A hybrid branch-and-bound and interior-point algorithm for stochastic mixed-integer nonlinear second-order cone programming

Hadjer Alioui[†] **· Baha Alzalg**[‡] [*]

Abstract One of the chief attractions of stochastic mixed-integer second-order cone programming is its diverse applications, especially in engineering (Alzalg and Alioui, *IEEE Access*, 10:3522-3547, 2022). The linear and nonlinear versions of this class of optimization problems are still unsolved yet. In this paper, we develop a hybrid optimization algorithm coupling branch-and-bound and primal-dual interior-point methods for solving two-stage stochastic mixed-integer nonlinear second-order cone programming. The adopted approach uses a branch-and-bound technique to handle the integer variables and an infeasible interior-point method to solve continuous relaxations of the resulting subproblems. The proposed hybrid algorithm is also implemented to data to show its efficiency.

Keywords: Mixed-integer programming · Stochastic programming · Nonlinear second-order cone programming · Interior-point methods · Branch-and-bound
Mathematics Subject Classification: 90C11 · 90C15 · 90C30 · 90C51 · 90C57

1 Introduction

The purpose of this paper is to study and solve two-stage stochastic mixed-integer nonlinear second-order cone programming (SMINLSOCP for short), in which conicity, uncertainty, integrality and nonlinearity all need to be faced and handled. See Figure 1 which shows conceptual relationships between SMINLSOCP and some special cases. An important special case of SMINLSOCP is stochastic mixed-integer (linear) second-order cone programming (SMISOCP for short). Both SMISOCP and SMINLSOCP are still algorithmically unsolved yet. Alzalg and Alioui [6] described some application models leading to two-stage SMISOCP. The applications of SMISOCP, which were the focus of [6], include random discrete facility location, portfolio optimization with CVaR and diversification constraints, stochastic joint uncapacitated location-inventory problems, optimal infrastructure problems for electric vehicles with battery swap technology, and optimal random berth allocation problems with uncertain handling time.

If we pull out the integrality from SMISOCP, we get stochastic second-order cone programming (SSOCP). SSOCPs are convex optimization problems and can be solved in three different approaches or methods: Benders' decomposition, deterministic equivalence, and Lagrangian dual. In the Benders' decomposition method, we divide the stochastic program into stages in which variables from earlier stages are used as constraints to solve the current subproblem. The work of Alzalg in [1,3] (see also Alzalg and Ariyawansa [7]) is based on this approach. In the deterministic equivalence method, we use the extended form of the stochastic program to create a huge one-stage problem with all constraints and all possible scenarios. The work of Alzalg in [2] and that of Alzalg et al. in [8] are based on this approach. In the Lagrangian dual

✉ [*]B. Alzalg (corresponding author)
b.alzalg@ju.edu.jo

H. Alioui
haj9180341@ju.edu.jo

† Department of Mathematics, M'Hamed Bougara University of Boumerdés, Algeria

‡ Department of Mathematics, The University of Jordan, Amman, Jordan

method, we relax the so-called nonancipativity constraints and use the barrier function method to improve the dual objective function's smoothness so that Newton search direction can be used. The work of Zhao [34] is based on this approach.

If we pull out the conicity from SMISOCP, we get stochastic mixed-integer linear programming (SMILP). SMILPs have also been studied widely; see for example [21, 26, 27]. Some solution methods for solving SMILP can be found in [17, 22, 25, 28, 29, 33]. Most of these methods are, in general, based on a combination of Benders decomposition and cutting-plane techniques and their modifications.

If we remove the uncertainty from SMISOCP, we get (deterministic) mixed-integer second-order cone programming (MISOCP). One very straightforward way to develop a method for solving MISOCPs is to use a branch-and-bound approach with an interior-point method designed especially for second-order cone programs. To be competitive in large-scale MISOCPs, it is important to reduce the number of nodes in the tree by using relaxation and cutting techniques designed for MISOCPs and to reduce the runtime at each node by using a second-order cone solver that can detect infeasibility and warmstart; see [10–13, 15]. Çezik and Iyengar [18] discussed cut generation for mixed-integer conic programming problems by extending some well-known strategies for mixed-integer linear programs to mixed-integer conic programs, including second-order cones. Atamtürk and Narayanan [9] introduced rounding cuts for MISOCPs by decomposing each second-order constraint into 2-dimensional polyhedral set (see Vielma et al. [31]). Drewes [19] proposed a hybrid branch-and-bound and branch-and-cut method for MISOCPs.

MISOCP is a subset of mixed-integer nonlinear programming (MINLP). For MINLPs, Benson and Shanno [10] present an exact primal-dual penalty method for this class of optimization problems by using an infeasible interior-point method to solve the resulting nonlinear subproblems and a branch-and-bound framework to handle the integer variables. See also the work of Benson and Vanderbei [15] which explains how to rewrite second-order cone and semidefinite problems so that they can be solved with a general-purpose interior-point method for nonlinear programming.

MINLP:	(Deterministic) mixed-integer nonlinear programming
MISOCP:	(Deterministic) mixed-integer second-order cone programming
SSOCP:	Stochastic second-order cone programming
SMILP:	Stochastic mixed-integer linear programming
SMISOCP:	Stochastic mixed-integer second-order cone programming
SMINLSOCP:	Stochastic mixed-integer nonlinear second-order cone programming

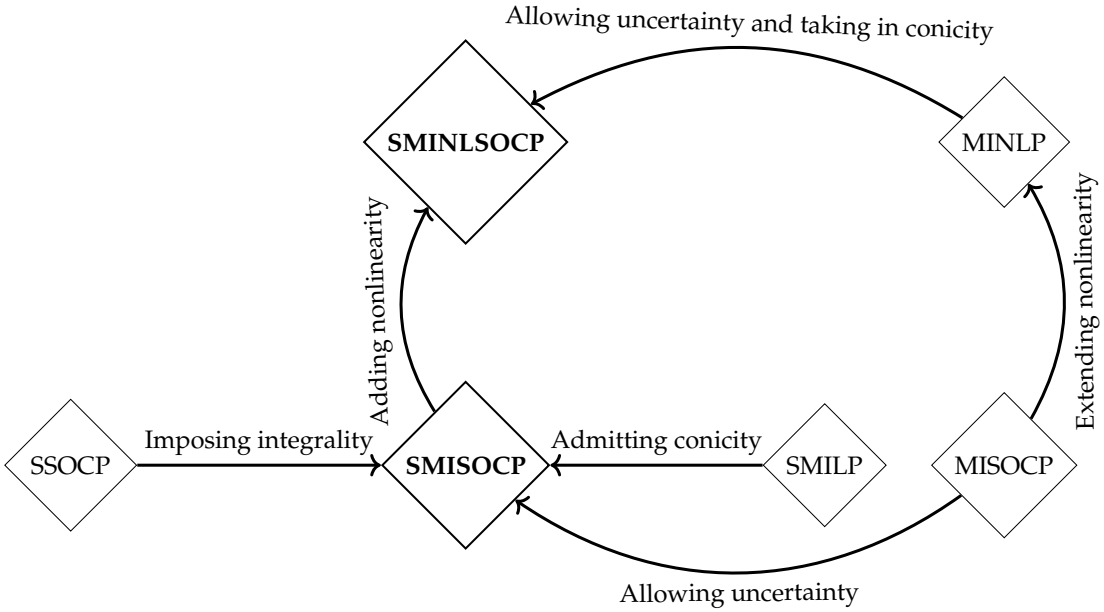

Figure 1: Conceptual relationships between SMISOCP and SMINLSOCP problems and other relevant optimization problems.

In this paper, we develop a combined branch-and-bound and interior-point algorithm for SMINLSOCPs, which uses a polynomial-time infeasible interior-point method for the inner-level second-order cone subproblems and a branch-and-bound technique for outer-level improvements. Our approach is suitable for the outer approximation framework, where each second-order cone subproblem in a sequence may differ from the others in a variety of ways due to its ability to handle any change to the problem. In addition, if the problem is binary stochastic second-order cone programming, the mathematical program with equilibrium constraints (MPEC) formulation is utilized. The MPEC reformulation could be solved before the start of the branch-and-bound code in order to prevent interrupting the algorithm's flow. Nevertheless, the solution of the continuous relaxation can serve as a "good warmstart" for solving the MPEC reformulation and may aid in locating a local optimum with a lower objective function value.

Warmstarting is the use of information obtained during the process of solving a problem to solve subsequent problems that are closely related. In the case of mixed-integer nonlinear programming problems, warmstarting means setting the initial solution (which has primal, dual, and slack variables) to the optimal solution of its parent. The major perceived difficulties of an interior-point method are the infeasibility detection, the lack of warmstarting, and the handling of fixed variables. Also, using the branch-and-bound approach may cause subproblems to become infeasible, so it is important to detect these instances early and reliably. During the branch-and-bound method, a problem's lower and upper bounds may become the same. Since these are fixed variables, the optimal set of Lagrange multipliers being unbounded.

This paper is organized as follows. In Section 2.2, we introduce some notations and definitions related to the second order cone. We also introduce our problem formulation in Section 2.2. Section 3 presents an overview of the resulting algorithm, including a brief description of both the branch-and-bound method and the infeasible interior-point method. In Section 4, we show how the cone constraint can be rewritten in a different way to make it smooth and convex. The inner- and outer-levels algorithmic improvements are discussed in detail in Sections 5 and 6. In Section 7, some numerical results on randomly generated SMISOCP problems are provided in order to assess the performance of the proposed algorithm.

2 Problem formulation

In this section, we present some notations and definitions that will be used and needed throughout the paper. Then we describe the SMINLSOCP problem formulation and its deterministic equivalence.

2.1 The second-order cone: Notations and related definitions

We use ", " and "; " to adjoin matrices and vectors in a row and column, respectively. To illustrate this, let $\boldsymbol{x}$ and $\boldsymbol{y}$ be vectors, then

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \left(\boldsymbol{x}^\mathsf{T}, \boldsymbol{y}^\mathsf{T}\right)^\mathsf{T} = (\boldsymbol{x}; \boldsymbol{y}).$$

We use $\mathcal{E}^n$ to denote the Euclidean space $\mathbb{R} \times \mathbb{R}^{n-1}$ whose vectors indexed from 0. For any vector $\boldsymbol{x} \in \mathcal{E}^n$ indexed from 0, we write $\bar{\boldsymbol{x}}$ for the sub-vector composed of entries 1 through $n-1$, so $\boldsymbol{x} \triangleq (x_0; \bar{\boldsymbol{x}})$.

The nth-dimensional second-order cone (also known as the Lorentz or quadratic cone) is defined as

$$\mathcal{E}^n_+ \triangleq \{\boldsymbol{x} = (x_0; \bar{\boldsymbol{x}}) \in \mathcal{E}^n : x_0 \geq \|\bar{\boldsymbol{x}}\|\},$$

where $\|\cdot\|$ represent the Euclidean norm. Figure 2 shows the 3rd-dimensional second-order cone $\mathcal{E}^3_+$.

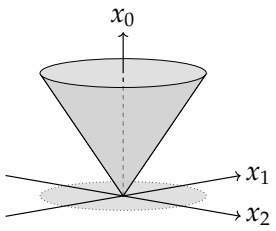

Figure 2: The 3rd-dimentional second-order cone $\mathcal{E}^3_+ \triangleq \left\{(x_0; x_1; x_2) \in \mathcal{E}^3 : x_0 \geq \sqrt{x_1^2 + x_2^2}\right\}$.

The interior of this cone is the set

$$\operatorname{int} \mathcal{E}^n_+ \triangleq \{\boldsymbol{x} = (x_0; \bar{\boldsymbol{x}}) \in \mathcal{E}^n : x_0 > \|\bar{\boldsymbol{x}}\|\}.$$

The cone $\mathcal{E}^n_+$ is pointed, closed, convex, and self dual (i.e., it equals its dual cone). In fact, $\mathcal{E}^n_+$ is considered one of the most important three symmetric cones (see [20]). If n is known from the context and $\boldsymbol{x} \in \mathcal{E}^n$, we use $\boldsymbol{x} \succeq \boldsymbol{0}$ (respectively, $\boldsymbol{x} \succ \boldsymbol{0}$) to indicate that $\boldsymbol{x} \in \mathcal{E}^n_+$ (respectively, $\boldsymbol{x} \in \operatorname{int} \mathcal{E}^n_+$).

As an important tool for our upcoming development, we associate with each $\boldsymbol{x} \in \mathcal{E}^n$ the arrow-shaped matrix, which is defined as

$$\operatorname{Arw}(\boldsymbol{x}) \triangleq \begin{bmatrix} x_0 & \bar{\boldsymbol{x}}^{\mathsf{T}} \\ \bar{\boldsymbol{x}} & x_0 I \end{bmatrix}.$$

Note that $\boldsymbol{x} \succeq \boldsymbol{0}$ (respectively, $\boldsymbol{x} \succ \boldsymbol{0}$) if and only if $\operatorname{Arw}(\boldsymbol{x})$ is positive semidefinite (respectively, $\operatorname{Arw}(\boldsymbol{x})$ is positive definite). We use $\boldsymbol{e}_n \triangleq (1; \boldsymbol{0})$ for the identity vector in $\mathcal{E}^n_+$. As another important tool for our development, we will use the Jordan multiplication $\circ : \mathcal{E}^n_+ \longrightarrow \mathcal{E}^n_+$, which is defined as

$$\boldsymbol{x} \circ \boldsymbol{y} \triangleq \begin{bmatrix} \boldsymbol{x}^{\mathsf{T}} \boldsymbol{y} \\ x_0 \bar{\boldsymbol{y}} + y_0 \bar{\boldsymbol{x}} \end{bmatrix} = \operatorname{Arw}(\boldsymbol{x}) \boldsymbol{y} = \operatorname{Arw}(\boldsymbol{x}) \operatorname{Arw}(\boldsymbol{y}) \boldsymbol{e}_n.$$

It is known that $(\mathcal{E}^n_+, \circ)$ is a Jordan algebra, and that it is a Euclidean Jordan algebra under the standard inner product (see [20] for definitions).

The spectral decomposition of $\boldsymbol{x} \in \mathcal{E}^n$ is a decomposition of $\boldsymbol{x}$ into its eigenvectors (also called idempotents), denoted as $\boldsymbol{c}_1(\boldsymbol{x})$ and $\boldsymbol{c}_2(\boldsymbol{x})$, together with its eigenvalues, denoted as $\lambda_1(\boldsymbol{x})$ and $\lambda_2(\boldsymbol{x})$. This decomposition is obtained as follows:

$$\boldsymbol{x} = \underbrace{(x_0 + \|\bar{\boldsymbol{x}}\|)}_{\lambda_1(\boldsymbol{x})} \underbrace{\left(\frac{1}{2}\right)\left(1; \frac{\bar{\boldsymbol{x}}}{\|\bar{\boldsymbol{x}}\|}\right)}_{\boldsymbol{c}_1(\boldsymbol{x})} + \underbrace{(x_0 - \|\bar{\boldsymbol{x}}\|)}_{\lambda_2(\boldsymbol{x})} \underbrace{\left(\frac{1}{2}\right)\left(1; -\frac{\bar{\boldsymbol{x}}}{\|\bar{\boldsymbol{x}}\|}\right)}_{\boldsymbol{c}_2(\boldsymbol{x})}.$$

The values $\operatorname{trace}(\boldsymbol{x}) \triangleq \lambda_1(\boldsymbol{x}) + \lambda_1(\boldsymbol{x}) = 2x_0$ and $\det(\boldsymbol{x}) \triangleq \lambda_1(\boldsymbol{x})\lambda_1(\boldsymbol{x}) = x_0^2 - \|\bar{\boldsymbol{x}}\|^2$ are called the trace and determinant of $\boldsymbol{x}$, respectively. We call $\boldsymbol{x} \in \mathcal{E}^n$ invertible (or nonsingular) if $\det(\boldsymbol{x}) \neq 0$ and call $\boldsymbol{x}^{-1}$ the inverse of $\boldsymbol{x}$, which is defined as $\boldsymbol{x}^{-1} \triangleq \lambda_1^{-1}(\boldsymbol{x})\boldsymbol{c}_1(\boldsymbol{x}) + \lambda_2^{-1}(\boldsymbol{x})\boldsymbol{c}_2(\boldsymbol{x}) = \frac{R\boldsymbol{x}}{\det(\boldsymbol{x})}$, where $R \triangleq [(1, \boldsymbol{0}^{\mathsf{T}}); (\boldsymbol{0}, -I)]$ is the reflection matrix in $\mathcal{E}^n$. Note that every positive definite vector in $\mathcal{E}^n$ is invertible and its inverse is also positive definite.

The logarithmic barrier function $f : \operatorname{int} \mathcal{E}^n_+ \longrightarrow \mathbb{R}$ is defined as $f(\boldsymbol{x}) \triangleq -\ln \det(\boldsymbol{x})$. This function will play an essential role in our algorithmic development.

The above notions and concepts are also used in the block sense as follows: Let $\boldsymbol{x} \triangleq (\boldsymbol{x}_1; \boldsymbol{x}_2; \ldots; \boldsymbol{x}_r)$, $\boldsymbol{y} \triangleq (\boldsymbol{y}_1; \boldsymbol{y}_2; \ldots, \boldsymbol{y}_r)$, where $\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathcal{E}^{n_i}$ for each $i = 1, 2, \ldots, r$. Then

(i) $\mathcal{E}_r \triangleq \mathcal{E}^l \times \mathcal{E}^{n_2} \times \cdots \times \mathcal{E}^{n_r}$, $\mathcal{E}_{r+} \triangleq \mathcal{E}^{n_1}_+ \times \mathcal{E}^{n_2}_+ \times \cdots \times \mathcal{E}^{n_r}_+$, and $\operatorname{int} \mathcal{E}_{r+} \triangleq \operatorname{int} \mathcal{E}^{n_1}_+ \times \operatorname{int} \mathcal{E}^{n_2}_+ \times \cdots \times \operatorname{int} \mathcal{E}^{n_r}_+$;

(ii) $\operatorname{Arw}(\boldsymbol{x}) \triangleq \operatorname{Arw}(\boldsymbol{x}_1) \oplus \operatorname{Arw}(\boldsymbol{x}_2) \oplus \cdots \oplus \operatorname{Arw}(\boldsymbol{x}_r)$ is the arrow-shaped in $\boldsymbol{x} \in \mathcal{E}_r$; [1]

[1]The direct sum of two square matrices X and Y is the block matrix $X \oplus Y \triangleq \begin{bmatrix} X & O \\ O & Y \end{bmatrix}$.

(iii) $\boldsymbol{x} \circ \boldsymbol{y} \triangleq (\boldsymbol{x}_1 \circ \boldsymbol{y}_1; \boldsymbol{x}_2 \circ \boldsymbol{y}_2; \cdots; \boldsymbol{x}_r \circ \boldsymbol{y}_r)$ is the Jordan product in $\mathcal{E}_r$;

(iv) $\boldsymbol{x}^T \boldsymbol{y} \triangleq \boldsymbol{x}_1^T \boldsymbol{y}_1 + \boldsymbol{x}_2^T \boldsymbol{y}_2 + \cdots + \boldsymbol{x}_r^T \boldsymbol{y}_r$ is the inner product in $\mathcal{E}_r$;

(v) $\boldsymbol{e} \triangleq (\boldsymbol{e}_{n_1}; \boldsymbol{e}_{n_2}; \ldots; \boldsymbol{e}_{n_r})$ is the identity of $\mathcal{E}_r$ (its dimension will be clear from the context);

(vi) $\boldsymbol{x}^1 \triangleq (\boldsymbol{x}_1^{-1}, \boldsymbol{x}_2^{-1}, \ldots, \boldsymbol{x}_r^{-1})$ is the inverse of $\boldsymbol{x}$ in $\mathcal{E}_r$ provided that $\det(\boldsymbol{x}_i) \neq 0$ for $i = 1, 2, \ldots, r$.

(vii) $\det(\boldsymbol{x}) \triangleq \prod_{i=1}^r \det(\boldsymbol{x}_i)$ is the determinant of $\boldsymbol{x} \in \mathcal{E}_r$;

(viii) $f(\boldsymbol{x}) \triangleq -\ln \det(\boldsymbol{x}) = -\sum_{i=1}^r \ln \det(\boldsymbol{x}_i)$ is the logarithmic barrier function of $\boldsymbol{x} \in \operatorname{int} \mathcal{E}_{r+}$.

If the block setting is known from the context and $\boldsymbol{x} \in \mathcal{E}_r$, we use $\boldsymbol{x} \succeq_r \boldsymbol{0}$ to indicate that $\boldsymbol{x} \in \mathcal{E}_{r+}$. This occurs if $\boldsymbol{x}$ is partitioned conformally as $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r)$ and each $\boldsymbol{x}_i \in \mathcal{E}^{n_i}$ satisfies $\boldsymbol{x}_i \succeq \boldsymbol{0}$ for $i = 1, 2, \ldots, r$. Similarly, we use $\boldsymbol{x} \succ_r \boldsymbol{0}$ to indicate that $\boldsymbol{x} \in \operatorname{int} \mathcal{E}_{r+}$.

We use O and I for zero and identity matrices respectively. We also write $\boldsymbol{1}$ for a vector with all entries equal to one. The dimensions of O, I and $\boldsymbol{1}$ will be clear from the context. For any mixed-integer vector $\boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}$ indexed from 1, we write $\ddot{\boldsymbol{x}}$ for the integer sub-vector composed of entries 1 through p, and write $\check{\boldsymbol{x}}$ for the real sub-vector composed of entries $p+1$ through n, so $\boldsymbol{x} \triangleq (\ddot{\boldsymbol{x}}; \check{\boldsymbol{x}}) \in \mathbb{Z}^p \times \mathbb{R}^{n-p}$. Finally, for any strictly positive vector $\boldsymbol{x} \in \mathbb{R}^n$, we define $\ln \boldsymbol{x} \triangleq \sum_{i=1}^n \ln x_i$, $\boldsymbol{x}^{-1} \triangleq (x_1^{-1}; x_2^{-1}; \ldots; x_n^{-1})$, and $X \triangleq \operatorname{Diag}(x_1, x_2, \ldots, x_n)$. That is, X denotes the $n \times n$ diagonal matrix whose diagonal entries are $x_1, x_2, \ldots, x_n$.

2.2 The SMINLSOCP problem formulation

Let m, n, p, q, l and d be positive integers. Let also $\ell_1, \ell_2, \ldots, \ell_r, d_1, d_2, \ldots, d_s$ be positive integers with $\ell = \sum_{i=1}^r \ell_i$ and $d = \sum_{j=1}^s d_j$. Let $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^\ell$ and $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^d$ be twice continuously differentiable functions, representing the constraint vectors of the first and second stages, respectively. Let also $f_1 : \mathbb{R}^n \to \mathbb{R}$ and $f_2 : \mathbb{R}^m \to \mathbb{R}$ be twice continuously differentiable functions, representing the first and second stage nonlinear objectives, respectively. Our problem is defined with deterministic data $f_1(\cdot)$ and $\boldsymbol{h}(\cdot)$ and with random data $f_2(\cdot, \omega)$ and $\boldsymbol{g}(\cdot, \omega)$ whose the realizations depend underlying outcome ω in an event space Ω with a known probability function $\mathbb{P}$. Given this data, the two-stage SMINLSOCP problem has the form:

$$\begin{array}{ll} \min & \boldsymbol{f}_1(\boldsymbol{x}) + \mathbb{E}\left[\varphi(\boldsymbol{x}, \omega)\right] \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) \succeq_r \boldsymbol{0}, \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, \end{array} \tag{1}$$

where $\boldsymbol{x} = (\ddot{\boldsymbol{x}}; \check{\boldsymbol{x}}) \in \mathbb{Z}^p \times \mathbb{R}^{n-p}$ is the first-stage decision variable, $\mathbb{E}\left[\varphi(\boldsymbol{x}, \omega)\right] \triangleq \int_\Omega \varphi(\boldsymbol{x}, \omega) \mathbb{P}(d\omega)$, and $\varphi(\boldsymbol{x}, \omega)$ is the minimum value of the problem

$$\begin{array}{ll} \min & f_2(\boldsymbol{y}, \omega) \\ \text{s.t.} & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \omega) \succeq_s \boldsymbol{0}, \\ & \boldsymbol{y} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}. \end{array} \tag{2}$$

Here $\boldsymbol{y} = (\ddot{\boldsymbol{y}}; \check{\boldsymbol{y}}) \in \mathbb{Z}^q \times \mathbb{R}^{m-q}$ is the second-stage decision variable. When $p = q = 0$ we have the two-stage SSOCP problem, and when $n = p$ and $m = q$ we have a two-stage stochastic integer programming problem.

We examine Problems (1) and (2) when the event space Ω is discrete and finite.

Let $\{f_2^{(k)}(\boldsymbol{y}^{(k)}), \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) : k = 1, \ldots, K\}$ be the set of the possible data of the random data $(f_2(\boldsymbol{y}, \omega), \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \omega))$ and let $\pi_k \triangleq \mathbb{P}((f_2^{(k)}(\boldsymbol{y}^{(k)}), \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)})) = (f_2(\boldsymbol{y}, \omega), \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \omega)))$ be the associated probability for $k = 1, 2, \ldots, K$. Given this, Problems (1) and (2) can be written as

$$\begin{array}{ll} \min & f_1(\boldsymbol{x}) + \sum_{k=1}^K \pi_k \varphi^{(k)}(\boldsymbol{x}) \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) \succeq_r \boldsymbol{0}, \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, \end{array} \tag{3}$$

where $\varphi^{(k)}(\boldsymbol{x})$, for $k = 1, 2, \ldots, K$, is the minimum value of the problem

$$\begin{array}{ll} \min & f_2^{(k)}(\boldsymbol{y}^{(k)}) \\ \text{s.t.} & \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) \succeq_s \boldsymbol{0}, \\ & \boldsymbol{y}^{(k)} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}. \end{array} \tag{4}$$

For more convenience, we redefine the function of Problem (4) as $f_2^{(k)}(\boldsymbol{y}^{(k)}) \triangleq \pi_k f_2^{(k)}(\boldsymbol{y}^{(k)})$ for $k = 1, 2, \ldots, K$. We write the SMINLSOCP problem (3, 4) as a large-scale mixed-intger nonlinear second-order cone programming problem with finite event space in primal standard form to get

$$\begin{array}{lccccccccccc} \min & f_1(\boldsymbol{x}) & + & f_2^{(1)}(\boldsymbol{y}^{(1)}) & + & f_2^{(2)}(\boldsymbol{y}^{(2)}) & + & \cdots & + & f_2^{(K)}(\boldsymbol{y}^{(K)}) & & \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) & & & & & & & & & \succeq_{\boldsymbol{r}} & \boldsymbol{0}, \\ & & & \boldsymbol{g}^{(1)}(\boldsymbol{x}, \boldsymbol{y}^{(1)}) & & & & & & & \succeq_s & \boldsymbol{0}, \\ & & & & & \boldsymbol{g}^{(1)}(\boldsymbol{x}, \boldsymbol{y}^{(2)}) & & & & & \succeq_s & \boldsymbol{0}, \\ & & & & & & & \ddots & & & \vdots & \\ & & & & & & & & & \boldsymbol{g}^{(K)}(\boldsymbol{x}, \boldsymbol{y}^{(K)}) & \succeq_s & \boldsymbol{0}, \\ & \boldsymbol{x} & & & & & & & & & \in & \mathbb{Z}^p \times \mathbb{R}^{n-p}, \\ & & & \boldsymbol{y}^{(1)}, & & \boldsymbol{y}^{(2)}, & & \ldots & & , \boldsymbol{y}^{(K)} & \in & \mathbb{Z}^q \times \mathbb{R}^{m-q}. \end{array} \tag{5}$$

We can also define

$$f\left(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(K)}\right) \triangleq f_1(\boldsymbol{x}) + \sum_{k=1}^{K} f_2^{(k)}\left(\boldsymbol{y}^{(k)}\right),$$

and rewrite Problem (5) compactly as

$$\begin{array}{lll} \min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(K)}) & \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) \succeq_{\boldsymbol{r}} \boldsymbol{0}, & \\ & \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) \succeq_s \boldsymbol{0}, & k = 1, 2, \ldots, K, \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, & \\ & \boldsymbol{y}^{(k)} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}, & k = 1, 2, \ldots, K. \end{array} \tag{6}$$

Problem (6) (or Problem (5)) is an SMINLSOCP problem in the primal form with a block diagonal structure. We will show that by exploiting the special structure of Problem (6), the computational work can be significantly reduced, especially when K is very large.

3 An overview of the proposed algorithm

In this section, we outline the proposed algorithm for solving Problem (6). We discuss two levels of iterations for our algorithm: Inner-level iterations and outer-level iterations. In inner-level iterations, we solve a continuous relaxation of the bound-modified problem. Here, we utilize the infeasible interior-point method developed by Alzalg et al. [8] for SSOCP and that developed by Venderbei and Shanno in [30] for nonlinear programming. In outer-level iterations, we apply a branch-and-bound technique based on Nemhauser and Wolsey [32] to handle the integrality of decision variables. We have chosen to include this section in order to define the framework and notations for the algorithmic improvements discussed in Sections 5 and 6.

3.1 Outer-level iterations: A branch-and-bound technique

In order to handle the integrality of the integer sub-vectors $\ddot{\boldsymbol{x}}, \ddot{\boldsymbol{y}}^{(1)}, \ddot{\boldsymbol{y}}^{(2)}, \ldots, \ddot{\boldsymbol{y}}^{(K)}$, we apply the branch-and-bound algorithm based on [32]. This algorithm builds a tree in which each node is associated with a relaxation of (6). Instead of integrality requirements, each node provides a set of lower and upper bounds for each $1 \le i \le p$, $1 \le j \le q$, and $1 \le k \le K$. In other words, the root node has the property that the constraints

$$\ddot{x}_i \in \mathbb{Z} \text{ or } \ddot{y}_j^{(k)} \in \mathbb{Z}, \text{ for } 1 \le i \le p, 1 \le j \le q, 1 \le k \le K,$$

are replaced by the bounds

$$l_i \le \ddot{x}_i \le u_i \text{ or } \tilde{l}_j^{(k)} \le \ddot{y}_j^{(k)} \le \tilde{u}_j^{(k)}, \text{ for } 1 \le i \le p, 1 \le j \le q, 1 \le k \le K,$$

respectively, where $l_i, u_i \in \mathbb{R}$ are (possibly infinite) lower and upper bounds of $\ddot{x}_i$ for each $i = 1, 2, \ldots, p$, and $\tilde{l}_j^{(k)}, \tilde{u}_j^{(k)} \in \mathbb{R}$ are (possibly infinite) lower and upper bounds of $\ddot{y}_j^{(k)}$ for each $j = 1, 2, \ldots, q$ and $k = 1, 2, \ldots, K$. In the branch-and-bound algorithm, the user can specify three algorithmic elements to fine-tune the algorithm's behavior. These elements are:

- *The search strategy*: This is the sequence in which the subproblems of the tree are examined. Generally, three basic search strategies are considered to determine the next node to branch from. "*The best first*" search strategy selects the node with the least lower bound value among the available nodes. "*The depth-first*" search strategy selects the active node with the lowest level in the tree, breaking ties either arbitrarily or by finding the node with the least lower bound value."*The breadth first*" selects the active node with the greatest level in the search tree by breaking ties arbitrarily or by finding the node with the least lower bound value.
- *The branching strategy*: An important question is how to branch or how to split a subproblem into smaller subproblems. The simplest scheme to branch the feasible set is to pick an integer variable with a fractional value in the bound-modified problem that can be solved in a recursive way at a node in the tree that cannot be explored.
- *The bounding strategy (pruning rules)*: These rules use pruned regions of the search space that are provably suboptimal. Solving the continuous relaxation problem of any subproblem gives an upper bound on its objective function. On very large models, an interior point approach may be best for solving the first problem.

The search and verification phases are two important phases for each branch-and-bound technique. In the search phase, the algorithm has not yet determined the optimum solution. During the verification phase, the optimal solution has already been found, but there are still unexplored subproblems in the tree that can be pruned. Now, we define $f^\star$ as the lowest value of the objective function among all feasible solutions to (6), but if there is a feasible solution to our problem, we put $f^\star = \infty$, so we have three possible outcomes:

(i) The problem (6) has an optimal solution $(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \boldsymbol{y}^{(2)^\star}, \ldots, \boldsymbol{y}^{(K)^\star})$ with $\ddot{\boldsymbol{x}}^\star \in \mathbb{Z}^p$ and $\ddot{\boldsymbol{y}}^{(k)^\star} \in \mathbb{Z}^q$ for $k = 1, 2, \ldots, K$. So, there is a feasible solution of our problem. Therefore, we do not pursue this node further (i.e., *fathoming by integrality*). In this case, we update $f^\star$ by letting it equal to $f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star})$ if $f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star}) < f^\star$.

(ii) The problem has an optimal solution $(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \boldsymbol{y}^{(2)^\star}, \ldots, \boldsymbol{y}^{(K)^\star})$ but $\ddot{x}_i^\star \notin \mathbb{Z}$ for some $i \in \{1, 2, \ldots, p\}$, or $\ddot{y}_j^{(k)^\star} \notin \mathbb{Z}$ for some $j \in \{1, 2, \ldots, q\}$ and some $k = 1, 2, \ldots, K$. If $f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star}) \geq f^\star$, each feasible solution its descendant may yield to (6) will have an optimal value that is at least as large as $f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star})$. Therefore, we do not pursue this node further (i.e., *fathoming by bounds*). If $f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star}) < f^\star$, we produce two new children for this node. One child has the bounds

$$l_i \leq \ddot{x}_i \leq \left\lfloor \ddot{x}_i^\star \right\rfloor \quad \text{or} \quad \tilde{l}_j^{(k)} \leq \ddot{y}_j^{(k)^\star} \leq \left\lfloor \ddot{y}_j^{(k)^\star} \right\rfloor,$$

and the other has the bounds

$$\left\lceil \ddot{x}_i^\star \right\rceil \leq \ddot{x}_i \leq u_i \quad \text{or} \quad \left\lceil \ddot{y}_j^{(k)^\star} \right\rceil \leq \ddot{y}_j^{(k)^\star} \leq \tilde{u}_j^{(k)}.$$

(iii) The problem at the generic node is infeasible, so the relaxation problem cannot produce feasible solutions to (6) because it has infeasible solutions. Therefore, we do not pursue this node further (i.e., *fathoming by infeasibility*).

Consequently, the branch-and-bound technique partitions the feasible region systematically while searching for feasible solutions to (6). If each node in the tree has been fathomed, Problem (6) has been resolved. Moreover, the optimal solution is the feasible solution with the lowest objective function value for Problem (6). So, the lowest objective function does not improve because the branch-and-bound tree is descended. On the other side, when solving the SMINLSOCP with nonconvex relaxation, a node's solution may be worse than that of its children. This occurs because an interior-point technique approaches only a local optimal solution, and these solutions may be in different neighborhoods. In this paper, we restrict our discussion and numerical testing to SMINLSOCPs with convex relaxations (i.e., the objective function f is convex and the constraint functions h_i and $g_j^{(k)}$, for $1 \leq i \leq \ell$, $1 \leq j \leq d$, and $1 \leq k \leq K$, are concave functions).

3.2 Inner-level iterations: An infeasible interior-point method

The described outer-level is based on repeatedly solving nonlinear second-order cone relaxations of Problem (6) with the addition of the bound constraints on discrete variables. We will use an infeasible interior-point method to solve a bound-modified problem at each step of the branch-and-bound algorithm. We rewrite Problem (6) as follows:

$$\begin{array}{lll} \min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(K)}) & \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) \succeq_r \boldsymbol{0}, & \\ & \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) \succeq_s \boldsymbol{0}, & k = 1, 2, \ldots, K, \\ & \boldsymbol{l} \leq \ddot{\boldsymbol{x}} \leq \boldsymbol{u}, & \\ & \tilde{\boldsymbol{l}}^{(k)} \leq \ddot{\boldsymbol{y}}^{(k)} \leq \tilde{\boldsymbol{u}}^{(k)}, & k = 1, 2, \ldots, K. \end{array} \tag{7}$$

We start by adding the slack variables $\boldsymbol{s}$ to the first second-order cone constraint and $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(K)}$ to the second set of second-order cone constraints. We also add the nonnegative slack vectors $\boldsymbol{w} \in \mathbb{R}^p$ and $\boldsymbol{v}^{(k)} \in \mathbb{R}^q$ to the lower bounds on $\ddot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{y}}^{(k)}$, respectively, and add the nonnegative slack vectors $\boldsymbol{t} \in \mathbb{R}^p, \boldsymbol{r}^{(k)} \in \mathbb{R}^q$ to the upper bounds on $\ddot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{y}}^{(k)}$, respectively, for $k = 1, 2, \ldots, K$. This yields:

$$\begin{array}{lll} \min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(K)}) & \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} = \boldsymbol{0}, & \\ & \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} = \boldsymbol{0}, & k = 1, 2, \ldots, K, \\ & \ddot{\boldsymbol{x}} - \boldsymbol{w} = \boldsymbol{l}, & \\ & \ddot{\boldsymbol{x}} + \boldsymbol{t} = \boldsymbol{u}, & \\ & \ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} = \tilde{\boldsymbol{l}}^{(k)}, & k = 1, 2, \ldots, K, \\ & \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} = \tilde{\boldsymbol{u}}^{(k)}, & k = 1, 2, \ldots, K, \\ & \boldsymbol{s} \succeq_r \boldsymbol{0}, \boldsymbol{z}^{(k)} \succeq_s \boldsymbol{0}, & k = 1, 2, \ldots, K, \\ & \boldsymbol{w}, \boldsymbol{t}, \boldsymbol{v}^{(k)}, \boldsymbol{r}^{(k)} \geq \boldsymbol{0}, & k = 1, 2, \ldots, K. \end{array} \tag{8}$$

It is worth noting that one or both of the bounds on $\ddot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{y}}^{(k)}$ may be infinite, for each $k = 1, 2, \ldots, K$, respectively. If any of the bounds is infinite, the relevant constraint can be just omitted. We incorporate the slack variables in the objective function of Problem (8) by adding logarithmic barrier terms and get:

$$\begin{array}{lll} \min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - \mu(\ln\det(\boldsymbol{s}) + \ln(\boldsymbol{w}) + \ln(\boldsymbol{t}) + \sum_{k=1}^{K}(\ln\det(\boldsymbol{z}^{(k)}) + \ln(\boldsymbol{v}^{(k)}) + \ln(\boldsymbol{r}^{(k)}))) & \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} = \boldsymbol{0}, & \\ & \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} = \boldsymbol{0}, & k = 1, 2, \ldots, K, \\ & \ddot{\boldsymbol{x}} - \boldsymbol{w} = \boldsymbol{l}, & \\ & \ddot{\boldsymbol{x}} + \boldsymbol{t} = \boldsymbol{u}, & \\ & \ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} = \tilde{\boldsymbol{l}}^{(k)}, & k = 1, 2, \ldots, K, \\ & \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} = \tilde{\boldsymbol{u}}^{(k)}, & k = 1, 2, \ldots, K, \end{array} \tag{9}$$

where $\mu > 0$ is a barrier parameter.

Let $\boldsymbol{\nu}, \boldsymbol{\gamma}, \boldsymbol{\psi}, \boldsymbol{\vartheta}^{(k)}, \boldsymbol{\lambda}^{(k)}$, and $\boldsymbol{\upsilon}^{(k)}$, $k = 1, 2, \ldots, K$, be the Lagrange multipliers (or dual variables) for Problem (9). Following our notations in Subsection 2.1, the matrices $W, \Gamma, T, \Psi, V^{(k)}, \Lambda^{(k)}, R^{(k)}$ and $\Upsilon^{(k)}$ stand for the diagonal matrices whose diagonal entries are the components of the vectors $\boldsymbol{w}, \boldsymbol{\gamma}, \boldsymbol{t}, \boldsymbol{\psi}, \boldsymbol{v}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{r}^{(k)}$ and $\boldsymbol{\upsilon}^{(k)}$, respectively, for $k = 1, 2, \ldots, K$. Let $A_{\ddot{x}}(\boldsymbol{x})$ and $A_{\check{x}}(\boldsymbol{x})$ be the transpose of the Jacobians of the constraint function $\boldsymbol{h}(\boldsymbol{x})$ with respect to $\ddot{\boldsymbol{x}}$ and $\check{\boldsymbol{x}}$, respectively. Let also $B_{\ddot{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$, $B_{\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$, $B_{\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ and $B_{\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ be the transpose of the Jacobians of the constraint function $\boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ with respect to $\ddot{\boldsymbol{x}}, \check{\boldsymbol{x}}, \ddot{\boldsymbol{y}}^{(k)}$ and $\check{\boldsymbol{y}}^{(k)}$, respectively, for each $k = 1, 2, \ldots, K$ (for ease of display, we will sometimes omit the use of function arguments).

The first-order optimality conditions of (9) are

$$
\begin{array}{rcll}
\nabla_{\ddot{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\ddot{x}}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{\nu} - \sum_{k=1}^{K} B_{\ddot{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\vartheta}^{(k)} - \boldsymbol{\gamma} - \boldsymbol{\psi} & = & \mathbf{0}, & \\
\nabla_{\check{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\check{x}}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{\nu} - \sum_{k=1}^{K} B_{\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)} & = & \mathbf{0}, & \\
\nabla_{\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)} - \boldsymbol{\lambda}^{(k)} - \boldsymbol{\upsilon}^{(k)} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
\nabla_{\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
\boldsymbol{s} \circ \boldsymbol{\nu} - 2\mu \boldsymbol{e} & = & \mathbf{0}, & \\
W\boldsymbol{\gamma} - \mu \mathbf{1} & = & \mathbf{0}, & \\
T\boldsymbol{\psi} - \mu \mathbf{1} & = & \mathbf{0}, & \\
\boldsymbol{z}^{(k)} \circ \boldsymbol{\vartheta}^{(k)} - 2\mu \boldsymbol{e} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
V^{(k)} \boldsymbol{\lambda}^{(k)} - \mu \mathbf{1} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
R^{(k)} \boldsymbol{\upsilon}^{(k)} - \mu \mathbf{1} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} & = & \mathbf{0}, & \\
\boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
\ddot{\boldsymbol{x}} - \boldsymbol{w} - \boldsymbol{l} & = & \mathbf{0}, & \\
\ddot{\boldsymbol{x}} + \boldsymbol{t} - \boldsymbol{u} & = & \mathbf{0}, & \\
\ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} - \tilde{\boldsymbol{l}}^{(k)} & = & \mathbf{0}, & k = 1, 2, \ldots, K, \\
\ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} - \tilde{\boldsymbol{u}}^{(k)} & = & \mathbf{0}, & k = 1, 2, \ldots, K.
\end{array}
\tag{10}
$$

We use Newton's method to solve System (10). Let

$$
\begin{array}{rcl}
\boldsymbol{\sigma}_{\ddot{x}} & \triangleq & \nabla_{\ddot{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\ddot{x}}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{\nu} - \sum_{k=1}^{K} B_{\ddot{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\vartheta}^{(k)} - \boldsymbol{\gamma} - \boldsymbol{\psi}, \\
\boldsymbol{\sigma}_{\check{x}} & \triangleq & \nabla_{\check{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\check{x}}(\boldsymbol{x})^{\mathsf{T}} \boldsymbol{\nu} - \sum_{k=1}^{K} B_{\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)}, \\
\boldsymbol{\sigma}_{\ddot{y}^{(k)}} & \triangleq & \nabla_{\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)} - \boldsymbol{\lambda}^{(k)} - \boldsymbol{\upsilon}^{(k)}, \\
\boldsymbol{\sigma}_{\check{y}^{(k)}} & \triangleq & \nabla_{\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\check{y}^{(K)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^{\mathsf{T}} \boldsymbol{\nu}^{(k)},
\end{array}
$$

and

$$
\begin{array}{lcllcllcl}
\boldsymbol{\beta}_{\boldsymbol{s}} & \triangleq & 2\mu \boldsymbol{e} - \boldsymbol{s} \circ \boldsymbol{\nu}, & \boldsymbol{\beta}_{\gamma} & \triangleq & \mu \boldsymbol{w}^{-1} - \boldsymbol{\gamma}, & \boldsymbol{\beta}_{\psi} & \triangleq & \mu \boldsymbol{t}^{-1} - \boldsymbol{\psi}, \\
\boldsymbol{\beta}_{\boldsymbol{z}^{(k)}} & \triangleq & 2\mu \boldsymbol{e} - \boldsymbol{z}^{(k)} \circ \boldsymbol{\vartheta}^{(\boldsymbol{k})}, & \boldsymbol{\beta}_{\boldsymbol{\lambda}^{(k)}} & \triangleq & \mu \boldsymbol{v}^{(\boldsymbol{k})^{-1}} - \boldsymbol{\lambda}^{(k)}, & \boldsymbol{\beta}_{\upsilon^{(k)}} & \triangleq & \mu \boldsymbol{r}^{(\boldsymbol{k})^{-1}} - \boldsymbol{\upsilon}^{(k)}, \\
\boldsymbol{\rho} & \triangleq & \boldsymbol{s} - \boldsymbol{h}(\boldsymbol{x}), & \boldsymbol{\varrho} & \triangleq & \boldsymbol{l} - \ddot{\boldsymbol{x}} + \boldsymbol{w}, & \boldsymbol{\tau} & \triangleq & \boldsymbol{u} - \boldsymbol{t} - \ddot{\boldsymbol{x}}, \\
\boldsymbol{\eta}^{(k)} & \triangleq & \boldsymbol{z}^{(k)} - \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), & \boldsymbol{\zeta}^{(k)} & \triangleq & \tilde{\boldsymbol{l}}^{(k)} - \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{v}^{(k)}, & \boldsymbol{\kappa}^{(k)} & \triangleq & \tilde{\boldsymbol{u}}^{(k)} - \boldsymbol{r}^{(k)} - \ddot{\boldsymbol{y}}^{(k)}.
\end{array}
$$

Let also

$$
C \triangleq \begin{bmatrix}
-I & & & & & & & & \\
I & & & & & & & & \\
 & -I & & & & & & & \\
 & & -I & & & & & & \\
 & & & I & & & & & \\
 & & & & -I & & & & \\
 & & & & & \ddots & & & \\
 & & & & & & -I & & \\
 & & & & & & I & & \\
 & & & & & & & -I &
\end{bmatrix},
H \triangleq \begin{bmatrix}
I & & & & & & & & & \\
 & I & & & & & & & & \\
 & & \mathrm{Arw}(\boldsymbol{s}) & & & & & & & \\
 & & & I & & & & & & \\
 & & & & I & & & & & \\
 & & & & & \mathrm{Arw}(\boldsymbol{z}^{(1)}) & & & & \\
 & & & & & & \ddots & & & \\
 & & & & & & & I & & \\
 & & & & & & & & I & \\
 & & & & & & & & & \mathrm{Arw}(\boldsymbol{z}^{(K)})
\end{bmatrix},
$$

$$
D \triangleq \begin{bmatrix}
W^{-1}\Gamma & & & & & & & & & \\
 & T^{-1}\Psi & & & & & & & & \\
 & & \mathrm{Arw}(\boldsymbol{\nu}) & & & & & & & \\
 & & & V^{(1)^{-1}}\Lambda^{(1)} & & & & & & \\
 & & & & R^{(1)^{-1}}\Upsilon^{(1)} & & & & & \\
 & & & & & \mathrm{Arw}(\boldsymbol{\vartheta}^{(1)}) & & & & \\
 & & & & & & \ddots & & & \\
 & & & & & & & V^{(K)-1}\Lambda^{(K)} & & \\
 & & & & & & & & R^{(K)-1}\Upsilon^{(K)} & \\
 & & & & & & & & & \mathrm{Arw}(\boldsymbol{\vartheta}^{(K)})
\end{bmatrix},
$$

$$J \triangleq \begin{bmatrix} I & O & O & O & \cdots & O & O \\ I & O & O & O & \cdots & O & O \\ A_{\ddot{x}} & A_{\check{x}} & O & O & \cdots & O & O \\ O & O & I & O & \cdots & O & O \\ O & O & I & O & \cdots & O & O \\ B_{\ddot{x}} & B_{\check{x}} & B_{\ddot{y}^{(1)}} & B_{\check{y}^{(1)}} & \cdots & O & O \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ O & O & O & O & \cdots & I & O \\ O & O & O & O & \cdots & I & O \\ B_{\ddot{x}} & B_{\check{x}} & O & O & \cdots & B_{\ddot{y}^{(K)}} & B_{\check{y}^{(K)}} \end{bmatrix}, \text{ and } G \triangleq \begin{bmatrix} G_{\ddot{x}\ddot{x}} & G_{\ddot{x}\check{x}} & G_{\ddot{x}\ddot{y}^{(1)}} & G_{\ddot{x}\check{y}^{(1)}} & \dots & G_{\ddot{x}\ddot{y}^{(K)}} & G_{\ddot{x}\check{y}^{(K)}} \\ G_{\check{x}\ddot{x}} & G_{\check{x}\check{x}} & G_{\check{x}\ddot{y}^{(1)}} & G_{\check{x}\ddot{y}^{(2)}} & \dots & G_{\check{x}\ddot{y}^{(K)}} & G_{\check{x}\check{y}^{(K)}} \\ G_{\ddot{x}\ddot{y}^{(1)}} & G_{\check{x}\ddot{y}^{(1)}} & G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} & G_{\ddot{y}^{(1)}\check{y}^{(1)}} & \dots & G_{\ddot{y}^{(1)}\ddot{y}^{(K)}} & G_{\ddot{y}^{(1)}\check{y}^{(K)}} \\ G_{\ddot{x}\check{y}^{(1)}} & G_{\check{x}\check{y}^{(1)}} & G_{\ddot{y}^{(1)}\check{y}^{(1)}} & G_{\check{y}^{(1)}\check{y}^{(2)}} & \dots & G_{\ddot{y}^{(K)}\check{y}^{(1)}} & G_{\check{y}^{(1)}\check{y}^{(K)}} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ G_{\ddot{x}\ddot{y}^{(K)}} & G_{\check{x}\ddot{y}^{(K)}} & G_{\ddot{y}^{(1)}\ddot{y}^{(K)}} & G_{\check{y}^{(1)}\ddot{y}^{(K)}} & \dots & G_{\ddot{y}^{(K)}\ddot{y}^{(K)}} & G_{\check{y}^{(K)}\ddot{y}^{(K)}} \\ G_{\ddot{x}\check{y}^{(K)}} & G_{\check{x}\check{y}^{(K)}} & G_{\ddot{y}^{(1)}\check{y}^{(K)}} & G_{\check{y}^{(1)}\ddot{y}^{(K)}} & \dots & G_{\ddot{y}^{(K)}\check{y}^{(K)}} & G_{\check{y}^{(K)}\check{y}^{(K)}} \end{bmatrix},$$

where the entries of the Hessian matrix G is obtained as follows for $k = 1, \dots, K$:

$$\begin{aligned}
G_{\ddot{x}\ddot{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{x}\ddot{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{i=1}^{\ell} \nu_i \nabla^2_{\ddot{x}\ddot{x}} h_i(\boldsymbol{x}) - \sum_{k=1}^{K} \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{x}\ddot{x}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\ddot{x}\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{x}\check{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{i=1}^{\ell} \nu_i \nabla^2_{\ddot{x}\check{x}} h_i(\boldsymbol{x}) - \sum_{k=1}^{K} \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{x}\check{x}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\ddot{x}\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{x}\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{x}\ddot{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\ddot{x}\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{x}\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{x}\check{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\check{x}\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\check{x}\check{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{i=1}^{\ell} \nu_i \nabla^2_{\check{x}\check{x}} h_i(\boldsymbol{x}) - \sum_{k=1}^{K} \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\check{x}\check{x}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\check{x}\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\check{x}\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\check{x}\ddot{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\check{x}\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\check{x}\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\check{x}\check{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\ddot{y}^{(k)}\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{y}^{(k)}\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{i=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{y}^{(k)}\ddot{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\ddot{y}^{(k)}\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\ddot{y}^{(k)}\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\ddot{\boldsymbol{y}}^{(k)}\boldsymbol{y}_2^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}), \\
G_{\check{y}^{(k)}\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) &= \nabla^2_{\check{y}^{(k)}\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \dots, \boldsymbol{y}^{(K)}) - \sum_{j=1}^{d} \vartheta_j^{(k)} \nabla^2_{\check{y}^{(k)}\check{y}^{(k)}} g_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}).
\end{aligned}$$

We also define

$$\boldsymbol{\Delta}_p \triangleq \begin{bmatrix} \boldsymbol{\Delta w} \\ \boldsymbol{\Delta t} \\ \boldsymbol{\Delta s} \\ \boldsymbol{\Delta v^{(1)}} \\ \boldsymbol{\Delta r^{(1)}} \\ \boldsymbol{\Delta z^{(1)}} \\ \vdots \\ \boldsymbol{\Delta v^{(K)}} \\ \boldsymbol{\Delta r^{(K)}} \\ \boldsymbol{\Delta z^{(K)}} \end{bmatrix}, \boldsymbol{\Delta}_d \triangleq \begin{bmatrix} \boldsymbol{\Delta \gamma} \\ \boldsymbol{\Delta \psi} \\ \boldsymbol{\Delta \nu} \\ \boldsymbol{\Delta \lambda^{(1)}} \\ \boldsymbol{\Delta v^{(1)}} \\ \boldsymbol{\Delta \vartheta^{(1)}} \\ \vdots \\ \boldsymbol{\Delta \lambda^{(K)}} \\ \boldsymbol{\Delta v^{(K)}} \\ \boldsymbol{\Delta \vartheta^{(K)}} \end{bmatrix}, \boldsymbol{\Delta}_{xy} \triangleq \begin{bmatrix} \boldsymbol{\Delta \ddot{x}} \\ \boldsymbol{\Delta \check{x}} \\ \boldsymbol{\Delta \ddot{y}}^{(1)} \\ \boldsymbol{\Delta \check{y}}^{(1)} \\ \vdots \\ \boldsymbol{\Delta \ddot{y}}^{(K)} \\ \boldsymbol{\Delta \check{y}}^{(K)} \end{bmatrix}, \boldsymbol{r}_p \triangleq \begin{bmatrix} \boldsymbol{\beta}_\gamma \\ \boldsymbol{\beta}_\psi \\ \boldsymbol{\beta}_s \\ \boldsymbol{\beta}_{\lambda^{(1)}} \\ \boldsymbol{\beta}_{v^{(1)}} \\ \boldsymbol{\beta}_{z^{(1)}} \\ \vdots \\ \boldsymbol{\beta}_{\lambda^{(K)}} \\ \boldsymbol{\beta}_{v^{(K)}} \\ \boldsymbol{\beta}_{z^{(K)}} \end{bmatrix}, \boldsymbol{r}_d \triangleq \begin{bmatrix} \varrho \\ \boldsymbol{\tau} \\ \boldsymbol{\rho} \\ \boldsymbol{\zeta}^{(1)} \\ \boldsymbol{\kappa}^{(1)} \\ \boldsymbol{\eta}^{(1)} \\ \vdots \\ \boldsymbol{\zeta}^{(K)} \\ \boldsymbol{\kappa}^{(K)} \\ \boldsymbol{\eta}^{(K)} \end{bmatrix}, \text{ and } \boldsymbol{r}_{xy} \triangleq \begin{bmatrix} \boldsymbol{\sigma}_{\ddot{x}} \\ \boldsymbol{\sigma}_{\check{x}} \\ \boldsymbol{\sigma}_{\ddot{y}^{(1)}} \\ \boldsymbol{\sigma}_{\check{y}^{(1)}} \\ \vdots \\ \boldsymbol{\sigma}_{\ddot{y}^{(K)}} \\ \boldsymbol{\sigma}_{\check{y}^{(K)}} \end{bmatrix}.$$

Note that the vectors $\boldsymbol{\Delta}_p, \boldsymbol{\Delta}_d, \boldsymbol{\Delta}_{xy}, \boldsymbol{r}_d, \boldsymbol{r}_p$ and $\boldsymbol{r}_{xy}$ are defined so that

$$\begin{array}{rcl} D\boldsymbol{\Delta}_p + H\boldsymbol{\Delta}_d &=& \boldsymbol{r}_p, \\ -G\boldsymbol{\Delta}_{xy} + J^\top\boldsymbol{\Delta}_d &=& \boldsymbol{r}_{xy}, \\ C\boldsymbol{\Delta}_p + J\boldsymbol{\Delta}_{xy} &=& \boldsymbol{r}_d, \end{array} \quad \text{or equivalently,} \begin{bmatrix} D & O & H \\ O & -G & J^\top \\ C & J & O \end{bmatrix} \begin{bmatrix} \Delta_p \\ \Delta_{xy} \\ \Delta_d \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}_p \\ \boldsymbol{r}_{xy} \\ \boldsymbol{r}_d \end{bmatrix}. \tag{11}$$

System (11) is the KKT conditions and can be resolved clearly without producing any additional fill-in. Thus, we omit $\Delta\boldsymbol{s}$, $\Delta\boldsymbol{w}$, $\Delta\boldsymbol{t}$, $\Delta\boldsymbol{z}^{(k)}$, $\Delta\boldsymbol{v}^{(k)}$, $\Delta\boldsymbol{r}^{(k)}$, $\Delta\boldsymbol{\gamma}$, $\Delta\boldsymbol{\psi}$, $\Delta\boldsymbol{\lambda}^{(k)}$, $\Delta\boldsymbol{v}^{(k)}$, for $k = 1, 2, \ldots, K$, and obtain the reduced KKT system given in (12).

$$\begin{bmatrix}
-(G_{\ddot{x}\ddot{x}} - D_{\boldsymbol{w}} + D_{\boldsymbol{t}}) & -G_{\ddot{x}\breve{x}} & -G_{\ddot{x}\ddot{y}^{(1)}} & -G_{\ddot{x}\breve{y}^{(1)}} & \cdots & -G_{\ddot{x}\ddot{y}^{(K)}} & -G_{\ddot{x}\breve{y}^{(K)}} & A_{\ddot{x}}^\top & B_{\ddot{x}}^\top & \cdots & B_{\ddot{x}}^\top \\
-G_{\ddot{x}\breve{x}} & -G_{\breve{x}\breve{x}} & -G_{\breve{x}\ddot{y}^{(1)}} & -G_{\breve{x}\breve{y}^{(1)}} & \cdots & -G_{\breve{x}\ddot{y}^{(K)}} & -G_{\breve{x}\breve{y}^{(K)}} & A_{\breve{x}}^\top & B_{\breve{x}}^\top & \cdots & B_{\breve{x}}^\top \\
-G_{\ddot{x}\ddot{y}^{(1)}} & -G_{\breve{x}\ddot{y}^{(1)}} & -(G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} - D_{\boldsymbol{v}^{(1)}} + D_{\boldsymbol{r}^{(1)}}) & -G_{\ddot{y}^{(1)}\breve{y}^{(1)}} & \cdots & -G_{\ddot{y}^{(1)}\ddot{y}^{(K)}} & -G_{\breve{x}\breve{y}^{(K)}} & & B_{\ddot{y}^{(1)}}^\top & & \\
-G_{\ddot{x}\breve{y}^{(1)}} & -G_{\breve{x}\breve{y}^{(1)}} & -G_{\ddot{y}^{(1)}\breve{y}^{(1)}} & -G_{\breve{y}^{(1)}\breve{y}^{(1)}} & \cdots & -G_{\ddot{y}^{(K)}\breve{y}^{(1)}} & -G_{\breve{y}^{(K)}\ddot{y}^{(1)}} & & B_{\breve{y}^{(1)}}^\top & & \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & \ddots & \\
-G_{\ddot{x}\ddot{y}^{(K)}} & -G_{\breve{x}\ddot{y}^{(K)}} & -G_{\ddot{y}^{(1)}\ddot{y}^{(K)}} & -G_{\breve{y}^{(1)}\ddot{y}^{(K)}} & \cdots & -(G_{\ddot{y}^{(K)}\ddot{y}^{(K)}} - D_{\boldsymbol{v}^{(k)}} + D_{\boldsymbol{r}^{(k)}}) & -G_{\breve{y}^{(K)}\ddot{y}^{(K)}} & & & & B_{\ddot{y}^{(K)}} \\
-G_{\ddot{x}\breve{y}^{(K)}} & -G_{\breve{x}\breve{y}^{(K)}} & -G_{\ddot{y}^{(1)}\breve{y}^{(K)}} & -G_{\breve{y}^{(1)}\breve{y}^{(K)}} & \cdots & -G_{\ddot{y}^{(K)}\breve{y}^{(K)}} & -G_{\breve{y}^{(K)}\breve{y}^{(K)}} & & & & B_{\breve{y}^{(K)}} \\
A_{\ddot{x}} & A_{\breve{x}} & & & & & & \mathrm{Arw}^{-1}(\boldsymbol{\nu})\mathrm{Arw}(\boldsymbol{s}) & & & \\
B_{\ddot{x}} & B_{\breve{x}} & B_{\ddot{y}^{(1)}} & B_{\breve{y}^{(1)}} & & & & & \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(1)})\mathrm{Arw}(\boldsymbol{z}^{(1)}) & & \\
\vdots & \vdots & & & \ddots & & & & & \ddots & \\
B_{\ddot{x}} & B_{\breve{x}} & & & & B_{\ddot{y}^{(K)}} & B_{\breve{y}^{(K)}} & & & & \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(K)})\mathrm{Arw}(\boldsymbol{z}^{(K)})
\end{bmatrix}
\begin{bmatrix} \Delta\ddot{\boldsymbol{x}} \\ \Delta\breve{\boldsymbol{x}} \\ \Delta\ddot{\boldsymbol{y}}^{(1)} \\ \Delta\breve{\boldsymbol{y}}^{(1)} \\ \vdots \\ \Delta\ddot{\boldsymbol{y}}^{(K)} \\ \Delta\breve{\boldsymbol{y}}^{(K)} \\ \Delta\boldsymbol{\nu} \\ \Delta\boldsymbol{\vartheta}^{(1)} \\ \vdots \\ \Delta\boldsymbol{\vartheta}^{(K)} \end{bmatrix}
=
\begin{bmatrix} \boldsymbol{\sigma}_{\ddot{x}} + D_{\boldsymbol{w}}\hat{\boldsymbol{\varrho}} - D_{\boldsymbol{t}}\hat{\boldsymbol{\tau}} \\ \boldsymbol{\sigma}_{\breve{x}} \\ \boldsymbol{\sigma}_{\ddot{y}^{(1)}} + D_{\boldsymbol{v}^{(1)}}\hat{\boldsymbol{\zeta}}^{(1)} - D_{\boldsymbol{r}^{(1)}}\hat{\boldsymbol{\kappa}}^{(1)} \\ \boldsymbol{\sigma}_{\breve{y}^{(1)}} \\ \vdots \\ \boldsymbol{\sigma}_{\ddot{y}^{(K)}} + D_{\boldsymbol{v}}^{(k)}\hat{\boldsymbol{\zeta}}^{(K)} - D_{\boldsymbol{r}^{(k)}}\hat{\boldsymbol{\kappa}}^{(K)} \\ \boldsymbol{\sigma}_{\breve{y}^{(K)}} \\ \boldsymbol{\rho} + \mathrm{Arw}^{-1}(\boldsymbol{\nu})\boldsymbol{\beta}_s \\ \boldsymbol{\eta}^{(1)} + \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(1)})\boldsymbol{\beta}_{\boldsymbol{z}^{(1)}} \\ \vdots \\ \boldsymbol{\eta}^{(K)} + \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(K)})\boldsymbol{\beta}_{\boldsymbol{z}^{(K)}} \end{bmatrix} \tag{12}$$

The matrices $D_{\boldsymbol{w}}, D_{\boldsymbol{t}}, D_{\boldsymbol{v}^{(k)}}, D_{\boldsymbol{r}^{(k)}}, k = 1, 2, \ldots, K$, that appeared in (12) are defined as

$$\begin{array}{lcllcl} D_{\boldsymbol{w}} & \triangleq & \Gamma W^{-1}, & D_{\boldsymbol{t}} & \triangleq & \Psi T^{-1}, \\ D_{\boldsymbol{v}^{(k)}} & \triangleq & V^{(k)^{-1}} \Lambda^{(k)}, & D_{\boldsymbol{r}^{(k)}} & \triangleq & R^{(k)^{-1}} \Upsilon^{(k)}, \end{array}$$

and the vectors $\hat{\boldsymbol{\varrho}}, \hat{\boldsymbol{\tau}}, \hat{\boldsymbol{\zeta}}^{(k)}$ and $\hat{\boldsymbol{\kappa}}^{(k)}, k = 1, 2, \ldots, K$, that appeared in (12) are defined as

$$\begin{array}{lcllcl} \hat{\boldsymbol{\varrho}} & \triangleq & \boldsymbol{\varrho} + D_{\boldsymbol{w}}^{-1} \boldsymbol{\beta}_{\gamma}, & \hat{\boldsymbol{\tau}} & \triangleq & \boldsymbol{\tau} - D_{\boldsymbol{t}}^{-1} \boldsymbol{\beta}_{\psi}, \\ \hat{\boldsymbol{\zeta}}^{(k)} & \triangleq & \boldsymbol{\zeta}^{(k)} + D_{\boldsymbol{v}^{(k)}}^{-1} \boldsymbol{\beta}_{\upsilon^{(k)}}, & \hat{\boldsymbol{\kappa}}^{(k)} & \triangleq & \boldsymbol{\kappa}^{(k)} - D_{\boldsymbol{r}^{(k)}}^{-1} \boldsymbol{\beta}_{\lambda}. \end{array}$$

Once we get the search directions $\Delta\ddot{\boldsymbol{x}}, \Delta\check{\boldsymbol{x}}, \Delta\ddot{\boldsymbol{y}}^{(k)}, \Delta\check{\boldsymbol{y}}^{(k)}, \Delta\boldsymbol{\nu}$, and $\Delta\boldsymbol{\vartheta}^{(k)}$ from (12) by applying block Gauss-Jordan elimination, we can get the step directions $\Delta\boldsymbol{\gamma}, \Delta\boldsymbol{\psi}, \Delta\boldsymbol{\lambda}^{(k)}, \Delta\boldsymbol{\upsilon}^{(k)}, \Delta\boldsymbol{s}, \Delta\boldsymbol{w}, \Delta\boldsymbol{t}, \Delta\boldsymbol{z}^{(k)}, \Delta\boldsymbol{v}^{(k)}$, and $\Delta\boldsymbol{r}^{(k)}$, for $k = 1, 2, \ldots, K$, by using the following formulas:

$$\begin{array}{lcllcl} \boldsymbol{\Delta s} & = & \mathrm{Arw}^{-1}(\boldsymbol{\nu})\big(\boldsymbol{\beta}_s - \boldsymbol{s} \circ \boldsymbol{\Delta\nu}\big), & \boldsymbol{\Delta z}^{(k)} & = & \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(k)})\left(\boldsymbol{\beta}_z^{(k)} - \boldsymbol{z}^{(k)} \circ \Delta\boldsymbol{\vartheta}^{(k)}\right), \\ \boldsymbol{\Delta w} & = & \Delta\ddot{\boldsymbol{x}} - \boldsymbol{\varrho}, & \boldsymbol{\Delta v}^{(k)} & = & \boldsymbol{\Delta}\ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{\zeta}^{(k)}, \\ \boldsymbol{\Delta t} & = & \boldsymbol{\tau} - \boldsymbol{\Delta}\ddot{\boldsymbol{x}}, & \boldsymbol{\Delta r}^{(k)} & = & \boldsymbol{\kappa}^{(k)} - \boldsymbol{\Delta}\ddot{\boldsymbol{y}}^{(k)}, \\ \boldsymbol{\Delta\gamma} & = & D_{\boldsymbol{w}}\big(\hat{\boldsymbol{\varrho}} - \boldsymbol{\Delta}\ddot{\boldsymbol{x}}\big), & \boldsymbol{\Delta\lambda}^{(k)} & = & D_{\boldsymbol{v}^{(k)}}\big(\hat{\boldsymbol{\zeta}}_{\lambda^{(k)}} - \boldsymbol{\Delta}\ddot{\boldsymbol{y}}^{(k)}\big), \\ \boldsymbol{\Delta\psi} & = & D_{\boldsymbol{t}}\big(\boldsymbol{\tau} + \boldsymbol{\Delta}\ddot{\boldsymbol{x}}\big), & \boldsymbol{\Delta r}^{(k)} & = & D_{\boldsymbol{r}^{(k)}}\big(\hat{\boldsymbol{\kappa}}^{(k)} + \boldsymbol{\Delta}\ddot{\boldsymbol{y}}^{(k)}\big), \end{array}$$

The algorithm is initialized with initial vectors $\boldsymbol{x}^{\langle 0\rangle}, \boldsymbol{s}^{\langle 0\rangle}, \boldsymbol{y}^{(k)\langle 0\rangle}, \boldsymbol{w}^{\langle 0\rangle}, \boldsymbol{t}^{\langle 0\rangle}, \boldsymbol{v}^{(k)\langle 0\rangle}$, and $\boldsymbol{r}^{(k)\langle 0\rangle}$, for $k = 1, 2, \ldots, K$, and the solution is approached iteratively through a sequence of points obtained from the reduced KKT system (12) according to:

$$\begin{array}{lcllcl} \ddot{\boldsymbol{x}}^{\langle j+1\rangle} & = & \ddot{\boldsymbol{x}}^{\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\ddot{\boldsymbol{x}}^{\langle j\rangle}, & \check{\boldsymbol{x}}^{\langle j+1\rangle} & = & \check{\boldsymbol{x}}^{\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\check{\boldsymbol{x}}^{\langle j\rangle}, \\ \ddot{\boldsymbol{y}}^{(k)\langle j+1\rangle} & = & \ddot{\boldsymbol{y}}^{(k)\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\ddot{\boldsymbol{y}}^{(k)\langle j\rangle}, & \check{\boldsymbol{y}}^{(k)\langle j+1\rangle} & = & \check{\boldsymbol{y}}^{(k)\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\check{\boldsymbol{y}}^{(k)\langle j\rangle}, \\ \boldsymbol{\nu}^{\langle j+1\rangle} & = & \boldsymbol{\nu}^{\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\nu}^{\langle j\rangle}, & \boldsymbol{\vartheta}^{(k)\langle j+1\rangle} & = & \boldsymbol{\vartheta}^{(k)\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\vartheta}^{(k)\langle j\rangle}, \\ \boldsymbol{\gamma}^{\langle j+1\rangle} & = & \boldsymbol{\gamma}^{\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\gamma}^{\langle j\rangle}, & \boldsymbol{\lambda}^{(k)\langle j+1\rangle} & = & \boldsymbol{\lambda}^{(k)\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\lambda}^{(k)\langle j\rangle}, \\ \boldsymbol{\psi}^{\langle j+1\rangle} & = & \boldsymbol{\psi}^{\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\psi}^{\langle j\rangle}, & \boldsymbol{\upsilon}^{(k)\langle j+1\rangle} & = & \boldsymbol{\upsilon}^{(k)\langle j\rangle} + \alpha_d^{\langle j\rangle} \Delta\boldsymbol{\upsilon}^{(k)\langle j\rangle}, \\ \boldsymbol{s}^{\langle j+1\rangle} & = & \boldsymbol{s}^{\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{s}^{\langle j\rangle}, & \boldsymbol{w}^{\langle j+1\rangle} & = & \boldsymbol{w}^{\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{w}^{\langle j\rangle}, \\ \boldsymbol{t}^{\langle j+1\rangle} & = & \boldsymbol{t}^{\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{t}^{\langle j\rangle}, & \boldsymbol{z}^{(k)\langle j+1\rangle} & = & \boldsymbol{z}^{(k)\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{z}^{(k)\langle j\rangle}, \\ \boldsymbol{v}^{(k)\langle j+1\rangle} & = & \boldsymbol{v}^{(k)\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{v}^{(k)\langle j\rangle}, & \boldsymbol{r}^{(k)\langle j+1\rangle} & = & \boldsymbol{r}^{(k)\langle j\rangle} + \alpha_p^{\langle j\rangle} \Delta\boldsymbol{r}^{(k)\langle j\rangle}, \end{array}$$

where the superscripts indicate the number of iterations. Here $0 < \alpha_p^{\langle j\rangle} \leq 1$ is the primal steplength selected to assure that $\boldsymbol{s}^{\langle j+1\rangle} \succ_r \boldsymbol{0}, \boldsymbol{z}^{(k)\langle j+1\rangle} \succ_s \boldsymbol{0}$, and that $\boldsymbol{w}^{\langle j+1\rangle}, \boldsymbol{t}^{\langle j+1\rangle}, \boldsymbol{v}^{(k)\langle j+1\rangle}, \boldsymbol{r}^{(k)\langle j+1\rangle} > \boldsymbol{0}$. Also, $0 < \alpha_d^{\langle j\rangle} \leq 1$ is the dual steplength selected to assure that $\boldsymbol{\nu}^{\langle j+1\rangle} \succ_r \boldsymbol{0}, \boldsymbol{\vartheta}^{(k)\langle j+1\rangle} \succ_s \boldsymbol{0}$, and that $\boldsymbol{\gamma}^{\langle j+1\rangle}, \boldsymbol{\psi}^{\langle j+1\rangle}, \boldsymbol{\lambda}^{(k)\langle j+1\rangle}, \boldsymbol{\upsilon}^{(k)\langle j+1\rangle} > \boldsymbol{0}$. A technique for controlling the steplength is discussed in more details in [16] (see also [4, 5]). Additionally, at each iteration, the value of the barrier parameter μ may be updated as a function of $(\boldsymbol{s} \circ \boldsymbol{\nu}, W^{\langle j+1\rangle}\boldsymbol{\gamma}^{\langle j+1\rangle}, T^{\langle j+1\rangle}\boldsymbol{\psi}^{\langle j+1\rangle})$ and $(\boldsymbol{z}^{(k)\langle j\rangle} \circ \boldsymbol{\vartheta}^{(k)}, V^{(k)\langle j+1\rangle}\boldsymbol{\lambda}^{(k)\langle j+1\rangle}, R^{(k)l+1}\boldsymbol{\upsilon}^{(k)\langle j+1\rangle})$.

When primal infeasibility, dual infeasibility, and average complementarity are all less than a certain tolerance level, the algorithm determines that an optimal solution has been obtained. For Problem (7), the primal infeasibility, dual infeasibility, and average complementary are as follow:

$$\begin{array}{lcl} \text{Primal infeasibility} & \triangleq & \max\limits_{1 \leq k \leq K} \left\{ \|\boldsymbol{\rho}\|_\infty, \|\boldsymbol{\varrho}\|_\infty, \|\boldsymbol{\tau}\|_\infty, \|\boldsymbol{\eta}^{(k)}\|_\infty, \|\boldsymbol{\zeta}^{(k)}\|_\infty, \|\boldsymbol{\kappa}^{(k)}\|_\infty \right\}, \\ \text{Dual infeasibility} & \triangleq & \max\limits_{1 \leq k \leq K} \left\{ \|\boldsymbol{\sigma}_{\ddot{x}}\|_\infty, \|\boldsymbol{\sigma}_{\check{x}}\|_\infty, \|\boldsymbol{\sigma}_{\ddot{y}^{(k)}}\|_\infty, \|\boldsymbol{\sigma}_{\check{y}^{(k)}}\|_\infty \right\}, \\ \text{Average complementary} & \triangleq & \dfrac{\boldsymbol{s}^\mathsf{T}\boldsymbol{\nu} + \boldsymbol{w}^\mathsf{T}\boldsymbol{\gamma} + \boldsymbol{t}^\mathsf{T}\boldsymbol{\psi}}{\ell + 2p} + \dfrac{\sum_{k=1}^K \boldsymbol{z}^{(k)\mathsf{T}}\boldsymbol{\vartheta}^{(k)}}{Kd} + \dfrac{\sum_{k=1}^K \boldsymbol{v}^{(k)\mathsf{T}}\boldsymbol{\lambda}^{(k)}}{Kq} + \dfrac{\sum_{k=1}^K \boldsymbol{r}^{(k)\mathsf{T}}\boldsymbol{\upsilon}^{(k)}}{Kq}. \end{array}$$

In the next section, we present different alternative formulations of the second-order cone constraints. These formulations yield convex and smooth problems.

4 Nondifferentiability at optimality

The form of the second-order cone constraints given in Problem (6) does fit the nonlinear paradigm. In this paper, we are using an interior point method where the constraints need to be twice continuously differentiable but the Euclidean norm fails that criterion. The inability of $\boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)})$ to be differentiated may pose a problem for any solver attempting to calculate the derivatives of constraint functions. In practice, if $\bar{\boldsymbol{h}}(\boldsymbol{x}) = \boldsymbol{0}$ and $\bar{\boldsymbol{g}}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) = \boldsymbol{0}$ for $k = 1, 2, \ldots, K$, then the second-order cone constraints will be nonsmooth.

The nonsmoothness that characterizes the second-order cone constraints can cause problems in two cases: There exists an intermediate solution at a nondifferentiable point, or a nondifferentiable point is where an optimal solution exists. It is easy to handle the first case in an interior-point algorithm by randomly generating the initial solution. Hence, the possibility of encountering a problematic intermediate solution is eliminated, but avoiding nondifferentiability in optimal solutions is more difficult. We discuss different ways to modify the second-order cone constraints in such a way that the problem can be written as a smooth and convex one.

Consider any constraint of the form

$$\|\bar{\boldsymbol{x}}\| \leq x_0, \tag{13}$$

where $\boldsymbol{x} \triangleq (x_0; \bar{\boldsymbol{x}}) \in \mathcal{E}^n$. Benson and Venderbei [15] (see also [24]) describe different alternate formulations of this second-order cone constraint. Now, we investigate many different formulations for second-order cone constraints to produce a convex and smooth problem. The first alternative is the perturbation procedure, where the resulting second-order cone constraints are smooth, but they may not be equivalent to the original constraints. The other are ways to rewrite the problem so that the new formulation is equivalent to the original problem. In what follows, we first present the mathematical setup of these treatments.

4.1 Smoothing by perturbation

Smoothing by perturbation is one way to ovoid the nondifferentiability problem by introducing a positive constant into the Euclidean norm. In other words, we modify the basic constraint in (13) with

$$\sqrt{\epsilon^2 + \|\bar{\boldsymbol{x}}\|^2} \leq x_0,$$

where ϵ is a very small constant (typically around 10^{-6}). The perturbation provides a second-order cone constraint and the resulting problem is smooth and convex. Even if ϵ might be small enough for the perturbation to be absorbed by the algorithm's numerical precision, this reformulation is not exactly the same as the original constraint. For this reason, we want to make the perturbation as small as possible. This may necessitate experimenting with various choices until we find the smallest value for which the algorithm can find the optimal solution to the problem. But it may take a long time to solve the same problem more than once, especially if the problem is large.

Nevertheless, resolving the same problem several times could be time-consuming. As a result, we can remove the constant ϵ from the second-order cone constraint and replace it with a variable, say v. Thus, the second-order cone constraint can be replaced with

$$\sqrt{v^2 + \|\bar{\boldsymbol{x}}\|^2} \leq x_0,$$

where $v > 0$. The restriction on positivity of v lets us solve the problem without losing any generality, and the strict inequality does not make much of a difference for the interior-point method.

4.2 Smoothing by reformulation

In spite of the fact that the perturbation technique works quite well in reality, one could say that it is advantageous to have a smooth problem as the original constraint. Obviously, it would also be desirable to preserve the favorable property of convexity in the problem. We present three different reformulation alternatives.

(i) *Smoothing by squaring*: The goal of this reformulation is to replace (13) with an equivalent constraint. The simplest reformulation is

$$\begin{array}{rcl} \|\bar{\boldsymbol{x}}\|^2 - x_0^2 & \leq & 0, \\ x_0 & \geq & 0. \end{array}$$

The function that is incorporated into the nonlinear program is

$$\gamma(\boldsymbol{x}) \triangleq \|\bar{\boldsymbol{x}}\|^2 - x_0^2, \tag{14}$$

where $\gamma(\boldsymbol{x})$ represents a smooth function everywhere. However, it is not convex. To see this note that its Hessian matrix is not positive definite:

$$\nabla\gamma(\boldsymbol{x}) = 2\begin{bmatrix} \bar{\boldsymbol{x}} \\ -x_0 \end{bmatrix}, \quad \nabla^2\gamma(\boldsymbol{x}) = 2\begin{bmatrix} I & 0 \\ 0 & -1 \end{bmatrix}.$$

As shown in [30], although the feasible region is convex, representing it as the intersection of nonconvex inequalities can result in a slow convergence to dual feasibility. Therefore, one would assume that this transform will not operate well.

(ii) *Convexification by exponentiation*: Using $\gamma(\cdot)$ defined in (14), we were able to obtain a second-order cone constraint in a smooth manner. The nonconvexity of γ can be solved by composing it with a smooth convex function that maps the negative halfline back onto itself. The exponential function is an example of a convex smooth function. So, let us use

$$\Phi(\boldsymbol{x}) \triangleq \mathrm{e}^{\gamma(\boldsymbol{x})} - 1.$$

To verify the convexity of $\Phi(\boldsymbol{x})$, note that

$$\begin{array}{rcl} \nabla\Phi(\boldsymbol{x}) & = & \mathrm{e}^{\gamma(\boldsymbol{x})/2}\begin{bmatrix} \bar{\boldsymbol{x}} \\ -x_0 \end{bmatrix}, \\ \nabla^2\Phi(\boldsymbol{x}) & = & \mathrm{e}^{\gamma(\boldsymbol{x})/2}\begin{bmatrix} I + \bar{\boldsymbol{x}}\boldsymbol{x}^{\mathsf{T}} & -x_0\bar{\boldsymbol{x}} \\ -x_0\bar{\boldsymbol{x}}^{\mathsf{T}} & 1 + x_0^2 \end{bmatrix} = \mathrm{e}^{\gamma(\boldsymbol{x})/2}\left(I + \begin{bmatrix} \bar{\boldsymbol{x}} \\ -x_0 \end{bmatrix}\begin{bmatrix} \bar{\boldsymbol{x}}^{\mathsf{T}} & -x_0 \end{bmatrix}\right). \end{array}$$

From the last expression of Hessian matrix, we find that $\nabla^2\Phi$ is positive definite.

Even though the exponential function provides a smooth and convex reformulation, it does not behave well in practice because of scaling troubles. For example, when $\|\bar{\boldsymbol{x}}\|$ is with order 10, $\mathrm{e}^{\|\bar{\boldsymbol{x}}\|^2}$ is very huge. So, this treatment is rarely effective in practice.

(iii) *Convexification by ratios*: Another strategy for avoiding the problem of nonconvexity during smoothing is to use

$$\delta(\boldsymbol{x}) \triangleq \frac{\|\bar{\boldsymbol{x}}\|}{x_0} - x_0.$$

To check a convexity of $\delta(\boldsymbol{x})$, note that

$$\begin{array}{rcl} \nabla\delta(\boldsymbol{x}) & = & \begin{bmatrix} 2x_0^{-1}\bar{\boldsymbol{x}} \\ -x_0^{-2}\|\bar{\boldsymbol{x}}\|^2 - 1 \end{bmatrix}, \\ \nabla^2\delta(\boldsymbol{x}) & = & 2x_0^{-1}\begin{bmatrix} I & -x_0^{-1}\bar{\boldsymbol{x}} \\ -\bar{\boldsymbol{x}}^{\mathsf{T}}x_0^{-1} & x_0^{-2}\|\bar{\boldsymbol{x}}\|^2 \end{bmatrix} = 2x_0^{-1}\begin{bmatrix} I \\ -x_0^{-1}\bar{\boldsymbol{x}}^{\mathsf{T}} \end{bmatrix}\begin{bmatrix} I & -x_0^{-1}\bar{\boldsymbol{x}} \end{bmatrix}. \end{array}$$

The Hessian here is a positive definite matrix. When an interior-point algorithm is used, the strict inequality constraint on x_0 does not apply. In fact, for many second-order cone programming problems, the affine formulation of x_0 has only a constant term, and the only nonsmooth region of δ is at $x_0 = 0$.

5 Inner-level algorithmic improvements

In [10], Benson introduces the exact primal-dual penalty method to mixed-integer nonlinear programming problems, which incorporate the works of Benson and Shanno [12] for linear programming and of that Benson and Shanno [13] for nonlinear programming. This method improves the efficiency of the interior-point algorithm by permitting them to identify infeasible problems and warmstart. The latter is the utilization of information obtained during the solution of a problem to solve subsequent problems that are closely related.

5.1 The penalty technique within the interior-point method framework

In this paper, we extend the primal-dual penalty method for the two-stage SMINLSOCP problems. In these problems, warmstarting will directly correspond to the initial solution setting (which has primal, dual, and slack variables) of an SMINLSOCP problem of the form (7) to the optimal solution within the branch-and-bound approach. Since there are complementary conditions at the optimal solution, some of the dual and slack variables vanish. In this case, the initialization variables are arbitrary and may adversely affect the algorithm's performance at the current node. Note that just reinitializing some variables may result in negative step directions for slack and dual variables. Because the interior-point algorithm shortens the steplengths α_p and α_d to keep such variables strictly positive, the algorithm may still become stuck.

Traditional penalty methods, which only offer primal relaxations, may still get stuck when used with a primal-dual interior-point method where the steplength α_d is based on the dual iterates. However, they also have other desired properties, such as the ability to detect infeasibility and regularizations that automatically meet constraints. The primal-dual penalty method introduced in [12] for linear programming and in [13] for nonlinear programming is a remedy that has been shown to work for warmstarts. This approach relaxes the second-order cone and the nonnegative constraints on the slack and dual variables and provides regularization for the matrix in the reduced KKT system (12). Thus, the optimal solution of one problem may be utilized to provide a warmstart for another, the regularization assures that the variables that need to move actually do make progress, and the algorithm does not become stuck due to the constraints.

The corresponding penalized problem to (7) has the form

$$
\begin{array}{lll}
\min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) + c_{\boldsymbol{s}}^{\mathsf{T}}\xi_{\boldsymbol{s}} + \boldsymbol{c}_{\boldsymbol{w}}^{\mathsf{T}}\xi_{\boldsymbol{w}} + \boldsymbol{c}_{\boldsymbol{t}}^{\mathsf{T}}\xi_{\boldsymbol{t}} + \sum_{k=1}^{K}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}}^{\mathsf{T}}\xi_{\boldsymbol{z}^{(k)}} + \boldsymbol{c}_{\boldsymbol{v}^{(k)}}^{\mathsf{T}}\xi_{\boldsymbol{v}^{(k)}} + \boldsymbol{c}_{\boldsymbol{r}^{(k)}}^{\mathsf{T}}\xi_{\boldsymbol{r}^{(k)}}) \\
\text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} = \boldsymbol{0}, \\
& \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} = \boldsymbol{0}, & k = 1, 2, \ldots, K, \\
& \ddot{\boldsymbol{x}} - \boldsymbol{w} = \boldsymbol{l}, \\
& \ddot{\boldsymbol{x}} + \boldsymbol{t} = \boldsymbol{u}, \\
& \ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} = \tilde{\boldsymbol{l}}^{(k)}, & k = 1, 2, \ldots, K, \\
& \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} = \tilde{\boldsymbol{u}}^{(k)}, & k = 1, 2, \ldots, K, \\
& \boldsymbol{s} + \xi_{\boldsymbol{s}} \succeq_r \boldsymbol{0}, \\
& \boldsymbol{b}_{\boldsymbol{v}} - \boldsymbol{s} \succeq_r \boldsymbol{0}, \\
& \boldsymbol{z}^{(k)} + \xi_{\boldsymbol{z}^{(k)}} \succeq_s \boldsymbol{0}, & k = 1, 2, \ldots, K, \\
& \boldsymbol{b}_{\vartheta^{(k)}} - \boldsymbol{z}^{(k)} \succeq_s \boldsymbol{0}, & k = 1, 2, \ldots, K, \\
& -\xi_{\boldsymbol{w}} \leq \boldsymbol{w} \leq \boldsymbol{b}_{\gamma}, \\
& -\xi_{\boldsymbol{t}} \leq \boldsymbol{t} \leq \boldsymbol{b}_{\psi}, \\
& -\xi_{\boldsymbol{v}^{(k)}} \leq \boldsymbol{v}^{(k)} \leq \boldsymbol{b}_{\lambda^{(k)}}, & k = 1, 2, \ldots, K \\
& -\xi_{\boldsymbol{r}^{(k)}} \leq \boldsymbol{r}^{(k)} \leq \boldsymbol{b}_{\boldsymbol{v}^{(k)}}, & k = 1, 2 \ldots, K, \\
& \xi_{\boldsymbol{w}}, \xi_{\boldsymbol{t}}, \xi_{\boldsymbol{v}^{(k)}}, \xi_{\boldsymbol{r}^{(k)}} \geq \boldsymbol{0}, & k = 1, 2 \ldots, K, \\
& \xi_{\boldsymbol{s}} \succeq_r \boldsymbol{0},\ \xi_{\boldsymbol{z}^{(k)}} \succeq_s \boldsymbol{0}, & k = 1, 2 \ldots, K,
\end{array}
\tag{15}
$$

where, for $k = 1, 2, \ldots, K$, $\xi_{\boldsymbol{s}}, \xi_{\boldsymbol{w}}, \xi_{\boldsymbol{t}}, \xi_{\boldsymbol{v}^{(k)}}, \xi_{\boldsymbol{r}^{(k)}}$ are the primal relaxation variables corresponding to the polyhedral cones, $\xi_{\boldsymbol{s}}, \xi_{\boldsymbol{z}^{(k)}}$ are the primal relaxation variables corresponding to the second-order cones, $\boldsymbol{c}_{\boldsymbol{s}}, \boldsymbol{c}_{\boldsymbol{w}}, \boldsymbol{c}_{\boldsymbol{t}}$ $\boldsymbol{c}_{\boldsymbol{z}^{(k)}}, \boldsymbol{c}_{\boldsymbol{v}^{(k)}}, \boldsymbol{c}_{\boldsymbol{r}^{(k)}}$ are the primal penalty parameters, and $\boldsymbol{b}_{\boldsymbol{v}}, \boldsymbol{b}_{\gamma}, \boldsymbol{b}_{\psi}$ $\boldsymbol{b}_{\vartheta^{(k)}}, \boldsymbol{b}_{\lambda^{(k)}}, \boldsymbol{b}_{\boldsymbol{v}^{(k)}}$ are the primal penalty parameters, respectively.

In this new form of the primal-dual penalty problem, all slack variables rather than the constraint functions and the bounds themselves are relaxed, as well as all upper bounds are added to the slack variables.

Following the development of Section 2.2, we propose an algorithm to solve Problem (15). The logarithmic barrier problem associated with (15) is

$$
\begin{array}{ll}
\min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) + \boldsymbol{c}_s^T \boldsymbol{\xi}_s + \boldsymbol{c}_w^T \boldsymbol{\xi}_w + \boldsymbol{c}_t^T \boldsymbol{\xi}_t + \sum_{k=1}^K (\boldsymbol{c}_{z^{(k)}}^T \boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{c}_{v^{(k)}}^T \boldsymbol{\xi}_{v^{(k)}} + \boldsymbol{c}_{r^{(k)}}^T \boldsymbol{\xi}_{r^{(k)}}) \\
& -\mu \ln \det(\boldsymbol{\xi}_s) - \mu \ln \det(\boldsymbol{\xi}_s + \boldsymbol{s}) - \mu \ln \det(\boldsymbol{b}_\nu - \boldsymbol{s}) - \mu \ln(\boldsymbol{\xi}_w) - \mu \ln(\boldsymbol{\xi}_w + \boldsymbol{w}) - \mu \ln (\boldsymbol{b}_\gamma - \boldsymbol{w}) - \mu \ln (\boldsymbol{\xi}_t) \\
& -\mu \ln (\boldsymbol{\xi}_t + \boldsymbol{t}) - \mu \ln(\boldsymbol{b}_\psi - \boldsymbol{t}) - \mu \sum_{k=1}^K (\ln \det(\boldsymbol{\xi}_{z^{(k)}}) + \ln \det(\boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{z}^{(k)}) + \ln \det(\boldsymbol{b}_{\vartheta^{(k)}} - \boldsymbol{z}^{(k)})) \\
& -\mu \sum_{k=1}^K (\ln (\boldsymbol{\xi}_{v^{(k)}}) + \ln(\boldsymbol{\xi}_{v^{(k)}} + \boldsymbol{v}^{(k)}) + \ln (\boldsymbol{b}_{\lambda^{(k)}} - \boldsymbol{v}^{(k)})) - \mu \sum_{k=1}^K (\ln (\boldsymbol{\xi}_{r^{(k)}}) + \ln(\boldsymbol{\xi}_{r^{(k)}} + \boldsymbol{r}^{(k)}) - \mu \ln (\boldsymbol{b}_{v^{(k)}} - \boldsymbol{r}^{(k)})) \\
\text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} = \boldsymbol{0}, \\
& \boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} = \boldsymbol{0}, \quad k = 1, \ldots, K, \\
& \ddot{\boldsymbol{x}} - \boldsymbol{w} = \boldsymbol{l}, \\
& \ddot{\boldsymbol{x}} + \boldsymbol{t} = \boldsymbol{u}, \\
& \ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} = \tilde{\boldsymbol{l}}^{(k)}, \qquad k = 1, \ldots, K, \\
& \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} = \tilde{\boldsymbol{u}}^{(k)}, \qquad k = 1, \ldots, K,
\end{array}
\tag{16}
$$

where μ is the barrier parameter, $\boldsymbol{\nu}$, $\boldsymbol{\gamma}$, $\boldsymbol{\psi}$, $\boldsymbol{\vartheta}^{(k)}$, $\boldsymbol{\lambda}^{(k)}$, $\boldsymbol{v}^{(k)}$, for $k = 1, 2, \ldots, K$, are dual variables associated with the relevant constraints

Following our notations in Subsection 2.1, the matrices $\Xi_s, \Xi_w, \Xi_t, \Xi_{z^{(k)}}, \Xi_{v^{(k)}}, \Xi_{r^{(k)}}, B_\nu, B_\gamma, B_\psi, B_{\vartheta^{(k)}}, B_{\lambda^{(k)}}, B_{v^{(k)}}$, $\Phi_\gamma, \Phi_\psi, \Phi_{\lambda^{(k)}}$ and $\Phi_{v^{(k)}}$ stand for the diagonal matrices whose diagonal entries are the components of the vectors $\boldsymbol{\xi}_s, \boldsymbol{\xi}_w, \boldsymbol{\xi}_t, \boldsymbol{\xi}_{z^{(k)}}, \boldsymbol{\xi}_{v^{(k)}}, \boldsymbol{\xi}_{r^{(k)}}, \boldsymbol{b}_\nu, \boldsymbol{b}_\gamma, \boldsymbol{b}_\psi, \boldsymbol{b}_{\vartheta^{(k)}}, \boldsymbol{b}_{\lambda^{(k)}}, \boldsymbol{b}_{v^{(k)}}, \boldsymbol{\phi}_\gamma, \boldsymbol{\phi}_\psi, \boldsymbol{\phi}_{\lambda^{(k)}}$ and $\boldsymbol{\phi}_{v^{(k)}}$, respectively, $k = 1, 2, \ldots, K$, where

$$
\begin{array}{llll}
\boldsymbol{\phi}_\nu \triangleq 2\mu(\boldsymbol{\xi}_s + \boldsymbol{s})^{-1}, & & \boldsymbol{\phi}_{\vartheta^{(k)}} \triangleq 2\mu(\boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{z}^{(k)})^{-1}, \\
\boldsymbol{\phi}_\gamma \triangleq \mu(\boldsymbol{b}_\gamma - \boldsymbol{w})^{-1}, & & \boldsymbol{\phi}_{\lambda^{(k)}} \triangleq \mu(\boldsymbol{b}_{\lambda^{(k)}} - \boldsymbol{v}^{(k)})^{-1}, \\
\boldsymbol{\phi}_\psi \triangleq \mu(\boldsymbol{b}_\psi - \boldsymbol{t})^{-1}, & & \boldsymbol{\phi}_{v^{(k)}} \triangleq \mu(b_{v^{(k)}} - \boldsymbol{r}^{(k)})^{-1}.
\end{array}
$$

The first-order necessary conditions for the lagrangian function of (16) are written as:

$$
\begin{array}{rcll}
\nabla_{\ddot{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\ddot{x}}(\boldsymbol{x})^T \boldsymbol{\nu} - \sum_{k=1}^K B_{\ddot{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^T \boldsymbol{\vartheta}^{(k)} - \boldsymbol{\gamma} - \boldsymbol{\psi} & = & \boldsymbol{0}, & \\
\nabla_{\check{x}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - A_{\check{x}}(\boldsymbol{x})^T \boldsymbol{\nu} - \sum_{k=1}^K B_{\check{x}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^T \boldsymbol{\vartheta}^{(k)} & = & \boldsymbol{0}, & \\
\nabla_{\ddot{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\ddot{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^T \boldsymbol{\vartheta}^{(k)} - \boldsymbol{\lambda}^{(k)} - \boldsymbol{v}^{(k)} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\nabla_{\check{y}^{(k)}} f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) - B_{\check{y}^{(k)}}(\boldsymbol{x}, \boldsymbol{y}^{(k)})^T \boldsymbol{\nu}^{(k)} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
(\boldsymbol{\phi}_\nu + \boldsymbol{\nu}) \circ (\boldsymbol{\xi}_s + \boldsymbol{s}) - 2\mu \boldsymbol{e} & = & \boldsymbol{0}, & \\
\boldsymbol{\xi}_s \circ (\boldsymbol{c}_s - \boldsymbol{\phi}_\nu) - 2\mu \boldsymbol{e} & = & \boldsymbol{0}, & \\
(\boldsymbol{\phi}_{\vartheta^{(k)}} + \boldsymbol{\vartheta}^{(k)}) \circ (\boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{z}^{(k)}) - 2\mu \boldsymbol{e} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\boldsymbol{\xi}_{z^{(k)}} \circ (\boldsymbol{c}_{z^{(k)}} - \boldsymbol{\phi}_{\vartheta^{(k)}}) - \mu \boldsymbol{e} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
(W + \Xi_w)(\boldsymbol{\gamma} + \boldsymbol{\phi}_\gamma) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
(T + \Xi_t)(\boldsymbol{\psi} + \boldsymbol{\phi}_\psi) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
(V^{(k)} + \Xi_{v^{(k)}})(\boldsymbol{\lambda}^{(k)} + \boldsymbol{\phi}_{\lambda^{(k)}}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & k = 1, \ldots, K \\
(R^{(k)} + \Xi_{r^{(k)}})(\boldsymbol{v}^{(k)} + \boldsymbol{\phi}_{v^{(k)}}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\Xi_w(\boldsymbol{c}_w - \boldsymbol{\gamma} - \boldsymbol{\phi}_\gamma) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
\Xi_t(\boldsymbol{c}_t - \boldsymbol{\psi} - \boldsymbol{\phi}_\psi) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
\Xi_{v^{(k)}}(\boldsymbol{c}_{v^{(k)}} - \boldsymbol{\lambda}^{(k)} - \boldsymbol{\phi}_{\lambda^{(k)}}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\Xi_{r^{(k)}}(\boldsymbol{c}_{r^{(k)}} - \boldsymbol{v}^{(k)} - \boldsymbol{\phi}_{v^{(k)}}) - \mu \boldsymbol{1} & = & \boldsymbol{1}, & k = 1, \ldots, K, \\
\boldsymbol{\phi}_\nu \circ (\boldsymbol{\xi}_s + \boldsymbol{s}) - 2\mu \boldsymbol{e} & = & \boldsymbol{0}, & \\
\boldsymbol{\phi}_{\vartheta^{(k)}} \circ (\boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{z}^{(k)}) - 2\mu \boldsymbol{e} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\Phi_\gamma(\boldsymbol{b}_\gamma - \boldsymbol{w}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
\Phi_\psi(\boldsymbol{b}_\psi - \boldsymbol{t}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & \\
\Phi_{\lambda^{(k)}}(\boldsymbol{b}_{\lambda^{(k)}} - \boldsymbol{v}^{(k)}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\Phi_{v^{(k)}}(\boldsymbol{b}_{v^{(k)}} - \boldsymbol{r}^{(k)}) - \mu \boldsymbol{1} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} & = & \boldsymbol{0}, & \\
\boldsymbol{g}^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\ddot{\boldsymbol{x}} + \boldsymbol{t} - \boldsymbol{u} & = & \boldsymbol{0}, & \\
\ddot{\boldsymbol{x}} - \boldsymbol{w} - \boldsymbol{l} & = & \boldsymbol{0}, & \\
\ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} - \tilde{\boldsymbol{u}}^{(k)} & = & \boldsymbol{0}, & k = 1, \ldots, K, \\
\ddot{\boldsymbol{y}}^{(k)} - \boldsymbol{v}^{(k)} - \tilde{\boldsymbol{l}}^{(k)} & = & \boldsymbol{0}, & k = 1, \ldots, K.
\end{array}
\tag{17}
$$

By applying Newton's method and eliminating the search directions $\Delta \boldsymbol{w}$, $\Delta \boldsymbol{t}$, $\Delta \boldsymbol{\gamma}$, $\Delta \boldsymbol{\psi}$, $\Delta \boldsymbol{\xi}_s$, $\Delta \boldsymbol{\xi}_w$, $\Delta \boldsymbol{\xi}_t$, $\Delta \boldsymbol{\phi}_\nu$, $\Delta \boldsymbol{\phi}_\gamma$, $\Delta \boldsymbol{\phi}_\psi$, $\Delta \boldsymbol{v}^{(k)}$, $\Delta \boldsymbol{r}^{(k)}$, $\Delta \boldsymbol{\lambda}^{(k)}$, $\Delta \boldsymbol{v}^{(k)}$, $\Delta \boldsymbol{\xi}_{z^{(k)}}$, $\Delta \boldsymbol{\xi}_{v^{(k)}}$, $\Delta \boldsymbol{\xi}_{r^{(k)}}$, $\Delta \boldsymbol{\phi}_{\vartheta^{(k)}}$, $\Delta \boldsymbol{\phi}_{\lambda^{(k)}}$, and $\Delta \boldsymbol{\phi}_{v^{(k)}}$ for $k = 1, 2, \ldots, K$, we get the following reduced KKT system which arises in the solution of the penalty problem (15):

$$
\begin{bmatrix}
-(G_{\ddot{x}\ddot{x}} - D_{\boldsymbol{w}} + D_{\boldsymbol{t}}) & -G_{\ddot{x}\dot{x}} & & -G_{\ddot{x}\ddot{y}^{(1)}} & & & -G_{\ddot{x}\ddot{y}^{(1)}} & \cdots & -G_{\ddot{x}\ddot{y}^{(K)}} & & A_{\ddot{x}}^{\mathsf{T}} & B_{\ddot{x}}^{\mathsf{T}} & & \cdots & B_{\ddot{x}}^{\mathsf{T}} \\
-G_{\dot{x}\ddot{x}} & -G_{\dot{x}\dot{x}} & & -G_{\dot{x}\ddot{y}^{(1)}} & & & -G_{\dot{x}\ddot{y}^{(1)}} & \cdots & -G_{\dot{x}\ddot{y}^{(K)}} & & A_{\dot{x}}^{\mathsf{T}} & B_{\dot{x}}^{\mathsf{T}} & & \cdots & B_{\dot{x}}^{\mathsf{T}} \\
-G_{\ddot{x}\ddot{y}^{(1)}} & -G_{\dot{x}\ddot{y}^{(1)}} & -(G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} - D_{\boldsymbol{v}^{(1)}} + D_{\boldsymbol{r}^{(1)}}) & -G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} & \cdots & & -G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} & \cdots & -G_{\ddot{y}^{(K)}\ddot{y}^{(1)}} & & & & B_{\ddot{y}^{(1)}}^{\mathsf{T}} & & \\
-G_{\ddot{x}\ddot{y}^{(1)}} & -G_{\dot{x}\ddot{y}^{(1)}} & -G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} & & \cdots & & -G_{\ddot{y}^{(1)}\ddot{y}^{(1)}} & \cdots & -G_{\ddot{y}^{(K)}\ddot{y}^{(1)}} & & & & B_{\dot{y}^{(1)}}^{\mathsf{T}} & & \\
\vdots & \vdots & \vdots & \vdots & \cdots & \cdots & \vdots & & \vdots & & & & & \ddots & \\
-G_{\ddot{x}\ddot{y}^{(K)}} & -G_{\dot{x}\ddot{y}^{(K)}} & -G_{\ddot{y}^{(1)}\ddot{y}^{(K)}} & & \cdots & -(G_{\ddot{y}^{(K)}\ddot{y}^{(K)}} - D_{\boldsymbol{v}^{(k)}} + D_{\boldsymbol{r}^{(k)}}) & & \cdots & -G_{\ddot{y}^{(K)}\ddot{y}^{(2)}} & & & & & & B_{\ddot{y}^{(K)}}^{\mathsf{T}} \\
-G_{\dot{x}\ddot{y}^{(K)}} & -G_{\dot{x}\dot{y}^{(K)}} & -G_{\ddot{y}^{(1)}\dot{y}^{(K)}} & & \cdots & & -G_{\ddot{y}^{(K)}\dot{y}^{(K)}} & & -G_{\dot{y}^{(K)}\dot{y}^{(K)}} & & & & & & B_{\dot{y}^{(K)}}^{\mathsf{T}} \\
A_{\ddot{x}}^{\mathsf{T}} & A_{\dot{x}}^{\mathsf{T}} & & & & & & & & E & & & & & \\
B_{\ddot{x}}^{\mathsf{T}} & B_{\dot{x}}^{\mathsf{T}} & & B_{\ddot{y}^{(1)}}^{\mathsf{T}} & & & & & & & F^{(1)} & & & & \\
\vdots & \vdots & & \vdots & & & & & & & & \ddots & & & \\
B_{\ddot{x}}^{\mathsf{T}} & B_{\dot{x}}^{\mathsf{T}} & & B_{\dot{y}^{(K)}}^{\mathsf{T}} & & & & & & & & & & & F^{(K)}
\end{bmatrix}
\begin{bmatrix}
\Delta \ddot{\boldsymbol{x}} \\ \Delta \dot{\boldsymbol{x}} \\ \Delta \ddot{\boldsymbol{y}}^{(1)} \\ \Delta \dot{\boldsymbol{y}}^{(1)} \\ \vdots \\ \Delta \ddot{\boldsymbol{y}}^{(K)} \\ \Delta \dot{\boldsymbol{y}}^{(K)} \\ \Delta \boldsymbol{\nu} \\ \Delta \boldsymbol{\vartheta}^{(1)} \\ \vdots \\ \Delta \boldsymbol{\vartheta}^{(K)}
\end{bmatrix}
=
\begin{bmatrix}
\boldsymbol{\sigma}_{\ddot{x}} + D_{\boldsymbol{w}} \boldsymbol{\varrho} - D_{\boldsymbol{t}} \boldsymbol{\tau} \\ \boldsymbol{\sigma}_{\dot{x}} \\ \boldsymbol{\sigma}_{\ddot{y}^{(2)}} + D_{v^{(2)}} \boldsymbol{\zeta}^{(2)} - D_{r^{(2)}} \boldsymbol{\kappa}^{(2)} \\ \boldsymbol{\sigma}_{\dot{y}^{(1)}} \\ \vdots \\ \boldsymbol{\sigma}_{\ddot{y}^{(K)}} + D_{v^{(K)}} \boldsymbol{\zeta}^{(K)} - D_{r^{(k)}} \boldsymbol{\kappa}^{(K)} \\ \boldsymbol{\sigma}_{\dot{y}^{(K)}} \\ \boldsymbol{\rho} + H_s^{-1} \boldsymbol{\beta}_s \\ \boldsymbol{\eta}^{(1)} + H_{z^{(1)}}^{-1} \boldsymbol{\beta}_{z^{(1)}} \\ \vdots \\ \boldsymbol{\eta}^{(K)} + H_{z^{(k)}}^{-1} \boldsymbol{\beta}_{z^{(K)}}
\end{bmatrix}
\tag{18}
$$

The matrices $D_{\boldsymbol{w}}, D_{\boldsymbol{t}}, D_{\boldsymbol{v}^{(k)}}, D_{\boldsymbol{r}^{(k)}}, E, F^{(k)}, H_{\boldsymbol{s}}$ and $H_{\boldsymbol{z}^{(k)}}$, $k = 1, 2, \ldots, K$, that appeared in (18) are defined as

$$
\begin{aligned}
D_{\boldsymbol{w}} &= \Big(\big((\Gamma + \Phi_{\boldsymbol{\gamma}})^{-1}(W + \Xi_{\boldsymbol{w}}) + \Xi_{\boldsymbol{w}}(C_{\boldsymbol{w}} - \Gamma - \Phi_{\boldsymbol{\gamma}})^{-1}\big)^{-1} + \Phi_{\boldsymbol{\gamma}}(B_{\boldsymbol{\gamma}} - W)^{-1}\Big)^{-1},\\
D_{\boldsymbol{t}} &= \Big(\big((\Psi + \Phi_{\boldsymbol{\psi}})^{-1}(T + \Xi_{\boldsymbol{t}}) + \Xi_{\boldsymbol{t}}(C_{\boldsymbol{t}} - \Psi - \Phi_{\boldsymbol{\psi}})^{-1}\big)^{-1} + \Phi_{\boldsymbol{\psi}}(B_{\boldsymbol{\psi}} - T)^{-1}\Big)^{-1},\\
D_{\boldsymbol{v}^{(k)}} &= \Big(\big((\Lambda^{(k)} + \Phi_{\boldsymbol{\lambda}^{(k)}})^{-1}(V^{(k)} + \Xi_{\boldsymbol{v}^{(k)}}) + \Xi_{\boldsymbol{v}^{(k)}}(C_{\boldsymbol{v}^{(k)}} - \Lambda^{(k)} - \Phi_{\boldsymbol{\lambda}^{(k)}})^{-1}\big)^{-1} + \Phi_{\boldsymbol{\lambda}^{(k)}}(B_{\boldsymbol{\lambda}^{(k)}} - V^{(k)})^{-1}\Big)^{-1},\\
D_{\boldsymbol{r}^{(k)}} &= \Big(\big((\Upsilon^{(k)} + \Phi_{\boldsymbol{\upsilon}^{(k)}})^{-1}(R^{(k)} + \Xi_{\boldsymbol{r}^{(k)}}) + \Xi_{\boldsymbol{r}^{(k)}}(C_{\boldsymbol{r}^{(k)}} - \Upsilon^{(k)} - \Phi_{\boldsymbol{\upsilon}^{(k)}})^{-1}\big)^{-1} + \Phi_{\boldsymbol{\upsilon}^{(k)}}(B_{\boldsymbol{\upsilon}^{(k)}} - R^{(k)})^{-1}\Big)^{-1},\\
E &\triangleq \Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}} + \boldsymbol{\nu})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{s}})\Big)^{-1}\Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big)\\
&\quad + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\Big(-\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\\
&\quad \mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}^{-1}(\boldsymbol{\phi}_{\boldsymbol{\nu}} + \boldsymbol{\nu}) + \Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}}\Big)^{-1}\Big)^{-1}\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s}),\\
F^{(k)} &\triangleq \Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{z}^{(k)}})\Big)^{-1}\\
&\quad \Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{(k)})\Big) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\Big(-\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\Big)\\
&\quad + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\\
&\quad + \mathrm{Arw}^{-1}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)}) + \Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}\Big)^{-1}\Big)^{-1}\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)}),\\
H_{\boldsymbol{s}} &\triangleq \Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}} + \boldsymbol{\nu})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{s}})\Big)^{-1}\Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big)\\
&\quad + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\Big(-\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\\
&\quad \mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}}) + \Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}}\Big)^{-1}\Big)^{-1},\\
H_{\boldsymbol{z}^{(k)}} &\triangleq \Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{z}^{(k)}})\Big)^{-1}\Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\\
&\quad \mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{(k)})\Big) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\Big(-\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\Big) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\\
&\quad \Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\\
&\quad + \Big(\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}\Big)^{-1}\Big)^{-1}\mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)}) + \mathrm{Arw}^{-1}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)}),
\end{aligned}
$$

and the vectors $\boldsymbol{\beta}_{\boldsymbol{s}}, \boldsymbol{\beta}_{\boldsymbol{\gamma}}, \boldsymbol{\beta}_{\boldsymbol{\psi}}, \boldsymbol{\beta}_{\boldsymbol{z}^{(k)}}, \boldsymbol{\beta}_{\boldsymbol{\lambda}^{(k)}}$, and $\boldsymbol{\beta}_{\boldsymbol{\upsilon}^{(k)}}$, $k = 1, 2, \ldots, K$, that appeared in (18) are defined as

$$
\begin{aligned}
\boldsymbol{\beta}_{\boldsymbol{s}} &= 2\mu\boldsymbol{e} - (\boldsymbol{\phi}_{\boldsymbol{s}} + \boldsymbol{\nu}) \circ (\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s}) - \Big(\mathrm{Arw}^{-1}(\boldsymbol{\phi}_{\boldsymbol{\nu}} + \boldsymbol{\nu})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big)^{-1}\\
&\quad + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{s}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{s}} + \boldsymbol{s})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\nu}})\Big)^{-1}\Big)(\boldsymbol{\xi}_{\boldsymbol{s}} \circ (\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}})),\\
\boldsymbol{\beta}_{\boldsymbol{\gamma}} &= ((\Gamma + \Phi_{\boldsymbol{\gamma}})^{-1}(T + \Xi_{\boldsymbol{w}}) + \Xi_{\boldsymbol{w}}(C_{\boldsymbol{w}} - \Gamma - \Phi_{\boldsymbol{w}})^{-1}\big)^{-1}\\
&\quad \Big(\mu(\Gamma + \Phi_{\gamma})^{-1}\boldsymbol{e} - \mu(C_{\boldsymbol{w}} - \Gamma - \Phi_{\gamma})^{-1})\boldsymbol{e} - \boldsymbol{w}\Big) - \Big(\mu(B_{\gamma} - W)^{-1}\boldsymbol{e} - \boldsymbol{\phi}_{\gamma}\Big),\\
\boldsymbol{\beta}_{\boldsymbol{\psi}} &= \Big((\Psi + \Phi_{\boldsymbol{\psi}})^{-1}(T + \Xi_{\boldsymbol{t}}) + \Xi_{\boldsymbol{t}}(C_{\boldsymbol{t}} - \Psi - \Phi_{\boldsymbol{t}})^{-1}\Big)^{-1}\\
&\quad \Big(\mu(\Psi + \Phi_{\psi})^{-1}\boldsymbol{e} - \mu(C_{\boldsymbol{t}} - \Psi - \Phi_{\psi})^{-1})\boldsymbol{e} - \boldsymbol{t}\Big) - \Big(\mu(B_{\psi} - T)^{-1}\boldsymbol{e} - \boldsymbol{\phi}_{\psi}\Big),\\
\boldsymbol{\beta}_{\boldsymbol{z}^{(k)}} &= 2\mu\boldsymbol{e} - (\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)}) \circ (\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)}) - \Big(\mathrm{Arw}^{-1}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}} + \boldsymbol{\vartheta}^{(k)})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}) + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\\
&\quad \mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\Big)^{-1} + \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\Big(\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\Big(\mathrm{Arw}(\boldsymbol{c}_{\boldsymbol{z}^{(k)}} - \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\\
&\quad \mathrm{Arw}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}})\mathrm{Arw}^{-1}(\boldsymbol{\xi}_{\boldsymbol{z}^{(k)}} + \boldsymbol{z}^{(k)})\mathrm{Arw}(\boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}})\Big)^{-1}\Big)(\boldsymbol{\xi}_{\boldsymbol{s}} \circ (\boldsymbol{c}_{\boldsymbol{s}} - \boldsymbol{\phi}_{\boldsymbol{\nu}})),\\
\boldsymbol{\beta}_{\boldsymbol{\lambda}^{(k)}} &= \Big((\Lambda^{(k)} + \Phi_{\boldsymbol{\lambda}^{(k)}})^{-1}(V^{(k)} + \Xi_{\boldsymbol{v}^{(k)}}) + \Xi_{\boldsymbol{v}^{(k)}}(C_{\boldsymbol{v}^{(k)}} - \Lambda^{(k)} - \Phi_{\boldsymbol{\lambda}^{(k)}})^{-1}\Big)^{-1}\\
&\quad \Big(\mu\big(\Lambda^{(k)} + \Phi_{\lambda^{(k)}}\big)^{-1}\boldsymbol{e} - \mu(C_{\boldsymbol{v}^{(k)}} - \Lambda^{(k)} - \Phi_{\lambda^{(k)}})^{-1})\boldsymbol{e} - \boldsymbol{v}^{(k)}) - \Big(\mu(B_{\lambda^{(k)}} - V^{(k)})^{-1}\boldsymbol{e} - \boldsymbol{\phi}_{\lambda^{(k)}}\Big),\\
\boldsymbol{\beta}_{\boldsymbol{\upsilon}^{(k)}} &= \Big((\Upsilon^{(k)} + \Phi_{\boldsymbol{\upsilon}^{(k)}})^{-1}(R^{(k)} + \Xi_{\boldsymbol{r}^{(k)}}) + \Xi_{\boldsymbol{r}^{(k)}}(C_{\boldsymbol{r}^{(k)}} - \Upsilon^{(k)} - \Phi_{\boldsymbol{\upsilon}^{(k)}})^{-1}\Big)^{-1}\\
&\quad \Big(\mu\big(\Upsilon + \Phi_{\upsilon^{(k)}}\big)^{-1}\boldsymbol{e} - \mu(C_{\boldsymbol{r}^{(k)}} - \Upsilon^{(k)} - \Phi_{\upsilon^{(k)}})^{-1})\boldsymbol{e} - \boldsymbol{r}^{(k)}) - \Big(\mu(B_{(k)} - R^{(k)})^{-1}\boldsymbol{e} - \boldsymbol{\phi}_{\upsilon^{(k)}}\Big).
\end{aligned}
$$

At each step $\alpha_p^{\langle j \rangle}$, the steplength is chosen to ensure that all the primal relaxation variables are positive variables and also ensure that

$$\begin{array}{lll} \boldsymbol{s}^{\langle j+1 \rangle} + \boldsymbol{\xi}_{\boldsymbol{s}}^{\langle j+1 \rangle} \succ_r \mathbf{0}, & \boldsymbol{w}^{\langle j+1 \rangle} + \boldsymbol{\xi}_{\boldsymbol{w}}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{t}^{\langle j+1 \rangle} + \boldsymbol{\xi}_{\boldsymbol{t}}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{z}_{(l+1)}^{(k)} + \boldsymbol{\xi}_{\boldsymbol{z}^{(k)}}^{\langle j+1 \rangle} \succ_s \mathbf{0}, & \boldsymbol{v}^{\langle j+1 \rangle} + \boldsymbol{\xi}_{\boldsymbol{v}}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{r}^{\langle j+1 \rangle} + \boldsymbol{\xi}_{\boldsymbol{r}}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{b}_{\nu} - \boldsymbol{s}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{b}_{\gamma} - \boldsymbol{w}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{b}_{\psi} - \boldsymbol{t}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{b}_{\vartheta^{(k)}} - \boldsymbol{z}_{(l+1)}^{(k)} > \mathbf{0}, & \boldsymbol{b}_{\lambda^{(k)}} - \boldsymbol{v}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{b}_{\upsilon^{(k)}} - \boldsymbol{r}^{\langle j+1 \rangle} > \mathbf{0}. \end{array} \tag{19}$$

Similarly, at each step $\alpha_d^{\langle j \rangle}$, the steplengh is chosen to assure that all the dual relaxation variables are positive and ensure that

$$\begin{array}{lll} \boldsymbol{\nu}^{\langle j+1 \rangle} + \boldsymbol{\phi}_{\nu}^{\langle j+1 \rangle} \succ_r \mathbf{0}, & \boldsymbol{\gamma}^{\langle j+1 \rangle} + \boldsymbol{\phi}_{\gamma}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{\psi}^{\langle j+1 \rangle} + \boldsymbol{\phi}_{\psi}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{\vartheta}_{(l+1)}^{(k)} + \boldsymbol{\phi}_{\vartheta^{(k)}}^{\langle j+1 \rangle} \succ_s \mathbf{0}, & \boldsymbol{\lambda}_{(l+1)}^{(k)} + \boldsymbol{\phi}_{\lambda^{(k)}}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{\upsilon}_{(l+1)}^{(k)} + \boldsymbol{\phi}_{\upsilon^{(k)}}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{c}_s - \boldsymbol{\phi}_{\nu}^{\langle j+1 \rangle} \succ_r \mathbf{0}, & \boldsymbol{c}_{\boldsymbol{w}} - \boldsymbol{\gamma}^{\langle j+1 \rangle} - \boldsymbol{\phi}_{\gamma}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{c}_{\boldsymbol{t}} - \boldsymbol{\psi}^{\langle j+1 \rangle} - \boldsymbol{\phi}_{\psi}^{\langle j+1 \rangle} > \mathbf{0}, \\ \boldsymbol{c}_{z^{(k)}} - \boldsymbol{\phi}_{\vartheta}^{\langle j+1 \rangle} \succ_s \mathbf{0}, & \boldsymbol{c}_{\boldsymbol{v}^{(k)}} - \boldsymbol{\lambda}_{(l+1)}^{(k)} - \boldsymbol{\phi}_{\lambda^{(k)}}^{\langle j+1 \rangle} > \mathbf{0}, & \boldsymbol{c}_{\boldsymbol{r}^{(k)}} - \boldsymbol{\upsilon}_{(l+1)}^{(k)} - \boldsymbol{\phi}_{\upsilon^{(k)}}^{\langle j+1 \rangle} > \mathbf{0}. \end{array} \tag{20}$$

Newton's method is used to solve the first-order conditions (17), while a line search is used to ensure progress toward optimality and to update the value of μ at each iteration (see [13, 14]). We also need to look at the penalty parameters to make sure that we have obtained a solution to the original SMINLSOCP problem or we can show a certificate of infeasibility. We point out that the sparcity structure of the reduced KKT system (18) resulting from the penalty problem (15) is similar that of (12).

The primal–dual penalty method is the optimal choice for the warmstarting problems of the interior-point method. On each node, we can use the optimal primal, dual, and slack variables of its parent as the initial solution and re-initialize the primal and dual relaxation variables to help the original variables move toward an optimum.

5.2 Dynamic penalty parameters updates and a primal–dual algorithm

The most essential aspect of setting and updating the penalty parameters is to make sure that they are larger than the components of the current subproblem for which they act as upper bounds. Let $(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)^\star}, \ldots, \boldsymbol{y}^{(K)^\star})$ be the optimal solution of an SMINLSOCP subproblem with the corresponding optimal slack variables $(\boldsymbol{s}^\star, \boldsymbol{w}^\star, \boldsymbol{t}^\star, \boldsymbol{z}^{(k)^\star}, \boldsymbol{v}^{(k)^\star}, \boldsymbol{r}^{(k)^\star})$ and optimal dual variables $(\boldsymbol{\nu}^\star, \boldsymbol{\gamma}^\star, \boldsymbol{\psi}^\star, \boldsymbol{\vartheta}^{(k)^\star}, \boldsymbol{\upsilon}^{(k)^\star})$, for $k = 1, 2, \ldots, K$. The penalty parameters are obtained as follow:

$$\begin{array}{llllllllll} \boldsymbol{b}_{\nu} & \triangleq & \boldsymbol{s}^\star + \boldsymbol{\kappa}_{\boldsymbol{s}}, & \boldsymbol{b}_{\gamma} & \triangleq & \boldsymbol{w}^\star + \boldsymbol{\kappa}_{\boldsymbol{wt}}, & \boldsymbol{b}_{\psi} & \triangleq & \boldsymbol{t}^\star + \boldsymbol{\kappa}_{\boldsymbol{wt}}, \\ \boldsymbol{b}_{\boldsymbol{\vartheta}^{(k)}} & \triangleq & \boldsymbol{z}^{(k)^\star} + \boldsymbol{\kappa}_{\boldsymbol{z}^{(k)}}, & \boldsymbol{b}_{\lambda^{(k)}} & \triangleq & \boldsymbol{v}^{(k)^\star} + \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}}, & \boldsymbol{b}_{\boldsymbol{\upsilon}^{(k)}} & \triangleq & \boldsymbol{r}^{(k)^\star} + \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}}, \\ \boldsymbol{c}_s & \triangleq & \boldsymbol{\nu}^\star + \boldsymbol{\phi}_{\nu}^{(0)} + \boldsymbol{\hbar}_{\nu}, & \boldsymbol{c}_{\boldsymbol{w}} & \triangleq & \boldsymbol{\gamma}^\star + \boldsymbol{\phi}_{\gamma}^{(0)} + \boldsymbol{\hbar}_{\gamma\psi}, & \boldsymbol{c}_{\boldsymbol{t}} & \triangleq & \boldsymbol{\psi}^\star + \boldsymbol{\phi}_{\psi}^{(0)} + \boldsymbol{\hbar}_{\gamma\psi} \\ \boldsymbol{c}_{z^{(k)}} & \triangleq & \boldsymbol{\vartheta}^{(k)^\star} + \boldsymbol{\phi}_{\vartheta^{(k)}}^{(0)} + \boldsymbol{\hbar}_{\vartheta^{(k)}}, & \boldsymbol{c}_{\boldsymbol{v}^{(k)}} & \triangleq & \boldsymbol{\lambda}^{(k)^\star} + \boldsymbol{\phi}_{\lambda^{(k)}}^{(0)} + \boldsymbol{\hbar}_{\lambda v_k}, & \boldsymbol{c}_{\boldsymbol{r}^{(k)}} & \triangleq & \boldsymbol{\upsilon}^{(k)^\star} + \boldsymbol{\phi}_{\upsilon^{(k)}}^{(0)} + \boldsymbol{\hbar}_{\lambda v_k}, \end{array} \tag{21}$$

where

$$\begin{array}{llllll} \boldsymbol{\kappa}_s & \triangleq & \max(\boldsymbol{h}(\boldsymbol{x}^\star), \boldsymbol{s}^\star, \mathbf{1}), & \boldsymbol{\kappa}_{\boldsymbol{wt}} & \triangleq & \max(\boldsymbol{w}^\star, \boldsymbol{t}^\star, \mathbf{1}), \\ \boldsymbol{\kappa}_{\boldsymbol{z}^{(k)}} & \triangleq & \max(\boldsymbol{g}^{(k)}(\boldsymbol{x}^\star, \boldsymbol{y}^{(k)^\star}), \boldsymbol{z}^{(k)^\star}, \mathbf{1}), & \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}} & \triangleq & \max(\boldsymbol{v}^{(k)^\star}, \boldsymbol{r}^{(k)^\star}, \mathbf{1}), \end{array}$$

and

$$\begin{array}{llllll} \boldsymbol{\hbar}_{\gamma\psi} & \triangleq & \max(\boldsymbol{\gamma}^\star, \boldsymbol{\psi}^\star, \nabla_{\ddot{x}} f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)\star}, \ldots, \boldsymbol{y}^{(K)\star}), A_{\ddot{x}}(\boldsymbol{x}^\star, \boldsymbol{y}^\star)^\mathsf{T} \boldsymbol{\nu}^\star, \mathbf{1}), & \boldsymbol{\hbar}_{\nu} & \triangleq & \max(\boldsymbol{\nu}^\star, \mathbf{1}), \\ \boldsymbol{\hbar}_{\boldsymbol{\lambda}^{(k)}\boldsymbol{\upsilon}^{(k)}} & \triangleq & \max(\boldsymbol{\lambda}^{(k)^\star}, \boldsymbol{\upsilon}^{(k)^\star}, \nabla_{\ddot{y}^{(k)}} f(\boldsymbol{x}^\star, \boldsymbol{y}^{(1)\star}, \ldots, \boldsymbol{y}^{(K)\star}), B_{\ddot{y}^{(k)}}(\boldsymbol{x}^\star, \boldsymbol{y}^{(k)^\star})^\mathsf{T} \boldsymbol{\vartheta}^{(k)^\star}, \mathbf{1}), & \boldsymbol{\hbar}_{\boldsymbol{\vartheta}^{(k)}} & \triangleq & \max(\boldsymbol{\vartheta}^{(k)^\star}, \mathbf{1}). \end{array}$$

The initial relaxation variables are chosen as follow:

$$\begin{array}{llllll} \boldsymbol{\xi}_{\boldsymbol{s}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\kappa}_s, \boldsymbol{\kappa}_{\boldsymbol{wt}}), & \boldsymbol{\phi}_{\boldsymbol{\nu}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\nu}, \boldsymbol{\hbar}_{\gamma\psi}), \\ \boldsymbol{\xi}_{\boldsymbol{w}}^{\langle 0 \rangle} & \triangleq & \max(\boldsymbol{l} - \ddot{\boldsymbol{x}}^\star, \mathbf{0}) + 10^{-4}\max(\boldsymbol{\kappa}_s, \boldsymbol{\kappa}_{\boldsymbol{wt}}), & \boldsymbol{\phi}_{\boldsymbol{\gamma}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\nu}, \boldsymbol{\hbar}_{\gamma\psi}), \\ \boldsymbol{\xi}_{\boldsymbol{t}}^{\langle 0 \rangle} & \triangleq & \max(\ddot{\boldsymbol{x}}^\star - \boldsymbol{u}, \mathbf{0}) + 10^{-4}\max(\boldsymbol{\kappa}_s, \boldsymbol{\kappa}_{\boldsymbol{wt}}), & \boldsymbol{\phi}_{\boldsymbol{\psi}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\nu}, \boldsymbol{\hbar}_{\gamma\psi}), \\ \boldsymbol{\xi}_{\boldsymbol{z}^{(k)}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\kappa}_{\boldsymbol{z}^{(k)}}, \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}}), & \boldsymbol{\phi}_{\boldsymbol{\vartheta}^{(k)}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\boldsymbol{\vartheta}^{(k)}}, \boldsymbol{\hbar}_{\boldsymbol{\lambda}^{(k)}\boldsymbol{\upsilon}^{(k)}}), \\ \boldsymbol{\xi}_{\boldsymbol{v}^{(k)}} & \triangleq & \max(\tilde{\boldsymbol{l}} - \ddot{\boldsymbol{y}}^{(k)^\star}, \mathbf{0}) + 10^{-4}\max(\boldsymbol{\kappa}_{\boldsymbol{z}^{(k)}}, \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}}), & \boldsymbol{\phi}_{\boldsymbol{\lambda}^{(k)}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\boldsymbol{\vartheta}^{(k)}}, \boldsymbol{\hbar}_{\boldsymbol{\lambda}^{(k)}\boldsymbol{\upsilon}^{(k)}}), \\ \boldsymbol{\xi}_{\boldsymbol{r}^{(k)\star}} & \triangleq & \max(\ddot{\boldsymbol{y}}^{(k)^\star} - \tilde{\boldsymbol{u}}^{(k)}, \mathbf{0}) + 10^{-4}\max(\boldsymbol{\kappa}_{\boldsymbol{z}^{(k)}}, \boldsymbol{\kappa}_{\boldsymbol{v}^{(k)}\boldsymbol{r}^{(k)}}), & \boldsymbol{\phi}_{\tilde{\boldsymbol{\upsilon}}^{(k)}}^{\langle 0 \rangle} & \triangleq & 10^{-4}\max(\boldsymbol{\hbar}_{\boldsymbol{\vartheta}^{(k)}}, \boldsymbol{\hbar}_{\boldsymbol{\lambda}^{(k)}\boldsymbol{\upsilon}^{(k)}}). \end{array} \tag{22}$$

These settings are sufficient after warmstart to start the penalty approach without moving the iterates too far from the current node. The relaxation is performed using a variable, so if a larger relaxation is needed, the relaxation variables will move accordingly.

Since the initial values of all penalty parameters might not be large enough to admit the optimal solution, we also need a way to update them. Given the relaxation, there is always an optimal solution for (15), and one way to update it in a *static* way is to find the optimal solution and then increase the penalty parameters if the relaxation variables are not close enough to zero. Even though this may require a lot of solutions, it greatly increases the number of iterations needed to find the optimal solution. Instead, we can use a *dynamic* updating strategy in which the penalty parameters are checked and updated at the end of each iteration as shown in Algorithm 5.1, which provides the details of the primal–dual penalty method with a dynamic update for two-stage SMINLSOCP problem. Concerning the time complexity of Algorithm 5.1, Section 5 in [8] discuses the polynomial convergence of this class of algorithms.

In the remaining part of this section we describe a termination criterion for infeasibility. In the preceding discussion, we determined what can go wrong when warmstarting an interior-point technique and presented the exact primal–dual penalty method as an alternative. Another issue for developing the inner-level algorithm within our framework was the efficient identification of infeasible nonlinear second-order cone programming subproblems. The primal-dual penalty method, which is provided as a solution for warmstarting, is also useful for detecting infeasibility. In addition, the upper bounds on the slack variables ensure that there is always an optimal solution to (15). Consequently, a provably convergent nonlinear second-order cone algorithm is guaranteed to find an optimal solution to Problem (15). If such a solution satisfies that $\xi_{s_i} \to a, \xi_{w_j} \to a$, or $\xi_{t_j} \to a$, for at least $i = 1, 2, \ldots, r$ and $j = 1, 2, \ldots, p$, or that $\xi_{z_i^{(k)}} \to a, \xi_{v_j^{(k)}} \to a$, or $\xi_{r_j^{(k)}} \to a$ for at least $i = 1, 2, \ldots, s$ and $j = 1, 2, \ldots, q$ for a scalar a as $c_{s_i} \to \infty$, $c_{w_j} \to \infty$, or $c_{t_j} \to \infty$, for at least $i = 1, 2, \ldots, r$ and $j = 1, 2, \ldots, p$, or $c_{z_i^{(k)}} \to \infty, c_{v_j^{(k)}} \to \infty$, or $c_{r_j^{(k)}} \to \infty$, for at least $i = 1, 2, \ldots, s$ and $j = 1, 2, \ldots, q$, then the original problem is infeasible.

Allowing a penalty parameter to be infinite is impractical. Still, a practical implementation is easy to make if the original objective function is removed and the penalty term is the only one that is minimized. This is the same as letting all penalty parameters become infinite. So, a feasibility restoration phase that can be used is the same one as Phase I of SNOPT[2] [23], and the resulting problem is

$$
\begin{array}{lll}
\min & \boldsymbol{c}_s^{\mathsf{T}}\boldsymbol{\xi}_s + \boldsymbol{c}_w^{\mathsf{T}}\boldsymbol{\xi}_w + \boldsymbol{c}_t^{\mathsf{T}}\boldsymbol{\xi}_t + \sum_{k=1}^{K}(\boldsymbol{c}_{z^{(k)}}^{\mathsf{T}}\boldsymbol{\xi}_{z^{(k)}} + \boldsymbol{c}_{v^{(k)}}^{\mathsf{T}}\boldsymbol{\xi}_{v^{(k)}} + \boldsymbol{c}_{r^{(k)}}^{\mathsf{T}}\boldsymbol{\xi}_{r^{(k)}}) & \\
\text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{s} = \boldsymbol{0}, & \\
& \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) - \boldsymbol{z}^{(k)} = \boldsymbol{0}, & k = 1, \ldots, K, \\
& \ddot{\boldsymbol{x}} - \boldsymbol{w} = \boldsymbol{l}, & \\
& \ddot{\boldsymbol{y}} - \boldsymbol{v}^{(k)} = \tilde{\boldsymbol{l}}^{(k)}, & k = 1, \ldots, K, \\
& \ddot{\boldsymbol{x}} + \boldsymbol{t} = \boldsymbol{u}, & \\
& \ddot{\boldsymbol{y}}^{(k)} + \boldsymbol{r}^{(k)} = \tilde{\boldsymbol{u}}^{(k)}, & k = 1, \ldots, K, \\
& \boldsymbol{s} + \boldsymbol{\xi}_s \succeq_r \boldsymbol{0}, & \\
& \boldsymbol{b}_v - \boldsymbol{s} \succeq_r \boldsymbol{0}, & \\
& \boldsymbol{z}^{(k)} + \boldsymbol{\xi}_{z^{(k)}} \succeq_s \boldsymbol{0}, & k = 1, \ldots, K, \\
& \boldsymbol{b}_{\vartheta^{(k)}} - \boldsymbol{z}^{(k)} \succeq_s \boldsymbol{0}, & k = 1, \ldots, K, \\
& -\boldsymbol{\xi}_w \le \boldsymbol{w} \le \boldsymbol{b}_\gamma, & \\
& -\boldsymbol{\xi}_t \le \boldsymbol{t} \le \boldsymbol{b}_\psi, & \\
& -\boldsymbol{\xi}_v \le \boldsymbol{v} \le \boldsymbol{b}_\lambda, & \\
& -\boldsymbol{\xi}_r \le \boldsymbol{r} \le \boldsymbol{b}_\upsilon, & \\
& \boldsymbol{\xi}_s, \boldsymbol{\xi}_{z^{(k)}} \ge \boldsymbol{0}, & k = 1, \ldots, K, \\
& \boldsymbol{\xi}_w, \boldsymbol{\xi}_t, \boldsymbol{\xi}_v, \boldsymbol{\xi}_r \ge \boldsymbol{0}. &
\end{array}
\tag{23}
$$

The purpose of solving Problem (23) is to minimize infeasibility. It differs from SNOPT in the sense that the slack variables continue to be bounded above by the dual penalty parameters. As these parameters are updated regularly, we can always find a feasible solution to (23). If the optimal objective function value is greater than the infeasibility tolerance, a certificate of infeasibility can be returned.

[2]SNOPT is for Sparse Nonlinear OPTimizer; a software package for solving large-scale nonlinear optimization problem.

Algorithm 5.1 A primal–dual penalty algorithm with a dynamic update for two-stage SMINLSOCP problem

begin

Initialize: $\boldsymbol{x} = \boldsymbol{x}^{\langle 0\rangle}$, $\boldsymbol{s} = \boldsymbol{s}^{\langle 0\rangle}$,$\boldsymbol{w} = \boldsymbol{w}^{\langle 0\rangle}$, $\boldsymbol{t} = \boldsymbol{t}^{\langle 0\rangle}$, $\boldsymbol{y} = \boldsymbol{y}^{(k)\langle 0\rangle}$,$\boldsymbol{z}^{(k)} = \boldsymbol{z}^{(k)\langle 0\rangle}$, $\boldsymbol{v}^{(k)} = \boldsymbol{v}^{(k)\langle 0\rangle}$, $\boldsymbol{r}^{(k)} = \boldsymbol{r}^{(k)\langle 0\rangle}$, for $k = 1, \ldots, K$;
Require: Penalty parameters as in (21), and relaxation variables as in (22);
Ensure: The tolerance $\tau = 10^{-8}$, the barrier paremeter $\mu = 0.3$;
Result: Primal and dual optimal solutions of Problem (15);

Step 1. Compute the search directions by solving System (18);

Step 2. Choose primal and dual steplenghs $\alpha_p^{\langle l\rangle}$ and $\alpha_d^{\langle l\rangle}$ that ensure (19) and (20);

Step 3. Update the penalty parameters, for $i = 1, \ldots, r$ and $j = 1, \ldots, p$: ▷ Dynamic updating

$$
\begin{array}{lll}
\textbf{If } s_i^{\langle l+1\rangle} - 0.9b_{v_i}^{\langle l\rangle} > 0, & \textbf{then} & b_{v_i}^{\langle l+1\rangle} \leftarrow 10b_{v_i}^{\langle l\rangle};\\
\textbf{If } w_j^{\langle l+1\rangle} - 0.9b_{\gamma_j}^{\langle l\rangle} > 0, & \textbf{then} & b_{\gamma_j}^{\langle l+1\rangle} \leftarrow 10b_{\gamma_j}^{\langle l\rangle};\\
\textbf{If } t_j^{\langle l+1\rangle} - 0.9b_{\psi_j}^{\langle l\rangle} > 0, & \textbf{then} & b_{\psi_j}^{\langle l+1\rangle} \leftarrow 10b_{\psi_j}^{\langle l\rangle};\\
\textbf{If } v_i^{\langle l+1\rangle} + \phi_{v_i}^{\langle l\rangle} - 0.9c_{s_i}^{\langle l\rangle} \geq 0, & \textbf{then} & c_{s_i}^{\langle l+1\rangle} \leftarrow 10c_{v_i}^{\langle l\rangle};\\
\textbf{If } \gamma_j^{\langle l+1\rangle} + \phi_{\gamma_j}^{\langle l\rangle} > 0.9c_{w_j}^{\langle l\rangle}, & \textbf{then} & \boldsymbol{c}_{\boldsymbol{w}_j}^{\langle l+1\rangle} \leftarrow 10\boldsymbol{c}_{w_j}^{\langle l\rangle};\\
\textbf{If } \psi_j^{\langle l+1\rangle} + \phi_{\psi_j}^{\langle l\rangle} > 0.9c_{t_j}^{\langle l\rangle}, & \textbf{then} & \boldsymbol{c}_{\boldsymbol{t}_j}^{\langle l+1\rangle} \leftarrow 10\boldsymbol{c}_{t_j}^{\langle l\rangle},
\end{array}
$$

and for $i = 1, \ldots, s$ and $j = 1, \ldots, q$: ▷ Dynamic updating

$$
\begin{array}{lll}
\textbf{If } z_i^{(k)\langle l+1\rangle} - 0.9b_{\vartheta_i}^{\langle l\rangle} > 0, & \textbf{then} & b_{\vartheta_i}^{\langle l+1\rangle} \leftarrow 10b_{\vartheta_i}^{\langle l\rangle};\\
\textbf{If } v_j^{(k)\langle l+1\rangle} - 0.9b_{\lambda_j^{(k)}}^{\langle l\rangle} > 0, & \textbf{then} & b_{\lambda_j^{(k)}}^{\langle l+1\rangle} \leftarrow 10b_{\lambda_j^{(k)}}^{\langle l\rangle};\\
\textbf{If } r_j^{\langle l+1\rangle} - 0.9b_{\upsilon_j^{(k)}}^{\langle l\rangle} > 0, & \textbf{then} & b_{\upsilon_j}^{\langle l+1\rangle} \leftarrow 10b_{\upsilon_j^{(k)}}^{\langle l\rangle};\\
\textbf{If } \vartheta_i^{(k)\langle l+1\rangle} + \phi_{\vartheta_i(k)}^{\langle l\rangle} - 0.9c_{z_i^{(k)}} > 0, & \textbf{then} & c_{z_i^{(k)}}^{\langle l+1\rangle} \leftarrow 10c_{z_i^{(k)}}^{\langle l\rangle};\\
\textbf{If } \lambda_j^{(k)\langle l+1\rangle} + \phi_{\lambda_j^{(k)}}^{\langle l\rangle} > 0.9c_{\boldsymbol{v}_j^{(k)}}^{\langle l\rangle}, & \textbf{then} & \boldsymbol{c}_{\boldsymbol{v}_j^{(k)}}^{\langle l+1\rangle} \leftarrow 10\boldsymbol{c}_{\boldsymbol{v}_j^{(k)}}^{\langle l\rangle};\\
\textbf{If } v_j^{(k)\langle l+1\rangle} + \phi_{\upsilon_j^{(k)}}^{\langle l\rangle} > 0.9c_{r_j}^{(k)}, & \textbf{then} & c_{r_j^{(k)}}^{\langle l+1\rangle} \leftarrow 10c_{\boldsymbol{r}_j^{(k)}}^{\langle l\rangle};
\end{array}
$$

Step 5. Obtain a new step from the previous step:

$$
\begin{array}{llll}
\ddot{\boldsymbol{x}}^{\langle l+1\rangle} \leftarrow \ddot{\boldsymbol{x}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\ddot{\boldsymbol{x}}^{\langle l\rangle}; & \check{\boldsymbol{x}}^{\langle l+1\rangle} \leftarrow \check{\boldsymbol{x}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\check{\boldsymbol{x}}^{\langle l\rangle};\\
\ddot{\boldsymbol{y}}^{\langle l+1\rangle} \leftarrow \ddot{\boldsymbol{y}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\ddot{\boldsymbol{y}}^{\langle l\rangle}; & \check{\boldsymbol{y}}^{\langle l+1\rangle} \leftarrow \check{\boldsymbol{y}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\check{\boldsymbol{y}}^{\langle l\rangle};\\
\boldsymbol{\nu}^{\langle l+1\rangle} \leftarrow \boldsymbol{\nu}^{\langle l\rangle} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\nu}^{\langle l\rangle}; & \boldsymbol{\vartheta}^{(k)\langle l+1\rangle} \leftarrow \boldsymbol{\vartheta}^{(k)\langle l\rangle} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\vartheta}^{(k)\langle l\rangle};\\
\boldsymbol{\gamma}^{\langle l+1\rangle} \leftarrow \boldsymbol{\gamma}^{\langle l\rangle} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\gamma}^{\langle l\rangle}; & \boldsymbol{\lambda}^{(k)\langle l\rangle} \leftarrow \boldsymbol{\lambda} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\lambda}^{(k)\langle l\rangle};\\
\boldsymbol{\psi}^{\langle l+1\rangle} \leftarrow \boldsymbol{\psi}^{\langle l\rangle} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\psi}^{\langle l\rangle}; & \boldsymbol{\upsilon}^{(k)\langle l+1\rangle} \leftarrow \boldsymbol{\upsilon}^{\langle l\rangle} + \alpha_d^{\langle l\rangle}\Delta\boldsymbol{\upsilon}^{(k)\langle l\rangle};\\
\boldsymbol{s}^{\langle l+1\rangle} \leftarrow \boldsymbol{s}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{s}^{\langle l\rangle}; & \boldsymbol{w}^{\langle l+1\rangle} \leftarrow \boldsymbol{w}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{w}^{\langle l\rangle};\\
\boldsymbol{t}^{\langle l+1\rangle} \leftarrow \boldsymbol{t}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{t}^{\langle l\rangle}; & \boldsymbol{z}^{(k)\langle l+1\rangle} \leftarrow \boldsymbol{z}^{(k)\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{z}^{(k)\langle l\rangle};\\
\boldsymbol{v}^{(k)\langle l+1\rangle} \leftarrow \boldsymbol{v}^{(k)\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{v}^{(k)\langle l\rangle}; & \boldsymbol{r}^{(k)\langle l+1\rangle} \leftarrow \boldsymbol{r}^{(k)\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{r}^{\langle l\rangle};\\
\boldsymbol{\xi}_s^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}_s^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_s^{\langle l\rangle}; & \boldsymbol{\xi}_{\boldsymbol{w}}^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_{\boldsymbol{w}}^{\langle l\rangle};\\
\boldsymbol{\xi}_{\boldsymbol{t}}^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}_{\boldsymbol{t}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_{\boldsymbol{t}}^{\langle l\rangle}; & \boldsymbol{\xi}_{z^{(k)}}^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}_{z^{(k)}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_{z^{(k)}}^{\langle l\rangle};\\
\boldsymbol{\xi}_{\boldsymbol{v}^{(k)}}^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}_{\boldsymbol{v}^{(k)}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_{\boldsymbol{v}^{(k)}}^{\langle l\rangle}; & \boldsymbol{\xi}_{\boldsymbol{r}^{(k)}}^{\langle l+1\rangle} \leftarrow \boldsymbol{\xi}_{\boldsymbol{r}^{(k)}}^{\langle l\rangle} + \alpha_p^{\langle l\rangle}\Delta\boldsymbol{\xi}_{\boldsymbol{r}^{(k)}}^{\langle l\rangle};
\end{array}
$$

Step 6. **If** (17) is satisfied within the tolerance τ, **then** STOP;
Return $(\boldsymbol{x}^{\langle l\rangle}, \boldsymbol{y}^{(k)\langle l\rangle}, \boldsymbol{s}^{\langle l\rangle}, \boldsymbol{w}^{\langle l\rangle}, \boldsymbol{t}^{\langle l\rangle}, \boldsymbol{z}^{(k)\langle l\rangle}, \boldsymbol{v}^{(k)\langle l\rangle}, \boldsymbol{r}^{(k)\langle l\rangle})$ as the primal optimal solution;
Return $(\boldsymbol{\nu}^{\langle l\rangle}, \boldsymbol{\vartheta}^{(k)\langle l\rangle}, \boldsymbol{\gamma}^{\langle l\rangle}, \boldsymbol{\psi}^{\langle l\rangle}, \boldsymbol{\lambda}^{(k)\langle l\rangle}, \boldsymbol{\upsilon}^{(k)\langle l\rangle})$ as the dual optimal solution;
Else Set $l \leftarrow l + 1$ and go to Step 1;

end

6 Outer-level algorithmic improvements

As discussed in Subsection 3.1 the concept of *fathoming by bounds* is one of pruning the branch-and-bound tree. If any optimal objective function value is not better than that of a feasible solution to Problem (6), then each node in the tree can be fathomed. As a result, it is important to find feasible solutions to Problem (6) quickly using one of the search techniques described in Subsection 3.1, as well as solutions with low objective function values, in order to obtain a significant reduction in the size of the branch-and-bound tree. Still, it is hard to find a balance between these goals, and the performance of the algorithm can vary greatly from one problem to the next.

In this part, we suggest the following approach based on the use of equilibrium constraints to find a feasible solution for binary SMINLSOCPs. The binary constraints are

$$\ddot{x}_i \in \{0,1\}, \quad \text{and} \quad \ddot{y}_j^{(k)} \in \{0,1\}, \quad \text{for } 1 \le i \le p, 1 \le j \le q, 1 \le k \le K, \tag{24}$$

which can also be rewritten as

$$\ddot{x}_i(1-\ddot{x}_i) = 0, \quad \text{and} \quad \ddot{y}_j^{(k)}(1-\ddot{y}_j^{(k)}) = 0, \quad \text{for } 1 \le i \le p, 1 \le j \le q, 1 \le k \le K, \tag{25}$$

In general, converting (24) into (25) is not recommended because the resulting problem is nonconvex, with each feasible solution leading to a local optimum of Problem (6). Consequently, if a general-purpose nonlinear programming algorithm gives a local optimum for the new problem, it can only claim to have established a feasible solution for the original problem, not necessarily the optimal solution. Moreover, the goal here is to find a solution that is feasible, using (25), which in turn may help prune the tree. The constraints in (25) can be expressed with equivalent constraints, specifically

$$-\ddot{x}_i(1-\ddot{x}_i) \ge 0, \quad \ddot{x}_i \ge 0, \quad \text{and} \quad (1-\ddot{x}_i) \ge 0, \quad \text{for } 1 \le i \le p, \tag{26}$$

and

$$\ddot{y}_j^{(k)}(1-\ddot{y}_j^{(k)}) \ge 0, \quad \ddot{y}_j^{(k)} \ge 0, \quad \text{and} \quad (1-\ddot{y}_j^{(k)}) \ge 0, \quad \text{for } \le j \le q. \tag{27}$$

When all the discrete variables are forced to be binary, Problem (7) can be viewed as an MPEC:

$$\begin{array}{lll}
\min & f(\boldsymbol{x}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(K)}) & \\
\text{s.t.} & \boldsymbol{h}_i(\boldsymbol{x}) \ge \boldsymbol{0}, & i = 1, \ldots, r, \\
& \boldsymbol{g}_j^{(k)}(\boldsymbol{x}, \boldsymbol{y}^{(k)}) \ge \boldsymbol{0}, & j = 1, \ldots, s, \quad k = 1, \ldots, K, \\
& -\ddot{x}_i(1-\ddot{x}_i) \ge 0, & i = 1, \ldots, p, \\
& -\ddot{y}_j^{(k)}(1-\ddot{y}_j^{(k)}) \ge 0, & j = 1, \ldots, q, \quad k = 1, 2, \ldots, K, \\
& 0 \le \ddot{x}_i \le 1, & i = 1, \ldots, p, \\
& 0 \le \ddot{y}_j^{(k)} \le 1, & j = 1, \ldots, q, \quad k = 1, 2, \ldots, K,
\end{array} \tag{28}$$

where Problem (28) differs from the SMINLSOCP problem (6) in only adding the constraints in (26) and (27). To obtain the reduced KKT system, we will use the same method described in the preceding development.

We denote the Lagrange multipliers of the constraints in (26) and (27) by $\boldsymbol{\pi}$ and $\boldsymbol{\theta}^{(k)}$ and their slack variables by $\tilde{\boldsymbol{s}}$ and $\tilde{\boldsymbol{z}}^{(k)}$. Following our notations in Subsection 2.1, the matrices $\ddot{X}$, $\ddot{Y}^{(k)}$, $\tilde{S}$, Π, $\tilde{Z}^{(k)}$, and $\Theta^{(k)}$ stand for the diagonal matrices whose diagonal entries are the components of the vectors $\ddot{\boldsymbol{x}}$, $\ddot{\boldsymbol{y}}^{(k)}$, $\tilde{\boldsymbol{s}}$, $\boldsymbol{\pi}$, $\tilde{\boldsymbol{z}}^{(k)}$, and $\boldsymbol{\theta}^{(k)}$, respectively, for $k = 1, 2, \ldots, K$.

Define the vectors $\tilde{\boldsymbol{\rho}}$, $\boldsymbol{\beta}_{\tilde{s}}$, $\tilde{\boldsymbol{\eta}}^{(k)}$, $\boldsymbol{\beta}_{\tilde{z}^{(k)}}$ as

$$\begin{array}{rclcrcl}
\tilde{\boldsymbol{\rho}} & \triangleq & \ddot{X}(\boldsymbol{1}-\ddot{\boldsymbol{x}}) + \tilde{\boldsymbol{s}}, & & \tilde{\boldsymbol{\eta}}^{(k)} & \triangleq & \ddot{Y}^{(k)}(\boldsymbol{1}-\ddot{\boldsymbol{y}}^{(k)}) + \tilde{\boldsymbol{z}}^{(k)}, \\
\boldsymbol{\beta}_{\tilde{s}} & \triangleq & \mu\tilde{\boldsymbol{s}}^{-1} - \boldsymbol{\pi}, & & \boldsymbol{\beta}_{\tilde{z}^{(k)}} & \triangleq & \mu\tilde{\boldsymbol{z}}^{(k)^{-1}} - \boldsymbol{\theta}^{(k)}.
\end{array}$$

The reduced KKT system to be resolved at each iteration is shown in System (29).

$$
\begin{bmatrix}
-(G_{\ddot{x}\ddot{x}}+\tilde{D}) & -G_{\ddot{x}\dot{x}} & -G_{\ddot{x}\dot{y}^{(1)}} & -G_{\ddot{x}\dot{y}^{(1)}\tilde{D}} & \cdots & -G_{\ddot{x}\dot{y}^{(K)}} & -G_{\ddot{x}\dot{y}^{(K)}} & A_{\ddot{x}}^{\mathsf{T}} & & B_{\ddot{x}}^{\mathsf{T}} & \cdots & B_{\ddot{x}}^{\mathsf{T}} \\
-G_{\dot{x}\ddot{x}} & -G_{\dot{x}\dot{x}} & -G_{\dot{x}\dot{y}^{(1)}} & -G_{\dot{x}\dot{y}^{(1)}} & \cdots & -G_{\dot{x}\dot{y}^{(K)}} & -G_{\dot{x}\dot{y}^{(K)}} & A_{\dot{x}}^{\mathsf{T}} & & B_{\dot{x}}^{\mathsf{T}} & \cdots & B_{\dot{x}}^{\mathsf{T}} \\
-G_{\dot{y}^{(1)}\ddot{x}} & -G_{\dot{y}^{(1)}\dot{x}} & -(G_{\dot{y}^{(1)}\dot{y}^{(1)}}+\tilde{H}^{(1)}) & -G_{\dot{y}^{(1)}\dot{y}^{(1)}} & \cdots & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & & & B_{\dot{y}^{(1)}}^{\mathsf{T}} & & \\
-G_{\dot{y}^{(1)}\ddot{x}} & -G_{\dot{y}^{(1)}\dot{x}} & -G_{\dot{y}^{(1)}\dot{y}^{(1)}} & -G_{\dot{y}^{(1)}\dot{y}^{(1)}} & \cdots & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & -G_{\dot{y}^{(K)}\dot{y}^{(1)}} & & & B_{\dot{y}^{(1)}}^{\mathsf{T}} & & \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \ddots & \\
-G_{\dot{x}\dot{y}^{(K)}} & -G_{\dot{x}\dot{y}^{(K)}} & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & \cdots & -(G_{\dot{y}^{(K)}\dot{y}^{(K)}}+\tilde{H}^{(K)}) & -G_{\dot{y}^{(K)}\dot{y}^{(K)}} & & & B_{\dot{y}^{(K)}} & & \\
-G_{\dot{x}\dot{y}^{(K)}} & -G_{\dot{x}\dot{y}^{(K)}} & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & -G_{\dot{y}^{(1)}\dot{y}^{(K)}} & \cdots & -G_{\dot{y}^{(K)}\dot{y}^{(K)}} & -G_{\dot{y}^{(K)}\dot{y}^{(K)}} & & & B_{\dot{y}^{(K)}} & & \\
A_{\ddot{x}}^{\mathsf{T}} & A_{\dot{x}}^{\mathsf{T}} & & & & & & & & & & \\
B_{\ddot{x}} & B_{\dot{x}} & B_{\dot{y}^{(1)}} & B_{\dot{y}^{(1)}} & \ddots & & & & & & & \\
\vdots & \vdots & & & & & & & & & & \\
B_{\ddot{x}} & B_{\dot{x}} & & & & B_{\dot{y}^{(K)}} & B_{\dot{y}^{(K)}} & & & & &
\end{bmatrix}
\begin{bmatrix}
\Delta\ddot{\boldsymbol{x}} \\
\Delta\dot{\boldsymbol{x}} \\
\Delta\dot{\boldsymbol{y}}^{(1)} \\
\Delta\dot{\boldsymbol{y}}^{(1)} \\
\vdots \\
\Delta\dot{\boldsymbol{y}}^{(K)} \\
\Delta\dot{\boldsymbol{y}}^{(K)} \\
\Delta\boldsymbol{\nu} \\
\Delta\boldsymbol{\vartheta}^{(1)} \\
\vdots \\
\Delta\boldsymbol{\vartheta}^{(K)}
\end{bmatrix}
=
\begin{bmatrix}
\boldsymbol{\sigma}_{\ddot{x}} + D_{u\ddot{x}}\hat{\boldsymbol{\varrho}} - D_t\hat{\boldsymbol{t}} - (2\hat{X} - I)(\boldsymbol{\pi} + \Pi\tilde{S}^{-1}\tilde{\boldsymbol{\rho}} + \boldsymbol{\beta}_{\tilde{s}}) \\
\boldsymbol{\sigma}_{\dot{x}} \\
\boldsymbol{\sigma}_{\dot{y}^{(1)}} + D_{v\dot{x}1}\hat{\boldsymbol{\zeta}}^{(1)} - D_{\rho\dot{x}1}\hat{\boldsymbol{\kappa}}^{(1)} - (2\hat{Y}^{(1)} - I)(\boldsymbol{\theta}^{(1)} + \Theta^{(1)}\tilde{Z}^{(1)-1}\tilde{\boldsymbol{\eta}}^{(1)} + \boldsymbol{\beta}_{\tilde{\eta}^{(1)}}) \\
\boldsymbol{\sigma}_{\dot{y}^{(1)}} \\
\vdots \\
\boldsymbol{\sigma}_{\dot{y}^{(K)}} + D_v^{(K)}\hat{\boldsymbol{\zeta}}^{(K)} - D_{\rho\dot{x}1}\hat{\boldsymbol{\kappa}}^{(K)} - (2\hat{Y}^{(K)} - I)(\boldsymbol{\theta}^{(K)} + \Theta^{(K)}\tilde{Z}^{(K)-1}\tilde{\boldsymbol{\eta}}^{(K)} + \boldsymbol{\beta}_{\tilde{\eta}^{(K)}}) \\
\boldsymbol{\sigma}_{\dot{y}^{(K)}} \\
\boldsymbol{\rho} + \mathrm{Arw}^{-1}(\boldsymbol{\nu})\boldsymbol{\beta}_{\tilde{s}} \\
\boldsymbol{\eta}^{(1)} + \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(1)})\boldsymbol{\beta}_{\tilde{z}^{(1)}} \\
\vdots \\
\boldsymbol{\eta}^{(K)} + \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(K)})\boldsymbol{\beta}_{\tilde{z}^{(K)}}
\end{bmatrix}
\tag{29}
$$

$$
\mathrm{Arw}^{-1}(\boldsymbol{\nu})\mathrm{Arw}(\boldsymbol{s}) \qquad \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(1)})\mathrm{Arw}(\boldsymbol{z}^{(1)}) \qquad \mathrm{Arw}^{-1}(\boldsymbol{\vartheta}^{(K)})\mathrm{Arw}(\boldsymbol{z}^{(K)})
$$

The matrices $\tilde{D}, \tilde{H}^{(1)}, \tilde{H}^{(2)}, \ldots, \tilde{H}^{(K)}$ that appeared in System (29) are defined as

$$\begin{array}{lcl} \tilde{D} & \triangleq & D_{\boldsymbol{w}} - D_{\boldsymbol{t}} - 2\Pi + (2\ddot{X} - I)\Pi\tilde{S}^{-1}(2\ddot{X} - I); \\ \tilde{H}^{(k)} & \triangleq & D_{\boldsymbol{v}^{(k)}} - D_{\boldsymbol{r}^{(k)}} - 2\Theta^{(k)} + (2\ddot{Y}^{(k)} - I)\Theta^{(k)}\tilde{Z}^{(k)^{-1}}(2\ddot{Y}^{(k)} - I),\ k = 1, 2, \ldots, K. \end{array}$$

We point out that (29) has a similar sparsity structure to (18). This confirms that the computational efforts to solve (28) and (7) are similar, and more importantly, that the two problems can be seamlessly swapped. For large problems, an effort equivalent to solving just one more nonlinear second-order cone relaxation can be well worth the massive reduction in the number of nodes in the branch-and-bound tree.

7 Numerical results

In order to see how the proposed algorithm in this paper operates, we have implemented it on a randomly generated mixed-integer stochastic programs over second-order cones. In fact, we have not found well-defined test problems for SMINLSOCP in the literature. However, we have constructed and solved particular problems in which the nonlinear is involved. Due to the fact that the number of SMINLSOCPs we have solved is limited, the results do not give the desired tendency. Therefore, to avoid any incautious actions, we have decided to exclude them in this numerical study. As mentioned in the introduction, both SMINLSOCP and its special case, SMISOCP, are still unsolved algorithmically. So, as an alternative, we have constructed and solved particular problems in which the objective function is linear and all constraints are also linear. Because the obtained results on SMISOCPs show the desired tendency, we have recorded them. In this section, we summarize these preliminary numerical experiments.

We obtained our numerical results using MATLAB version R2022a on Windows 11th operating system, which was carried out on a PC with Intel (R) Core (TM) i7-1165G7 at 2.80 GHz and 8 GB physical memory. As presented in Section 3, our approach is principally a branch-and-bound code that writes a continuous relaxation of the bound modified problem of the form (7) at any node in the branch-and-bound tree.

In our implementation, the random generated problem is an SMISOCP problem, in which some coefficients are randomly generated with discrete distribution. More precisely, we consider

$$\begin{array}{llllll} \min & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{x} + \mathbb{E}\left[\varphi(\boldsymbol{x}, \omega)\right] & \text{where } \varphi(\boldsymbol{x}, \omega) \text{ is} & \min & \boldsymbol{d}(\omega)^{\mathsf{T}}\boldsymbol{y} & \\ \text{s.t.} & A\boldsymbol{x} \geq \boldsymbol{0}, & \text{the minimum value} & \text{s.t.} & T(w)\boldsymbol{x} + W(\omega)\boldsymbol{y} \geq \boldsymbol{0}, & (30) \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, & \text{of the problem} & & \boldsymbol{y} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}, & \end{array}$$

and $\mathbb{E}[\varphi(\boldsymbol{x}, \omega)] = \int_\Omega \varphi(\boldsymbol{x}, \omega) P(d\omega)$. If $\pi_k = \mathbb{P}(T(\omega), W(\omega), \boldsymbol{d}(\omega)) = (T^{(k)}, W^{(k)}, d^{(k)})$ is the associated probability, for $k = 1, 2, \ldots, K$, the equivalent deterministic formulation of Problem (30) is the problem:

$$\begin{array}{llllllllllll} \min & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{x} & + & \pi_1 \boldsymbol{d}^{(1)\mathsf{T}}\boldsymbol{y}^{(1)} & + & \pi_2 \boldsymbol{d}^{(2)\mathsf{T}}\boldsymbol{y}^{(2)} & + & \cdots & + & \pi_K \boldsymbol{d}^{(K)\mathsf{T}}\boldsymbol{y}^{(K)} & & \\ \text{s.t.} & A\boldsymbol{x} & & & & & & & & & \geq & \boldsymbol{0}, \\ & T^{(1)}\boldsymbol{x} & + & W^{(1)}\boldsymbol{y}^{(1)} & & & & & & & \geq & \boldsymbol{0}, \\ & T^{(2)}\boldsymbol{x} & & & + & W^{(2)}\boldsymbol{y}^{(2)} & & & & & \geq & \boldsymbol{0}, \\ & \vdots & & & & & & \ddots & & & & \vdots \\ & T^{(K)}\boldsymbol{x} & & & & & & & + & W^{(K)}\boldsymbol{y}^{(K)} & \geq & \boldsymbol{0}, \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, & & \boldsymbol{y}^{(1)} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}, & & \boldsymbol{y}^{(2)} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}, & & \ldots & & \boldsymbol{y}^{(K)} \in \mathbb{Z}^q \times \mathbb{R}^{m-q}. & & \end{array} \quad (31)$$

We assume that the probability value π_k is absorbed by $\boldsymbol{d}^{(k)\mathsf{T}}\boldsymbol{y}^{(k)}$ (i.e., $\boldsymbol{d}^{(k)\mathsf{T}}\boldsymbol{y}^{(k)} \leftarrow \pi_k \boldsymbol{d}^{(k)\mathsf{T}}\boldsymbol{y}^{(k)}$) for $k = 1, 2, \ldots, K$. The modified continuous problem corresponding to (31) is the problem:

$$\begin{array}{llllllllllll} \min & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{x} & + & \boldsymbol{d}^{(1)\mathsf{T}}\boldsymbol{y}^{(1)} & + & \boldsymbol{d}^{(2)\mathsf{T}}\boldsymbol{y}^{(2)} & + & \cdots & + & \boldsymbol{d}^{(K)\mathsf{T}}\boldsymbol{y}^{(K)} & & \\ \text{s.t.} & A\boldsymbol{x} & & & & & & & & & \geq & \boldsymbol{0}, \\ & T^{(1)}\boldsymbol{x} & + & W^{(1)}\boldsymbol{y}^{(1)} & & & & & & & \geq & \boldsymbol{0}, \\ & T^{(2)}\boldsymbol{x} & & & + & W^{(2)}\boldsymbol{y}^{(2)} & & & & & \geq & \boldsymbol{0}, \\ & \vdots & & & & & & \ddots & & & & \vdots \\ & T^{(K)}\boldsymbol{x} & & & & & & & + & W^{(K)}\boldsymbol{y}^{(K)} & \geq & \boldsymbol{0}, \\ & \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}, & & \tilde{\boldsymbol{l}}^{(1)} \leq \boldsymbol{y}^{(1)} \leq \tilde{\boldsymbol{u}}^{(1)}, & & \tilde{\boldsymbol{l}}^{(2)} \leq \boldsymbol{y}^{(2)} \leq \tilde{\boldsymbol{u}}^{(2)}, & & \ldots & & \tilde{\boldsymbol{l}}^{(K)} \leq \boldsymbol{y}^{(K)} \leq \tilde{\boldsymbol{u}}^{(u)}. & & \end{array} \quad (32)$$

Table 1: Some input data for the SMISOCP problem (32) when $K = 5$.

A	$\mathrm{T}^{(1)}$	$\mathrm{T}^{(2)}$	$\mathrm{T}^{(3)}$	$\mathrm{T}^{(4)}$	$\mathrm{T}^{(5)}$
$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
	$\mathrm{W}^{(1)}$	$\mathrm{W}^{(2)}$	$\mathrm{W}^{(3)}$	$\mathrm{W}^{(4)}$	$\mathrm{W}^{(5)}$
	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
$\boldsymbol{c}$	$\boldsymbol{d}^{(1)}$	$\boldsymbol{d}^{(2)}$	$\boldsymbol{d}^{(3)}$	$\boldsymbol{d}^{(4)}$	$\boldsymbol{d}^{(5)}$
$\begin{pmatrix} 0.374 \\ -0.586 \\ 0.902 \end{pmatrix}$	$\begin{pmatrix} -0.670 \\ 0.175 \\ -0.250 \end{pmatrix}$	$\begin{pmatrix} 0.855 \\ 0.732 \\ 0.584 \end{pmatrix}$	$\begin{pmatrix} 0.078 \\ 0.892 \\ -0.002 \end{pmatrix}$	$\begin{pmatrix} -0.960 \\ -0.397 \\ 0.856 \end{pmatrix}$	$\begin{pmatrix} -0.523 \\ -0.482 \\ 0.203 \end{pmatrix}$

Firstly, we generate a list of numbers uniformly distributed between -1 and 1 to obtain an initialization of $\boldsymbol{x}^{\langle 0\rangle}, \boldsymbol{y}^{(k)^{\langle 0\rangle}}, \boldsymbol{s}^{\langle 0\rangle}, \boldsymbol{z}^{(k)^{\langle 0\rangle}}, \boldsymbol{\nu}^{\langle 0\rangle}, \boldsymbol{\vartheta}^{(k)^{\langle 0\rangle}}, k = 1, 2, \ldots, K$. All the slack variables and multipliers $\boldsymbol{w}^{\langle 0\rangle}, \boldsymbol{t}^{\langle 0\rangle}, \boldsymbol{v}^{(k)^{\langle 0\rangle}}, \boldsymbol{r}^{(k)^{\langle 0\rangle}}, \boldsymbol{\gamma}^{\langle 0\rangle}$, $\boldsymbol{\psi}^{\langle 0\rangle}, \boldsymbol{\lambda}^{(k)^{\langle 0\rangle}}, \boldsymbol{\upsilon}^{(k)^{\langle 0\rangle}}, k = 1, 2, \ldots, K$, are initialized with a default vector of $\mathbf{1}$. For simplicity, we consider the sample case of $K = 5$ scenarios for the SMISOCP problem (32). We consider $\pi_1 = \pi_2 = \cdots = \pi_5 = \frac{1}{5}$ as the associated probabilities, and take the matrices $A, T^{(k)}$, and $W^{(k)}$, and the vectors $\boldsymbol{c}$ and $\boldsymbol{d}^{(k)}$ for $k = 1, 2, \ldots, 5$, to be as in Table 1.

After running the generic primal-dual penalty approach, it converts (7) to (15) and solves any relaxation until all nodes are fathomed. For Problem (32) in the branch-and-bound tree, the root solution is reached after 40 iterations with CPU = 0.3125(s) and the value of the objective function is 21.920. Table 2 presents the optimal solution for (32). After obtaining this solution at the root node in the tree, we start by pruning the branch-and-bound tree using a fathoming-by-bounds algorithm, which is explained in Subsection 3.1. Any node in the tree can be fathomed if its optimal objective function value is not better than that of a feasible solution to (32). Figure 3 shows a schematic generation of a branch-and-bound tree.

While using the penalty primal-dual method, we examine the performance of our approach on warm-starting after variable bound perturbations. The importance of the primal-dual interior point method is the ability to warmstart using the solution of the initial problem (32) shown in Table 2 in order to solve a closely related (possibly perturbed) problem.

As shown in Table 3, the branch-and-bound code can solve Problem (31) in a reasonable time. We found that CPU = 1.5312(s) with the optimal objective function value 10.654.

Table 2: The optimal solution after applying the primal-dual penalty method when $K = 5$.

$\boldsymbol{x}$	$\boldsymbol{y}^{(1)}$	$\boldsymbol{y}^{(2)}$	$\boldsymbol{y}^{(3)}$	$\boldsymbol{y}^{(4)}$	$\boldsymbol{y}^{(5)}$
$\begin{pmatrix} 0.990 \\ -0.508 \\ 0.637 \end{pmatrix}$	$\begin{pmatrix} 0.597 \\ -0.468 \\ -0.052 \end{pmatrix}$	$\begin{pmatrix} 0.721 \\ 0.405 \\ -0.213 \end{pmatrix}$	$\begin{pmatrix} 0.928 \\ 0.050 \\ 0.021 \end{pmatrix}$	$\begin{pmatrix} 0.874 \\ 0.210 \\ 0.587 \end{pmatrix}$	$\begin{pmatrix} 0.746 \\ -0.465 \\ -0.036 \end{pmatrix}$
$\boldsymbol{\nu}$	$\boldsymbol{\vartheta}^{(1)}$	$\boldsymbol{\vartheta}^{(2)}$	$\boldsymbol{\vartheta}^{(3)}$	$\boldsymbol{\vartheta}^{(4)}$	$\boldsymbol{\vartheta}^{(5)}$
$\begin{pmatrix} 0.751 \\ 0.177 \\ 0.670 \end{pmatrix}$	$\begin{pmatrix} 0.941 \\ -0.240 \\ 0.589 \end{pmatrix}$	$\begin{pmatrix} -0.467 \\ -0.301 \\ -0.318 \end{pmatrix}$	$\begin{pmatrix} 0.955 \\ -0.360 \\ -0.805 \end{pmatrix}$	$\begin{pmatrix} 0.8431 \\ -0.360 \\ 0.562 \end{pmatrix}$	$\begin{pmatrix} 0.891 \\ -0.613 \\ 0.191 \end{pmatrix}$

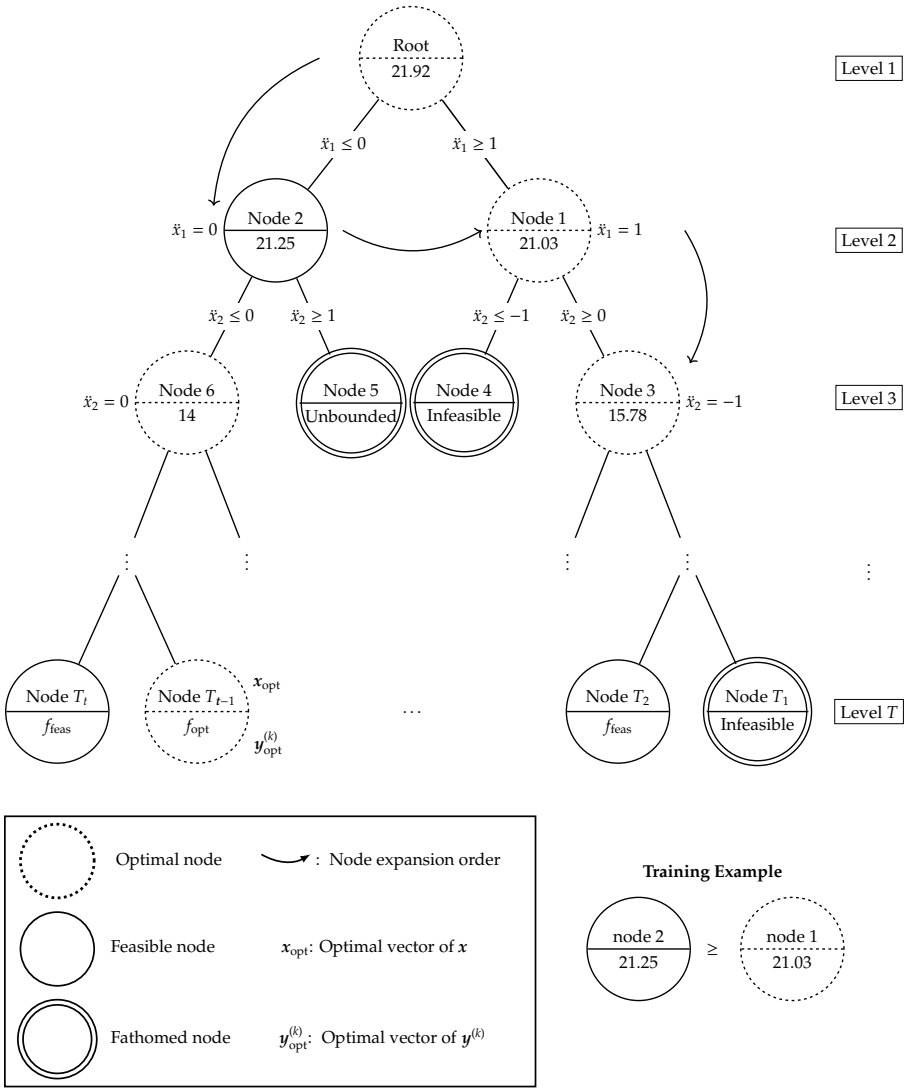

Figure 3: A schematic creation of a branch-and-bound tree while solving underlying subproblems ($K = 5$).

Table 3: The optimal solution after applying the branch-and-bound method when $K = 5$.

x	$y^{(1)}$	$y^{(2)}$	$y^{(3)}$	$y^{(4)}$	$y^{(5)}$
$\begin{pmatrix} 1.000 \\ 1.000 \\ 0.037 \end{pmatrix}$	$\begin{pmatrix} 1.000 \\ -1.000 \\ 0.020 \end{pmatrix}$	$\begin{pmatrix} 1.000 \\ 0.000 \\ 0.808 \end{pmatrix}$	$\begin{pmatrix} 1.000 \\ 1.000 \\ 0.089 \end{pmatrix}$	$\begin{pmatrix} 1.000 \\ 0.000 \\ 0.946 \end{pmatrix}$	$\begin{pmatrix} 1.000 \\ 1.000 \\ -0.003 \end{pmatrix}$
ν	$\vartheta^{(1)}$	$\vartheta^{(2)}$	$\vartheta^{(3)}$	$\vartheta^{(4)}$	$\vartheta^{(5)}$
$\begin{pmatrix} 0.997 \\ 0.554 \\ -0.822 \end{pmatrix}$	$\begin{pmatrix} 0.531 \\ -0.055 \\ 0.311 \end{pmatrix}$	$\begin{pmatrix} -0.573 \\ -0.072 \\ -0.173 \end{pmatrix}$	$\begin{pmatrix} 0.866 \\ -0.304 \\ -0.344 \end{pmatrix}$	$\begin{pmatrix} 0.8734 \\ -0.149 \\ -0.227 \end{pmatrix}$	$\begin{pmatrix} 0.467 \\ -0.286 \\ -0.263 \end{pmatrix}$

We also consider K scenarios of SMISOCPs, with $K = 5, 10, 15, 20$, and $\pi_k = \frac{1}{K}$ is the associated probability for $k = 1, 2, \ldots, K$. We take $A = I_{\ell \times n}$, $T^{(k)} = I_{d \times n}$, $W^{(k)} = I_{d \times m}$, while vectors $c \in \mathbb{R}^n$ and $d^{(k)} \in \mathbb{R}^m$ are chosen randomly for $k = 1, 2, \ldots, K$. The variables $x^{\langle 0 \rangle}$, $y^{(k)\langle 0 \rangle}$, $s^{\langle 0 \rangle}$, $z^{(k)\langle 0 \rangle}$, $\nu^{\langle 0 \rangle}$, $\vartheta^{(k)\langle 0 \rangle}$, $k = 1, 2, \ldots, K$, are initialized with identity vectors $e = (1; \mathbf{0})$ of appropriate dimensions, and variables $w^{\langle 0 \rangle}$, $t^{\langle 0 \rangle}$, $v^{(k)\langle 0 \rangle}$, $r^{(k)\langle 0 \rangle}$, $\gamma^{\langle 0 \rangle}$, $\psi^{\langle 0 \rangle}$, $\lambda^{(k)\langle 0 \rangle}$, $v^{(k)\langle 0 \rangle}$, $k = 1, 2, \ldots, K$, are initialized with default vectors of ones, i.e., $\mathbf{1}$ with appropriate dimensions.

Table 4: Numerical results after applying the proposed algorithm on randomly-generated problems.

Problem size and number of realizations							Inner-level performance		Outer-level performance	
n	m	p	q	K	$n + Km$	$p + Kq$	Iter	CPU(s)	Iter	CPU(s)
5	10	3	6	5	55	33	8.6	1.4187	4.0	1.4937
8	14	5	9	5	78	50	9.4	2.5281	6.6	1.9652
11	18	7	12	5	101	67	12.2	2.6765	9.6	2.8531
14	22	9	15	5	124	84	9.8	3.3639	8.4	4.1844
17	26	11	18	5	147	101	14.2	10.2531	12.4	12.9693
20	30	13	21	5	170	118	18.6	12.3181	15.8	16.7695
5	10	3	6	10	105	63	25.8	5.6625	41.2	6.0914
8	14	5	9	10	148	95	33.8	7.9156	46.2	16.3870
11	18	7	12	10	191	127	34.2	13.0086	48.6	19.3718
14	22	9	15	10	234	159	54.2	7.2135	62.8	22.0031
17	26	11	18	10	277	191	52.4	15.3152	45.8	35.0062
20	30	13	21	10	320	223	60.8	20.4367	56.6	43.8759
5	10	3	6	15	155	93	47.0	12.6090	69.4	18.9134
8	14	5	9	15	218	140	61.2	19.0422	73.4	30.4730
11	18	7	12	15	281	187	68.4	14.4531	60.8	21.9968
14	22	9	15	15	344	234	84.2	17.8499	100.2	35.1898
17	26	11	18	15	407	281	68.4	19.7344	91.0	40.2656
20	30	13	21	15	470	328	81.6	28.5601	100.2	56.8956
5	10	3	6	20	205	123	93.0	19.725	91.4	36.4468
8	14	5	9	20	288	185	70.8	21.2656	90.6	19.3454
11	18	7	12	20	371	247	100.0	26.9387	98.8	28.9565
14	22	9	15	20	454	309	132.2	25.5687	123.4	36.4937
17	26	11	18	20	537	371	130.4	46.6125	133.6	90.6613
20	30	13	21	20	620	344	180.2	65.9354	140.8	105.6543

The solutions are reached and the numerical results of branch-and-bound-code are recorded in Table 4. In our test problems, "CPU(s)" denotes the average CPU time (in seconds) of 5 runs recorded, and "Iter" denotes the average number of iterations of 5 runs recorded while obtaining approximate optimal solutions of the underlying problem. Figures 4 and 5 visualize the numerical results in Table 4 for $K = 5, 10, 15, 20$.

Computationally, it is not possible to update penalty parameters an infinite number of times. So, in the code, we updated the penalty parameter a certain number of times before discarding the original objective function and minimizing the infeasibility. If the penalty objective function is close to zero, the problem continues with the current solution with the largest value of the penalty parameter encountered. Otherwise, a certificate of infeasibility can be returned. We have excluded a small number of problems that are unbounded or infeasible.

For nondifferentiability in the user-supplied at the optimal solution, our results show also that a general-purpose solution works well for problems with second-order cone constraints, especially when a perturbation or reformulation method is used to deal with problems that are not smooth. The perturbation variable seems to work well for problems with small blocks, but the ratio reformulation gives a much sparser Hessian for problems with larger blocks and makes it clear how to solve them. So, we used the ratio reformulation technique described in Section 4 to smooth out the second-order cone constraints.

To sum up, the computational results in Table 4 and Figures 4 and 5 show that a hybrid algorithm coupling branch-and-bound and primal-dual interior-point methods is efficient overall. In all instances under study, we found that the number of iterations never exceeds 190, and the CPU time never exceeds 110 seconds. Also, from our results, it is clear that the average CPU time and number of iterations increase when the number of scenarios K increases.

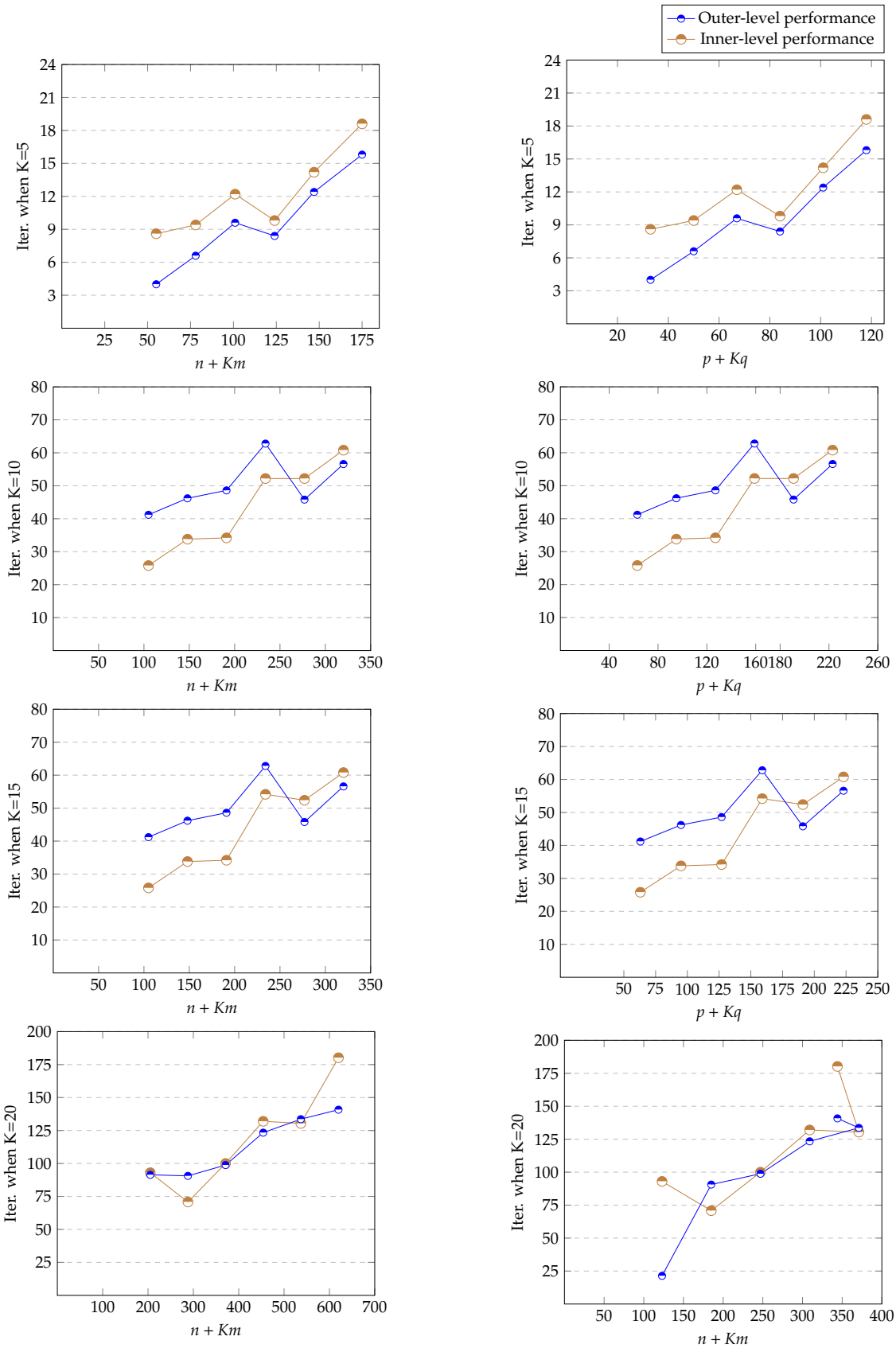

Figure 4: A representation of the average number of iterations for the inner- and outer-levels using line graphs.

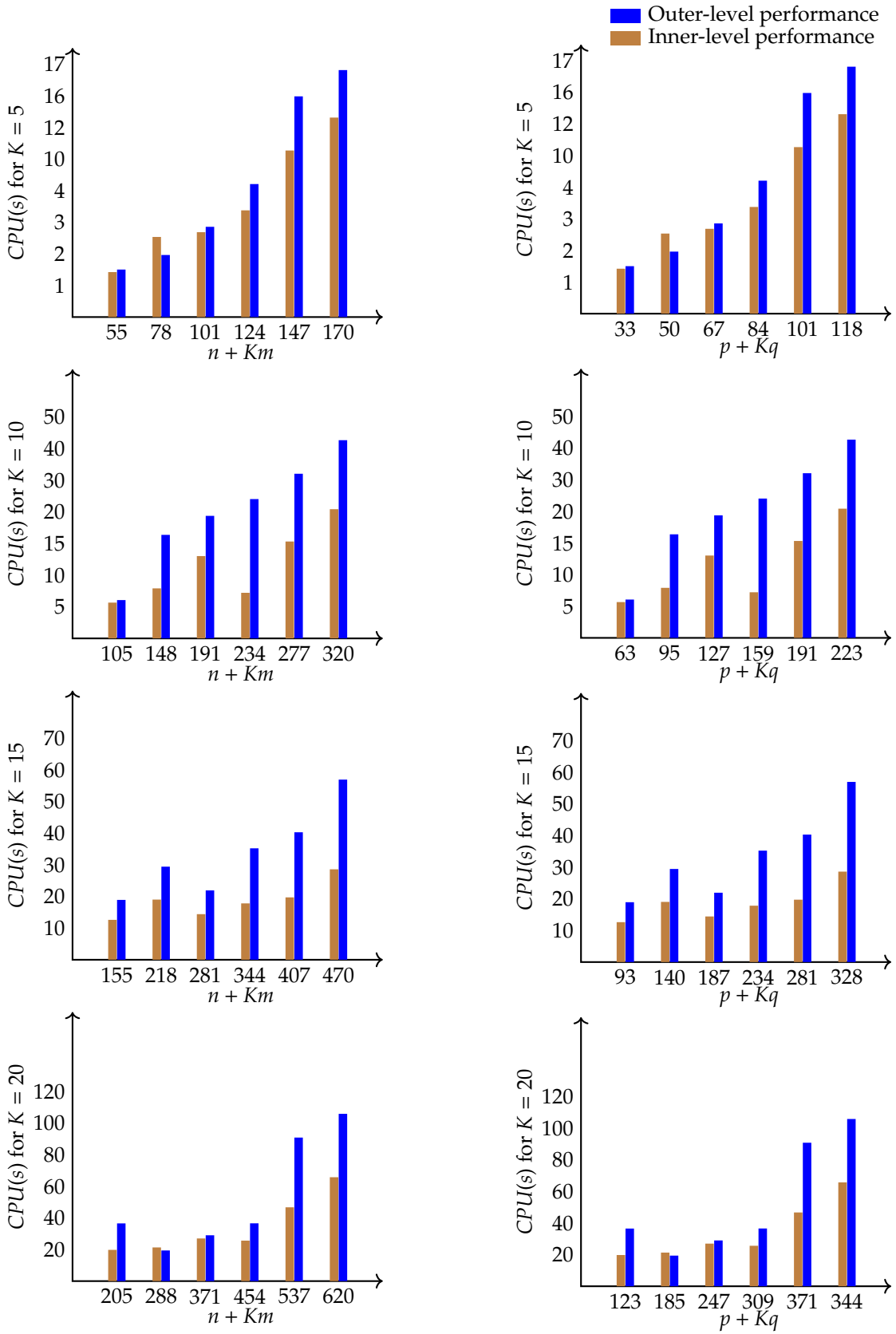

Figure 5: A representation of the average CPU(s) of inner- and outer-level iterations using bar charts.

8 Concluding remarks

Two-stage stochastic mixed-integer nonlinear second-order cone programming has many practical applications [6]. In this paper, we have developed a hybrid optimization algorithm coupling branch-and-bound and infeasible primal-dual interior-point methods for solving this class of optimization problems. To cut down on the size of the branch-and-bound tree, we have used a method based on constraints and equilibrium. We have addressed some challenging issues such as setting and updating penalty parameters as well as detecting infeasibility. To overcome nondifferentiability at optimality, we have also described a reformulation of the problem with second-order cone constraints in order to solve it using a general-purpose interior-point algorithm for nonlinear programming. The proposed algorithm has also been implemented to data and has proven its efficiency. Although this framework is attractive from the decomposition and sparsity point of views (especially for large problems), it should be incorporated within further work to develop and enhance a deeper level of implementations. This includes introducing heuristics for generating feasible solutions and developing a truly meaningful test suite for warmstarting problems. We commend the interested researchers for continuing the work on this important class of mixed-integer optimization problems to address such practical implementation issues in the proposed algorithm and to look into other algorithms to solve them.

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.
Data availability Data sharing not applicable to this article as no datasets were generated/analyzed during this study.

References

1. B. Alzalg. Decomposition-based interior point methods for stochastic quadratic second-order cone programming. *Applied Mathematics and Computation*, 249:1–18, 2014.
2. B. Alzalg. Homogeneous self-dual algorithms for stochastic second-order cone programming. *Journal of Optimization Theory and Applications*, 163(1):148–164, 2014.
3. B. Alzalg. Volumetric barrier decomposition algorithms for stochastic quadratic second-order cone programming. *Applied Mathematics and Computation*, 265:494–508, 2015.
4. B. Alzalg. A primal-dual interior-point method based on various selections of displacement step for symmetric optimization. *Computational Optimization and Applications*, 72(2):363–390, 2019.
5. B. Alzalg. A logarithmic barrier interior-point method based on majorant functions for second-order cone programming. *Optimization Letters*, 14(3):729–746, 2020.
6. B. Alzalg and H. Alioui. Applications of stochastic mixed-integer second-order cone optimization. *IEEE Access*, 10:3522–3547, 2021.
7. B. Alzalg and K. Ariyawansa. Logarithmic barrier decomposition-based interior point methods for stochastic symmetric programming. *Journal of Mathematical Analysis and Applications*, 409(2):973–995, 2014.
8. B. Alzalg, K. Badarneh, and A. Ababneh. An infeasible interior-point algorithm for stochastic second-order cone optimization. *Journal of Optimization Theory and Applications*, 181(1):324–346, 2019.
9. A. Atamtürk and V. Narayanan. Cuts for conic mixed-integer programming. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 16–29. Springer, 2007.
10. H. Y. Benson. Mixed integer nonlinear programming using interior-point methods. *Optimization Methods and Software*, 26(6):911–931, 2011.
11. H. Y. Benson, A. Sen, D. F. Shanno, and R. J. Vanderbei. Interior-point algorithms, penalty methods and equilibrium problems. *Computational Optimization and Applications*, 34(2):155–182, 2006.

12. H. Y. Benson and D. F. Shanno. An exact primal–dual penalty method approach to warmstarting interior-point methods for linear programming. *Computational Optimization and Applications*, 38(3):371–399, 2007.

13. H. Y. Benson and D. F. Shanno. Interior-point methods for nonconvex nonlinear programming: regularization and warmstarts. *Computational Optimization and Applications*, 40(2):143–189, 2008.

14. H. Y. Benson, D. F. Shanno, and R. J. Vanderbei. A comparative study of large-scale nonlinear optimization algorithms. In *High performance algorithms and software for nonlinear optimization*, pages 95–127. Springer, 2003.

15. H. Y. Benson and R. J. Vanderbei. Solving problems with semidefinite and related constraints using interior-point methods for nonlinear programming. *Mathematical Programming*, 95(2):279–302, 2003.

16. H. Y. Benson, R. J. Vanderbei, and D. F. Shanno. Interior-point methods for nonconvex nonlinear programming: Filter methods and merit functions. *Computational Optimization and Applications*, 23(2):257–272, 2002.

17. C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83(1):451–464, 1998.

18. M. T. Cezik and G. Iyengar. Cuts for mixed 0-1 conic programming. *Mathematical Programming*, 104(1):179–202, 2005.

19. S. Drewes and S. Ulbrich. *Mixed integer second order cone programming*. Verlag Dr. Hut Germany, 2009.

20. J. Faraut. Analysis on symmetric cones. *Oxford mathematical monographs*, 1994.

21. D. Gade, G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets, and D. L. Woodruff. Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. *Mathematical Programming*, 157(1):47–67, 2016.

22. D. Gade, S. Küçükyavuz, and S. Sen. Decomposition algorithms with parametric gomory cuts for two-stage stochastic integer programs. *Mathematical Programming*, 144(1):39–64, 2014.

23. P. E. Gill, W. Murray, and M. A. Saunders. User's guide for snopt 5.3: A fortran package for large-scale nonlinear programming. 1997.

24. J. Lee and S. Leyffer. *Mixed integer nonlinear programming*, volume 154. Springer Science & Business Media, 2011.

25. Y. Qi and S. Sen. The ancestral benders' cutting plane algorithm with multi-term disjunctions for mixed-integer recourse decisions in stochastic programming. *Mathematical Programming*, 161(1):193–235, 2017.

26. S. Sen. Stochastic mixed-integer programming algorithms: Beyond benders' decomposition. 2011.

27. S. Sen and J. L. Higle. The c 3 theorem and a d 2 algorithm for large scale stochastic mixed-integer programming: Set convexification. *Mathematical Programming*, 104(1):1–20, 2005.

28. S. Sen and H. D. Sherali. Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming. *Mathematical Programming*, 106(2):203–223, 2006.

29. H. D. Sherali and B. M. Fraticelli. A modification of benders' decomposition algorithm for discrete subproblems: An approach for stochastic programs with integer recourse. *Journal of Global Optimization*, 22(1):319–342, 2002.

30. R. J. Vanderbei and D. F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13(1):231–252, 1999.

31. J. P. Vielma, S. Ahmed, and G. L. Nemhauser. A lifted linear programming branch-and-bound algorithm for mixed-integer conic quadratic programs. *INFORMS Journal on Computing*, 20(3):438–450, 2008.

32. L. A. Wolsey and G. L. Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

33. M. Zhang and S. Kucukyavuz. Finitely convergent decomposition algorithms for two-stage stochastic pure integer programs. *SIAM Journal on Optimization*, 24(4):1933–1951, 2014.

34. G. Zhao. A lagrangian dual method with self-concordant barriers for multi-stage stochastic convex programming. *Mathematical Programming*, 102(1):1–24, 2005.