

The CVXOPT cone linear program solver

L. Vandenberghe

August 2009; updated September 12, 2009

1 Introduction

This document describes the primal-dual interior-point algorithm used in the `conelp` solver of CVXOPT version 1.1.1 and some details of its implementation.

The algorithm is for a pair of primal and dual cone linear programs (cone LPs)

$$\begin{array}{ll}
 \text{P:} & \text{minimize} & c^T x \\
 & \text{subject to} & Gx + s = h \\
 & & Ax = b \\
 & & s \succeq 0 \\
 \text{D:} & \text{maximize} & -h^T z - b^T y \\
 & \text{subject to} & G^T z + A^T y + c = 0 \\
 & & z \succeq 0.
 \end{array} \tag{1}$$

The inequalities $s \succeq 0$, $z \succeq 0$ are generalized inequalities with respect to a self-dual convex cone C . We restrict C to be a Cartesian product $C = C_1 \times C_2 \times \cdots \times C_K$, where each cone C_k can be a nonnegative orthant, second-order cone, or positive semidefinite cone.

Notation We will often represent symmetric matrices as vectors that contain the lower triangular entries of the matrix. This operation is denoted **vec**: if $U \in \mathbf{S}^p$ (the symmetric matrices of order p), then

$$\mathbf{vec}(U) = (U_{11}, \sqrt{2}U_{21}, \dots, \sqrt{2}U_{p1}, U_{22}, \sqrt{2}U_{32}, \dots, \sqrt{2}U_{p2}, \dots, U_{p-1,p-1}, \sqrt{2}U_{p,p-1}, U_{pp}).$$

The scaling of the off-diagonal entries ensures that inner products are preserved, *i.e.*, $\mathbf{tr}(UV) = \mathbf{vec}(U)^T \mathbf{vec}(V)$ for all U, V . The inverse operation is denoted **mat**: if u is a vector of length $p(p+1)/2$, then

$$\mathbf{mat}(u) = \begin{bmatrix} u_1 & u_2/\sqrt{2} & \cdots & u_p/\sqrt{2} \\ u_2/\sqrt{2} & u_{p+1} & \cdots & u_{2p-1}/\sqrt{2} \\ \vdots & \vdots & & \vdots \\ u_p/\sqrt{2} & u_{2p-1}/\sqrt{2} & \cdots & u_{p(p+1)/2} \end{bmatrix}.$$

The scaling ensures that $u^T v = \mathbf{tr}(\mathbf{mat}(u) \mathbf{mat}(v))$.

The image of \mathbf{S}_+^p (the positive semidefinite matrices of order p) under the **vec** operation is denoted \mathcal{S}_p :

$$\mathcal{S}_p = \{\mathbf{vec}(U) \mid U \in \mathbf{S}_+^p\} = \{u \in \mathbf{R}^{p(p+1)/2} \mid \mathbf{mat}(u) \succeq 0\}.$$

The second-order cone in \mathbf{R}^p is denoted

$$\mathcal{Q}_p = \{(u_0, u_1) \in \mathbf{R}^p \mid \|u_1\|_2 \leq u_0\}.$$

The notation \mathbf{R}_+^p is used for the cone of nonnegative p -vectors.

Throughout these notes we define *degree* m of the cone C in (1) as

$$m = m_1 + \cdots + m_K, \quad m_k = \begin{cases} p & C_k = \mathbf{R}_+^p \\ 1 & C_k = \mathcal{Q}_p \\ p & C_k = \mathcal{S}_p. \end{cases} \quad (2)$$

2 Self-dual embedding

The algorithm is based on a self-dual reformulation of the cone LPs [YTM94, dKRT97]. In this section we first describe a homogeneous embedding, and explain how it can be used to detect primal and dual infeasibility. We then give a slightly larger extended embedding that has the advantage of being strictly feasible.

2.1 Homogeneous self-dual embedding

The primal and dual cone LPs can be embedded in a self-dual cone LP

$$\begin{aligned} & \text{minimize} && 0 \\ & \text{subject to} && \begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix} \\ & && (s, \kappa, z, \tau) \succeq 0. \end{aligned} \quad (3)$$

This problem is always feasible, since $(s, \kappa, x, y, z, \tau) = 0$ is a feasible point. Moreover any feasible point is optimal. We also note that the equality constraint implies that

$$s^T z + \kappa \tau = \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix}^T \begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix}^T \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix} = 0$$

at all feasible points. In particular, this shows that there are no strictly feasible points.

Now suppose $(s, \kappa, x, y, z, \tau)$ is a solution of (3) with $\kappa + \tau > 0$.

- If $\tau > 0$, $\kappa = 0$, we can divide x, y, z by τ to obtain a solution of the Karush-Kuhn-Tucker (KKT) conditions for (1),

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix} = 0 \quad (s, z) \succeq 0, \quad z^T s = 0. \quad (4)$$

- If $\tau = 0$, $\kappa > 0$, then $h^T z + b^T y + c^T x < 0$, so we must have $h^T z + b^T y < 0$ or $c^T x < 0$ or both. If $h^T z + b^T y < 0$, this provides a proof of primal infeasibility, since

$$G^T z + A^T y = 0, \quad z \succeq 0, \quad h^T z + b^T y < 0. \quad (5)$$

If $c^T x < 0$, this provides a proof of dual infeasibility, since

$$Gx + s = 0, \quad Ax = 0, \quad s \succeq 0, \quad c^T x < 0. \quad (6)$$

If $\tau = \kappa = 0$, no conclusion can be made about (1).

2.2 Extended self-dual embedding

As an extension, we can define another self-dual cone LP

$$\begin{aligned}
& \text{minimize} && (m+1)\theta \\
& \text{subject to} && \begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T & c & q_x \\ -A & 0 & 0 & b & q_y \\ -G & 0 & 0 & h & q_z \\ -c^T & -b^T & -h^T & 0 & q_\tau \\ -q_x^T & -q_y^T & -q_z^T & -q_\tau & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ m+1 \end{bmatrix} \\
& && (s, \kappa, z, \tau) \succeq 0.
\end{aligned} \tag{7}$$

Here m is the degree of the cone, defined in (2), and

$$\begin{bmatrix} q_x \\ q_y \\ q_z \\ q_\tau \end{bmatrix} = \frac{m+1}{s_0^T z_0 + 1} \left(\begin{bmatrix} 0 \\ 0 \\ s_0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} \right) \tag{8}$$

where x_0, s_0, y_0, z_0 can be chosen arbitrarily with $(s_0, z_0) \succ 0$. This LCP is always strictly feasible and

$$(s, \kappa, x, y, z, \tau, \theta) = (s_0, 1, x_0, y_0, z_0, 1, \frac{s_0^T z_0 + 1}{m+1})$$

is a strictly feasible point. By taking the inner product of both sides of the equality constraint in (7) with (x, y, z, τ, θ) we see that the constraint implies that

$$\theta = \frac{s^T z + \kappa \tau}{m+1}, \tag{9}$$

so $\theta \geq 0$ for all feasible points.

It is easily verified that (7) is self-dual, *i.e.*, its dual problem is formally the same (if we change the objective to a maximization). Therefore, at optimum the solution must satisfy a complementarity condition with itself, and we can write the optimality conditions for (7) as

$$\begin{aligned}
\begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \\ 0 \end{bmatrix} &= \begin{bmatrix} 0 & A^T & G^T & c & q_x \\ -A & 0 & 0 & b & q_y \\ -G & 0 & 0 & h & q_z \\ -c^T & -b^T & -h^T & 0 & q_\tau \\ -q_x^T & -q_y^T & -q_z^T & -q_\tau & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ m+1 \end{bmatrix}, & \tag{10} \\
& (s, \kappa, z, \tau) \succeq 0, \quad z^T s + \kappa \tau = 0. & \tag{11}
\end{aligned}$$

Combined with (9), this implies that at the optimum $\theta = 0$ and the extended embedding reduces to the homogeneous embedding. If $(s, \kappa, x, y, z, \tau, \theta)$ is an optimal solution with $\kappa + \tau > 0$, we can therefore extract from it an optimal solution of (1), or a proof of primal or dual infeasibility.

for all $v \succ 0$, $w \succ 0$. Thus, $H(H(v)^{-1}w)^{1/2}$ is the symmetric square root of $H(v)H(w)H(v)$. The identity is straightforward in the case of the nonnegative orthant and the positive semidefinite cone, so we prove it only for the second-order cone $C_k = \mathcal{Q}_p$. First, with $u_k = H_k(v_k)^{-1}w_k$,

$$\begin{aligned} u_k &= 2(v_k^T w_k)v_k - (v_k^T Jv_k)Jw_k \\ u_k^T J u_k &= 4(v_k^T w_k)^2 v_k^T Jv_k + (v_k^T Jv_k)^2 w_k^T Jw_k - 4(v_k^T Jv_k)(v_k^T w_k)^2 \\ &= (v_k^T Jv_k)^2 w_k^T Jw_k. \end{aligned}$$

Hence

$$\begin{aligned} H_k(u_k)^{-1} &= 2u_k u_k^T - (u_k^T J u_k)J \\ &= 8(v_k^T w_k)^2 v_k v_k^T - 4(v_k^T w_k)(v_k^T Jv_k)(v_k w_k^T J + Jw_k v_k^T) + 2(v_k^T Jv_k)Jw_k w_k^T J \\ &\quad - (v_k^T Jv_k)^2 (w_k^T Jw_k)J \\ &= (2v_k v_k^T - (v_k^T Jv_k)J)(2w_k w_k^T - (w_k^T Jw_k)J)(2v_k v_k^T - (v_k^T Jv_k)J) \\ &= H_k(v_k)^{-1} H_k(w_k)^{-1} H_k(v_k)^{-1}. \end{aligned}$$

4 The central path

4.1 Definition

The central path of (7) is defined as the solution of

$$\begin{aligned} \begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \\ 0 \end{bmatrix} &= \begin{bmatrix} 0 & A^T & G^T & c & q_x \\ -A & 0 & 0 & b & q_y \\ -G & 0 & 0 & h & q_z \\ -c^T & -b^T & -h^T & 0 & q_\tau \\ -q_x^T & -q_y^T & -q_z^T & -q_\tau & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ m+1 \end{bmatrix}, & (13) \\ (s, \kappa, z, \tau) \succ 0, \quad z = -\mu g(s), \quad \tau = \mu/\kappa & (14) \end{aligned}$$

where μ is a nonnegative parameter. It follows from the equalities in (14) and the property $s^T g(s) = -m$ that $\mu = (s^T z + \kappa\tau)/(m+1)$. By taking the inner product with (x, y, z, τ, θ) on both sides of the equality (13) we also see that $\theta = \mu$ at points on the central path. We can therefore parametrize the central path more simply as

$$\begin{aligned} \begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \end{bmatrix} &= \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix} + \mu \begin{bmatrix} q_x \\ q_y \\ q_z \\ q_\tau \end{bmatrix} & (15) \\ (s, \kappa, z, \tau) \succ 0, \quad z = -\mu g(s), \quad \tau = \mu/\kappa. & (16) \end{aligned}$$

The last equality in (13) was dropped because it is redundant: by taking the inner product of both sides of (15) with $(0, 0, z, \tau)$, we get

$$z^T s + \tau\kappa = \mu(q_x^T x + q_y^T y + q_z^T z + q_\tau\tau),$$

and hence the last equation in (13).

We will use (15)–(16) to parametrize the central path. Alternatively, we can interpret (15)–(16) as a nonstandard definition of the central path for the homogeneous embedding (3).

4.2 Symmetrization

We have $z_k = -\mu g_k(s_k)$ if and only if $s_k \circ z_k = \mu \mathbf{e}_k$, where

$$u \circ v = \begin{cases} (u_1 v_1, \dots, u_p v_p) & C_k = \mathbf{R}_+^p \\ (u^T v, u_0 v_1 + v_0 u_1) & C_k = \mathcal{Q}_p \\ (1/2) \mathbf{vec}(\mathbf{mat}(u) \mathbf{mat}(v) + \mathbf{mat}(v) \mathbf{mat}(u)) & C_k = \mathcal{S}_p \end{cases}$$

and

$$\mathbf{e}_k = \begin{cases} (1, 1, \dots, 1) & C_k = \mathbf{R}_+^p \\ (1, 0, \dots, 0) & C_k = \mathcal{Q}_p \\ \mathbf{vec}(I_p) & C_k = \mathcal{S}_p. \end{cases}$$

Note that $\mathbf{e}^T(u \circ v) = u^T v$.

The condition $z = -\mu g(s)$ in the definition of the central path can therefore be replaced by the symmetric relation $s \circ z = \mu \mathbf{e}$ where $s \circ z = (s_1 \circ z_1, \dots, s_K \circ z_K)$, $\mathbf{e} = (\mathbf{e}_1, \dots, \mathbf{e}_K)$. This gives

$$\begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix} + \mu \begin{bmatrix} q_x \\ q_y \\ q_z \\ q_\tau \end{bmatrix} \quad (17)$$

$$(s, \kappa, z, \tau) \succ 0, \quad z \circ s = \mu \mathbf{e}, \quad \kappa \tau = \mu \quad (18)$$

as a symmetric equivalent of the central path equations (15)–(16).

4.3 Inverse, square, and square root

We give some useful properties of certain powers associated with the \circ -product.

The inverse, square, and square root of u are defined by the relations $u^{-1} \circ u = \mathbf{e}$, $u^2 = u \circ u$, $u^{1/2} \circ u^{1/2} = u$. For $C_k = \mathbf{R}_+^p$ these operations are the componentwise vector operations. For $C_k = \mathcal{S}_p$ they are given by the matrix inverse, square, and symmetric square root. For $C_k = \mathcal{Q}_p$, we have

$$u_k^{-1} = \frac{1}{u_k^T J u_k} J u_k, \quad u_k^2 = \begin{bmatrix} u_k^T u_k \\ 2u_{k0} u_{k1} \end{bmatrix}, \quad u_k^{1/2} = \frac{1}{\sqrt{2(u_{k0} + \sqrt{u_k^T J u_k})}} \begin{bmatrix} u_{k0} + \sqrt{u_k^T J u_k} \\ u_{k1} \end{bmatrix}.$$

Note that

$$(u_k^{-1})^T J u_k^{-1} = (u_k^T J u_k)^{-1}, \quad (u_k^{1/2})^T J u_k^{1/2} = (u_k^T J u_k)^{1/2}.$$

The following general properties of the logarithmic barrier are also useful.

$$H(u)^{-1} = H(u^{-1}), \quad H(u)^{1/2} = H(u^{1/2}), \quad (19)$$

$$H(u)u = u^{-1}, \quad H(u^{1/2})u = \mathbf{e}, \quad (H(u)v)^{-1} = H(u)^{-1}v^{-1}. \quad (20)$$

5 Nesterov-Todd scaling

A primal-dual scaling W is a linear transformation

$$\tilde{s} = W^{-T}s, \quad \tilde{z} = Wz$$

that leaves the cone invariant and preserves the central path, *i.e.*,

$$s \succ 0 \iff \tilde{s} \succ 0, \quad z \succ 0 \iff \tilde{z} \succ 0, \quad s \circ z = \mu \mathbf{e} \iff \tilde{s} \circ \tilde{z} = \mu \mathbf{e}.$$

If W is a scaling we can write the central path equations (17)–(18) equivalently as

$$\begin{bmatrix} 0 \\ 0 \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \tau \end{bmatrix} + \mu \begin{bmatrix} q_x \\ q_y \\ q_z \\ q_\tau \end{bmatrix} \quad (21)$$

$$(s, \kappa, z, \tau) \succ 0, \quad (Wz) \circ (W^{-T}s) = \mu \mathbf{e}, \quad \kappa\tau = \mu \quad (22)$$

Interior-point algorithms are based on linearizing these equations, using a primal-dual scaling that changes at each iteration depending on the values of the current iterates $(\hat{s}, \hat{\kappa}), (\hat{z}, \hat{\tau})$.

5.1 Definition

The *Nesterov-Todd scaling* at \hat{s}, \hat{z} is derived from the unique scaling point $w \in C = C_1 \times \cdots \times C_K$ that satisfies

$$H(w)\hat{s} = \hat{z}$$

[NT97, NT98]. A general expression for the scaling point is

$$w = H(\hat{s}^{-1/2}) \left(H(\hat{s}^{-1/2})\hat{z} \right)^{-1/2} = H(\hat{z}^{1/2}) \left(H(\hat{z}^{-1/2})\hat{s} \right)^{1/2}. \quad (23)$$

To see this, define $u = H(\hat{s}^{-1/2})\hat{z}$. From (12) and (19)–(20),

$$\begin{aligned} H(w)\hat{s} &= H(\hat{s}^{1/2})H(u^{-1/2})H(\hat{s}^{1/2})\hat{s} \\ &= H(\hat{s}^{1/2})H(u^{-1/2})\mathbf{e} \\ &= H(\hat{s}^{1/2})u \\ &= \hat{z}. \end{aligned}$$

A Nesterov-Todd scaling is obtained by factoring $H(w)$ as $H(w)^{-1} = W^T W$, where W is a scaling matrix. Hence $W\hat{z} = W^{-T}\hat{s}$ and we will denote this vector as λ :

$$\lambda = W^{-T}\hat{s} = W\hat{z}.$$

Note that there may exist more than one suitable factorization $H(w)^{-1} = W^T W$. For example, for the second-order and semidefinite cones, one can choose the symmetric scaling $W = H(w)^{-1/2} = H(w^{-1/2})$, or a nonsymmetric scaling.

5.2 Nonnegative orthant

Scaling Any positive diagonal matrix W_k can be used as a scaling for $C_k = \mathbf{R}_+^p$.

Nesterov-Todd scaling point The Nesterov-Todd scaling point at \hat{s}_k, \hat{z}_k is

$$w_k = \hat{s}_k^{1/2} \circ \hat{z}_k^{-1/2}.$$

Nesterov-Todd scaling The Nesterov-Todd scaling is

$$W_k = \mathbf{diag}(w_k) = \mathbf{diag}(\hat{s}_k^{1/2} \circ \hat{z}_k^{-1/2}).$$

The scaled variable $\lambda_k = W_k^{-1} \hat{s}_k = W_k \hat{z}_k$ is

$$\lambda_k = \hat{z}_k^{1/2} \circ \hat{s}_k^{1/2}$$

and satisfies $\lambda_k^T \lambda_k = \hat{s}_k^T \hat{z}_k$.

We use the same scaling for the last (scalar) block in the embedding:

$$\tilde{\kappa} = (\hat{\tau}/\hat{\kappa})^{1/2} \kappa, \quad \tilde{\tau} = (\hat{\kappa}/\hat{\tau})^{1/2} \tau.$$

5.3 Second-order cone

Scaling Any matrix W_k that satisfies

$$W_k J W_k^T = \beta^2 J \tag{24}$$

where $\beta \neq 0$, can be used as scaling matrix for $C_k = \mathcal{Q}_p$. (The matrix $(1/\beta)W_k$ is sometimes called a *hypernormal matrix* [RS88].) Note that W_k is necessarily nonsingular, and

$$W_k^T J W_k = J W_k^{-1} (W_k J W_k^T) J W_k = \beta^2 J. \tag{25}$$

Examples of symmetric scaling matrices are Hessians $H_k(u)$ or inverse Hessians $H_k(u)^{-1}$ of the second-order cone barrier (in this case $\beta = 1/(u^T J u)$, resp., $\beta = u^T J u$). The matrix

$$\frac{1}{u^T J u} H_k^{-1}(u) = \frac{2}{u^T J u} u u^T - J$$

is also called a *hyperbolic Householder matrix* [RS88]. A product of scaling matrices (for example, $W_k J$), is also a (generally nonsymmetric) scaling matrix.

We now verify that if W_k satisfies (24), then the second-order cone and the central path are preserved under multiplication with W_k . Let \mathbf{e}_k be the first unit vector and $v = W_k^T \mathbf{e}_k = (v_0, v_1)$ the first row of W_k . This is a nonnegative vector since, from (24),

$$v^T J v = (W_k^T \mathbf{e}_k)^T J (W_k^T \mathbf{e}_k) = \mathbf{e}_k^T W_k J W_k^T \mathbf{e}_k = \beta^2 \geq 0.$$

Suppose $x = (x_0, x_1) \in \mathcal{Q}_p$ and $\tilde{x} = W_k x$. Then

$$\tilde{x}_0 = v^T x = v_0 x_0 + v_1^T x_1 \geq v_0 x_0 - \|v_1\|_2 \|x_1\|_2 \geq 0$$

by the Cauchy-Schwarz inequality, and

$$\tilde{x}^T J \tilde{x} = x^T W_k^T J W_k x = \beta^2 x^T J x \geq 0.$$

Conversely, using (25) we see that if \tilde{x} is in the second-order cone, then $x = W_k^{-1}\tilde{x}$ is also in the second-order cone. A similar argument shows that multiplications with W_k^T and W_k^{-T} preserve the second-order cone. Furthermore, if z_k and s_k are on the central path, *i.e.*,

$$z_k = -\mu g_k(s_k) = \frac{\mu}{s_k^T J s_k} J s_k,$$

then $\tilde{z}_k = W_k z_k$, $\tilde{s}_k = W_k^{-T} s_k$ are on the transformed central path, with the same parameter μ :

$$\tilde{z}_k = \frac{\mu}{\tilde{s}_k^T J \tilde{s}_k} W_k J W_k^T \tilde{s}_k = \frac{\mu}{\tilde{s}_k^T J \tilde{s}_k} J \tilde{s}_k = -\mu g_k(\tilde{s}_k).$$

Nesterov-Todd scaling point The Nesterov-Todd scaling point w_k is uniquely defined by

$$H(w_k)^{-1} \hat{z}_k = \hat{s}_k.$$

Let \bar{z}_k and \bar{s}_k be the normalized vectors

$$\bar{z}_k = \frac{1}{(\hat{z}_k^T J \hat{z}_k)^{1/2}} \hat{z}_k, \quad \bar{s}_k = \frac{1}{(\hat{s}_k^T J \hat{s}_k)^{1/2}} \hat{s}_k,$$

and define

$$\gamma = \left(\frac{1 + \bar{z}_k^T \bar{s}_k}{2} \right)^{1/2}, \quad \bar{w}_k = \frac{1}{2\gamma} (\bar{s}_k + J \bar{z}_k). \quad (26)$$

We have

$$\bar{w}_k^T J \bar{w}_k = 1, \quad \bar{w}_k^T \bar{z}_k = \bar{w}_k^T J \bar{s}_k = \gamma.$$

From this it is easy to see that

$$\left(2\bar{w}_k \bar{w}_k^T - J \right) \bar{z}_k = \bar{s}_k, \quad \left(2J \bar{w}_k \bar{w}_k^T J - J \right) \bar{s}_k = \bar{z}_k.$$

In other words the hyperbolic Householder transformation defined by \bar{w}_k maps \bar{z}_k to \bar{s}_k .

In terms of the unnormalized variables, this means that if we define

$$w_k^T J w_k = \left(\frac{\hat{s}_k^T J \hat{s}_k}{\hat{z}_k^T J \hat{z}_k} \right)^{1/2}, \quad w_k = \left(w_k^T J w_k \right)^{1/2} \bar{w}_k,$$

then

$$H(w_k)^{-1} \hat{z}_k = \left(2w_k w_k^T - (w_k^T J w_k) J \right) \hat{z}_k = (w_k^T J w_k) \left(2\bar{w}_k \bar{w}_k^T - J \right) \hat{z}_k = \hat{s}_k.$$

Symmetric Nesterov-Todd scaling Let \bar{w}_k be as in (26), and define

$$v_k = \bar{w}_k^{1/2} = \frac{1}{(2(\bar{w}_k^0 + 1))^{1/2}} (\bar{w}_k + \mathbf{e}_k).$$

We have $v_k^T J v_k = 1$, so the matrices

$$\bar{W}_k = 2v_k v_k^T - J, \quad \bar{W}_k^{-1} = 2J v_k v_k^T J - J$$

are hyperbolic Householder matrices. More explicitly, written in terms of \bar{w}_k ,

$$\bar{W}_k = \begin{bmatrix} \bar{w}_{k0} & \bar{w}_{k1}^T \\ \bar{w}_{k1} & I + (\bar{w}_{k0} + 1)^{-1} \bar{w}_{k1} \bar{w}_{k1}^T \end{bmatrix}, \quad \bar{W}_k^{-1} = \begin{bmatrix} \bar{w}_{k0} & -\bar{w}_{k1}^T \\ -\bar{w}_{k1} & I + (\bar{w}_{k0} + 1)^{-1} \bar{w}_{k1} \bar{w}_{k1}^T \end{bmatrix}.$$

\bar{W}_k is the Householder transformation that maps $J\bar{w}_k$ to \mathbf{e}_k , and therefore

$$\bar{W}_k(2J\bar{w}_k\bar{w}_k^T J - J)\bar{W}_k = (2\mathbf{e}_k\mathbf{e}_k^T - J) = I.$$

In other words, $\bar{W}_k = (2\bar{w}_k\bar{w}_k^T - J)^{1/2}$. In terms of the unnormalized variables, $H(w_k)^{-1} = W_k^T W_k$ where

$$W_k = (w_k^T J w_k)^{1/2} \bar{W}_k = \begin{pmatrix} \hat{s}_k^T J \hat{s}_k \\ \hat{z}_k^T J \hat{z}_k \end{pmatrix}^{1/4} \bar{W}_k.$$

To find expressions for the scaled variables, we define

$$\bar{\lambda}_k = \bar{W}_k \bar{z}_k = \bar{W}_k^{-1} \bar{s}_k = J \bar{W}_k J \bar{s}_k.$$

We have $\bar{\lambda}_k^T J \bar{\lambda}_k = 1$ and

$$\bar{\lambda}_{k0} = \gamma, \quad \bar{\lambda}_k - J \bar{\lambda}_k = \bar{W}_k (\bar{z}_k - J \bar{s}_k).$$

The last expression provides a way to evaluate $\bar{\lambda}_k$ directly from \hat{s}_k and \hat{z}_k :

$$\begin{aligned} \bar{\lambda}_{k1} &= \frac{1}{2} \left(\bar{W}_k (\bar{z}_k - J \bar{s}_k) \right)_1 \\ &= \frac{1}{2} \left(\bar{z}_{k1} + \bar{s}_{k1} + \frac{\bar{z}_{k0} - \bar{s}_{k0}}{\bar{w}_{k0} + 1} \bar{w}_{k1} \right) \\ &= \frac{1}{\bar{s}_{k0} + \bar{z}_{k0} + 2\gamma} \left((\gamma + \bar{z}_{k0}) \bar{s}_{k1} + (\gamma + \bar{s}_{k0}) \bar{z}_{k1} \right). \end{aligned}$$

The unnormalized scaled variable is

$$\lambda_k = W_k \hat{z}_k = W_k^{-1} \hat{s}_k = \left((\hat{s}_k^T J \hat{s}_k) (\hat{z}_k^T J \hat{z}_k) \right)^{1/4} \bar{\lambda}_k.$$

5.4 Semidefinite cone

Scaling Any nonsingular congruence transformation can be used as a scaling for $C_k \in \mathcal{S}_p$:

$$W_k v = \mathbf{vec}(R^T \mathbf{mat}(v) R), \quad W_k^{-T} u = \mathbf{vec}(R^{-1} \mathbf{mat}(u) R^{-T}).$$

Nesterov-Todd scaling point The scaling point at \hat{s}_k, \hat{z}_k is the symmetric matrix for which

$$\mathbf{mat}(w_k) \hat{Z}_k \mathbf{mat}(w_k) = \hat{S}_k$$

where $\hat{S}_k = \mathbf{mat}(\hat{s}_k)$ and $\hat{Z}_k = \mathbf{mat}(\hat{z}_k)$. From (23),

$$w_k = \mathbf{vec} \left(\hat{S}_k^{1/2} \left(\hat{S}_k^{1/2} \hat{Z}_k \hat{S}_k^{1/2} \right)^{-1/2} \hat{S}_k^{1/2} \right).$$

Nonsymmetric Nesterov-Todd scaling The scaling point w_k can be computed in factored form $w_k = \text{vec}(R_k R_k^T)$, where R_k diagonalizes $\text{mat}(\hat{z}_k)$ and $\text{mat}(\hat{s}_k)$:

$$R_k^T \text{mat}(\hat{z}_k) R_k = R_k^{-1} \text{mat}(\hat{s}_k) R_k^{-T} = \text{mat}(\lambda_k),$$

with $\text{mat}(\lambda_k)$ diagonal. Note that $W_k^{-T} \hat{s}_k = W_k \hat{z}_k = \lambda_k$ and $\lambda_k^T \lambda_k = \hat{s}_k^T \hat{z}_k$.

The scaling matrix R_k can be computed as follows. We first compute Cholesky factorizations

$$S_k = \text{mat}(\hat{s}_k) = L_1 L_1^T, \quad Z_k = \text{mat}(\hat{z}_k) = L_2 L_2^T.$$

Next, we compute the SVD

$$L_2^T L_1 = U \Lambda_k V^T$$

and take $\lambda_k = \text{vec}(\text{diag}(\Lambda_k))$. Finally, we form

$$R_k = L_1 V \Lambda_k^{-1/2} = L_2^{-T} U \Lambda_k^{1/2}.$$

It can be verified that $R_k^T S_k^{-1} R_k = \Lambda_k^{-1}$ and $R_k^T Z_k R_k = \Lambda_k$ and that the inverse of R_k is given by $R_k^{-1} = \Lambda_k^{1/2} V^T L_1^{-1} = \Lambda_k^{-1/2} U^T L_2^T$.

5.5 Compositions of scaling matrices

If V is a scaling matrix, then

$$V^T H(w)^{-1} V = H(V^T w)^{-1}.$$

This is easy to see for the nonnegative orthant and the semidefinite cone. To verify the property for $C_k = \mathcal{Q}_p$, assume $V_k^T J V_k = V_k J V_k^T = \beta^2 J$. Then

$$\begin{aligned} V_k^T H_k(w_k)^{-1} V_k &= V_k^T (2w_k w_k^T V_k - (w_k^T J w_k) J) V_k \\ &= V_k^T 2w_k w_k^T V_k - (w_k^T J w_k) \beta^2 J \\ &= 2V_k^T w_k w_k^T V_k - (w_k^T V_k J V_k^T w_k) J \\ &= H_k(V_k^T w_k)^{-1}. \end{aligned} \tag{27}$$

6 Path-following algorithm

The algorithm computes search directions by linearizing the central path equations (21)–(22) around the current iterate $(\hat{s}, \hat{\kappa}, \hat{x}, \hat{y}, \hat{z}, \hat{\tau})$.

6.1 Outline

We start at initial values $(\hat{s}, \hat{\kappa}, \hat{x}, \hat{y}, \hat{z}, \hat{\tau}) = (s_0, 1, x_0, y_0, z_0, 1)$, where $s_0 \succ 0$, $z_0 \succ 0$, and define (q_x, q_y, q_z, q_τ) as in (8). We also compute the Nesterov-Todd scaling W at \hat{s} , \hat{z} , and the scaled variable $\lambda := W^{-T} \hat{s} = W \hat{z}$.

1. *Evaluate residuals, gap, and stopping criteria.* Compute

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \hat{s} \\ \hat{\kappa} \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix} \tag{28}$$

and

$$\hat{\mu} = \frac{\hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau}}{m+1} = \frac{\lambda^T \lambda + \hat{\kappa} \hat{\tau}}{m+1}.$$

Terminate if $(s, x, y, z) = (\hat{s}/\hat{\tau}, \hat{x}/\hat{\tau}, \hat{y}/\hat{\tau}, \hat{z}/\hat{\tau})$ satisfies (approximately) the optimality conditions (4), or (\hat{z}, \hat{y}) is an (approximate) certificate of primal infeasibility (5), or (\hat{s}, \hat{x}) is an (approximate) certificate of dual infeasibility (6).

2. *Affine direction.* Solve the linear equations

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s_a \\ \Delta \kappa_a \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x_a \\ \Delta y_a \\ \Delta z_a \\ \Delta \tau_a \end{bmatrix} = - \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \quad (29)$$

$$\lambda \circ (W \Delta z_a + W^{-T} \Delta s_a) = -\lambda \circ \lambda, \quad \hat{\kappa} \Delta \tau_a + \hat{\tau} \Delta \kappa_a = -\hat{\kappa} \hat{\tau}. \quad (30)$$

3. *Step size and centering parameter.* Compute

$$\begin{aligned} \alpha &= \sup \{ \alpha \in [0, 1] \mid (\hat{s}, \hat{\kappa}, \hat{z}, \hat{\tau}) + \alpha(\Delta s_a, \Delta \kappa_a, \Delta z_a, \Delta \tau_a) \succeq 0 \} \\ &= \sup \{ \alpha \in [0, 1] \mid (\lambda, \hat{\kappa}, \lambda, \hat{\tau}) + \alpha(W^{-T} \Delta s_a, \Delta \kappa_a, W \Delta z_a, \Delta \tau_a) \succeq 0 \} \\ \sigma &= \left(\frac{(\hat{s} + \alpha \Delta s_a)^T (\hat{z} + \alpha \Delta z_a)^T + (\hat{\kappa} + \alpha \Delta \kappa_a)(\hat{\tau} + \alpha \Delta \tau_a)}{\hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau}} \right)^3 \\ &= (1 - \alpha)^3. \end{aligned} \quad (31)$$

4. *Combined direction.* Solve the linear equation

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \\ \Delta \kappa \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix} = -(1 - \sigma) \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \quad (32)$$

$$\lambda \circ (W \Delta z + W^{-T} \Delta s) = -\lambda \circ \lambda - (W^{-T} \Delta s_a) \circ (W \Delta z_a) + \sigma \hat{\mu} \mathbf{e}, \quad (33)$$

$$\hat{\kappa} \Delta \tau + \hat{\tau} \Delta \kappa = -\hat{\kappa} \hat{\tau} - \Delta \kappa_a \Delta \tau_a + \sigma \hat{\mu}. \quad (34)$$

5. *Update iterates and scaling matrices.*

$$(\hat{s}, \hat{\kappa}, \hat{x}, \hat{y}, \hat{z}, \hat{\tau}) := (\hat{s}, \hat{\kappa}, \hat{x}, \hat{y}, \hat{z}, \hat{\tau}) + \alpha(\Delta s, \Delta \kappa, \Delta x, \Delta y, \Delta z, \Delta \tau)$$

where

$$\alpha = \sup \left\{ \alpha \in [0, 1] \mid (\lambda, \hat{\kappa}, \lambda, \hat{\tau}) + \frac{\alpha}{0.99} (W^{-T} \Delta s, \Delta \kappa, W \Delta z, \Delta \tau) \succeq 0 \right\}.$$

Compute the scaling matrix W for \hat{s}, \hat{z} , and the scaled variable $\lambda := W^{-T} \hat{s} = W \hat{z}$.

6.2 Discussion

We discuss steps 2–4 in more detail. We first derive some useful properties of the affine scaling direction computed in step 2. The first equation in (30) is equivalent to

$$\hat{s} \circ \Delta z_a + \hat{z} \circ \Delta s_a = -\hat{s} \circ \hat{z}$$

in unscaled coordinates. Taking the inner product with \mathbf{e} on both sides gives

$$\hat{s}^T \Delta z_a + \hat{z}^T \Delta s_a = -\hat{s}^T \hat{z}, \quad \hat{\kappa} \Delta \tau_a + \hat{\tau} \Delta \kappa_a = -\hat{\kappa} \hat{\tau}. \quad (35)$$

Furthermore,

$$\begin{aligned} \Delta z_a^T \Delta s_a + \Delta \tau_a \Delta \kappa_a &= - \begin{bmatrix} \Delta x_a \\ \Delta y_a \\ \Delta z_a \\ \Delta \tau_a \end{bmatrix}^T \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \\ &= - \begin{bmatrix} \Delta x_a \\ \Delta y_a \\ \Delta z_a \\ \Delta \tau_a \end{bmatrix}^T \left(\begin{bmatrix} 0 \\ 0 \\ \hat{s} \\ \hat{\kappa} \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix} \right) \\ &= -\hat{s}^T \Delta z_a - \hat{\kappa} \Delta \tau_a - \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}^T \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x_a \\ \Delta y_a \\ \Delta z_a \\ \Delta \tau_a \end{bmatrix} \\ &= -\hat{s}^T \Delta z_a - \hat{\kappa} \Delta \tau_a - \hat{z}^T \Delta s_a - \hat{\tau} \Delta \kappa_a + \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}^T \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \\ &= -\hat{s}^T \Delta z_a - \hat{\kappa} \Delta \tau_a - \hat{z}^T \Delta s_a - \hat{\tau} \Delta \kappa_a + \hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau} \\ &= 0. \end{aligned} \quad (36)$$

Lines 1 and 4 follow from (29) and the skew-symmetry of the coefficient matrix. Line 6 follows from (35). Line 5 follows from the skew-symmetry of the coefficient matrix in the definition of the residuals (28):

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}^T \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} = \hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau} = (m+1)\hat{\mu}. \quad (37)$$

The simple expression for σ in (31) follows by plugging in (35) and (36) in the definition.

The combined direction computed in step 4 has similar properties. From (33) and (34) we see that

$$\begin{aligned} \hat{s}^T \Delta z + \hat{\kappa} \Delta \tau + \hat{z}^T \Delta s + \hat{\tau} \Delta \kappa &= -\hat{s}^T \hat{z} - \hat{\kappa} \hat{\tau} - \Delta s_a^T \Delta z_a - \Delta \kappa_a \Delta \tau_a + \sigma \hat{\mu} (m+1) \\ &= -(1-\sigma)(\hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau}) \end{aligned} \quad (38)$$

and

$$\begin{aligned}
\Delta z^T \Delta s + \Delta \tau \Delta \kappa &= -(1 - \sigma) \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix}^T \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \\
&= -(1 - \sigma) \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix}^T \left(\begin{bmatrix} 0 \\ 0 \\ \hat{s} \\ \hat{\kappa} \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix} \right) \\
&= -(1 - \sigma) \left(\hat{s}^T \Delta z + \hat{\kappa} \Delta \tau + \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}^T \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix} \right) \\
&= -(1 - \sigma) \left(\hat{s}^T \Delta z + \hat{\kappa} \Delta \tau + \hat{z}^T \Delta s + \hat{\tau} \Delta \kappa + (1 - \sigma) \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\tau} \end{bmatrix}^T \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix} \right) \\
&= -(1 - \sigma) \left(\hat{s}^T \Delta z + \hat{\kappa} \Delta \tau + \hat{z}^T \Delta s + \hat{\tau} \Delta \kappa + (1 - \sigma) (\hat{s}^T \hat{z} + \hat{\kappa} \hat{\tau}) \right) \\
&= 0.
\end{aligned} \tag{39}$$

Next we show by induction that

$$(r_x, r_y, r_z, r_\tau) = \hat{\mu}(q_x, q_y, q_z, q_\tau)$$

at the beginning of each iteration. In the first iteration, this is true by definition of (q_x, q_y, q_z, q_τ) . Suppose it is satisfied by the current iterates. Then

$$\begin{aligned}
(r_x^+, r_y^+, r_z^+, r_\tau^+) &= (1 - \alpha(1 - \sigma))(r_x, r_y, r_z, r_\tau) \\
&= (1 - \alpha(1 - \sigma))\hat{\mu}(q_x, q_y, q_z, q_\tau) \\
&= \hat{\mu}^+(q_x, q_y, q_z, q_\tau),
\end{aligned}$$

because, from (38) and (39), $\hat{\mu}^+ = (1 - \alpha(1 - \sigma))\hat{\mu}$. Using this property, we can write (32) as

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \\ \Delta \kappa \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix} = \sigma \hat{\mu} \begin{bmatrix} q_x \\ q_y \\ q_z \\ q_\tau \end{bmatrix} - \begin{bmatrix} r_x \\ r_y \\ r_z \\ r_\tau \end{bmatrix}.$$

This shows that (32) can be interpreted as the central path equation (21) with $(s, \kappa, x, y, z, \tau)$ replaced by $(\hat{s} + \Delta s, \hat{\kappa} + \Delta \kappa, \hat{x} + \Delta x, \hat{y} + \Delta y, \hat{z} + \Delta z, \hat{\tau} + \Delta \tau)$, and $\mu = \sigma \hat{\mu}$. Making the same substitution in the nonlinear central path equations (22) gives

$$(W^{-T}(\hat{z} + \Delta z)) \circ (W(\hat{s} + \Delta s)) = \sigma \hat{\mu} \mathbf{e}, \quad (\hat{\kappa} + \Delta \kappa)(\hat{\tau} + \Delta \tau) = \sigma \hat{\mu}.$$

Expanding the products, using $W^{-T}\hat{z} = W\hat{s} = \lambda$, and approximating the second-order terms as

$$(W^{-T}\Delta s) \circ (W\Delta z) \approx (W^{-T}\Delta s_a) \circ (W\Delta z_a), \quad \Delta\tau\Delta\kappa \approx \Delta\kappa_a\Delta\tau_a$$

gives (33) and (34). (The righthand side terms in (33) and (34) that involve the affine direction components are known as the Mehrotra correction terms [Meh92, Wri97].)

In summary, we see that in step 4 a search direction is computed by linearizing the central path equations (21)–(22) around the current iterates with $\mu = \sigma\hat{\mu}$. Step 2 is the linearization for $\mu = 0$. Step 3 is a heuristic for choosing σ , based on the result for $\mu = 0$.

7 Initialization

If primal and dual starting points \hat{x} , \hat{s} , \hat{y} , \hat{z} are not specified by the user, they are selected as follows. The initial primal variable \hat{x} is the solution of the constrained least-squares problem

$$\begin{aligned} &\text{minimize} && \|Gx - h\|_2^2 \\ &\text{subject to} && Ax = b. \end{aligned}$$

The initial value of \hat{s} is computed from the residual $\tilde{s} = G\hat{x} - h$, as

$$\hat{s} = \begin{cases} \tilde{s} & \alpha_p < 0 \\ \tilde{s} + (1 + \alpha_p)\mathbf{e} & \text{otherwise} \end{cases}$$

where $\alpha_p = \inf\{\alpha \mid \tilde{s} + \alpha\mathbf{e} \succeq 0\}$. The values \hat{x} , \tilde{s} can be computed by solving the linear equation

$$\begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -I \end{bmatrix} \begin{bmatrix} \hat{x} \\ y \\ -\tilde{s} \end{bmatrix} = \begin{bmatrix} 0 \\ b \\ h \end{bmatrix}.$$

The initial dual variables \hat{y} , \hat{z} are computed by solving a least-norm problem

$$\begin{aligned} &\text{minimize} && \|z\|_2^2 \\ &\text{subject to} && G^T z + A^T y + c = 0. \end{aligned}$$

If the solution is \hat{y} , \tilde{z} , then we use \hat{y} as initial value of y , and

$$\hat{z} = \begin{cases} \tilde{z} & \alpha_d < 0 \\ \tilde{z} + (1 + \alpha_d)\mathbf{e} & \text{otherwise,} \end{cases}$$

where $\alpha_d = \inf\{\alpha \mid \tilde{z} + \alpha\mathbf{e} \succeq 0\}$, as the initial value of z . The least-norm problem is equivalent to the the linear equation

$$\begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -I \end{bmatrix} \begin{bmatrix} x \\ \hat{y} \\ \tilde{z} \end{bmatrix} = \begin{bmatrix} -c \\ 0 \\ 0 \end{bmatrix}.$$

8 Step length computation

Steps 3 and 5 require the computation of the maximum α such that

$$\lambda + \alpha \Delta \tilde{s} \succeq 0, \quad \lambda + \alpha \Delta \tilde{z} \succeq 0,$$

where $\Delta \tilde{s} = W^{-T} \Delta s$ and $\Delta \tilde{z} = W \Delta z$. To facilitate the calculation we compute, for each cone,

$$\rho_k = H(\lambda_k)^{1/2} \Delta \tilde{s}_k, \quad \sigma_k = H(\lambda_k)^{1/2} \Delta \tilde{z}_k.$$

The maximum step size then follows as $\alpha = \min_k \alpha_k$ where

$$\alpha_k = \sup \{ \alpha \mid \mathbf{e}_k + \alpha \rho_k \succeq 0, \mathbf{e}_k + \alpha \sigma_k \succeq 0 \}.$$

8.1 Nonnegative orthant

For the nonnegative orthant $C_k = \mathbf{R}_+^p$,

$$\rho_k = \lambda_k^{-1} \circ \Delta \tilde{s}_k, \quad \sigma_k = \lambda_k^{-1} \circ \Delta \tilde{z}_k, \quad \alpha_k = \max \left\{ 0, -\min_i \rho_{ki}, -\min_i \sigma_{ki} \right\}^{-1}.$$

8.2 Second-order cone

For the second-order cone $C_k = \mathcal{Q}_p$,

$$\begin{aligned} \rho_k &= \frac{1}{(\lambda_k^T J \lambda_k)^{1/2}} \begin{bmatrix} \bar{\lambda}_k^T J \Delta \tilde{s}_k \\ \Delta \tilde{s}_{k1} - (\bar{\lambda}_k^T J \Delta \tilde{s}_k + \Delta \tilde{s}_{k0})(\bar{\lambda}_{k0} + 1)^{-1} \bar{\lambda}_{k1} \end{bmatrix} \\ \sigma_k &= \frac{1}{(\lambda_k^T J \lambda_k)^{1/2}} \begin{bmatrix} \bar{\lambda}_k^T J \Delta \tilde{z}_k \\ \Delta \tilde{z}_{k1} - (\bar{\lambda}_k^T J \Delta \tilde{z}_k + \Delta \tilde{z}_{k0})(\bar{\lambda}_{k0} + 1)^{-1} \bar{\lambda}_{k1} \end{bmatrix} \end{aligned}$$

where $\bar{\lambda}_k = \lambda_k / (\lambda_k^T J \lambda_k)^{1/2}$. The maximum step size is

$$\alpha_k = \max \{ 0, \|\rho_{k1}\|_2 - \rho_{k0}, \|\sigma_{k1}\|_2 - \sigma_{k0} \}^{-1}.$$

8.3 Semidefinite cone

For the semidefinite cone $C_k = \mathbf{S}_+^p$,

$$\rho_k = \mathbf{vec}(\Lambda_k^{-1/2} \Delta \tilde{S}_k \Lambda_k^{-1/2}), \quad \sigma_k = \mathbf{vec}(\Lambda_k^{-1/2} \Delta \tilde{Z}_k \Lambda_k^{-1/2}),$$

where $\Lambda_k = \mathbf{mat}(\lambda_k)$, $\Delta \tilde{S}_k = \mathbf{mat}(\Delta \tilde{s}_k)$, $\Delta \tilde{Z}_k = \mathbf{mat}(\Delta \tilde{z}_k)$. We determine α_k by taking two eigenvalue decompositions

$$\mathbf{mat}(\rho_k) = Q_s \mathbf{diag}(\gamma_s) Q_s^T, \quad \mathbf{mat}(\sigma_k) = Q_z \mathbf{diag}(\gamma_z) Q_z^T.$$

The maximum step size is

$$\alpha_k = \max \left\{ 0, -\min_i \gamma_{si}, -\min_i \gamma_{zi} \right\}^{-1}.$$

9 Updating the scaling matrix

At the end of each iteration, we update the scaling point, scaling matrix, and scaled variables. The current scaling point w and scaling W satisfy

$$H(w)\hat{s} = \hat{z}, \quad W\hat{z} = W^{-T}\hat{s} = \lambda.$$

We need to compute a scaling point w^+ and scaling W^+ such that

$$H(w^+)(\hat{s} + \alpha\Delta s) = \hat{z} + \alpha\Delta z, \quad W^+(\hat{z} + \alpha\Delta z) = (W^+)^{-T}(\hat{s} + \alpha\Delta s) = \lambda^+.$$

This can be done as follows. We first compute the scaling point q for the scaled coordinates:

$$H(q)\hat{s}^+ = \hat{z}^+, \quad \hat{s}^+ = \lambda + \alpha\Delta\tilde{s}, \quad \hat{z}^+ = \lambda + \alpha\Delta\tilde{z}.$$

The new scaling point is $w^+ = W^T q$. This follows from (27):

$$\begin{aligned} H(W^T q)^{-1}(\hat{z} + \alpha\Delta z) &= W^T H(q)^{-1} W(\hat{z} + \alpha\Delta z) \\ &= W^T H(q)^{-1}(\lambda + \alpha\Delta\tilde{z}) \\ &= W^T(\lambda + \alpha\Delta\tilde{s}) \\ &= \hat{s} + \alpha\Delta s. \end{aligned}$$

9.1 Nonnegative orthant

If $C_k = \mathbf{R}_+^p$, the update is straightforward:

$$\begin{aligned} w_k^+ &= (\lambda_k + \alpha\Delta\tilde{s}_k)^{1/2} \circ (\lambda_k + \alpha\Delta\tilde{z}_k)^{-1/2} \circ w_k \\ \lambda_k^+ &= (\lambda_k + \alpha\Delta\tilde{z}_k)^{1/2} \circ (\lambda_k + \alpha\Delta\tilde{s}_k)^{1/2}. \end{aligned}$$

9.2 Second-order cone

Updated NT scaling point If $C_k = \mathcal{Q}_p$, we compute the scaling point q_k for the scaled variables, which satisfies

$$H_k(q_k)^{-1}\tilde{s}_k^+ = \tilde{z}_k^+, \quad (40)$$

as in section 5.3: $q_k = (q_k^T J q_k)^{1/2} \bar{q}_k$ where

$$q_k^T J q_k = \left(\frac{(\tilde{s}_k^+)^T J \tilde{s}_k^+}{(\tilde{z}_k^+)^T J \tilde{z}_k^+} \right)^{1/2}, \quad \bar{q}_k = \frac{1}{2\gamma^+} (\tilde{s}_k^+ + J\tilde{z}_k^+), \quad \gamma^+ = \left(\frac{1 + (\tilde{z}_k^+)^T \tilde{s}_k^+}{2} \right)^{1/2}$$

and \tilde{z}_k^+ and \tilde{s}_k^+ are the normalized scaled variables (in the current scaling)

$$\tilde{z}_k^+ = \frac{1}{((\tilde{z}_k^+)^T J \tilde{z}_k^+)^{1/2}} \tilde{z}_k^+, \quad \tilde{s}_k^+ = \frac{1}{((\tilde{s}_k^+)^T J \tilde{s}_k^+)^{1/2}} \tilde{s}_k^+.$$

Note that

$$\bar{q}_k^T J \bar{q}_k = 1, \quad \bar{q}_k^T \tilde{z}_k^+ = \bar{q}_k^T J \tilde{s}_k^+ = \gamma^+.$$

The new scaling point then follows as

$$w_k^+ = W_k^T q_k = \left((w_k^T J w_k)(q_k^T J q_k) \right)^{1/2} (2v_k v_k^T - J) \bar{q}_k.$$

Updated scaling matrix It follows that the parameters of the new scaling matrix

$$W_k^+ = \left((w_k^+)^T J w_k^+ \right)^{1/2} \bar{W}_k^+, \quad \bar{W}_k^+ = 2v_k^+(v_k^+)^T - J,$$

can be determined as follows.

1. The new scaling factor is

$$(w_k^+)^T J w_k^+ = (w_k^T J w_k)(q_k^T J q_k), \quad q_k^T J q_k = \left(\begin{array}{c} (\tilde{s}_k^+)^T J \tilde{s}_k^+ \\ (\tilde{z}_k^+)^T J \tilde{z}_k^+ \end{array} \right)^{1/2}.$$

2. The unitary vector v_k^+ , which defines the updated Householder transformation, is

$$v_k^+ := (\bar{w}_k^+)^{1/2} = \frac{1}{(2(\bar{w}_{k0}^+ + 1))^{1/2}} (\bar{w}_k^+ + \mathbf{e}_k)$$

with

$$\bar{w}_k^+ = (2v_k v_k^T - J) \bar{q}_k, \quad \bar{q}_k = \frac{1}{2\gamma^+} (\bar{s}_k^+ + J \bar{z}_k^+), \quad \gamma^+ = \left(\frac{1 + (\bar{z}_k^+)^T \bar{s}_k^+}{2} \right)^{1/2}.$$

Updated scaled variable The updated scaled variable

$$\lambda_k^+ = \left((\lambda_k^+)^T J (\lambda_k^+) \right)^{1/2} \bar{\lambda}_k^+$$

can be computed from the updated scaled variables $\tilde{s}_k^+, \tilde{z}_k^+$ as follows. The norm is easy to compute:

$$\begin{aligned} (\lambda_k^+)^T J \lambda_k^+ &= \left((s_k^+)^T J s_k^+ \right)^{1/2} \left((z_k^+)^T J z_k^+ \right)^{1/2} \\ &= \left((\tilde{s}_k^+)^T W_k J W_k \tilde{s}_k^+ \right)^{1/2} \left((\tilde{z}_k^+)^T W_k^{-1} J W_k^{-1} \tilde{z}_k^+ \right)^{1/2} \\ &= \left((\tilde{s}_k^+)^T J \tilde{s}_k^+ \right)^{1/2} \left((\tilde{z}_k^+)^T J \tilde{z}_k^+ \right)^{1/2}. \end{aligned}$$

The normalized vector $\bar{\lambda}_k^+$ is defined as

$$\bar{\lambda}_k^+ = \bar{W}_k^+ \bar{z}_k^+ = (\bar{W}_k^+)^{-1} \bar{s}_k^+ = J \bar{W}_k^+ J \bar{s}_k^+.$$

Its first component is

$$\bar{\lambda}_{k0}^+ = (\bar{w}_k^+)^T \bar{z}_k^+ = \bar{q}_k^T \bar{W}_k \bar{z}_k^+ = \bar{q}_k^T \bar{z}_k^+ = \gamma^+.$$

The rest follows from

$$(I - J) \bar{\lambda}_k^+ = \bar{W}_k^+ \left(\bar{z}_k^+ - J \bar{s}_k^+ \right) = \bar{W}_k^+ \bar{W}_k^{-1} \left(\tilde{z}_k^+ - J \tilde{s}_k^+ \right) = \bar{W}_k^+ J \bar{W}_k \left(J \tilde{z}_k^+ - \tilde{s}_k^+ \right).$$

Define $u_k = \tilde{s}_k^+ - J\tilde{z}_k^+$. Using the fact that $\bar{q}_k^T J u_k = 0$, we get

$$\begin{aligned}
2\bar{\lambda}_{k1}^+ &= - \left(\left(\frac{1}{\bar{w}_{k0}^+ + 1} (\bar{w}_k^+ + \mathbf{e}_k)(\bar{w}_k^+ + \mathbf{e}_k)^T - J \right) J \bar{W}_k u_k \right)_1 \\
&= - \frac{1}{\bar{w}_{k0}^+ + 1} \bar{w}_{k1}^+ (\bar{W}_k \bar{q}_k + \mathbf{e}_k)^T J \bar{W}_k u_k + (\bar{W}_k u_k)_1 \\
&= - \frac{1}{\bar{w}_{k0}^+ + 1} \bar{w}_{k1}^+ (\bar{q}_k^T J u_k + \mathbf{e}_k^T \bar{W}_k u_k) + (\bar{W}_k u_k)_1 \\
&= - \frac{\bar{w}_k^T u_k}{\bar{w}_{k0}^+ + 1} \bar{w}_{k1}^+ + (\bar{W}_k u_k)_1 \\
&= \left(\bar{W}_k \left(- \frac{\bar{w}_k^T u_k}{\bar{w}_{k0}^+ + 1} \bar{q}_k + u_k \right) \right)_1 \\
&= \left(\bar{W}_k \left(- \frac{2v_{k0}(v_k^T u_k) - u_{k0}}{\bar{w}_{k0}^+ + 1} \bar{q}_k + u_k \right) \right)_1 \\
\bar{\lambda}_{k1}^+ &= \left(\bar{W}_k \left(- \frac{v_{k0}(v_k^T u_k) - u_{k0}/2}{\bar{w}_{k0}^+ + 1} \bar{q}_k + \frac{1}{2} u_k \right) \right)_1 \\
&= \left(\bar{W}_k \left(\frac{1 - d/\gamma^+}{2} \tilde{s}_k - \frac{1 + d/\gamma^+}{2} J\tilde{z}_k \right) \right)_1
\end{aligned}$$

where

$$d = \frac{v_{k0}(v_k^T u_k) - u_{k0}/2}{\bar{w}_{k0}^+ + 1} = \frac{v_{k0}(v_k^T u_k) - u_{k0}/2}{2v_{k0}(v_k^T \bar{q}_k) - \bar{q}_{k0} + 1}.$$

9.3 Semidefinite cone

If $C_k = \mathcal{S}_p$, we use the eigenvalue decompositions

$$\Lambda_k^{-1/2} \Delta \tilde{S}_k \Lambda_k^{-1/2} = Q_s \mathbf{diag}(\gamma_s) Q_s^T, \quad \Lambda_k^{-1/2} \Delta \tilde{Z}_k \Lambda_k^{-1/2} = Q_z \mathbf{diag}(\gamma_z) Q_z^T,$$

where $\Lambda_k = \mathbf{mat}(\lambda_k)$, $\Delta \tilde{S}_k = \mathbf{mat}(\Delta \tilde{s}_k)$, $\Delta \tilde{Z}_k = \mathbf{mat}(\Delta \tilde{z}_k)$, to factor the new iterates in the old scaling coordinates as

$$R_k^{-1} S_k^+ R_k^{-T} = \Lambda_k + \alpha \Delta \tilde{S}_k = L_1 L_1^T, \quad R_k^T Z_k^+ R_k = \Lambda_k + \alpha \Delta \tilde{Z}_k = L_2 L_2^T,$$

with $L_1 = \Lambda_k^{1/2} Q_s (I + \alpha \mathbf{diag}(\gamma_s))^{1/2}$, $L_2 = \Lambda_k^{1/2} Q_z (I + \alpha \mathbf{diag}(\gamma_z))^{1/2}$. We then take an SVD

$$L_2^T L_1 = U \Lambda_k^+ V^T.$$

The scaling matrix that satisfies

$$(R_k^+)^{-1} S_k^+ (R_k^+)^{-T} = (R_k^+)^T Z_k^+ R_k^+ = \Lambda_k^+$$

is given by

$$R_k^+ = R_k L_1 V (\Lambda_k^+)^{-1/2} = R_k L_2^{-T} U (\Lambda_k^+)^{1/2}.$$

Its inverse is

$$(R_k^+)^{-1} = (\Lambda_k^+)^{1/2} V^T L_1^{-1} R_k^{-1} = (\Lambda_k^+)^{-1/2} U^T L_2^T R_k^{-1}.$$

10 Newton equations

The most expensive computation in each iteration of the algorithm is the solution of the linear equations in steps 2 and 4. These equations differ only in the righthand side and are of the form

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \\ \Delta \kappa \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & 0 & h \\ -c^T & -b^T & -h^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix} = - \begin{bmatrix} d_x \\ d_y \\ d_z \\ d_\tau \end{bmatrix} \quad (41)$$

$$\lambda \circ (W \Delta z + W^{-T} \Delta s) = -d_s, \quad \hat{\kappa} \Delta \tau + \hat{\tau} \Delta \kappa = -d_\kappa. \quad (42)$$

We will refer to the equations as Newton equations because they can be interpreted as linearizations of the central path conditions. In this and the next section we describe how CVXOPT solves the Newton equations. We first explain how to reduce them to a smaller 3×3 block equation (KKT system).

10.1 4×4 Block system

Eliminating Δs and $\Delta \kappa$ from (42) gives

$$\begin{bmatrix} 0 & A^T & G^T & c \\ -A & 0 & 0 & b \\ -G & 0 & W^T W & h \\ -c^T & -b^T & -h^T & \hat{\kappa}/\hat{\tau} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta \tau \end{bmatrix} = \begin{bmatrix} d_x \\ d_y \\ d_z - W^T (\lambda \diamond d_s) \\ d_\tau - d_\kappa/\hat{\tau} \end{bmatrix} \quad (43)$$

$$\Delta s = -W^T (\lambda \diamond d_s + W \Delta z), \quad \Delta \kappa = -(d_\kappa + \hat{\kappa} \Delta \tau)/\hat{\tau}. \quad (44)$$

Here $u \diamond v$ denotes the inverse of $u \circ v$ taken as a linear function of v , *i.e.*, $u \circ (u \diamond v) = v$ for all v . For $C_k = \mathbf{R}_+^p$, $\lambda_k \diamond v = \mathbf{diag}(\lambda_k)^{-1} v$. For $C_k = \mathcal{Q}_p$,

$$\begin{aligned} \lambda_k \diamond v &= \begin{bmatrix} \lambda_{k0} & \lambda_{k1}^T \\ \lambda_{k1} & \lambda_{k0} I \end{bmatrix}^{-1} \begin{bmatrix} v_0 \\ v_1 \end{bmatrix} \\ &= \frac{1}{\lambda_{k0}^2 - \lambda_{k1}^T \lambda_{k1}} \begin{bmatrix} \lambda_{k0} & -\lambda_{k1}^T \\ -\lambda_{k1} & \lambda_{k0}^{-1} ((\lambda_{k0}^2 - \lambda_{k1}^T \lambda_{k1}) I + \lambda_{k1} \lambda_{k1}^T) \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \end{bmatrix} \end{aligned}$$

If $C_k = \mathcal{S}_p$, with $\Lambda = \mathbf{mat}(\lambda_k)$, $\lambda_k \diamond v$ is the solution of

$$\frac{1}{2} (\Lambda \mathbf{mat}(x) + \mathbf{mat}(x) \Lambda) = \mathbf{mat}(v),$$

i.e., $\lambda_k \diamond v = \mathbf{vec}(\mathbf{mat}(v) \circ \Gamma)$ where $\Gamma_{ij} = 2/(\Lambda_{ii} + \Lambda_{jj})$.

Note that for the affine scaling Newton equation (step 2), the righthand side of (43) simplifies to

$$d_z - W^T (\lambda \diamond d_s) = r_z - W^T \lambda = r_z - \hat{s}, \quad d_\tau - d_\kappa/\hat{\tau} = r_\tau - \hat{\kappa}.$$

Also, note that $u = W^T (\lambda \diamond d_s)$ is the solution of $z \circ u = d_s$. Hence, the search directions depend only on the product $W^T W$ and not on the scaling W itself as the righthand side of (43) may suggest.

10.2 3×3 Block system

To solve the 4×4 block system (43)–(44), we solve two KKT systems

$$\begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & W^T W \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta y_1 \\ \Delta z_1 \end{bmatrix} = - \begin{bmatrix} c \\ b \\ h \end{bmatrix} \quad (45)$$

and

$$\begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & W^T W \end{bmatrix} \begin{bmatrix} \Delta x_2 \\ \Delta y_2 \\ \Delta z_2 \end{bmatrix} = \begin{bmatrix} d_x \\ d_y \\ d_z - W^T(\lambda \diamond d_s) \end{bmatrix} \quad (46)$$

and make a linear combination with

$$\begin{aligned} \Delta \tau &= \frac{d_\tau - d_\kappa/\hat{\tau} + c^T \Delta x_2 + b^T \Delta y_2 + h^T \Delta z_2}{\hat{\kappa}/\hat{\tau} - c^T \Delta x_1 - b^T \Delta y_1 - h^T \Delta z_1} \\ &= \frac{d_\tau - d_\kappa/\hat{\tau} + c^T \Delta x_2 + b^T \Delta y_2 + h^T \Delta z_2}{\hat{\kappa}/\hat{\tau} + \Delta z_1^T W^T W \Delta z_1} \end{aligned}$$

to get

$$\Delta x = \Delta x_2 + \Delta \tau \Delta x_1, \quad \Delta y = \Delta y_2 + \Delta \tau \Delta y_1, \quad \Delta z = \Delta z_2 + \Delta \tau \Delta z_1.$$

11 KKT equation solvers

Each iteration of the interior-point method requires the solution of three KKT systems

$$\begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -W^T W \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}. \quad (47)$$

The first is the equation (45). The other two are the equation (46) with the righthand sides in step 2 and step 4. In addition, if the problem includes second-order cone or semidefinite constraints, one step of iterative refinement is applied when solving (41) and (42). This increases the number of KKT systems solves per iteration by two. In this section we describe the two default methods for solving the KKT system (47).

11.1 Cholesky factorization

The equation (48) can be reduced to

$$\begin{bmatrix} G^T W^{-1} W^{-T} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b_x + G^T W^{-1} W^{-T} b_z \\ b_y \end{bmatrix}. \quad (48)$$

From x, y , the solution z follows as $Wz = W^{-T}(Gx - b_z)$.

If $G^T W^{-1} W^{-T} G$ is nonsingular, we can solve (48) via a Cholesky factorization $G^T W^{-1} W^{-T} G = LL^T$. We solve

$$AL^{-T}L^{-1}A^T y = AL^{-T}L^{-1} \left(b_x + G^T W^{-1} W^{-T} b_z \right) - b_y,$$

using a Cholesky factorization of $AL^{-T}L^{-1}A^T$ to obtain y , and then

$$LL^T x = b_x + G^T W^{-1} W^{-T} b_z - A^T y$$

to obtain x .

If $G^T W^{-1} W^{-T} T G$ is singular, we first write (48) as

$$\begin{bmatrix} G^T W^{-1} W^{-T} G + A^T A & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b_x + G^T W^{-1} W^{-T} b_z + A^T b_y \\ b_y \end{bmatrix}$$

We compute the Cholesky factorization $G^T W^{-1} W^{-T} G + A^T A = LL^T$, and solve

$$AL^{-T}L^{-1}A^T y = AL^{-T}L^{-1} \left(b_x + G^T W^{-1} W^{-T} b_z + A^T b_y \right) - b_y,$$

using a Cholesky factorization of $AL^{-T}L^{-1}A^T$ to obtain y , and then

$$LL^T x = b_x + G^T W^{-1} W^{-T} b_z + A^T (b_y - y)$$

to obtain x .

This method is the default for linear programs. The CHOLMOD sparse Cholesky factorization algorithms are used for sparse matrices, and the LAPACK algorithm for dense matrices. No attempts are made to separate G and A in dense and sparse submatrices and to exploit such structure.

11.2 QR factorization

In this method we write the KKT system as

$$\begin{bmatrix} 0 & A^T & \tilde{G}^T \\ A & 0 & 0 \\ \tilde{G} & 0 & -I \end{bmatrix} \begin{bmatrix} x \\ y \\ Wz \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ W^{-T} b_z \end{bmatrix}. \quad (49)$$

We use two QR factorizations

$$A^T = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix}, \quad \tilde{G} Q_2 = Q_3 R_3.$$

The solution x , y , Wz is computed in the following steps:

$$\begin{aligned} w &= W^{-T} b_z - \tilde{G} Q_1 R_1^{-T} b_y \\ u &= R_3^{-T} Q_2^T b_x + Q_3^T w \\ Wz &= Q_3 u - w \\ y &= R_1^{-1} \left(Q_1^T b_x - Q_1^T \tilde{G}^T (Wz) \right) \\ x &= Q_1 R_1^{-T} b_y + Q_2 R_3^{-1} u. \end{aligned}$$

To verify this, we first use the QR factorization of A^T to write (49) as

$$\begin{bmatrix} 0 & 0 & R_1 & Q_1^T \tilde{G}^T \\ 0 & 0 & 0 & Q_2^T \tilde{G}^T \\ R_1^T & 0 & 0 & 0 \\ \tilde{G} Q_1 & \tilde{G} Q_2 & 0 & -I \end{bmatrix} \begin{bmatrix} Q_1^T x \\ Q_2^T x \\ y \\ Wz \end{bmatrix} = \begin{bmatrix} Q_1^T b_x \\ Q_2^T b_x \\ b_y \\ W^{-T} b_z \end{bmatrix}.$$

From the third equation, we have $Q_1^T x = R_1^{-T} b_y$. If we define $u = R_3 Q_2^T x$, we can write the three remaining equations in the variables u, y, Wz as

$$\begin{aligned} R_1 y &= Q_1^T b_x - Q_1^T \tilde{G}^T(Wz) \\ Q_2^T \tilde{G}^T(Wz) &= Q_2^T b_x \\ Wz &= \tilde{G} Q_1(Q_1^T x) + \tilde{G} Q_2(Q_2^T x) - W^{-T} b_z \\ &= \tilde{G} Q_1 R_1^{-T} b_y + Q_3 u - W^{-T} b_z \\ &= Q_3 u - w. \end{aligned}$$

From the last equation, we eliminate Wz to get an equation in u :

$$R_3^T u = Q_2^T b_x + R_3^T Q_3 w.$$

This method is the default method for problems with second-order cone or semidefinite constraints. The LAPACK dense QR factorization routines are used, so no sparsity is exploited.

References

- [AG03] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming Series B*, 95:3–51, 2003.
- [ART03] E. D. Andersen, C. Roos, and T. Terlaky. On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming*, 95(2):249–277, 2003.
- [dKRT97] E. de Klerk, C. Roos, and T. Terlaky. Initialization in semidefinite programming via a self-dual skew-symmetric embedding. *Operations Research Letters*, 20(5):213–221, 1997.
- [Meh92] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, November 1992.
- [NT97] Yu. E. Nesterov and M. J. Todd. Self-scaled cones and interior-point methods for convex programming. *Mathematics of Operations Research*, 22:1–42, 1997.
- [NT98] Yu. E. Nesterov and M. J. Todd. Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8(2):324–364, May 1998.
- [RS88] C. M. Rader and A. O. Steinhardt. Hyperbolic Householder transforms. *SIAM Journal on Matrix Analysis and Applications*, 9(2):269–290, 1988.
- [Stu00] J. F. Sturm. Similarity and other spectral relations for symmetric cones. *Linear Algebra and Its Applications*, 312:135–154, 2000.
- [Stu02] J. F. Sturm. Implementation of interior point methods for mixed semidefinite and second order cone optimization problems. *Optimization Methods and Software*, 17(6):1105–1154, 2002.
- [Stu03] J. F. Sturm. Avoiding numerical cancellation in the interior point method for solving semidefinite programs. *Mathematical Programming Series B*, 95:219–247, 2003.

- [Tsu99] T. Tsuchiya. A convergence analysis of the scaling-invariant primal-dual path-following algorithms for second-order cone programming. *Optimization Methods and Software*, 11-12:141–182, 1999.
- [TTT03] R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming Series B*, 95:189–217, 2003.
- [Wri97] S. J. Wright. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, 1997.
- [YTM94] Y. Ye, M. J. Todd, and S. Mizuno. An $O(\sqrt{n}L)$ -iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67, 1994.