathbf{p}_\mathbf{x})_{i+1}^k - \mathbf{s}_{i+1}^k \circ (\mathbf{q}_\mathbf{s})_{i+1}^k \\ &\overset{(26)}{=} \mathbf{x}_{i+1}^k \circ \left(\mathbf{x}_{i+1}^k + \mathbf{s}_{i+1}^k - (\mathbf{q}_\mathbf{s})_{i+1}^k\right) - \mathbf{s}_{i+1}^k \circ (\mathbf{q}_\mathbf{s})_{i+1}^k \\ &= \left(\mathbf{x}_{i+1}^k - (\mathbf{q}_\mathbf{s})_{i+1}^k\right) \circ \left(\mathbf{x}_{i+1}^k + \mathbf{s}_{i+1}^k\right).\end{aligned}$$

Since $\mathbf{x}_{i+1}^k + \mathbf{s}_{i+1}^k > 0$, we conclude that $\mathbf{x}_{i+1}^k = (\mathbf{q}_\mathbf{s})_{i+1}^k$ which, combining with (28), leads to

$$\frac{\mu^k}{\beta} \cdot \mathbf{e} = \mathbf{x}_{i+1}^k \circ \mathbf{s}_{i+1}^k.$$

It also directly follows from (26) that $\mathbf{s}_{i+1}^k = (\mathbf{p_x})_{i+1}^k$. We use the same arguments to conclude

$$\tau_{i+1}^k = (\mathbf{q}_\kappa)_{i+1}^k, \qquad \kappa_{i+1}^k = (\mathbf{p}_\tau)_{i+1}^k, \qquad \tau_{i+1}^k \kappa_{i+1}^k = \frac{\mu^k}{\beta}.$$

Finally, we show that

$$\xi_{i+1}^k = (\mathbf{p}_\theta)_{i+1}^k = -n-1, \qquad \theta_{i+1}^k = (\mathbf{q}_\xi)_{i+1}^k = \frac{\mu^k}{\beta}.$$

Indeed, from (11), (21) and (25) we have

$$(\mathbf{p}_\theta)_{i+1}^k = (\mathbf{p}_\theta)_i^k - \tilde{\theta}_{i+1}^k + \theta_{i+1}^k = -n-1 = \xi_{i+1}^k.$$

Furthermore, combining (16) and (17) we have

$$(\mathbf{p}_\theta)_i^k = \xi_i^k = -n-1,$$

which implies that

$$\theta_{i+1}^k - \tilde{\theta}_{i+1}^k = (\mathbf{p}_\theta)_{i+1}^k - (\mathbf{p}_\theta)_i^k = 0. \tag{29}$$

Therefore, we conclude that

$$\begin{aligned}(\mathbf{q}_\xi)_{i+1}^k &\overset{(11)}{=} (\mathbf{q}_\xi)_i^k - \tilde{\xi}_{i+1}^k + \xi_{i+1}^k \overset{(13)}{=} (\mathbf{q}_\xi)_i^k - \theta_i^k - \xi_i^k + \tilde{\theta}_{i+1}^k + \xi_{i+1}^k \\ &\overset{(17),(25)}{=} (\mathbf{q}_\xi)_i^k - \theta_i^k + \tilde{\theta}_{i+1}^k \overset{(29)}{=} (\mathbf{q}_\xi)_i^k - \theta_i^k + \theta_{i+1}^k \overset{(16)}{=} \theta_{i+1}^k.\end{aligned}$$

This completes the proof. □

Observe that Theorem 3.1 simplifies Algorithm 1 by eliminating the dual variables $\mathbf{p}$ and $\mathbf{q}$. They are replaced by $\mathbf{v}$ and $\mathbf{u}$, respectively. Moreover, note that $\mathbf{u}$ and $\mathbf{v}$ are separable in (10). As a result, we can update $\mathbf{u}$ by

$$\mathbf{u}_{i+1}^k = \operatorname*{argmin}_{\mathbf{u}} \left[B(\mathbf{u}, \mu^k) + \frac{\beta}{2} \left\| \mathbf{u} - \tilde{\mathbf{u}}_{i+1}^k + \mathbf{v}_i^k \right\|^2 \right], \tag{30}$$

where

$$B(\mathbf{u}, \mu) := \beta(n+1)\theta - \mu \log(\mathbf{x}) - \mu \log(\tau), \tag{31}$$

and update $\mathbf{v}$ by

$$\mathbf{v}_{i+1}^k = \mathbf{v}_i^k - \tilde{\mathbf{u}}_{i+1}^k + \mathbf{u}_{i+1}^k, \tag{32}$$

which follows from the update for $\mathbf{p}_{i+1}^k$. Note that $\tilde{\mathbf{v}}_i^k$ can now be eliminated from the algorithm.

Problem (30) admits closed-form solutions given by

$$\mathbf{y}_{i+1}^k = \operatorname*{argmin}_{\mathbf{y}} \left[\frac{1}{2}\left\|\mathbf{y} - \tilde{\mathbf{y}}_{i+1}^k + \mathbf{r}_i^k\right\|^2\right] = \tilde{\mathbf{y}}_{i+1}^k, \tag{33}$$

$$\begin{aligned}\mathbf{x}_{i+1}^k &= \operatorname*{argmin}_{\mathbf{x}} \left[-\frac{\mu^k}{\beta}\log(\mathbf{x}) + \frac{1}{2}\left\|\mathbf{x} - \tilde{\mathbf{x}}_{i+1}^k + \mathbf{s}_i^k\right\|^2\right] \\ &= \frac{1}{2}\left[\left(\tilde{\mathbf{x}}_{i+1}^k - \mathbf{s}_i^k\right) + \sqrt{\left(\tilde{\mathbf{x}}_{i+1}^k - \mathbf{s}_i^k\right)\circ\left(\tilde{\mathbf{x}}_{i+1}^k - \mathbf{s}_i^k\right) + \frac{4\mu^k}{\beta}}\right],\end{aligned} \tag{34}$$

$$\begin{aligned}\tau_{i+1}^k &= \operatorname*{argmin}_{\tau} \left[-\frac{\mu^k}{\beta}\log(\tau) + \frac{1}{2}\left\|\tau - \tilde{\tau}_{i+1}^k + \kappa_i^k\right\|^2\right] \\ &= \frac{1}{2}\left[\left(\tilde{\tau}_{i+1}^k - \kappa_i^k\right) + \sqrt{\left(\tilde{\tau}_{i+1}^k - \kappa_i^k\right)\circ\left(\tilde{\tau}_{i+1}^k - \kappa_i^k\right) + \frac{4\mu^k}{\beta}}\right],\end{aligned} \tag{35}$$

$$\theta_{i+1}^k = \tilde{\theta}_{i+1}^k, \tag{36}$$

where the last step is from (29). By eliminating $\mathbf{p}_i^k$ and $\mathbf{q}_i^k$, (9) reduces to

$$\left(\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k\right) = \prod_{Q\mathbf{u}=\mathbf{v}}\left(\mathbf{u}_i^k + \mathbf{v}_i^k, \mathbf{u}_i^k + \mathbf{v}_i^k\right).$$

It is easy to show (by the KKT condition) that the solution is given by

$$\tilde{\mathbf{u}}_{i+1}^k = \left(I + Q^\top Q\right)^{-1}(I - Q)\left(\mathbf{u}_i^k + \mathbf{v}_i^k\right) = (I + Q)^{-1}\left(\mathbf{u}_i^k + \mathbf{v}_i^k\right), \tag{37}$$

because matrix Q is skew-symmetric. Moreover, we only need to invert (or factorize) $I + Q$ once at the beginning of the algorithm. In this sense, ABIP is matrix inversion free.

Therefore, we have shown that (9), (10) and (11) can be simply implemented by means of (37), (30) and (32) respectively, and the solutions of (30) are given by (33), (34), (35) and (36).

We use the following criterion to terminate the inner loop of Algorithm 1:

$$\left\|Q\mathbf{u}_i^k - \mathbf{v}_i^k\right\|^2 \le (\mu^k)^3. \tag{38}$$

Finally, we present this specific implementation ABIP as Algorithm 2.

Remark 3.2 *We denote* $(\mathbf{u}_k^*, \mathbf{v}_k^*)$ *as the optimal solution to* (6) *when* $\mu = \mu^k$, *which also satisfies the following optimality conditions of* (6)*:*

$$\begin{cases} Q\mathbf{u} - \mathbf{v} &= 0, \\ \mathbf{x}\circ\mathbf{s} &= \frac{\mu^k}{\beta}\mathbf{e}, \\ \tau\kappa &= \frac{\mu^k}{\beta}, \\ \theta &= \frac{\mu^k}{\beta}, \\ \mathbf{r} &= 0, \\ \xi &= -n-1, \\ (\mathbf{x},\mathbf{s},\tau,\kappa) &> 0. \end{cases} \tag{40}$$

Moreover, $(\mathbf{u}_k^*, \mathbf{v}_k^*)$ *is uniquely defined. In fact,* (40) *defines a central path (cf. [37, 5, 29, 36]) of the homogeneous self-dual embedded model ([48]).*

Algorithm 2 The Simplified ABIP Method for Linear Programming

Set $\mu^0 = \beta > 0$ and $\gamma \in (0,1)$.
Set $\mathbf{r}_0^0 = \mathbf{y}_0^0 = 0$, $\left(\mathbf{x}_0^0, \tau_0^0, \mathbf{s}_0^0, \kappa_0^0\right) = (\mathbf{e}, 1, \mathbf{e}, 1) > 0$, $\theta_0^0 = 1$, and $\xi_0^0 = -n-1$ with $\mathbf{x}_0^0 \circ \mathbf{s}_0^0 = \frac{\mu^0}{\beta}\mathbf{e}$, and $\tau_0^0 \kappa_0^0 = \frac{\mu^0}{\beta}$.
for $k = 0, 1, 2, \ldots$ **do**
  for $i = 0, 1, 2, \ldots$ **do**
    if the **inner** termination criterion (38) is satisfied **then**
      break.
    end if
    Update $\tilde{\mathbf{u}}_{i+1}^k$ by using (37);
    Update $\mathbf{u}_{i+1}^k$ by using (33), (34), (35) and (36);
    Update $\mathbf{v}_{i+1}^k$ by using (32).
    if the **final** termination criterion is satisfied **then**
      return.
    end if
  end for
  Set $\mu^{k+1} = \gamma \cdot \mu^k$;
  Set $\mathbf{r}_0^{k+1} = 0$, $\xi_0^{k+1} = -n-1$ and

$$\left(\mathbf{y}_0^{k+1}, \mathbf{x}_0^{k+1}, \mathbf{s}_0^{k+1}, \tau_0^{k+1}, \kappa_0^{k+1}, \theta_0^{k+1}\right) = \sqrt{\gamma} \cdot \left(\mathbf{y}_{i+1}^k, \mathbf{x}_{i+1}^k, \mathbf{s}_{i+1}^k, \tau_{i+1}^k, \kappa_{i+1}^k, \theta_{i+1}^k\right). \tag{39}$$

end for

From Theorem 3.1 we have

$$\begin{cases} Q\tilde{\mathbf{u}}_i^k - \tilde{\mathbf{v}}_i^k &= 0, \\ \mathbf{x}_i^k \circ \mathbf{s}_i^k &= \frac{\mu^k}{\beta}\mathbf{e}, \\ \tau_i^k \kappa_i^k &= \frac{\mu^k}{\beta}, \\ \mathbf{r}_i^k &= 0, \\ \xi_i^k &= -n-1, \\ \left(\mathbf{x}_i^k, \mathbf{s}_i^k, \tau_i^k, \kappa_i^k\right) &> 0, \end{cases}$$

and

$$\theta_i^k = \tilde{\theta}_i^k = \frac{\left(\tilde{\mathbf{x}}_i^k\right)^\top \tilde{\mathbf{s}}_i^k + \tilde{\tau}_i^k \tilde{\kappa}_i^k + \left(\tilde{\mathbf{y}}_i^k\right)^\top \tilde{\mathbf{r}}_i^k}{-\tilde{\xi}_i^k}.$$

Together with the feasibility condition that $\left\|\left(\tilde{\mathbf{u}}_i^k, \tilde{\mathbf{v}}_i^k\right) - \left(\mathbf{u}_i^k, \mathbf{v}_i^k\right)\right\| \to 0$ when $i \to +\infty$ (See Lemma 3.4), we conclude that the optimal solution to problem (8) is on the central path. This implies that ABIP is indeed a central path following algorithm, in view of the classical primal-dual central path following scheme.

Corollary 3.3 *Following a similar argument as in Theorem 3.1, it is easy to prove:*

$$\left(\mathbf{p}_k^*, \mathbf{q}_k^*\right) = \left(\mathbf{v}_k^*, \mathbf{u}_k^*\right), \tag{41}$$

where $\left(\mathbf{p}_k^*, \mathbf{q}_k^*\right)$ denotes the optimal dual solution of (8).

3.3 Iteration Complexity Analysis

In this subsection we analyze the iteration complexity of ABIP. The following identity will be frequently used in our analysis:

$$(a_1 - a_2)^\top (a_3 - a_4) = \frac{1}{2}\left(\|a_4 + a_2\|^2 - \|a_4 + a_1\|^2 + \|a_3 + a_1\|^2 - \|a_3 + a_2\|^2\right), \forall a_1, a_2, a_3, a_4. \tag{42}$$

To prove the main result, we need several technical lemmas.

Lemma 3.4 *Given $k \geq 1$, the sequence $\left\{\|\mathbf{u}_i^k - \mathbf{u}_k^*\|^2 + \|\mathbf{v}_i^k - \mathbf{v}_k^*\|^2\right\}_{i\geq 0}$ is monotonically decreasing and converges to* 0.

Proof. We observe from the optimality condition of problem (8) that

$$\beta(\mathbf{p}_k^*, \mathbf{q}_k^*) \in \partial \mathbb{1}(Q\mathbf{u} = \mathbf{v})[\mathbf{u}_k^*, \mathbf{v}_k^*], \quad -\beta(\mathbf{p}_k^*, \mathbf{q}_k^*) \in \partial B\left(\mathbf{u}_k^*, \mathbf{v}_k^*, \mu^k\right).$$

By using the convexity of $\mathbb{1}(Q\mathbf{u} = \mathbf{v})$ and (41) we have

$$0 \leq \beta\left(\mathbf{u}_k^* - \tilde{\mathbf{u}}_{i+1}^k, \mathbf{v}_k^* - \tilde{\mathbf{v}}_{i+1}^k\right)^\top (\mathbf{p}_k^*, \mathbf{q}_k^*) = \beta\left(\mathbf{u}_k^* - \tilde{\mathbf{u}}_{i+1}^k, \mathbf{v}_k^* - \tilde{\mathbf{v}}_{i+1}^k\right)^\top (\mathbf{v}_k^*, \mathbf{u}_k^*),$$

and using the convexity of $B(u, v, \mu)$ with respect to (u, v) and (41) we have

$$\beta(n+1)(\theta_k^* - \theta_{i+1}^k) \leq -\beta\left(\mathbf{u}_k^* - \mathbf{u}_{i+1}^k, \mathbf{v}_k^* - \mathbf{v}_{i+1}^k\right)^\top (\mathbf{p}_k^*, \mathbf{q}_k^*) = -\beta\left(\mathbf{u}_k^* - \mathbf{u}_{i+1}^k, \mathbf{v}_k^* - \mathbf{v}_{i+1}^k\right)^\top (\mathbf{v}_k^*, \mathbf{u}_k^*).$$

Summing up the above two inequalities leads to

$$\beta(n+1)(\theta_k^* - \theta_{i+1}^k) \leq \beta\left(\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k, \mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right)^\top (\mathbf{v}_k^*, \mathbf{u}_k^*). \tag{43}$$

Combining (11) and the optimality conditions of (9) and (10) yields

$$\left(\mathbf{p}_{i+1}^k, \mathbf{q}_{i+1}^k\right) - \left(\mathbf{u}_{i+1}^k, \mathbf{v}_{i+1}^k\right) + \left(\mathbf{u}_i^k, \mathbf{v}_i^k\right) \in \partial \mathbb{1}(Q\mathbf{u} = \mathbf{v})\left[\tilde{\mathbf{u}}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k\right], \tag{44}$$

$$-\beta\left(\mathbf{p}_{i+1}^k, \mathbf{q}_{i+1}^k\right) \in \partial B\left(\mathbf{u}_{i+1}^k, \mathbf{v}_{i+1}^k, \mu^k\right). \tag{45}$$

From the convexity of $\mathbb{1}(Q\mathbf{u} = \mathbf{v})$, we have

$$\begin{aligned} 0 &\overset{(44)}{\leq} \left(\tilde{\mathbf{u}}_{i+1}^k - \mathbf{u}_k^*, \tilde{\mathbf{v}}_{i+1}^k - \mathbf{v}_k^*\right)^\top \left(\mathbf{p}_{i+1}^k - \mathbf{u}_{i+1}^k + \mathbf{u}_i^k, \mathbf{q}_{i+1}^k - \mathbf{v}_{i+1}^k + \mathbf{v}_i^k\right) \\ &\overset{(42)}{=} \left(\tilde{\mathbf{u}}_{i+1}^k - \mathbf{u}_k^*, \tilde{\mathbf{v}}_{i+1}^k - \mathbf{v}_k^*\right)^\top \left(\mathbf{p}_{i+1}^k, \mathbf{q}_{i+1}^k\right) + \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2\right. \\ &\quad \left. - \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) + \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right). \end{aligned} \tag{46}$$

From the convexity of $B(\mathbf{u}, \mathbf{v}, \mu^k)$ with respect to $(\mathbf{u}, \mathbf{v})$ we have

$$\begin{aligned} \beta(n+1)\left(\theta_{i+1}^k - \theta_k^*\right) &= B\left(\mathbf{u}_{i+1}^k, \mathbf{v}_{i+1}^k, \mu^k\right) - B\left(\mathbf{u}_k^*, \mathbf{v}_k^*, \mu^k\right) \\ &\overset{(45)}{\leq} -\beta\left(\mathbf{u}_{i+1}^k - \mathbf{u}_k^*, \mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right)^\top \left(\mathbf{p}_{i+1}^k, \mathbf{q}_{i+1}^k\right). \end{aligned} \tag{47}$$

Adding (46) and (47) and using (12) yields

$$\begin{aligned} & \beta(n+1)(\theta_{i+1}^k - \theta_k^*) \\ \leq\ & \beta\left(\tilde{\mathbf{u}}_{i+1}^k - \mathbf{u}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k - \mathbf{v}_{i+1}^k\right)^\top \left(\mathbf{v}_{i+1}^k, \mathbf{u}_{i+1}^k\right) \\ & + \frac{\beta}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 - \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\ & + \frac{\beta}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right). \end{aligned} \tag{48}$$

Further adding (43) and (48) we have

$$\begin{aligned} 0 \leq\ & \left(\tilde{\mathbf{u}}_{i+1}^k - \mathbf{u}_{i+1}^k, \tilde{\mathbf{v}}_{i+1}^k - \mathbf{v}_{i+1}^k\right)^\top \left(\mathbf{v}_{i+1}^k - \mathbf{v}_k^*, \mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right) \\ & + \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 - \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\ & + \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right) \\ \overset{(11),(12)}{=}\ & \left(\mathbf{v}_i^k - \mathbf{v}_{i+1}^k, \mathbf{u}_i^k - \mathbf{u}_{i+1}^k\right)^\top \left(\mathbf{v}_{i+1}^k - \mathbf{v}_k^*, \mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right) \\ & + \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 - \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\ & + \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right) \\ \overset{(42)}{=}\ & \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right) - \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_{i+1}^k\right\|^2\right. \\ & \left. + \left\|\mathbf{v}_i^k - \mathbf{v}_{i+1}^k\right\|^2\right) + \frac{1}{2}\left(\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 - \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\ & + \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right). \end{aligned} \tag{49}$$

Recall that (11) implies

$$\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 = \left\|\mathbf{p}_i^k - \mathbf{p}_{i+1}^k\right\|^2 + \left\|\mathbf{q}_i^k - \mathbf{q}_{i+1}^k\right\|^2 = \left\|\mathbf{u}_i^k - \mathbf{u}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_{i+1}^k\right\|^2. \tag{50}$$

Now, we use (50) and (49) to obtain

$$\frac{1}{2}\left(\left\|\mathbf{u}_i^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_i^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \leq \left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2. \tag{51}$$

Therefore, we conclude that $\left\{\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2\right\}_{i\geq 0}$ is a monotonically decreasing sequence. Now that $\left\{\left(\mathbf{u}_i^k, \mathbf{v}_i^k\right)\right\}_{i\geq 0}$ is a bounded sequence, there must exist a subsequence $\left\{\left(\mathbf{u}_{i_j}^k, \mathbf{v}_{i_j}^k\right)\right\}_{j\geq 0}$ that converges to a limit point $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$. Since

$$-\beta\left(\mathbf{v}_{i_j}^k, \mathbf{u}_{i_j}^k\right) = -\beta\left(\mathbf{p}_{i_j}^k, \mathbf{q}_{i_j}^k\right) \in \partial B\left(\mathbf{u}_{i_j}^k, \mathbf{v}_{i_j}^k, \mu^k\right),$$

by letting $j \to +\infty$ we have

$$\begin{cases} \bar{\mathbf{x}} \circ \bar{\mathbf{s}} &= \left(\mu^k/\beta\right) \cdot \mathbf{e}, \\ \bar{\tau}\bar{\kappa} &= \mu^k/\beta, \\ \bar{\mathbf{r}} &= 0, \\ \bar{\xi} &= -n-1, \\ (\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\tau}, \bar{\kappa}) &> 0. \end{cases}$$

Furthermore, from (51) we have

$$\left\|\mathbf{u}_{i_j}^k - \tilde{\mathbf{u}}_{i_j+1}^k\right\|^2 + \left\|\mathbf{v}_{i_j}^k - \tilde{\mathbf{v}}_{i_j+1}^k\right\|^2 \to 0,$$

which implies that $(\tilde{\mathbf{u}}_{i_j+1}^k, \tilde{\mathbf{v}}_{i_j+1}^k)$ converges to $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$. Therefore, we have $Q\bar{\mathbf{u}} - \bar{\mathbf{v}} = 0$ and

$$\bar{\theta} = \frac{(\bar{\mathbf{x}})^\top \bar{\mathbf{s}} + \bar{\tau}\bar{\kappa} + (\bar{\mathbf{y}})^\top \bar{\mathbf{r}}}{-\bar{\xi}} = \frac{\mu^k}{\beta}.$$

Due to the uniqueness of the central path solution, we have $(\bar{\mathbf{u}}, \bar{\mathbf{v}}) = (\mathbf{u}_k^*, \mathbf{v}_k^*)$, which implies that

$$\left\|\mathbf{u}_{i_j}^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_{i_j}^k - \mathbf{v}_k^*\right\|^2 \longrightarrow 0, \quad \text{as } j \to +\infty.$$

Therefore, we conclude that

$$\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 \longrightarrow 0, \quad \text{as } i \to +\infty.$$

This completes the proof. □

Lemma 3.5 *The sequence* $\left\{\left\|\mathbf{u}_0^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_0^k - \mathbf{v}_k^*\right\|^2\right\}_{k\geq 0}$ *is uniformly bounded, i.e., there exists a constant* $C > 0$ *that does not depend on* k *or* μ^k *such that*

$$\left\|\mathbf{u}_0^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_0^k - \mathbf{v}_k^*\right\|^2 \leq C. \tag{52}$$

Moreover, the iterates $\left\{\left(\mathbf{u}_i^k, \mathbf{v}_i^k\right)\right\}$*, for* $k \geq 1$*,* $i = 0, 1, \ldots, N_k$*, are uniformly bounded, i.e., there exists a constant* $D > 0$ *that does not depend on* k *or* μ^k *such that*

$$\left\|\mathbf{u}_i^k\right\|^2 + \left\|\mathbf{v}_i^k\right\|^2 \leq D, \forall i = 0, 1, \ldots, N_k, \tag{53}$$

where N_k *denotes the number of inner iterations in the* k*-th outer loop.*

Proof. We recall an important fact that the set of the central path points $\{\mathbf{u}_k^*, \mathbf{v}_k^*\}$ with μ^k/β, i.e., the solution of (40), is uniformly bounded, where $0 < \mu^k \leq \mu^0$. That is, there exists a constant C_1 that does not depend on k or μ^k such that

$$\|\mathbf{u}_k^*\|^2 + \|\mathbf{v}_k^*\|^2 \leq C_1, \quad \forall k \geq 1. \tag{54}$$

Note that (52) leads to (53) immediately. To see this, note that combining (52) and Lemma 3.4 yields

$$\left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 \leq C, \forall i = 0, 1, \ldots, N_k$$

which together with (54) implies

$$\left\|\mathbf{u}_i^k\right\|^2 + \left\|\mathbf{v}_i^k\right\|^2 \leq 2C + 2C_1, \forall i = 0, 1, \ldots, N_k$$

proving (53) with $D = 2C + 2C_1$.

We prove (52) by induction. Indeed, for $k = 0$, since we choose $\mu^0 = \beta$, the initial point we choose in Algorithm 2 satisfies (40) automatically, i.e., $(\mathbf{u}_0^*, \mathbf{v}_0^*) = (\mathbf{u}_0^0, \mathbf{v}_0^0)$, and (38), i.e., $\left\|Q\mathbf{u}_0^0 - \mathbf{v}_0^0\right\|^2 \leq (\mu^0)^3$. Thus, we have

$$\left\|\mathbf{u}_0^0 - \mathbf{u}_0^*\right\|^2 + \left\|\mathbf{v}_0^0 - \mathbf{v}_0^*\right\|^2 = 0, \qquad \left\|\mathbf{u}_i^0\right\|^2 + \left\|\mathbf{v}_i^0\right\|^2 \leq C_1, \forall i = 0, 1, \ldots, N_0.$$

For $k \geq 1$, the induction assumption is that there exists a constant $C > 0$ that does not depend on k or μ^k such that

$$\left\|\mathbf{u}_0^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_0^k - \mathbf{v}_k^*\right\|^2 \leq C.$$

holds true. Then we have

$$\left\|\mathbf{u}_i^k\right\|^2 + \left\|\mathbf{v}_i^k\right\|^2 \leq D, \forall i = 0, 1, \ldots, N_k.$$

Recall that $(\mathbf{u}_k^*, \mathbf{v}_k^*)$ satisfies (40) which together with the above inequality and (54) results in

$$S_i^k \succeq \frac{\mu^k}{\beta\sqrt{D}}\mathbb{I}, \quad S_k^* \succeq \frac{\mu^k}{\beta\sqrt{D}}\mathbb{I}, \quad \kappa_i^k \geq \frac{\mu^k}{\beta\sqrt{D}}, \quad \kappa_k^* \geq \frac{\mu^k}{\beta\sqrt{D}},$$

from which we get

$$\left\|\mathbf{x}_i^k - \mathbf{x}_k^*\right\|^2 = \left(\frac{\mu^k}{\beta}\right)^2 \left\|\frac{1}{\mathbf{s}_i^k} - \frac{1}{\mathbf{s}_k^*}\right\|^2 = \left(\frac{\mu^k}{\beta}\right)^2 \left\|\left(S_i^k S_k^*\right)^{-1} \left(\mathbf{s}_i^k - \mathbf{s}_k^*\right)\right\|^2 \leq \left(\frac{\beta D}{\mu^k}\right)^2 \left\|\mathbf{s}_i^k - \mathbf{s}_k^*\right\|^2 \quad (55)$$

and

$$\left|\tau_i^k - \tau_k^*\right|^2 \leq \left(\frac{\beta D}{\mu^k}\right)^2 \left|\kappa_i^k - \kappa_k^*\right|^2. \quad (56)$$

Note that (38) holds for $i = N_k$. Therefore, we have

$$\begin{aligned} & \left(\theta_{N_k}^k - \theta_k^*\right)^2 \quad (57) \\ = \; & \frac{1}{\|\bar{\mathbf{b}}\|^2} \cdot \left\|-\bar{\mathbf{b}}\theta_{N_k}^k + \mathbf{b}\tau_{N_k}^k - A\mathbf{x}_{N_k}^k + \mathbf{r}_{N_k}^k - \mathbf{b}\left(\tau_{N_k}^k - \tau_k^*\right) + A\left(\mathbf{x}_{N_k}^k - \mathbf{x}_k^*\right) - \left(\mathbf{r}_{N_k}^k - \mathbf{r}_k^*\right)\right\|^2 \\ \overset{(38)}{\leq} \; & \frac{3}{\|\bar{\mathbf{b}}\|^2} \cdot \max\left\{1, \|\mathbf{b}\|^2, \|A\|^2\right\} \left(\left(\mu^k\right)^3 + \left\|\mathbf{x}_k^* - \mathbf{x}_{N_k}^k\right\|^2 + \left(\tau_k^* - \tau_{N_k}^k\right)^2\right), \end{aligned}$$

and

$$\begin{aligned} & \left\|\mathbf{y}_{N_k}^k - \mathbf{y}_k^*\right\|^2 \quad (58) \\ = \; & \left\|(AA^\top)^{-1}A\left(A^\top\mathbf{y}_{N_k}^k - A^\top\mathbf{y}_k^*\right)\right\|^2 \\ = \; & \left\|(AA^\top)^{-1}A\left(A^\top\mathbf{y}_{N_k}^k + \mathbf{s}_{N_k}^k - \mathbf{c}\tau_{N_k}^k + \bar{\mathbf{c}}\theta_{N_k}^k + \left(\mathbf{s}_k^* - \mathbf{s}_{N_k}^k\right) - \mathbf{c}\left(\tau_k^* - \tau_{N_k}^k\right) + \bar{\mathbf{c}}\left(\theta_k^* - \theta_{N_k}^k\right)\right)\right\|^2 \\ \overset{(38)}{\leq} \; & \frac{4\lambda_{\max}\left(A^\top A\right)}{\lambda_{\min}^2\left(AA^\top\right)} \cdot \max\left\{1, \|\mathbf{c}\|^2, \|\bar{\mathbf{c}}\|^2\right\} \cdot \left(\left(\mu^k\right)^3 + \left\|\mathbf{s}_k^* - \mathbf{s}_{N_k}^k\right\|^2 + \left(\tau_k^* - \tau_{N_k}^k\right)^2 + \left(\theta_k^* - \theta_{N_k}^k\right)^2\right). \end{aligned}$$

By defining function $\mathcal{P}(\mathbf{u},\mathbf{v}) = \|Q\mathbf{u}-\mathbf{v}\|^2$, and noticing that $\mathcal{P}(\mathbf{u},\mathbf{v})$ is a convex function and $\nabla^2_{\mathbf{v}}\mathcal{P}$ is an identity matrix, we have

$$\frac{1}{2}\left\|\mathbf{v}^k_{N_k}-\mathbf{v}^*_k\right\|^2 \leq \mathcal{P}\left(\mathbf{u}^k_{N_k},\mathbf{v}^k_{N_k},\mu^k\right)-\mathcal{P}\left(\mathbf{u}^*_k,\mathbf{v}^*_k,\mu^k\right) = \left\|Q\mathbf{u}^k_{N_k}-\mathbf{v}^k_{N_k}\right\|^2 \leq (\mu^k)^3. \tag{59}$$

By noting $(\mu^k)^3 \leq (\mu^0)^2\mu^k$, and combining (55)-(59), we have, there exists a constant C_2 that does not depend on k or μ^k such that

$$\left\|\mathbf{u}^k_{N_k}-\mathbf{u}^*_k\right\|^2+\left\|\mathbf{v}^k_{N_k}-\mathbf{v}^*_k\right\|^2 \leq C_2\mu^k. \tag{60}$$

Therefore, we conclude that

$$\begin{aligned}
& \left\|\mathbf{u}^{k+1}_0-\mathbf{u}^*_{k+1}\right\|^2+\left\|\mathbf{v}^{k+1}_0-\mathbf{v}^*_{k+1}\right\|^2 \\
= \quad & \left\|\mathbf{y}^{k+1}_0-\mathbf{y}^*_{k+1}\right\|^2+\left\|\mathbf{x}^{k+1}_0-\mathbf{x}^*_{k+1}\right\|^2+\left(\tau^{k+1}_0-\tau^*_{k+1}\right)^2+\left(\theta^{k+1}_0-\theta^*_{k+1}\right)^2 \\
& +\left\|\mathbf{s}^{k+1}_0-\mathbf{s}^*_{k+1}\right\|^2+\left(\kappa^{k+1}_0-\kappa^*_{k+1}\right)^2 \\
\overset{(39)}{\leq} \quad & 2\gamma\left(\left\|\mathbf{y}^k_{N_k}-\mathbf{y}^*_k\right\|^2+\left\|\mathbf{x}^k_{N_k}-\mathbf{x}^*_k\right\|^2+\left(\tau^k_{N_k}-\tau^*_k\right)^2+\left(\theta^k_{N_k}-\theta^*_k\right)^2+\left\|\mathbf{s}^k_{N_k}-\mathbf{s}^*_k\right\|^2+\left(\kappa^k_{N_k}-\kappa^*_k\right)^2\right) \\
& +2\left(\|\sqrt{\gamma}\mathbf{y}^*_k-\mathbf{y}^*_{k+1}\|^2+\|\sqrt{\gamma}\mathbf{x}^*_k-\mathbf{x}^*_{k+1}\|^2+(\sqrt{\gamma}\tau^*_k-\tau^*_{k+1})^2+(\sqrt{\gamma}\theta^*_k-\theta^*_{k+1})^2\right. \\
& \left.+\|\sqrt{\gamma}\mathbf{s}^*_k-\mathbf{s}^*_{k+1}\|^2+(\sqrt{\gamma}\kappa^*_k-\kappa^*_{k+1})^2\right) \\
\leq \quad & 2\gamma\left(\left\|\mathbf{u}^k_{N_k}-\mathbf{u}^*_k\right\|^2+\left\|\mathbf{v}^k_{N_k}-\mathbf{v}^*_k\right\|^2\right)+4\gamma\left(\|\mathbf{u}^*_k\|^2+\|\mathbf{v}^*_k\|^2\right)+4\gamma\left(\|\mathbf{u}^*_{k+1}\|^2+\|\mathbf{v}^*_{k+1}\|^2\right) \\
\overset{(54),(60)}{\leq} \quad & 2\gamma C_2\mu^k+8\gamma C_1 \leq 2\gamma C_2\mu^0+8\gamma C_1.
\end{aligned}$$

Therefore, letting $C = 2\gamma C_2\mu^0+8\gamma C_1$ proves (52). This completes the proof. □

Lemma 3.6 *The number of iterations (denoted by N_k) needed in the inner loop of Algorithm 2 is*

$$N_k \leq \log\left(\frac{2C\left(1+\|Q\|^2\right)}{\left(\mu^k\right)^3}\right)\left[\log\left(1+\min\left\{\frac{1}{C_3},\frac{\mu^k}{4DC_3\beta}\right\}\right)\right]^{-1}, \tag{61}$$

where C_3 is defined as

$$C_3 = \left[1+\frac{12\lambda_{\max}\left(A^\top A\right)}{\lambda^2_{\min}\left(AA^\top\right)}\cdot\max\left\{1,\|\mathbf{c}\|^2,\|\bar{\mathbf{c}}\|^2\right\}\right]\cdot\left[1+\frac{6}{\|\bar{\mathbf{b}}\|^2}\cdot\max\left\{\|\mathbf{b}\|^2,\|A\|^2\right\}\right] > 1. \tag{62}$$

Proof. It follows from Lemma 3.5 that $B\left(\mathbf{u}^k_i,\mathbf{v}^k_i,\mu^k\right)$ is strongly convex with respect to $(\mathbf{x},\mathbf{s},\tau,\kappa)$. More specifically, we have

$$\begin{aligned}
\nabla^2_{\mathbf{x}}B\left(\mathbf{u}^k_i,\mathbf{v}^k_i,\mu^k\right) = \mathrm{Diag}\left(\frac{\mu^k}{\mathbf{x}^k_i.*\mathbf{x}^k_i}\right) &\succeq \frac{\mu^k}{D}\cdot\mathbb{I}, \\
\nabla^2_{\mathbf{s}}B\left(\mathbf{u}^k_i,\mathbf{v}^k_i,\mu^k\right) = \mathrm{Diag}\left(\frac{\mu^k}{\mathbf{s}^k_i.*\mathbf{s}^k_i}\right) &\succeq \frac{\mu^k}{D}\cdot\mathbb{I}, \\
\nabla^2_{\tau}B\left(\mathbf{u}^k_i,\mathbf{v}^k_i,\mu^k\right) = \frac{\mu^k}{\left(\tau^k_i\right)^2} &\succeq \frac{\mu^k}{D}, \\
\nabla^2_{\kappa}B\left(\mathbf{u}^k_i,\mathbf{v}^k_i,\mu^k\right) = \frac{\mu^k}{\left(\kappa^k_i\right)^2} &\succeq \frac{\mu^k}{D}.
\end{aligned}$$

Therefore, (48) is changed to

$$
\begin{aligned}
&\beta(n+1)(\theta_{i+1}^k-\theta_k^*)+\frac{\mu^k}{2D}\left(\left\|\mathbf{x}_{i+1}^k-\mathbf{x}_k^*\right\|^2+\left\|\mathbf{s}_{i+1}^k-\mathbf{s}_k^*\right\|^2+\left(\tau_{i+1}^k-\tau_k^*\right)^2+\left(\kappa_{i+1}^k-\kappa_k^*\right)^2\right)\\
\leq\;& \beta\left(\tilde{\mathbf{u}}_{i+1}^k-\mathbf{u}_{i+1}^k,\tilde{\mathbf{v}}_{i+1}^k-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{v}_{i+1}^k,\mathbf{u}_{i+1}^k\right)\\
&+\frac{\beta}{2}\left(\left\|\mathbf{u}_i^k-\mathbf{u}_k^*\right\|^2+\left\|\mathbf{v}_i^k-\mathbf{v}_k^*\right\|^2-\left\|\mathbf{u}_i^k-\tilde{\mathbf{u}}_{i+1}^k\right\|^2-\left\|\mathbf{v}_i^k-\tilde{\mathbf{v}}_{i+1}^k\right\|^2\right)\\
&+\frac{\beta}{2}\left(\left\|\mathbf{u}_{i+1}^k-\tilde{\mathbf{u}}_{i+1}^k\right\|^2+\left\|\mathbf{v}_{i+1}^k-\tilde{\mathbf{v}}_{i+1}^k\right\|^2-\left\|\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right\|^2-\left\|\mathbf{v}_{i+1}^k-\mathbf{v}_k^*\right\|^2\right). \qquad (63)
\end{aligned}
$$

By summing (43) and (63), using (50) and observing

$$
\begin{aligned}
&\left(\mathbf{u}_{i+1}^k-\tilde{\mathbf{u}}_{i+1}^k,\mathbf{v}_{i+1}^k-\tilde{\mathbf{v}}_{i+1}^k\right)^\top(\mathbf{v}_k^*,\mathbf{u}_k^*)+\left(\tilde{\mathbf{u}}_{i+1}^k-\mathbf{u}_{i+1}^k,\tilde{\mathbf{v}}_{i+1}^k-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{v}_{i+1}^k,\mathbf{u}_{i+1}^k\right)\\
=\;&\left(\tilde{\mathbf{u}}_{i+1}^k-\mathbf{u}_{i+1}^k,\tilde{\mathbf{v}}_{i+1}^k-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{v}_{i+1}^k-\mathbf{v}_k^*,\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right)\\
=\;&\left(\mathbf{p}_i^k-\mathbf{p}_{i+1}^k,\mathbf{q}_i^k-\mathbf{q}_{i+1}^k\right)^\top\left(\mathbf{v}_{i+1}^k-\mathbf{v}_k^*,\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right)\\
=\;&\left(\mathbf{v}_i^k-\mathbf{v}_{i+1}^k,\mathbf{u}_i^k-\mathbf{u}_{i+1}^k\right)^\top\left(\mathbf{v}_{i+1}^k-\mathbf{v}_k^*,\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right)\\
=\;&\frac{1}{2}\left(\left\|\mathbf{u}_i^k-\mathbf{u}_k^*\right\|^2+\left\|\mathbf{v}_i^k-\mathbf{v}_k^*\right\|^2-\left\|\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right\|^2-\left\|\mathbf{v}_{i+1}^k-\mathbf{v}_k^*\right\|^2\right)-\frac{1}{2}\left(\left\|\mathbf{u}_i^k-\mathbf{u}_{i+1}^k\right\|^2+\left\|\mathbf{v}_i^k-\mathbf{v}_{i+1}^k\right\|^2\right),
\end{aligned}
$$

we obtain

$$
\begin{aligned}
&\frac{1}{2}\left(\left\|\mathbf{u}_i^k-\tilde{\mathbf{u}}_{i+1}^k\right\|^2+\left\|\mathbf{v}_i^k-\tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \qquad (64)\\
&+\frac{\mu^k}{2D\beta}\left(\left\|\mathbf{x}_{i+1}^k-\mathbf{x}_k^*\right\|^2+\left\|\mathbf{s}_{i+1}^k-\mathbf{s}_k^*\right\|^2+\left(\tau_{i+1}^k-\tau_k^*\right)^2+\left(\kappa_{i+1}^k-\kappa_k^*\right)^2\right)\\
\leq\;&\left\|\mathbf{u}_i^k-\mathbf{u}_k^*\right\|^2+\left\|\mathbf{v}_i^k-\mathbf{v}_k^*\right\|^2-\left\|\mathbf{u}_{i+1}^k-\mathbf{u}_k^*\right\|^2-\left\|\mathbf{v}_{i+1}^k-\mathbf{v}_k^*\right\|^2. \qquad (65)
\end{aligned}
$$

Moreover, from (45) we have

$$
0\leq B(\mathbf{u},\mathbf{v},\mu^k)-B\left(\mathbf{u}_{i+1}^k,\mathbf{v}_{i+1}^k,\mu^k\right)+\beta\left(\mathbf{u}-\mathbf{u}_{i+1}^k,\mathbf{v}-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{p}_{i+1}^k,\mathbf{q}_{i+1}^k\right), \qquad (66)
$$

and

$$
0\leq B(\mathbf{u},\mathbf{v},\mu^k)-B\left(\mathbf{u}_i^k,\mathbf{v}_i^k,\mu^k\right)+\beta\left(\mathbf{u}-\mathbf{u}_i^k,\mathbf{v}-\mathbf{v}_i^k\right)^\top\left(\mathbf{p}_i^k,\mathbf{q}_i^k\right). \qquad (67)
$$

Letting $(\mathbf{u},\mathbf{v})=\left(\mathbf{u}_i^k,\mathbf{v}_i^k\right)$ in (66) and $(\mathbf{u},\mathbf{v})=\left(\mathbf{u}_{i+1}^k,\mathbf{v}_{i+1}^k\right)$ in (67), and adding them up lead to

$$
\begin{aligned}
0\;\leq\;&-\left(\mathbf{u}_i^k-\mathbf{u}_{i+1}^k,\mathbf{v}_i^k-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{p}_i^k-\mathbf{p}_{i+1}^k,\mathbf{q}_i^k-\mathbf{q}_{i+1}^k\right)\\
=\;&\left(\mathbf{u}_i^k-\mathbf{u}_{i+1}^k,\mathbf{v}_i^k-\mathbf{v}_{i+1}^k\right)^\top\left(\mathbf{u}_{i+1}^k-\tilde{\mathbf{u}}_{i+1}^k,\mathbf{v}_{i+1}^k-\tilde{\mathbf{v}}_{i+1}^k\right)\\
=\;&\frac{1}{2}\left(\left\|\mathbf{u}_i^k-\tilde{\mathbf{u}}_{i+1}^k\right\|^2-\left\|\mathbf{u}_{i+1}^k-\tilde{\mathbf{u}}_{i+1}^k\right\|^2-\left\|\mathbf{u}_i^k-\mathbf{u}_{i+1}^k\right\|^2\right)\\
&+\frac{1}{2}\left(\left\|\mathbf{v}_i^k-\tilde{\mathbf{v}}_{i+1}^k\right\|^2-\left\|\mathbf{v}_{i+1}^k-\tilde{\mathbf{v}}_{i+1}^k\right\|^2-\left\|\mathbf{v}_i^k-\mathbf{v}_{i+1}^k\right\|^2\right),
\end{aligned}
$$

which implies that

$$\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2 \leq \left\|\mathbf{u}_{i}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2. \tag{68}$$

Combining (65) and (68) yields

$$\begin{aligned}
& \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\
& +\frac{\mu^k}{2D\beta}\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2 + \left(\kappa_{i+1}^k - \kappa_k^*\right)^2\right) \\
\leq\ & \left\|\mathbf{u}_{i}^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_{i}^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2.
\end{aligned} \tag{69}$$

Furthermore, we have (by denoting $C_4 := 6\lambda_{\max}(A^\top A)\max\{1, \|\mathbf{c}\|^2, \|\bar{\mathbf{c}}\|^2\}/\lambda_{\min}^2(AA^\top)$)

$$\begin{aligned}
& \left\|\mathbf{y}_{i+1}^k - \mathbf{y}_k^*\right\|^2 \\
\leq\ & 2\left(\left\|\mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k\right\|^2 + \left\|\tilde{\mathbf{y}}_{i+1}^k - \mathbf{y}_k^*\right\|^2\right) \\
=\ & 2\left\|\mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k\right\|^2 + 2\left\|(AA^\top)^{-1}A\left(A^\top\tilde{\mathbf{y}}_{i+1}^k - A^\top\mathbf{y}_k^*\right)\right\|^2 \\
=\ & 2\left\|\mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k\right\|^2 + 2\left\|(AA^\top)^{-1}A\left(\mathbf{c}\tilde{\tau}_{i+1}^k - \tilde{\mathbf{s}}_{i+1}^k - \bar{\mathbf{c}}\tilde{\theta}_{i+1}^k - \mathbf{c}\tau_k^* + \mathbf{s}_k^* + \bar{\mathbf{c}}\theta_k^*\right)\right\|^2 \\
\leq\ & C_4\left(\left\|\tilde{\mathbf{s}}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tilde{\tau}_{i+1}^k - \tau_k^*\right)^2 + \left(\tilde{\theta}_{i+1}^k - \theta_k^*\right)^2\right) + 2\left\|\mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k\right\|^2 \\
\leq\ & 2C_4\left(\left\|\mathbf{s}_{i+1}^k - \tilde{\mathbf{s}}_{i+1}^k\right\|^2 + \left(\tau_{i+1}^k - \tilde{\tau}_{i+1}^k\right)^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) \\
& +C_4\left(\theta_{i+1}^k - \theta_k^*\right)^2 + 2\left\|\mathbf{y}_{i+1}^k - \tilde{\mathbf{y}}_{i+1}^k\right\|^2,
\end{aligned} \tag{70}$$

and (by denoting $C_5 := 3\max\{\|\mathbf{b}\|^2, \|A\|^2\}/\|\bar{\mathbf{b}}\|^2$)

$$\begin{aligned}
& \left(\theta_{i+1}^k - \theta_k^*\right)^2 \\
=\ & \left(\tilde{\theta}_{i+1}^k - \theta_k^*\right)^2 = \frac{1}{\|\bar{\mathbf{b}}\|^2}\cdot\left\|\mathbf{b}\tilde{\tau}_{i+1}^k - A\tilde{\mathbf{x}}_{i+1}^k + \tilde{\mathbf{r}}_{i+1}^k - \mathbf{b}\tau_k^* + A\mathbf{x}_k^* - \mathbf{r}_k^*\right\|^2 \\
\leq\ & C_5\left(\left\|\tilde{\mathbf{x}}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tilde{\tau}_{i+1}^k - \tau_k^*\right)^2 + \left\|\tilde{\mathbf{r}}_{i+1}^k - \mathbf{r}_k^*\right\|^2\right) \\
=\ & C_5\left(\left\|\tilde{\mathbf{x}}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tilde{\tau}_{i+1}^k - \tau_k^*\right)^2 + \left\|\tilde{\mathbf{r}}_{i+1}^k - \mathbf{r}_{i+1}^k\right\|^2\right) \\
\leq\ & 2C_5\left(\left\|\mathbf{x}_{i+1}^k - \tilde{\mathbf{x}}_{i+1}^k\right\|^2 + \left(\tau_{i+1}^k - \tilde{\tau}_{i+1}^k\right)^2 + \left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) + C_5\left\|\tilde{\mathbf{r}}_{i+1}^k - \mathbf{r}_{i+1}^k\right\|^2.
\end{aligned} \tag{71}$$

By summing up (70) and (71), we have

$$
\begin{aligned}
& \left\|\mathbf{y}_{i+1}^k - \mathbf{y}_k^*\right\|^2 + \left(\theta_{i+1}^k - \theta_k^*\right)^2 \qquad (72)\\
\leq\ & 2(C_4+C_5)\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\
& + 2C_4\left(\left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2 + \left(\theta_{i+1}^k - \theta_k^*\right)^2\right) + 2C_5\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) \\
\overset{(71)}{\leq}\ & 2(C_4+C_5)\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) + 2C_4\left(\left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) \\
& + 2C_5\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) + 4C_4C_5\left(\left\|\mathbf{x}_{i+1}^k - \tilde{\mathbf{x}}_{i+1}^k\right\|^2 + \left(\tau_{i+1}^k - \tilde{\tau}_{i+1}^k\right)^2\right) \\
& + 4C_4C_5\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) + 2C_4C_5\left\|\tilde{\mathbf{r}}_{i+1}^k - \mathbf{r}_{i+1}^k\right\|^2 \\
\overset{(62)}{\leq}\ & C_3\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) + C_3\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right).
\end{aligned}
$$

Noticing

$$\|\mathbf{r}_{i+1}^k - \mathbf{r}_k^*\|^2 = (\xi_{i+1}^k - \xi_k^*)^2 = 0,$$

we have

$$
\begin{aligned}
& \min\left\{\frac{1}{2C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\left(\left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2\right) \\
\leq\ & \frac{\mu^k}{4DC_3\beta}\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2 + \left(\kappa_{i+1}^k - \kappa_k^*\right)^2\right) \\
& + \min\left\{\frac{1}{2C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\left(\left\|\mathbf{y}_{i+1}^k - \mathbf{y}_k^*\right\|^2 + \left(\theta_{i+1}^k - \theta_k^*\right)^2\right) \\
\leq\ & \frac{\mu^k}{4D\beta}\cdot\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2 + \left(\kappa_{i+1}^k - \kappa_k^*\right)^2\right) \\
& + \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\
& + \frac{\mu^k}{4D\beta}\cdot\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2\right) \\
\leq\ & \frac{1}{2}\left(\left\|\mathbf{u}_{i+1}^k - \tilde{\mathbf{u}}_{i+1}^k\right\|^2 + \left\|\mathbf{v}_{i+1}^k - \tilde{\mathbf{v}}_{i+1}^k\right\|^2\right) \\
& + \frac{\mu^k}{2D\beta}\left(\left\|\mathbf{x}_{i+1}^k - \mathbf{x}_k^*\right\|^2 + \left\|\mathbf{s}_{i+1}^k - \mathbf{s}_k^*\right\|^2 + \left(\tau_{i+1}^k - \tau_k^*\right)^2 + \left(\kappa_{i+1}^k - \kappa_k^*\right)^2\right) \\
\overset{(69)}{\leq}\ & \left\|\mathbf{u}_i^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_i^k - \mathbf{v}_k^*\right\|^2 - \left\|\mathbf{u}_{i+1}^k - \mathbf{u}_k^*\right\|^2 - \left\|\mathbf{v}_{i+1}^k - \mathbf{v}_k^*\right\|^2,
\end{aligned}
$$

where the second inequality is due to $C_3 > 1$ and (72). Therefore, we obtain that

$$
\begin{aligned}
\left\|\mathbf{u}_{N_k}^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_{N_k}^k - \mathbf{v}_k^*\right\|^2 &\leq \left(1 + \min\left\{\frac{1}{2C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\right)^{-N_k}\left(\left\|\mathbf{u}_0^k - \mathbf{u}_k^*\right\|^2 + \left\|\mathbf{v}_0^k - \mathbf{v}_k^*\right\|^2\right) \\
&\leq C\left(1 + \min\left\{\frac{1}{2C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\right)^{-N_k},
\end{aligned}
$$

On the other hand, we have

$$\left\|Q\mathbf{u}_{N_k}^k - \mathbf{v}_{N_k}^k\right\|^2 = \left\|Q(\mathbf{u}_{N_k}^k - \mathbf{u}_k^*) - (\mathbf{v}_{N_k}^k - \mathbf{v}_k^*)\right\|^2 \leq 2\|Q\|^2 \left\|\mathbf{u}_{N_k}^k - \mathbf{u}_k^*\right\|^2 + 2\left\|\mathbf{v}_{N_k}^k - \mathbf{v}_k^*\right\|^2.$$

Therefore, The number of iterations (denoted by N_k) needed in the inner loop of Algorithm 2 should satisfy that

$$2C\left(1+\|Q\|^2\right)\left(1+\min\left\{\frac{1}{2C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\right)^{-N_k} \leq \left(\mu^k\right)^3.$$

which proves (61). □

Now we are ready to present the main result of the iteration complexity of Algorithm 2.

Theorem 3.7 *The total IPM and ADMM iteration complexities of ABIP are respectively*

$$T_{\mathsf{IPM}} = O\left(\log\left(\frac{1}{\epsilon}\right)\right), \qquad T_{\mathsf{ADMM}} = O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right).$$

Proof. Note that ABIP consists of two types of loops: inner loops and outer loops. In the outer loop, a log-barrier penalty problem is formed with a penalty parameter μ^k, and in the inner loop, this log-barrier penalty problem is solved by a two-block ADMM. The outer loop is terminated when $\mu^k < \epsilon$, with a pre-given tolerance $\epsilon > 0$. It is then easy to see that the number of outer loops, i.e., the number of interior point iterations, is

$$T_{\mathsf{IPM}} = \left\lceil \frac{\log(\mu^0/\epsilon)}{\log(1/\gamma)} \right\rceil.$$

For the total number of ADMM steps, we have

$$\begin{aligned} T_{\mathsf{ADMM}} = \sum_{k=1}^{T_{\mathsf{IPM}}} N_k &= \sum_{k=1}^{T_{\mathsf{IPM}}} \log\left(\frac{2C\left(1+\|Q\|^2\right)}{\left(\mu^k\right)^3}\right)\left[\log\left(1+\min\left\{\frac{1}{C_3}, \frac{\mu^k}{4DC_3\beta}\right\}\right)\right]^{-1} \\ &= \sum_{k=1}^{T_{\mathsf{IPM}}} \log\left(\frac{2C\left(1+\|Q\|^2\right)/(\mu^0)^3}{\left(\gamma^k\right)^3}\right)\left[\log\left(1+\min\left\{\frac{1}{C_3}, \frac{\mu^0\gamma^k}{4DC_3\beta}\right\}\right)\right]^{-1} \\ &= O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right). \end{aligned}$$

This completes the proof. □

4 Implementation Details

4.1 Termination Criteria

So far we have not discussed how to terminate the outer loop of ABIP yet. In our implementation, we chose the one that is used in SCS [33]. Specifically, we run the algorithm until it finds a primal-dual optimal solution or a certificate of primal or dual infeasibility of the original LP pair (1), up to some tolerance. The detailed procedure is as follows. If $\tau_i^k > 0$ in $\mathbf{u}_i^k$, then let

$$\left(\frac{\mathbf{x}_i^k}{\tau_i^k}, \frac{\mathbf{s}_i^k}{\tau_i^k}, \frac{\mathbf{y}_i^k}{\tau_i^k}\right)$$

be the candidate solution which is guaranteed to satisfy the feasibility condition. It thus suffices to check if the residuals

$$\mathsf{pres}_i^k = \frac{A\mathbf{x}_i^k}{\tau_i^k} - \mathbf{b},\ \mathsf{dres}_i^k = \frac{A^\top \mathbf{y}_i^k}{\tau_i^k} + \frac{\mathbf{s}_i^k}{\tau_i^k} - \mathbf{c},\ \mathsf{dgap}_i^k = \frac{\mathbf{c}^\top \mathbf{x}_i^k}{\tau_i^k} - \frac{\mathbf{b}^\top \mathbf{y}_i^k}{\tau_i^k},$$

are small. More specifically, we terminate the algorithm if

$$\left\|\mathsf{pres}_i^k\right\| \le \epsilon_{\mathsf{pres}}\left(1 + \|\mathbf{b}\|\right),\ \left\|\mathsf{dres}_i^k\right\| \le \epsilon_{\mathsf{dres}}\left(1 + \|\mathbf{c}\|\right),\ \left\|\mathsf{dgap}_i^k\right\| \le \epsilon_{\mathsf{dgap}}\left(1 + \left|\mathbf{c}^\top \mathbf{x}\right| + \left|\mathbf{b}^\top \mathbf{y}\right|\right),$$

are met. The quantities ϵ_{pres}, ϵ_{dres} and ϵ_{dgap} are the primal residual, dual residue and duality gap tolerances, respectively.

On the other hand, if the current iterates satisfy that

$$\left\|A^\top \mathbf{y}_i^k + \mathbf{s}_i^k\right\| \le \epsilon_{\mathsf{pinfeas}}\left(\frac{\mathbf{b}^\top \mathbf{y}_i^k}{\|\mathbf{b}\|}\right), \tag{73}$$

then $\frac{\mathbf{y}_i^k}{\mathbf{b}^\top \mathbf{y}_i^k}$ is an approximate certificate for the primal infeasibility with the tolerance $\epsilon_{\mathsf{pinfeas}}$; or if the current iterates satisfy that

$$\left\|A\mathbf{x}_i^k\right\| \le \epsilon_{\mathsf{dinfeas}}\left(\frac{-\mathbf{c}^\top \mathbf{x}_i^k}{\|\mathbf{c}\|}\right), \tag{74}$$

then $-\frac{\mathbf{x}_i^k}{\mathbf{c}^\top \mathbf{x}_i^k}$ is an approximate certificate for the dual infeasibility with the tolerance $\epsilon_{\mathsf{dinfeas}}$.

4.2 Over Relaxation

In practice, we implemented some techniques that can accelerate the algorithm. One of them is to incorporate a relaxation parameter in the ADMM [15]. Specifically, when applied to Algorithm 2, we replace all $\tilde{\mathbf{u}}_{i+1}^k$ in the $\mathbf{u}$- and $\mathbf{v}$-updates with

$$\alpha\tilde{\mathbf{u}}_{i+1}^k + (1-\alpha)\mathbf{u}_i^k,$$

where $\alpha \in [0, 2]$ is a relaxation parameter. In that case, (32), (33), (34), (35) and (36) become

$$\mathbf{v}_{i+1}^k = \mathbf{v}_i^k - \alpha\tilde{\mathbf{u}}_{i+1}^k - (1-\alpha)\mathbf{u}_i^k + \mathbf{u}_{i+1}^k,$$

and

$$\begin{aligned}
\mathbf{y}_{i+1}^k &= \alpha\tilde{\mathbf{y}}_{i+1}^k + (1-\alpha)\mathbf{y}_i^k, \\
\mathbf{x}_{i+1}^k &= \frac{1}{2}\left[\left(\alpha\tilde{\mathbf{x}}_{i+1}^k + (1-\alpha)\mathbf{x}_i^k - \mathbf{s}_i^k\right) + \sqrt{\left(\alpha\tilde{\mathbf{x}}_{i+1}^k + (1-\alpha)\mathbf{x}_i^k - \mathbf{s}_i^k\right) \circ \left(\alpha\tilde{\mathbf{x}}_{i+1}^k + (1-\alpha)\mathbf{x}_i^k - \mathbf{s}_i^k\right) + \frac{4\mu^k}{\beta}}\right], \\
\tau_{i+1}^k &= \frac{1}{2}\left[\left(\alpha\tilde{\tau}_{i+1}^k + (1-\alpha)\tau_i^k - \kappa_i^k\right) + \sqrt{\left(\alpha\tilde{\tau}_{i+1}^k + (1-\alpha)\tau_i^k - \kappa_i^k\right) \circ \left(\alpha\tilde{\tau}_{i+1}^k + (1-\alpha)\tau_i^k - \kappa_i^k\right) + \frac{4\mu^k}{\beta}}\right], \\
\theta_{i+1}^k &= \alpha\tilde{\theta}_{i+1} + (1-\alpha)\theta_i^k,
\end{aligned}$$

When $\alpha = 1$, this reduces to the corresponding update in Algorithm 2 given above. When $\alpha > 1$, this is known as *over-relaxation*; when $\alpha < 1$, this is known as *under-relaxation*. Previous works [13, 34] suggest that the performance of ADMM can be improved significantly if one sets $\alpha \approx 1.5$.

4.3 Barzilai-Borwein Spectral Method for Selecting β

The performance of ADMM highly depends on the choice of β. One way to accelerate ADMM is to adaptively adjust β (see also [26, 41]). In practice, we generated a sequence of $\{\beta^k\}_{k\geq 0}$, where β^k is only used in the k-th outer iteration. Intuitively, the speed of traveling along the central path is determined by μ^k and β, implying that adjusting β in each outer iteration based on the iterates is equivalent to a predictor-corrector method [30]. The way we adaptively adjust β is based on the Barzilai-Borwein spectral method [4, 43], which is proven to be superior than the residue balancing approach [41]. Indeed, for each $k \geq 0$, we select β^k using **spectral stepsize estimation** and **safeguarding** at the beginning of the k-th outer iteration.

Spectral stepsize estimation: We calculate the first three iterates, i.e., $\left(\tilde{\mathbf{u}}_i^k, \mathbf{u}_i^k, \mathbf{v}_i^k\right)_{i=0}^{i=2}$, using a fixed $\beta^k > 0$ and an initial point $\left(\mathbf{y}_0^k, \mathbf{x}_0^k, \mathbf{s}_0^k, \tau_0^k, \kappa_0^k, \theta_0^k\right)$ obtained by (39) and $\mathbf{r}_0^k = 0$, $\xi_0^k = -n-1$. Then we estimate the curvature, i.e.,

$$\hat{\mathbf{v}}_1^k = \mathbf{v}_0^k - \alpha\tilde{\mathbf{u}}_1^k - (1-\alpha)\mathbf{u}_0^k + \alpha\mathbf{u}_1^k, \quad \hat{\mathbf{v}}_2^k = \mathbf{v}_1^k - \alpha\tilde{\mathbf{u}}_2^k - (1-\alpha)\mathbf{u}_1^k + \alpha\mathbf{u}_2^k,$$

and

$$\Delta\hat{\mathbf{v}}^k = \hat{\mathbf{v}}_2^k - \hat{\mathbf{v}}_1^k, \quad \Delta\tilde{\mathbf{u}}^k = \alpha\left(\tilde{\mathbf{u}}_2^k - \tilde{\mathbf{u}}_1^k\right) + (1-\alpha)\left(\mathbf{u}_1^k - \mathbf{u}_0^k\right).$$

As is typical in Barzilai-Borwein stepsize gradient methods [4], the spectral stepsizes φ_{SD}^k and φ_{MG}^k have the closed-form solutions as

$$\varphi_{\mathsf{SD}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\hat{\mathbf{v}}^k\rangle}{\langle\Delta\hat{\mathbf{v}}^k, \Delta\tilde{\mathbf{u}}^k\rangle}, \quad \varphi_{\mathsf{MG}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\tilde{\mathbf{u}}^k\rangle}{\langle\Delta\tilde{\mathbf{u}}^k, \Delta\tilde{\mathbf{u}}^k\rangle},$$

where SD and MG refer to *steepest descent* and *minimum gradient*, respectively, and $\varphi_{\mathsf{SD}}^k \geq \varphi_{\mathsf{MG}}^k$ due to the Cauchy-Schwarz inequality. We then consider the hybrid stepsize rule proposed in [50],

$$\varphi^k = \begin{cases} \varphi_{\mathsf{MG}}^k, & \text{if } 2\varphi_{\mathsf{MG}}^k > \varphi_{\mathsf{SD}}^k, \\ \varphi_{\mathsf{SD}}^k - \varphi_{\mathsf{MG}}^k/2, & \text{otherwise.} \end{cases} \tag{75}$$

Similarly, we calculate

$$\Delta\mathbf{u}^k = -\left(\mathbf{u}_2^k - \mathbf{u}_1^k\right),$$

and the spectral stepsizes ψ_{SD}^k and ψ_{MG}^k as

$$\psi_{\mathsf{SD}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\hat{\mathbf{v}}^k\rangle}{\langle\Delta\hat{\mathbf{v}}^k, \Delta\mathbf{u}^k\rangle}, \quad \psi_{\mathsf{MG}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\mathbf{u}^k\rangle}{\langle\Delta\mathbf{u}^k, \Delta\mathbf{u}^k\rangle},$$

and consider the hybrid stepsize rule again,

$$\psi^k = \begin{cases} \psi_{\mathsf{MG}}^k, & \text{if } 2\psi_{\mathsf{MG}}^k > \psi_{\mathsf{SD}}^k, \\ \psi_{\mathsf{SD}}^k - \psi_{\mathsf{MG}}^k/2, & \text{otherwise.} \end{cases} \tag{76}$$

Safeguarding: We suggest a safeguarding heuristic by accessing the quality of the curvature estimates, i.e., only update the stepsize if the curvature estimates satisfy a reliability criterion. More specifically, we consider the following quantities defined in [43]:

$$\varphi_{\mathsf{cor}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\tilde{\mathbf{u}}^k\rangle}{\|\Delta\hat{\mathbf{v}}^k\|\,\|\Delta\tilde{\mathbf{u}}^k\|}, \quad \psi_{\mathsf{cor}}^k = \frac{\langle\Delta\hat{\mathbf{v}}^k, \Delta\mathbf{u}^k\rangle}{\|\Delta\hat{\mathbf{v}}^k\|\,\|\Delta\mathbf{u}^k\|}.$$

The spectral stepsizes are updated only if the estimation is sufficiently reliable, i.e.,

$$\hat{\beta}^k = \begin{cases} \sqrt{\varphi^k \psi^k}, & \text{if } \varphi^k_{\mathsf{cor}} > \epsilon_{\mathsf{cor}} \text{ and } \psi^k_{\mathsf{cor}} > \epsilon_{\mathsf{cor}}, \\ \varphi^k, & \text{if } \varphi^k_{\mathsf{cor}} > \epsilon_{\mathsf{cor}} \text{ and } \psi^k_{\mathsf{cor}} \leq \epsilon_{\mathsf{cor}}, \\ \psi^k, & \text{if } \varphi^k_{\mathsf{cor}} \leq \epsilon_{\mathsf{cor}} \text{ and } \psi^k_{\mathsf{cor}} > \epsilon_{\mathsf{cor}}, \\ \beta^k, & \text{otherwise}, \end{cases} \tag{77}$$

where $\epsilon_{\mathsf{cor}} > 0$ is a quality threshold for the curvature estimates. Notice that $\hat{\beta}^k = \beta^k$ when both curvature estimates are deemed inaccurate while $\hat{\beta}^k \neq \beta^k$ but $\hat{\beta}^k \approx \beta^k$ implies that $\hat{\beta}^k$ and β^k are both suitable to be used in the k-th outer iteration.

Near-optimal selection: We select a near-optimal threshold $\epsilon_{\mathsf{penalty}} > 0$ and set

1. If $\hat{\beta}^k \neq \beta^k$ and $\left|\hat{\beta}^k - \beta^k\right| \leq \epsilon_{\mathsf{penalty}}$, then $\frac{\beta^k + \hat{\beta}^k}{2}$ will be used in the k-th outer iteration.
2. If $\hat{\beta}^k \neq \beta^k$ and $\left|\hat{\beta}^k - \beta^k\right| > \epsilon_{\mathsf{penalty}}$, then $\beta^k = \hat{\beta}^k$ and we redo the spectral step-size estimation and safeguarding with the same initial point.
3. If $\hat{\beta}^k = \beta^k$, then the spectral step-size estimation and safeguarding will be continued based on the subsequent iterates.

4.4 Presolving and Postsolving

Now we discuss issues in analyzing large and sparse LPs prior to solving them with ABIP. Firstly, we remove several computational expensive sub-procedures, e.g., forcing, dominated and duplicate rows and columns procedures, as used in [3, 22, 31, 25]. Secondly, we use Dulmage-Mendelsohn decomposition [35] to remove all the dependent rows in A and reformulate the original problem in the form of problem (1).

We consider LPs formulated in the following form:

$$\begin{array}{ll} \min & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{w}, \end{array}$$

where A has some linearly dependent rows. Before solving them with ABIP, we run a simple presolve procedure. More specifically, we detect and remove empty rows, singleton rows, fixed variables and empty columns, together with removing all the linearly dependent rows, i.e.,

(P1) An empty row: $\exists i: \ a_{ij} = 0, \ \forall j$. Either the i-th constraint is redundant or infeasible.

(P2) An empty column: $\exists j: \ a_{ij} = 0, \ \forall i$. x_j is fixed at one of its bounds or the problem is unbounded.

(P3) An infeasible variable: $\exists j: \ l_j > w_j$. The problem is trivially infeasible.

(P4) A fixed variable: $\exists j: \ l_j = w_j$. x_j can be substituted out of the problem.

(P5) A free variable: $\exists j: \ l_j = -\infty, \ w_j = \infty$. x_j can be substituted by two nonnegative variables x_j^+ and x_j^-.

(P6) A singleton row: $\exists (i,k): \ a_{ij} = 0, \ \forall j \neq k, \ a_{jk} \neq 0$. The i-th constraint fixes variable $x_j = \frac{b_i}{a_{ik}}$.

(P7) Dulmage-Mendelsohn decomposition. All the linearly dependent rows are detected and removed.

Then we have a reduced LP problem as follows,

$$\begin{array}{ll} \min & \tilde{\mathbf{c}}^\top \tilde{\mathbf{x}} \\ \text{s.t.} & \tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}, \\ & \tilde{\mathbf{l}} \leq \tilde{\mathbf{x}} \leq \tilde{\mathbf{w}}, \end{array}$$

where $\tilde{\mathbf{x}}, \tilde{\mathbf{l}}, \tilde{\mathbf{w}} \in \mathbb{R}^{\tilde{n}}$ and $\tilde{A} \in \mathbb{R}^{\tilde{m}+\tilde{n}}$ has full row rank. The last step is to reformulate the above problem as in the form of problem (1). After the presolving, it is guaranteed that $\tilde{\mathbf{l}} > -\infty$. Now we can define $U = \{j : \tilde{\mathbf{w}}_j < +\infty\}$ and $\bar{\mathbf{x}} = \tilde{\mathbf{x}} - \tilde{\mathbf{l}}$, and obtain the desired LP problem,

$$\begin{array}{ll} \min & \tilde{\mathbf{c}}^\top \bar{\mathbf{x}} \\ \text{s.t.} & \tilde{A}\bar{\mathbf{x}} = \tilde{\mathbf{b}} - \tilde{A}\tilde{\mathbf{l}}, \\ & \tilde{\mathbf{x}}_U + \tilde{\mathbf{s}} = \tilde{\mathbf{w}}_U - \tilde{\mathbf{l}}_U, \\ & \tilde{\mathbf{x}} \geq 0, \ \tilde{\mathbf{s}} \geq 0. \end{array}$$

After the presolve procedure, the reduced problem is ready to be solved by ABIP. A postsolve procedure is used to convert the solution to the reduced problem back to the solution to the original problem.

Remark 4.1 *Since our algorithm is a purely first-order algorithm, we also conduct the scale procedure after the presolve procedure to make the problems more well-conditioned. We refer to [33] for more details.*

5 Numerical Experiments

In this section, we report experimental results of ABIP on solving randomly generated LPs, 114 instances from NETLIB collection[1] and 6 instances from UCI collection[2]. We compare ABIP with several state-of-the-art solvers, e.g., SDPT3 [39], MOSEK [2], DSDP-CG [6] and SCS [33].

Our ABIP code is written in C with a MATLAB interface. The current version of ABIP is single-threaded and computes the projections onto the subspace using a direct method, which uses a single-threaded sparse permuted $\mathrm{LDL}^\top$ decomposition from the SuiteSparse package [12, 11, 1].

In the experimental results reported below, we use the termination criteria for ABIP in Sections 4.1 with default values

$$\epsilon_{\mathsf{pres}} = \epsilon_{\mathsf{dres}} = \epsilon_{\mathsf{dgap}} = \epsilon_{\mathsf{pinfeas}} = \epsilon_{\mathsf{dinfeas}} = 10^{-3}, \tag{78}$$

and that for Barzilai-Borwein spectral method in Section 4.3 with default values

$$\epsilon_{\mathsf{cor}} = 0.2, \quad \epsilon_{\mathsf{penalty}} = 0.1.$$

The over-relaxation parameter is set to $\alpha = 1.8$. Moreover, we set the maximum number of ADMM steps of ABIP to 10^6. If the target accuracy in (78) is not achieved in 10^6 ADMM steps, we terminate the code and claim that ABIP fails to solve this instance and use "—" in the table to indicate the failure.

[1] http://users.clas.ufl.edu/hager/coap/format.html
[2] https://archive.ics.uci.edu/ml/datasets.html

The decreasing ratio γ is adjusted according to the value of the barrier parameter μ, primal/dual feasibility violations and the duality gap. More specifically, we increase γ as the iterate approaches the optimal solution. The objective value reported for all methods in the experiments below is the one after postsolving. The time required to do any preprocessing, i.e., presolving, postsolving, do/undo scaling and matrix factorization are included in the total solve times. All the experiments were carried out on a laptop with Linux system and 8 2.60GHz cores and 16Gb of RAM. The single-threaded version does not make use of the multiple cores.

For the other four solvers, we use the following stopping criteria. For SCS, we change the default α from 1.5 to 1.8, change the stopping tolerance from 10^{-5} to 10^{-3}, and set the maximum number ADMM steps to 10^6. These changes are made to ensure a more fair comparison, because we found that the default parameter setting needs much longer time to converge. For SDPT3, the maximum number of interior point steps is set as the default value 100. For DSDP-CG, the maximum number of interior point steps is set as 100. For MOSEK, we use the default settings. For all these solvers, we claim that they fail to solve an instance (denoted by "—" in the tables) if after the codes terminate, the target accuracy in (78) is not achieved.

5.1 Random LP Instances

In this subsection we test the five solvers on randomly generated dense LPs. First, we generate a Gaussian random vector $\mathbf{x} \in \mathbb{R}^n$ and split its entries randomly into three groups. The first group consists 60% of entries and their values are set to zero. The second group consists of 10% entries and their values are zoom in for ten times larger. The rest of the entries are in the third group and their values are zoomed out for ten times smaller. We then generate vector $\mathbf{s} \in \mathbb{R}^n$ such that $\mathbf{x}_i \mathbf{s}_i = 0$ for all i, and nonzero entries of $\mathbf{s}$ follow standard normal distribution. This ensures the complementary slackness and zero duality gap. Next we generate the data matrix $A \in \mathbb{R}^{m \times n}$ and dual solution $\mathbf{y} \in \mathbb{R}^m$ with entries following standard normal distribution. Finally, we set $\mathbf{b} = A\mathbf{x}$ and $\mathbf{c} = A^\top \mathbf{y} + \mathbf{s}$, which ensures primal and dual feasibility. The solution to the problem is not necessarily unique, but the optimal value is given by $\mathbf{c}^\top \mathbf{x} = \mathbf{b}^\top \mathbf{y}$.

Results: We report the comparison results of the five solver in Table 1. These results show that ABIP compares favorably to the other four solvers. For the last example which is very large, our ABIP even outperforms the commercial solver MOSEK.

5.2 NETLIB LP Collections

In this subsection, we report the performance of all five solvers on 114 feasible instances from NETLIB collection[3].

Problem instances: NETLIB is a collection of LPs from real applications. It has been recognized as the standard testing data set for LP. The problem statistics of the 114 feasible instances before and after presolving is summarized in Tables 2 and 3.

Results: A summary of the numerical resutls on NETLIB is given in Table 4. We observe that SCS is not very robust comparing with other solvers because it only successfully solves 89 problems. MOSEK is the most robust one while ABIP, SDPT3 and DSDP-CG are comparable, and they all significantly outperform SCS. This phenomenon can be explained by the superior robustness of the interior-point methods over the pure first-order methods. Furthermore, the promising performance of ABIP strongly supports the usage of the first-order interior-point method on very large LP problems. Tables 5 and 6 provide the CPU times of the five solvers for these LP instances from NETLIB.

[3] http://www.netlib.org/lp/

5.3 Sparse Inverse Covariance Estimation

In this subsection, we compare the five solvers on solving the following problem which arises from machine learning applications:

$$\begin{aligned} \min_{\Omega \in \mathbb{R}^{d\times d}} \quad & \|\Omega\|_1 \\ \text{s.t.} \quad & \|\Sigma\Omega - I_d\|_\infty \le \lambda, \end{aligned} \tag{79}$$

where $\Sigma \in \mathbb{R}^{d\times d}$ denotes a sample covariance matrix, and $\lambda > 0$ denotes some noisy tolerance. This problem, known as sparse inverse covariance estimation (SICE), is widely studied in high-dimensional statistics and machine learning [8]. For given Σ, SICE (79) aims to find a perturbed inverse convariance matrix which is also sparse. Note that SICE (79) is separable for columns of $\Omega = (\beta_1, \beta_2, \cdots, \beta_d)$, and thus can be decomposed to d problems as follows:

$$\begin{aligned} \min_{\beta_j \in \mathbb{R}^{d}} \quad & \|\beta_j\|_1 \\ \text{s.t.} \quad & \|\Sigma\beta_j - e_j\|_\infty \le \lambda. \end{aligned} \tag{80}$$

(80) can be written as a standard LP as follows:

$$\begin{array}{ll} \min & \mathbf{e}^\top \beta^+ + \mathbf{e}^\top \beta^- \\ \text{s.t.} & S\beta^+ - S\beta^- + w^+ = \lambda\mathbf{e} + e_j, \\ & w^+ + w^- = 2\lambda\mathbf{e}, \\ & \beta^+, \beta^-, w^+, w^- \ge 0, \end{array} \tag{81}$$

where the number of variables is $n = 4d$ and the number of constraints is $m = 2d$. In our experiment, we set $\lambda = \frac{3}{2}\sqrt{\frac{\log(d)}{N}}$ where N is the number of sampled data in the original data.

Problem instances: We obtained Σ from the UCI Machine Learning Repository[4]. The statistics of the 6 selected instances is summarized in Table 7.

Results: Detailed numerical results are reported in Table 8. From this table we see that MOSEK is the best among all the solvers possibly because of its preprocessing procedure of detecting dependent columns. We also observe that ABIP and SCS are comparable and the speedup over SDPT3 and DSDP-CG is more significant as the problem size increases. Moreover, in these data sets ABIP is more robust than SCS, SDPT3 and DSDP-CG as SCS fails on ucihapt and SDPT3 and DSDP-CG fail on gisette.

6 Conclusions

In this paper we present a novel implementation of the primal-dual interior point method to solve linear programs via self-dual embedding. In our approach, we use the ADMM to track the central path. Therefore, the new approach is a first-order implementation of the interior point method (IPM). As such, it inherits intrinsic properties of the IPM. We present a theoretical analysis showing that the overall complexity of ADMM steps is $O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$, and our extensive numerical experiments show that the new algorithm is stable in performance and scalable in size. We believe that there are still rooms for improvements in terms of the numerical stability by incorporating more preconditioning techniques, and we also plan to incorporate the power of distributive computing in the future.

Acknowledgements. We would like to thank Zaiwen Wen for fruitful discussions regarding this project and many helps on the codes.

[4]https://archive.ics.uci.edu/ml/datasets.html

References

[1] P. R. Amestoy, T. A. Davis, and I. S. Duff. Algorithm 837: AMD, an approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software (TOMS)*, 30(3):381–388, 2004.

[2] E. D. Andersen. The homogeneous and self-dual model and algorithm for linear optimization. Technical report, Technical report TR-1-2009, 2009. MOSEK ApS., 2009.

[3] E. D. Andersen and K. D. Andersen. Presolving in linear programming. *Mathematical Programming*, 71(2):221–245, 1995.

[4] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[5] D. A. Bayer and J. C. Lagarias. The nonlinear geometry of linear programming. I. Affine and projective scaling trajectories. *Transactions of the American Mathematical Society*, 314(2):499–526, 1989.

[6] S. J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM J. Optim.*, 10(2):443–461, 2000.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[8] T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

[9] P. L. Combettes and Jean-Christophe Pesquet. A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):564–574, 2007.

[10] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.

[11] T. A. Davis. Algorithm 849: A concise sparse Cholesky factorization package. *ACM Transactions on Mathematical Software (TOMS)*, 31(4):587–591, 2005.

[12] T. A. Davis. *Direct methods for sparse linear systems*. SIAM Fundamentals of Algorithms. SIAM, Philadephia (2006), 2006.

[13] J. Eckstein. Parallel alternating direction multiplier decomposition of convex programs. *Journal of Optimization Theory and Applications*, 80(1):39–62, 1994.

[14] J. Eckstein. Augmented Lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results. *Preprint*, 2012.

[15] J. Eckstein and D. P. Bertsekas. On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992.

[16] M. Fortin and R. Glowinski. *Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems*. North-Holland Pub. Co., 1983.

[17] K. Fountoulakis, J. Gondzio, and P. Zhlobich. Matrix-free interior point method for compressed sensing problems. *Mathematical Programming Computation*, 6:1–31, 2014.

[18] D. Gabay. Applications of the method of multipliers to variational inequalities. In M. Fortin and R. Glowinski, editors, *Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems*. North-Hollan, Amsterdam, 1983.

[19] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM, Philadelphia, Pennsylvania, 1989.

[20] R. Glowinski and A. Marrocco. Sur l'approximation par èlèments finis et la rèsolution par pènalisation-dualitè d'une classe de problèmes de dirichlet non linèaires. *RAIRO*, R-2:41–76, 1975.

[21] T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. *SIAM J. Imaging Sci.*, 2:323–343, 2009.

[22] J. Gondzio. Presolve analysis of linear programs prior to applying an interior point method. *INFORMS Journal on Computing*, 9(1):73–91, 1997.

[23] J. Gondzio. Interior point methods 25 years later. *European Journal of Operational Research*, 218(3):587–601, 2012.

[24] J. Gondzio. Matrix-free interior point method. *Computational Optimization and Applications*, 51:457–480, 2012.

[25] N. Gould and P. L. Toint. Preprocessing for quadratic programming. *Mathematical Programming*, 100(1):95–132, 2004.

[26] B. S. He, H. Yang, and S. L. Wang. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Journal of Optimization Theory and applications*, 106(2):337–356, 2000.

[27] N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[28] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16:964–979, 1979.

[29] N. Megiddo. *Progress in mathematical programming*. Springer, 1989.

[30] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[31] C. Mészáros and U. H. Suhl. Advanced preprocessing techniques for linear and quadratic programming. *OR Spectrum*, 25(4):575–595, 2003.

[32] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM Studies in Applied Mathematics, 1994.

[33] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

[34] B. O'Donoghue, G. Stathopoulos, and S. Boyd. A splitting method for optimal control. *IEEE Transactions on Control Systems Technology*, 21(6):2432–2442, 2013.

[35] A. Pothen and C.-J. Fan. Computing the block triangular form of a sparse matrix. *ACM Transactions on Mathematical Software (TOMS)*, 16(4):303–324, 1990.

[36] J. Renegar. A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, 40(1):59–93, 1988.

[37] G. Sonnevend. An "analytical centre" for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming. *System modelling and optimization*, pages 866–875, 1986.

[38] J. F. Sturm and S. Zhang. On a wide region of centers and primal-dual interior point algorithms for linear programming. *Mathematics of Operations Research*, 22:408–431, 1997.

[39] R. H. Tütüncü, K.-C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical programming*, 95(2):189–217, 2003.

[40] S. Wang and N. Shroff. A new alternating direction method for linear programming. In *NIPS*, pages 1479–1487, 2017.

[41] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2(3):203–230, 2010.

[42] X. Xu, P-F. Hung, and Y. Ye. A simplified homogeneous and self-dual linear programming algorithm and its implementation. *Annals of Operations Research*, 62(1):151–171, 1996.

[43] Z. Xu, M. Figueiredo, and T. Goldstein. Adaptive ADMM with spectral penalty parameter selection. In *AISTATS*, pages 718–727, 2017.

[44] J. Yang and Y. Zhang. Alternating direction algorithms for l_1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2011.

[45] J. Yang, Y. Zhang, and W. Yin. A fast alternating direction method for tvl1-l2 signal reconstruction from partial fourier data. *IEEE Journal of Selected Topics in Signal Processing Special Issue on Compressed Sensing*, 4(2):288–297, 2010.

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

[47] Y. Ye. *Interior Point Algorithms: Theory and Analysis*. Wiley & Sons, New York, 1997.

[48] Y. Ye, M. J. Todd, and S. Mizuno. An $\mathcal{O}(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67, 1994.

[49] I. E-H. Yen, K. Zhong, C-J. Hsieh, P. K. Ravikumar, and I. S. Dhillon. Sparse linear programming via primal and dual augmented coordinate descent. In *NIPS*, pages 2368–2376, 2015.

[50] B. Zhou, L. Gao, and Y-H. Dai. Gradient methods with adaptive step-sizes. *Computational Optimization and Applications*, 35(1):69–86, 2006.

(m,n)	solver	obj	pres	dres	dgap	time
	ABIP	-2.53e+04	9.14e-05	5.74e-04	9.14e-04	1.34e+01
	SCS	-2.54e+04	9.32e-04	6.74e-04	7.55e-04	9.89e+01
(500, 5000)	SDPT3	-2.54e+04	1.75e-08	2.96e-05	1.89e-05	3.11e+01
	DSDP-CG	-2.54e+04	2.47e-09		1.01e-04	5.05e+01
	MOSEK	-2.54e+04	4.71e-11	7.36e-13	6.15e-11	**1.17e+00**
	ABIP	-1.25e+05	8.71e-05	4.03e-04	9.96e-04	2.42e+01
	SCS	-1.25e+05	9.86e-04	6.02e-04	1.70e-04	3.40e+01
(500, 10000)	SDPT3	-1.26e+05	5.63e-08	5.59e-05	4.57e-05	5.01e+01
	DSDP-CG	-1.26e+05	1.12e-10		1.14e-04	6.01e+01
	MOSEK	-1.26e+05	1.02e-11	4.81e-13	6.84e-13	**2.19e+00**
	ABIP	-1.80e+05	7.00e-05	2.57e-04	9.97e-04	6.21e+01
	SCS	-1.80e+05	8.58e-04	4.18e-04	2.29e-04	5.11e+01
(500, 20000)	SDPT3	-1.80e+05	7.88e-08	1.47e-04	1.25e-04	1.75e+02
	DSDP-CG	-1.80e+05	2.78e-11		1.10e-04	1.19e+02
	MOSEK	-1.80e+05	6.96e-12	1.68e-13	1.11e-12	**4.49e+00**
	ABIP	6.09e+04	3.92e-05	3.99e-04	9.90e-04	1.38e+02
	SCS	6.06e+04	9.51e-04	8.97e-04	3.49e-04	1.49e+02
(1000, 5000)	SDPT3	6.05e+04	5.11e-09	1.19e-06	9.25e-07	1.84e+02
	DSDP-CG	6.05e+04	9.98e-12		3.51e-04	3.19e+02
	MOSEK	6.05e+04	4.76e-10	4.91e-10	1.41e-11	**4.39e+00**
	ABIP	7.52e+04	4.08e-05	3.52e-04	9.79e-04	9.00e+01
	SCS	7.47e+04	9.89e-04	6.97e-04	4.46e-04	3.40e+02
(1000, 10000)	SDPT3	7.47e+04	3.54e-08	4.66e-05	5.37e-05	2.26e+02
	DSDP-CG	7.47e+04	2.44e-10	-	3.29e-05	3.93e+02
	MOSEK	7.47e+04	2.69e-11	2.04e-12	3.57e-11	**6.42e+00**
	ABIP	-1.37e+05	2.04e-04	2.57e-04	9.14e-04	1.89e+02
	SCS	-1.38e+05	9.94e-04	6.67e-04	5.27e-04	1.60e+02
(1000, 20000)	SDPT3	-1.38e+05	3.79e-08	1.30e-04	1.11e-04	6.68e+02
	DSDP-CG	-1.38e+05	2.24e-11		5.22e-05	4.82e+02
	MOSEK	-1.38e+05	5.91e-11	6.00e-11	1.21e-11	**1.13e+01**
	ABIP	2.23e+05	8.21e-04	8.16e-04	9.88e-04	1.91e+02
	SCS	2.23e+05	6.89e-04	7.01e-04	3.40e-04	1.67e+02
(5000, 6000)	SDPT3	2.23e+05	2.67e-06	7.81e-05	3.33e-03	1.13e+04
	DSDP-CG	2.23e+05	5.67e-11		3.67e-04	1.18e+04
	MOSEK	2.23e+05	3.02e-12	2.78e-12	2.94e-13	**1.51e+02**
	ABIP	8.99e+05	5.10e-06	1.15e-04	9.99e-04	2.76e+03
	SCS	8.99e+05	9.90e-04	8.76e-04	8.83e-04	1.18e+03
(5000, 10000)	SDPT3	8.98e+05	2.55e-08	2.55e-07	3.02e-07	1.58e+04
	DSDP-CG	8.97e+05	5.55e-12		4.46e-04	2.19e+04
	MOSEK	8.98e+05	3.51e-09	3.28e-09	5.18e-10	**2.02e+02**
	ABIP	9.44e+05	1.16e-05	1.43e-04	9.97e-04	8.44e+03
	SCS	9.44e+05	9.14e-04	6.77e-04	8.91e-04	4.67e+03
(5000, 20000)	SDPT3	9.43e+05	6.54e-08	1.99e-05	1.67e-05	1.69e+04
	DSDP-CG	9.43e+05	2.12e-13		4.44e-04	2.60e+04
	MOSEK	9.43e+05	2.30e-09	2.30e-09	2.73e-11	**2.65e+02**
	ABIP	5.51e+05	5.40e-04	5.21e-04	9.34e-04	1.27e+03
	SCS	5.50e+05	5.89e-04	6.08e-04	9.82e-04	**1.12e+03**
(10000, 12000)	SDPT3	5.52e+05	1.01e-05	5.87e-04	2.90e-01	2.70e+04
	DSDP-CG	5.52e+05	6.90e-11		4.04e-04	8.18e+04
	MOSEK	5.52e+05	2.24e-11	1.55e-12	1.68e-12	1.17e+03
	ABIP	3.40e+06	2.76e-06	8.62e-05	1.00e-03	1.57e+04
	SCS	3.40e+06	9.83e-04	7.89e-04	1.88e-04	6.86e+03
(10000, 20000)	SDPT3	3.92e+06	1.46e-02	5.05e-03	3.06e-03	8.59e+04
	DSDP-CG	3.39e+06	6.14e-12		3.97e-04	2.11e+05
	MOSEK	3.39e+06	6.39e-09	5.98e-09	2.08e-10	**1.57e+03**
	ABIP	-1.46e+06	6.93e-04	7.02e-04	6.54e-04	**2.72e+03**
	SCS	-1.46e+06	7.73e-04	8.16e-04	8.12e-04	3.67e+03
(15000, 20000)	SDPT3	-	-	-	-	—
	DSDP-CG	-	-	-	-	—
	MOSEK	-1.46e+06	7.84e-10	2.54e-10	1.52e-10	2.85e+03

Table 1: The performance of all five solvers on randomly generated LPs. CPU times are in seconds. Note: DSDP-CG does not provide the dual residuals.

Problem	Before Presolving				After Presolving			
	Rows	Cols	Nonzeros	Sparsity	Rows	Cols	Nonzeros	Sparsity
25FV47	821	1876	10705	0.00695	798	1854	10580	0.00715
80BAU3B	2262	12061	23264	0.00085	5221	14502	28620	0.00038
ADLITTLE	56	138	424	0.05487	55	137	417	0.05534
AFIRO	27	51	102	0.07407	27	51	102	0.07407
AGG	488	615	2862	0.00954	488	615	2862	0.00954
AGG2	516	758	4740	0.01212	516	758	4740	0.01212
AGG3	516	758	4756	0.01216	516	758	4756	0.01216
BANDM	305	472	2494	0.01732	269	436	2137	0.01822
BEACONFD	173	295	3408	0.06678	148	270	3105	0.07770
BLEND	74	114	522	0.06188	74	114	522	0.06188
BNL1	643	1586	5532	0.00542	632	1576	5522	0.00554
BNL2	2324	4486	14996	0.00144	2268	4430	14914	0.00148
BOEING1	351	726	3827	0.01502	592	967	4309	0.00753
BOEING2	166	305	1358	0.02682	213	352	1478	0.01971
BORE3D	233	334	1448	0.01861	210	311	1346	0.02061
BRANDY	220	303	2202	0.03303	149	259	2015	0.05221
CAPRI	271	482	1896	0.01452	397	605	2137	0.00890
CRE_A	3516	7248	18168	0.00071	3428	7248	18168	0.00073
CRE_B	9648	77137	260785	0.00035	7240	77137	260785	0.00047
CRE_C	3068	6411	15977	0.00081	2986	6411	15977	0.00083
CRE_D	8926	73948	246614	0.00037	6476	73948	246614	0.00051
CYCLE	1903	3371	21234	0.00331	1878	3381	20958	0.00330
CZPROB	929	3562	10708	0.00324	737	3141	9454	0.00408
D2Q06C	2171	5831	33081	0.00261	2171	5831	33081	0.00261
D6CUBE	415	6184	37704	0.01469	404	6184	37704	0.01509
DEGEN2	444	757	4201	0.01250	444	757	4201	0.01250
DEGEN3	1503	2604	25432	0.00650	1503	2604	25432	0.00650
DFL001	6071	12230	35632	0.00048	6084	12243	35658	0.00048
E226	223	472	2768	0.02630	220	469	2737	0.02653
ETAMACRO	400	816	2537	0.00777	488	823	2306	0.00574
FFFFF800	524	1028	6401	0.01188	501	1005	6283	0.01248
FINNIS	497	1064	2760	0.00522	528	1050	2599	0.00469
FIT1D	24	1049	13427	0.53333	1050	2075	15479	0.00710
FIT1P	627	1677	9868	0.00938	1026	2076	10666	0.00501
FIT2D	25	10524	129042	0.49047	10525	21024	150042	0.00068
FIT2P	3000	13525	50284	0.00124	10500	21025	65284	0.00030
FORPLAN	161	492	4634	0.05850	157	485	4583	0.06019
GANGES	1309	1706	6937	0.00311	1534	1931	7387	0.00249
GFRD_PNC	616	1160	2445	0.00342	858	1402	2929	0.00243
GREENBEA	2392	5598	31070	0.00232	2608	5714	31014	0.00208
GREENBEB	2392	5598	31070	0.00232	2608	5706	30966	0.00208
GROW7	140	301	2612	0.06198	420	581	3172	0.01300
GROW15	300	645	5620	0.02904	900	1245	6820	0.00609
GROW22	440	946	8252	0.01983	1320	1826	10012	0.00415
ISRAEL	174	316	2443	0.04443	174	316	2443	0.04443
KB2	43	68	313	0.10705	52	77	331	0.08267
KEN_7	2426	3602	8404	0.00096	4558	5734	12374	0.00047
KEN_11	14694	21349	49058	0.00016	29751	36406	78567	0.00007
KEN_13	28632	42659	97246	0.00008	60813	74840	159749	0.00004
KEN_18	105127	154699	358171	0.00002	231314	280886	606657	0.00001
LOTFI	153	366	1136	0.02029	151	364	1123	0.02043
MAROS	846	1966	10137	0.00609	835	1921	10060	0.00627
MAROS_R7	3136	9408	144848	0.00491	3136	9408	144848	0.00491
MODSZK1	687	1620	3168	0.00285	686	1622	3170	0.00285
NESM	662	3105	13470	0.00655	2250	4518	16436	0.00162
OSA_07	1118	25067	144812	0.00517	1118	25067	144812	0.00517
OSA_14	2337	54797	317097	0.00248	2337	54797	317097	0.00248

Table 2: The statistics of feasible LPs in NETLIB before and after presolving: part 1.

Problem	Before Presolving				After Presolving			
	Rows	Cols	Nonzeros	Sparsity	Rows	Cols	Nonzeros	Sparsity
OSA_30	4350	104374	604488	0.00133	4350	104374	604488	0.00133
OSA_60	10280	243246	1408073	0.00056	10280	243246	1408073	0.00056
PDS_02	2953	7716	16571	0.00073	4768	9531	20190	0.00044
PDS_06	9881	29351	63220	0.00022	18604	38074	80556	0.00011
PDS_10	16558	49932	107605	0.00013	32079	65453	138482	0.00007
PDS_20	33874	108175	232647	0.00006	67599	141976	299853	0.00003
PEROLD	625	1506	6148	0.00653	891	1796	7663	0.00479
PILOT	1441	4860	44375	0.00634	2481	5697	44380	0.00314
PILOT4	410	1123	5264	0.01143	649	1420	7720	0.00838
PILOT87	2030	6680	74949	0.00553	3608	8038	75635	0.00261
PILOT_JA	940	2267	14977	0.00703	1263	2383	14017	0.00466
PILOT_WE	722	2928	9265	0.00438	1016	3224	10125	0.00309
PILOTNOV	975	2446	13331	0.00559	1291	2582	13140	0.00394
QAP8	912	1632	7296	0.00490	912	1632	7296	0.00490
QAP12	3192	8856	38304	0.00136	3192	8856	38304	0.00136
QAP15	6330	22275	94950	0.00067	6330	22275	94950	0.00067
RECIPE	91	204	687	0.03701	153	245	785	0.02094
SC50A	50	78	160	0.04103	49	77	159	0.04214
SC50B	50	78	148	0.03795	48	76	146	0.04002
SC105	105	163	340	0.01987	105	163	340	0.01987
SC205	205	317	665	0.01023	205	317	665	0.01023
SCAGR7	129	185	465	0.01948	129	185	465	0.01948
SCAGR25	471	671	1725	0.00546	471	671	1725	0.00546
SCFXM1	330	600	2732	0.01380	322	592	2707	0.01420
SCFXM2	660	1200	5469	0.00691	644	1184	5419	0.00711
SCFXM3	990	1800	8206	0.00460	966	1776	8131	0.00474
SCORPION	388	466	1534	0.00848	375	453	1460	0.00859
SCRS8	490	1275	3288	0.00526	485	1270	3262	0.00530
SCSD1	77	760	2388	0.04081	77	760	2388	0.04081
SCSD6	147	1350	4316	0.02175	147	1350	4316	0.02175
SCSD8	397	2750	8584	0.00786	397	2750	8584	0.00786
SCTAP1	300	660	1872	0.00945	300	660	1872	0.00945
SCTAP2	1090	2500	7334	0.00269	1090	2500	7334	0.00269
SCTAP3	1480	3340	9734	0.00197	1480	3340	9734	0.00197
SEBA	515	1036	4360	0.00817	1029	1550	5388	0.00338
SHARE1B	117	253	1179	0.03983	112	248	1148	0.04133
SHARE2B	96	162	777	0.04996	96	162	777	0.04996
SHELL	536	1777	3558	0.00374	613	1604	3212	0.00327
SHIP04L	402	2166	6380	0.00733	356	2162	6368	0.00827
SHIP04S	402	1506	4400	0.00727	268	1414	4124	0.01088
SHIP08L	778	4363	12882	0.00380	688	4339	12810	0.00429
SHIP08S	778	2467	7194	0.00375	416	2171	6306	0.00698
SHIP12L	1151	5533	16276	0.00256	838	5329	15664	0.00351
SHIP12S	1151	2869	8284	0.00251	466	2293	6556	0.00614
SIERRA	1227	2735	8001	0.00238	3238	4731	11983	0.00078
STAIR	356	614	4003	0.01831	362	544	3843	0.01951
STANDATA	359	1274	3230	0.00706	463	1362	3381	0.00536
STANDGUB	361	1383	3338	0.00669	464	1470	3489	0.00512
STANDMPS	467	1274	3878	0.00652	571	1362	4029	0.00518
STOCFOR1	117	165	501	0.02595	109	157	471	0.02752
STOCFOR2	2157	3045	9357	0.00142	2157	3045	9357	0.00142
STOCFOR3	16675	23541	72721	0.00019	16675	23541	72721	0.00019
TRUSS	1000	8806	27836	0.00316	1000	8806	27836	0.00316
TUFF	333	628	4561	0.02181	317	642	4599	0.02260
VTP_BASE	198	346	1051	0.01534	258	389	1065	0.01061
WOOD1P	244	2595	70216	0.11089	244	2595	70216	0.11089
WOODW	1098	8418	37487	0.00406	1098	8418	37487	0.00406

Table 3: The statistics of feasible LPs in NETLIB before and after presolving: part 2.

Solver	**Num of Instances**		**Time(s)**	
	Solved	Unsolved	Mean	Std
ABIP	102	12	31.92	189.27
SCS	89	25	36.36	205.10
SDPT3	104	10	19.56	112.70
DSDP-CG	105	9	136.44	770.84
MOSEK	114	0	0.25	1.13

Table 4: Summary of the performance of all five solvers on NETLIB. The CPU times (in seconds) are summarized for 76 instances that can be solved by all five solvers.

Problem	ABIP	SCS	SDPT3	DSDP-CG	MOSEK
25FV47	6.10e+00	2.41e+01	1.40e+00	1.09e+00	9.67e-02
80BAU3B	5.43e+01	9.44e+00	6.37e+00	3.25e+01	1.64e-01
ADLITTLE	1.46e-02	4.01e-02	2.48e-01	6.93e-03	2.67e-03
AFIRO	3.19e-03	8.08e-04	1.18e-01	1.55e-03	1.72e-03
AGG	4.79e-01	—	3.31e-01	1.62e-01	6.44e-03
AGG2	1.97e+00	2.08e+01	2.52e-01	2.23e-01	1.34e-02
AGG3	4.23e+00	9.99e+01	2.67e-01	2.37e-01	1.10e-02
BANDM	4.13e-01	2.26e+00	2.00e-01	7.83e-02	7.04e-03
BEACONFD	5.21e+00	6.29e+00	1.03e-01	5.37e-02	4.52e-03
BLEND	2.59e-02	2.91e-02	7.90e-02	1.18e-02	3.03e-03
BNL1	7.07e-01	4.93e+00	3.17e-01	5.13e-01	2.48e-02
BNL2	3.75e+00	5.05e+00	2.53e+00	6.17e+00	1.26e-01
BOEING1	2.38e-01	1.21e+01	3.49e-01	3.18e-01	1.59e-02
BOEING2	3.18e-01	1.07e+00	1.99e-01	8.41e-02	6.63e-03
BORE3D	1.45e+01	3.29e+01	1.51e-01	4.71e-02	2.55e-03
BRANDY	2.46e+00	3.55e+00	2.24e-01	—	6.40e-03
CAPRI	4.54e-01	—	4.16e-01	1.32e-01	6.65e-03
CRE_A	3.27e-01	1.50e+00	1.37e+00	1.17e+01	5.68e-02
CRE_B	9.16e+01	3.37e+01	5.36e+01	—	5.46e-01
CRE_C	6.11e-01	1.96e+00	1.25e+00	1.12e+01	5.13e-02
CRE_D	1.23e+02	3.44e+01	5.50e+01	—	4.65e-01
CYCLE	4.87e+01	9.76e+01	3.08e+00	4.93e+00	5.78e-02
CZPROB	1.36e+00	3.20e+00	3.61e-01	1.08e+00	1.88e-02
D2Q06C	4.83e+02	—	2.48e+00	1.08e+01	2.16e-01
D6CUBE	2.91e+01	2.26e+00	5.39e-01	3.00e+00	9.27e-02
DEGEN2	1.02e-01	7.61e-02	1.77e-01	1.66e-01	1.56e-02
DEGEN3	9.15e-01	5.46e-01	8.97e-01	3.25e+00	1.02e-01
DFL001	—	1.86e+01	9.03e+01	8.41e+01	1.33e+00
E226	1.46e+00	4.38e+01	2.75e-01	1.17e-01	7.10e-03
ETAMACRO	5.31e+00	1.60e+01	3.84e-01	1.96e-01	2.15e-02
FFFFF800	5.99e+01	—	4.94e-01	4.28e-01	1.78e-02
FINNIS	3.08e-01	5.42e-01	2.63e-01	2.26e-01	1.16e-02
FIT1D	5.19e+01	—	2.60e-01	1.36e+00	1.24e-02
FIT1P	2.19e+00	3.00e+00	8.39e-01	1.50e+00	9.09e-02
FIT2D	1.11e+03	—	1.95e+00	1.42e+02	1.02e-01
FIT2P	4.46e+00	1.43e+01	2.39e+00	1.69e+03	8.59e-02
FORPLAN	3.64e+01	4.46e+00	2.39e-01	9.79e-02	1.16e-02
GANGES	4.06e-01	2.71e-01	2.57e-01	7.94e-01	1.70e-02
GFRD_PNC	4.05e-02	4.95e-02	2.96e-01	2.75e-01	1.22e-02
GREENBEA	1.11e+02	3.59e+02	3.20e+00	1.21e+01	1.94e-01
GREENBEB	4.72e+00	1.35e+01	3.36e+00	1.27e+01	7.85e-02
GROW7	2.79e-01	3.21e-02	1.60e-01	6.64e-02	6.34e-03
GROW15	5.78e-01	5.84e-02	1.52e-01	2.65e-01	1.06e-02
GROW22	9.54e-01	1.27e-01	1.95e-01	—	1.54e-02
ISRAEL	3.62e+00	—	1.92e-01	7.81e-02	1.24e-02
KB2	1.27e+00	8.48e+00	7.30e-02	6.02e-03	3.27e-03
KEN_7	9.74e-01	3.76e-01	3.49e-01	5.44e+00	1.82e-02
KEN_11	1.12e+01	6.25e+00	—	—	1.34e-01
KEN_13	3.82e+02	1.84e+01	—	—	4.21e-01
KEN_18	1.04e+02	2.10e+02	—	—	1.99e+00
LOTFI	6.69e+00	—	2.62e-01	3.62e-02	7.88e-03
MAROS	—	—	7.30e-01	1.01e+00	2.34e-02
MAROS_R7	4.26e+01	2.11e+00	3.45e+00	1.69e+01	3.00e-01
MODSZK1	5.60e+00	2.58e+00	4.40e-01	3.42e-01	4.59e-02
NESM	1.07e+00	1.28e+00	5.27e-01	3.69e+00	4.54e-02
OSA_07	4.76e+01	9.66e+00	1.35e+00	2.34e+01	9.76e-02
OSA_14	3.12e+01	3.91e+01	3.72e+00	1.18e+02	2.13e-01

Table 5: CPU time (in seconds) of all five solvers on NETLIB: part 1.

Problem	ABIP	SCS	SDPT3	DSDP-CG	MOSEK
OSA_30	4.11e+01	6.13e+01	7.15e+00	4.66e+02	4.92e-01
OSA_60	3.44e+02	2.98e+02	—	—	1.30e+00
PDS_02	1.19e+00	3.70e-01	4.02e+00	1.30e+01	4.08e-02
PDS_06	1.02e+01	2.44e+00	8.98e+01	2.69e+02	3.88e-01
PDS_10	2.85e+01	6.99e+00	3.76e+02	1.03e+03	7.54e-01
PDS_20	1.14e+02	4.37e+01	9.12e+02	6.46e+03	2.75e+00
PEROLD	—	—	—	9.46e-01	5.19e-02
PILOT	8.86e+00	—	3.17e+00	1.60e+01	3.87e-01
PILOT4	—	—	—	6.13e-01	4.53e-02
PILOT87	1.65e+03	1.76e+03	1.13e+01	3.70e+01	8.10e-01
PILOT_JA	—	—	—	2.57e+00	9.26e-02
PILOT_WE	6.95e+00	6.57e+01	—	1.69e+00	7.38e-02
PILOTNOV	—	—	1.51e+00	2.73e+00	8.96e-02
QAP8	1.18e+00	1.33e-01	3.83e-01	4.11e-01	7.95e-02
QAP12	1.66e+01	5.11e+00	7.35e+00	1.44e+01	1.44e+00
QAP15	7.94e+01	2.44e+01	3.27e+01	1.09e+02	9.42e+00
RECIPE	1.10e-01	6.72e-03	8.46e-02	1.71e-02	2.56e-03
SC50A	1.35e-02	2.35e-03	5.35e-02	5.22e-03	1.77e-03
SC50B	1.71e-02	2.88e-03	4.61e-02	5.03e-03	1.45e-03
SC105	8.39e-03	6.65e-03	6.82e-02	1.63e-02	3.42e-03
SC205	2.64e-02	2.87e-02	9.63e-02	2.98e-02	3.54e-03
SCAGR7	1.81e-02	1.08e-02	9.71e-02	1.57e-02	5.09e-03
SCAGR25	1.02e-01	6.12e-02	1.95e-01	7.59e-02	2.25e-02
SCFXM1	5.81e+00	—	2.97e-01	1.11e-01	9.33e-03
SCFXM2	8.77e+00	—	3.41e-01	3.95e-01	1.87e-02
SCFXM3	1.15e+01	—	4.18e-01	7.96e-01	2.75e-02
SCORPION	7.85e-02	3.84e-02	1.24e-01	6.63e-02	4.37e-03
SCRS8	5.31e-01	9.04e-01	3.54e-01	2.11e-01	1.01e-02
SCSD1	5.91e-02	1.36e-02	6.49e-02	1.96e-02	4.21e-03
SCSD6	8.91e-02	6.36e-02	7.95e-02	5.81e-02	8.33e-03
SCSD8	1.52e-01	1.09e-01	1.27e-01	1.46e-01	1.16e-02
SCTAP1	5.66e-01	6.57e-01	1.57e-01	6.81e-02	7.24e-03
SCTAP2	2.53e+00	3.20e-01	1.97e-01	7.35e-01	1.51e-02
SCTAP3	1.93e+00	6.98e-01	2.91e-01	1.52e+00	2.00e-02
SEBA	1.53e+00	—	4.68e-01	1.02e+00	1.46e-02
SHARE1B	3.22e+00	—	1.37e-01	3.06e-02	4.86e-03
SHARE2B	1.02e+01	—	7.22e-02	1.57e-02	3.20e-03
SHELL	1.82e-01	7.88e-02	2.59e-01	2.68e-01	1.38e-02
SHIP04L	1.91e-01	7.42e-02	1.72e-01	2.69e-01	1.06e-02
SHIP04S	2.90e-01	6.57e-02	1.46e-01	1.31e-01	9.86e-03
SHIP08L	4.25e-01	2.20e-01	2.13e-01	1.02e+00	1.42e-02
SHIP08S	3.54e-01	1.29e-01	1.97e-01	2.97e-01	1.07e-02
SHIP12L	1.76e+00	1.11e+00	2.45e-01	1.34e+00	2.11e-02
SHIP12S	4.72e-01	3.05e-01	2.30e-01	2.70e-01	1.17e-02
SIERRA	2.87e+02	—	1.62e+00	3.92e+00	2.34e-02
STAIR	4.20e-01	2.17e+00	2.42e-01	1.29e-01	1.35e-02
STANDATA	6.17e-01	1.53e-01	2.66e-01	2.28e-01	8.40e-03
STANDGUB	9.92e-01	1.63e-01	2.73e-01	2.45e-01	7.89e-03
STANDMPS	1.03e+00	1.42e-01	2.52e-01	3.48e-01	1.19e-02
STOCFOR1	—	1.96e-02	7.05e-02	1.35e-02	3.75e-03
STOCFOR2	—	1.69e+00	2.88e-01	2.87e+00	2.75e-02
STOCFOR3	—	—	2.62e+00	1.78e+02	2.11e-01
TRUSS	2.08e+00	7.76e+00	3.99e-01	1.41e+00	7.72e-02
TUFF	1.06e-01	—	—	2.68e-01	1.22e-02
VTP_BASE	—	—	—	5.13e-02	2.69e-03
WOOD1P	—	5.90e-01	6.34e-01	—	3.56e-02
WOODW	—	—	8.04e-01	6.99e+00	4.63e-02

Table 6: CPU time (in seconds) of all five solvers on NETLIB: part 2.

Name	Problem Statistics						
	Samples (N)	Variables (d)	Threshold (λ)	Rows (m)	Cols (n)	Nonzeros	Sparsity
gisette	13500	5000	0.0377	10000	20000	50015000	0.2501
isolet	7797	618	0.0431	1236	2472	765702	0.2506
sEMG	1800	3000	0.1000	6000	12000	18009000	0.2501
sEMGday	3600	2500	0.0699	5000	10000	12507500	0.2501
ucihapt	10929	561	0.0361	1122	2244	631125	0.2507
ucihar	10299	561	0.0372	1122	2244	631125	0.2507

Table 7: The statistics of 6 instances in UCI collection.

Name	Method	obj	pres	dres	dgap	time
gisette	ABIP	1.24e-05	2.51e-04	9.99e-04	1.01e-04	8.78e+02
	SCS	1.25e-05	8.89e-04	9.54e-04	4.13e-08	2.31e+03
	SDPT3	4.86e-04	5.66e-11	**2.39e-03**	9.41e-04	5.66e+04
	DSDP-CG	-2.09e+06	**1.77e+00**	-	5.99e-04	1.94e+04
	MOSEK	1.25e-05	7.11e-13	4.63e-15	2.89e-13	7.46e+01
isolet	ABIP	6.37e+02	1.01e-04	1.00e-03	2.03e-06	6.22e+02
	SCS	6.37e+02	7.34e-04	8.85e-04	1.28e-05	8.68e+02
	SDPT3	6.37e+02	1.67e-08	1.45e-06	4.62e-08	2.89e+01
	DSDP-CG	6.37e+02	3.70e-10	-	1.65e-04	4.29e+01
	MOSEK	6.37e+02	2.89e-08	1.92e-10	2.43e-10	6.95e-01
sEMG	ABIP	1.06e+02	4.10e-04	3.57e-04	1.00e-03	1.04e+03
	SCS	1.07e+02	9.48e-04	8.61e-04	5.27e-05	2.85e+02
	SDPT3	1.07e+02	2.35e-08	5.38e-06	4.06e-06	3.52e+03
	DSDP-CG	1.07e+02	2.38e-12	-	3.18e-04	6.13e+03
	MOSEK	1.07e+02	3.05e-09	5.76e-15	9.95e-10	2.99e+01
sEMGday	ABIP	1.88e+02	3.95e-04	8.89e-04	1.00e-03	3.70e+02
	SCS	1.89e+02	4.18e-04	9.99e-04	1.70e-05	2.43e+02
	SDPT3	1.89e+02	1.23e-08	2.43e-06	1.21e-06	2.04e+03
	DSDP-CG	1.89e+02	6.63e-11	-	2.76e-04	3.34e+03
	MOSEK	1.89e+02	1.85e-09	1.10e-13	1.03e-09	1.76e+01
ucihapt	ABIP	3.64e+03	9.90e-05	1.00e-03	8.14e-07	1.44e+03
	SCS	3.63e+03	**4.35e-03**	**5.29e-03**	6.43e-06	4.87e+03
	SDPT3	3.63e+03	7.73e-07	3.79e-07	4.25e-08	4.20e+01
	DSDP-CG	3.63e+03	1.58e-07	-	1.82e-04	3.51e+01
	MOSEK	3.63e+03	1.06e-10	1.56e-11	3.17e-14	7.46e-01
ucihar	ABIP	2.05e+03	2.47e-05	1.00e-03	1.71e-06	3.63e+02
	SCS	2.05e+03	3.55e-04	9.99e-04	1.88e-06	1.00e+03
	SDPT3	2.05e+03	3.99e-08	1.67e-09	1.08e-13	4.65e+01
	DSDP-CG	2.05e+03	4.42e-08	-	3.95e-04	3.23e+01
	MOSEK	2.05e+03	2.16e-08	6.57e-10	5.77e-11	5.87e-01

Table 8: The performance of all solvers on UCI. CPU times are in seconds.