Step by step design of an interior-point solver in self-dual conic optimization

Jean Charles Gilbert

Step by step design of an interior-point solver in self-dual conic optimization – Application to the Shor relaxation of some small OPF problems

Jean Charles Gilbert, INRIA

December 17, 2016

These notes present a project in numerical optimization dealing with the implementation of an interior-point method for solving a self-dual conic optimization (SDCO) problem. The cone is the Cartesian product of semidefinite cones of various dimensions (imposing matrices to be positive semidefinite) and of a positive orthant (imposing variables to be nonnegative). Therefore, the solved problem encompasses semidefinite and linear optimization.

The project was given in a course entitled *Advanced Continuous Optimization II* at the University Paris-Saclay, in 2016-1017. The solver is designed step by step during a series of 5 sessions of 4 hours each. Each session corresponds to a chapter of these notes. The correctness of the SDCO solver is verified during each session on small academic problems, having diverse properties. During the last session, the developed piece of software is used to solve a few small size Shor relaxations of QCQP (quadratically constrained quadratic programming) problems, modeling various instances of the OPF (optimal power flow) problem.

These notes are actually carrying on with those of the year 2015-2016 [11], by adding some features to the preceeding developped solver. The main one is that the primal variable is no longer a single positive semidefinite symmetric matrix but is formed of several positive semidefinite symmetric matrices and a nonnegative vector.

The goal of the project is *not* to design an SDCO solver that would beat the best existing solver but to help the students to understand and demystify what there is inside such a piece of software. As a side outcome, this course also shows that a rather performent SDCO solver can be realized in a relatively short time. In addition the student will be able to improve the developed piece of software, in case this is required by professionel needs. For a review of SDCO codes and more details on their development, we refer the reader to [42, 1], in particular [25, 40].

Table of Contents

1 The NT direction

The goal of this first session is to compute the Nesterov-Todd (NT) direction of an interior point algorithm to solve a self-dual conic optimization problem.

1.1 The problem to solve

1.1.1 Space structures

We denote by $\mathbb{E}$ the Euclidean vector space of the variables defining the primal form of the convex optimization considered in all this project; see section 1.1.2. It is equipped with a scalar product denoted by $\langle \cdot, \cdot \rangle_{\mathbb{E}}$. Conic optimization is formulated thanks to a cone of $\mathbb{E}$, which is denoted by K. By definition a *cone* K of $\mathbb{E}$ is a set verifying $\mathbb{R}_{++}K \subset K$, where $R_{++} := \{x \in \mathbb{R}^n : x > 0\}$ is the *positive orthant* (vectorial inequalities must be understand componentwise; hence $x > 0$ means that $x_i > 0$ for all $i \in [1:n]$). The *dual cone* of K is the closed convex cone defined by

$$K^+ := \{y \in \mathbb{E} : \langle y, x \rangle_{\mathbb{E}} \geqslant 0 \text{ for all } x \in K\}.$$

The cone K is said to be *self-dual* if $K^+ = K$. Hence, a self-dual cone is necessarily closed and convex. The unknowns of the optimization problem we consider below are elements of a vector space $\mathbb{E}$ that are constrained to belong to some self-dual cone K and to satisfy some affine constraint.

More specifically, the vector space $\mathbb{E}$ considered in this project is the Cartesian product

$$\mathbb{E} := \mathcal{S}^{n_1} \times \cdots \times \mathcal{S}^{n_s} \times \mathbb{R}^{n_l},$$

where, for $j \in [1:s]$, $\mathcal{S}^{n_j}$ is the *space of symmetric real matrices* of order n_j; it has dimension $n_j(n_j+1)/2$. This means that the unknowns of the optimization problem below are s symmetric matrices of order $n_1, \ldots, n_s$ and a vector of variables of dimension n_l. An element $x \in \mathbb{E}$ will be denoted by

$$x = (x^{s,1}, \ldots, x^{s,s}, x^l),$$

where $x^{s,j} \in \mathcal{S}^{n_j}$, for $j \in [1:s]$, and $x^l \in \mathbb{R}^{n_l}$ (unusually, we have chosen to represent matrices by lower case letters, because in $\mathbb{E}$ they are assembled with a vector denoted by a lower case letter).

The scalar product on some $\mathcal{S}^{n_j}$ is defined and denoted by

$$\langle \cdot, \cdot \rangle_{\mathcal{S}^{n_j}} : (a, b) \in \mathcal{S}^{n_j} \times \mathcal{S}^{n_j} \mapsto \langle a, b \rangle_{\mathcal{S}^{n_j}} := \operatorname{tr} ab = \sum_{k,l=1}^{n_j} a_{kl} b_{kl},$$

where $\operatorname{tr} a := \sum_i a_{ii}$ denotes the *trace* of the matrix $a \in \mathbb{R}^{n_j \times n_j}$. The associated norm is the so-called *Frobenius norm*; its value at $a \in \mathcal{S}^{n_j}$ is denoted by

$$\|a\|_{\mathcal{S}^{n_j}} = \left(\sum_{k,l=1}^{n_j} a_{kl}^2 \right)^{1/2}.$$

The Euclidean scalar product is supposed to equip $\mathbb{R}^{n_l}$. Finally the scalar product on $\mathbb{E}$ takes at $(x, s) \in \mathbb{E} \times \mathbb{E}$ the value

$$\langle x, s \rangle_{\mathbb{E}} := \sum_{j=1}^{s} \langle x^{s,j}, s^{s,j} \rangle_{\mathcal{S}^{n_j}} + (x^l)^{\mathsf{T}} (s^l).$$

The associated norm is

$$\|x\|_{\mathbb{E}} := \left(\sum_{j=1}^{s} \|x^{s,j}\|_{\mathcal{S}^{n_j}}^2 + \|x^l\|_2^2 \right)^{1/2}.$$

A product of two symmetric matrices is not necessary symmetric. Since these products will appear below, it is natural to introduce the vector space

$$\mathbb{F} := \mathbb{R}^{n_1 \times n_1} \times \cdots \times \mathbb{R}^{n_s \times n_s} \times \mathbb{R}^{n_l},$$

where $\mathbb{R}^{n_j \times n_j}$ denotes the set of square real matrices of order n_j. Of course, $\mathbb{E} \subset \mathbb{F}$. The dimension of $\mathbb{F}$ is given by

$$n_{\mathbb{F}} := \sum_{j=1}^{s} n_j^2 + n_l. \tag{1.1}$$

The space $\mathbb{F}$ has its own scalar product, which takes at $(x, s) \in \mathbb{F} \times \mathbb{F}$ the value

$$\langle x, s \rangle_{\mathbb{F}} := \sum_{j=1}^{s} \operatorname{tr}(x^{s,j})^{\mathsf{T}} (s^{s,j}) + (x^l)^{\mathsf{T}} (s^l). \tag{1.2}$$

The associated norm is $\|x\|_{\mathbb{F}} := \langle x, x \rangle_{\mathbb{F}}^{1/2}$. Clearly, when x and $s \in \mathbb{E}$, $\langle x, s \rangle_{\mathbb{F}} = \langle x, s \rangle_{\mathbb{E}}$. The vector space $\mathbb{F}$ appears for example, when one defines the *Hadamard product* of two vectors x and $s \in \mathbb{E}$, which is the vector

$$x \bullet s = (x^{s,1} s^{s,1}, \ldots, x^{s,s} s^{s,s}, x^l \cdot s^l) \in \mathbb{F},$$

where $x^{s,j} s^{s,j}$ is the product of the matrices $x^{s,j}$ and $s^{s,j}$ and $x^l \cdot s^l$ is the standard Hadamard product of two vectors, which is $(x^l \cdot s^l)_i = x_i^l s_i^l$ for all $i \in [1 : n_l]$. The fact

that $x \bullet s$ may not be in $\mathbb{E}$ when x and $s \in \mathbb{E}$ will be a source of trouble in the sequel. The *double Hadamard product*

$$x \bullet s \bullet v$$

is equivalently defined by $(x \bullet s) \bullet v$ or $x \bullet (s \bullet v)$.

The self-dual cone K of $\mathbb{E}$ considered in this project is a Cartesian product of cones, namely

$$K := \mathcal{S}_+^{n_1} \times \cdots \times \mathcal{S}_+^{n_s} \times \mathbb{R}_+^{n_l},$$

where $\mathcal{S}_+^{n_j}$ is the of $\mathcal{S}^{n_j}$ and $\mathbb{R}_+^{n_l} := \{x \in \mathbb{R}^{n_l} : x \geqslant 0\}$ is the *nonnegative orthant* (vector inequalities have to be understood componentwise: $x \geqslant 0$ iff $x_i \geqslant 0$ for all i). The cone K is self-dual since this is the case for $\mathcal{S}_+^{n_j}$ and $\mathbb{R}_+^{n_l}$. This choice of K implies that the unknown matrices in $\mathcal{S}^{n_j}$ of the optimization problem are forced to be positive semidefinite and that the unknown vector in $\mathbb{R}^{n_l}$ is forced to have nonnegative components. We associate with K the operator $\min_K : \mathbb{E} \to \mathbb{R}$ defined at $x \in \mathbb{E}$ by

$$\min_K(x) = \min\left(\lambda_{\min}(x^{s,1}), \ldots, \lambda_{\min}(x^{s,s}), \min_{i \in [1\,:\,n_l]} x_i^l\right), \tag{1.3}$$

where $\lambda_{\min}(x^{s,j})$ is the smallest eigenvalue of the symmetric matrix $x^{s,j}$. Therefore $x \in K$ if and only if $\min_K(x) \geqslant 0$. The *strict feasible sets* of the optimization problems (P) and (D) defined below make use of the *strict cone*

$$K^s := \mathcal{S}_{++}^{n_1} \times \cdots \times \mathcal{S}_{++}^{n_s} \times \mathbb{R}_{++}^{n_l},$$

where $\mathcal{S}_{++}^{n_j}$ is the cone of positive definite matrices of $\mathcal{S}^{n_j}$ and $\mathbb{R}_{++}^{n_l} := \{x \in \mathbb{R}^{n_l} : x > 0\}$ is the *positive orthant* (strict inequalities also act componentwise: $x > 0$ iff $x_i > 0$ for all i). Clearly, $x \in K^s$ if and only if $\min_K(x) > 0$. For $x = (x^{s,1}, \ldots, x^{s,s}, x^l)$ in K^s, one can define

$$\begin{aligned} x^{-1} &:= ((x^{s,1})^{-1}, \ldots, (x^{s,s})^{-1}, (x^l)^{-1}) \in K^s, \\ x^{1/2} &:= ((x^{s,1})^{1/2}, \ldots, (x^{s,s})^{1/2}, (x^l)^{1/2}) \in K^s, \end{aligned}$$

where the exponent -1 (resp. $1/2$) refers to the inverse (resp. square root) of a positive definite matrix or the componentwise inverse (resp. square root) of a positive vector.

A particular element of K^s, which is used continually below, is

$$e := (I^{n_1}, \ldots, I^{n_s}, e^l),$$

where I^{n_j} is the identity matrix of order n_j and e^l is the vector of all ones in $\mathbb{R}^{n_l}$. Observe that

$$\langle x \bullet s, e\rangle_{\mathbb{E}} = \langle x, s\rangle_{\mathbb{E}} \qquad \text{and} \qquad n_c := \|e\|_{\mathbb{E}}^2 = \sum_{j=1}^{s} n_j + n_l. \tag{1.4}$$

The number n_c is called the *complexity module* of the problem (hence the index c). Note the difference with the dimension $n_{\mathbb{F}}$ of the space $\mathbb{F}$, defined in (1.1), in which the square of the dimensions n_j appears instead. We will see with (2.9) that the number of iterations to reach optimality at a given precision depends on the square root of n_c.

1.1.2 The primal and dual SDCO problems

The primal (P) and dual (D) *self-dual conic optimization* (SDCO) problems read

$$\boxed{(P) \quad \left\{ \begin{array}{l} \inf \langle c, x \rangle_{\mathbb{E}} \\ A(x) = b \\ x \in K \end{array} \right.} \quad \text{and} \quad \boxed{(D) \quad \left\{ \begin{array}{l} \sup\ b^{\mathsf{T}} y \\ A^*(y) + s = c \\ s \in K, \end{array} \right.} \tag{1.5}$$

where $c \in \mathbb{E}$, $A : \mathbb{E} \to \mathbb{R}^m$ is a linear map, $b \in \mathbb{R}^m$, and $A^* : \mathbb{R}^m \to \mathbb{E}$ is the adjoint of A when $\mathbb{R}^m$ is equipped with the Euclidean scalar product. These problems are Lagrangian dual to each other. They are convex in the sense that their objective is convex and their feasible set is also convex. When $s = 0$ and $n_l \neq 0$, one recovers the *linear optimization* (LO) problem; when $s = 1$ and $n_l = 0$, one recovers the standard *semidefinite optimization* (SDO) problem.

The *feasible sets* of (P) and (D) are respectively denoted by

$$\mathcal{F}_P := \{x \in K : A(x) = b\} \quad \text{and} \quad \mathcal{F}_D := \{(y, s) \in \mathbb{R}^m \times K : A^*(y) + s = c\},$$

and their *strictly feasible sets* are denoted by

$$\mathcal{F}_P^s := \{x \in K^s : A(x) = b\} \quad \text{and} \quad \mathcal{F}_D^s := \{(y, s) \in \mathbb{R}^m \times K^s : A^*(y) + s = c\}.$$

Accordingly, a point $x \in \mathcal{F}_P^s$ is said to be *strictly feasible for* (P) and a pair $(y, s) \in \mathcal{F}_D^s$ is said to be *strictly feasible for* (D). We also introduce the following Cartesian products

$$\mathcal{F} := \mathcal{F}_P \times \mathcal{F}_D \quad \text{and} \quad \mathcal{F}^s := \mathcal{F}_P^s \times \mathcal{F}_D^s.$$

The solution sets of these problems are denoted by

$$\mathrm{Sol}(P) \qquad \text{and} \qquad \mathrm{Sol}(D)$$

and their optimal values by

$$\mathrm{val}(P) \qquad \text{and} \qquad \mathrm{val}(D).$$

The *duality gap* is zero if $\mathrm{val}(D) = \mathrm{val}(P)$ (with possible infinite values) and, otherwise, is the nonnegative difference $\mathrm{val}(P) - \mathrm{val}(D) \geqslant 0$.

By the Riesz-Fréchet theorem, there are elements $a_i \in \mathbb{E}$ such that for all $x \in \mathbb{E}$, there holds

$$A(x) = \begin{pmatrix} \langle a_1, x \rangle_{\mathbb{E}} \\ \vdots \\ \langle a_m, x \rangle_{\mathbb{E}} \end{pmatrix}. \tag{1.6}$$

Each $a_i \in \mathbb{E}$ is therefore an assembling of s matrices $a_i^{s,j} \in \mathcal{S}^{n_j}$, for $j \in [1\,{:}\,s]$, and a vector in $\mathbb{R}^{n_l}$. It will be also assumed that the Euclidean scalar product is used on $\mathbb{R}^m$, in which case $A^*(y)$ is a weighted sum of the vectors a_i (see question 1.1):

$$A^*(y) = \sum_{i=1}^{m} y_i a_i \in \mathbb{E}. \tag{1.7}$$

1.1.3 The central path

It is known that when $\mathcal{F}^s \neq \varnothing$, $z := (x, y, s)$ is a primal-dual solution to (P) and (D) if and only if

$$\begin{cases} A^*(y) + s = c, \quad s \in K \\ A(x) = b, \quad x \in K \\ \langle x, s \rangle_{\mathbb{E}} = 0. \end{cases} \tag{1.8}$$

These may be considered as the necessary and sufficient optimality conditions of the convex problems (P) or (D), under the "constraint qualification condition" $\mathcal{F}^s \neq \varnothing$ (it is known, indeed, that "$\mathcal{F}^s \neq \varnothing$ and the surjectivity of A" is equivalent to Robinson's constraint qualification [34] at one or any point of the feasible sets of the problems (P) and (D) [10]). It can be shown (see question 1.2) that for x and $s \in K$, there holds

$$\langle x, s \rangle_{\mathbb{E}} = 0 \qquad \Longleftrightarrow \qquad x \bullet s = 0. \tag{1.9}$$

Therefore, (1.8) also reads

$$\begin{cases} A^*(y) + s = c, \quad s \in K \\ A(x) = b, \quad x \in K \\ x \bullet s = 0. \end{cases} \tag{1.10}$$

We prefer (1.10) to (1.8), since then the "number of equations" in (1.10) is equal to the "number of unknowns". Indeed, the triple (x, y, s) lies in $\mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ and the equation values are in the same space when $x \bullet s \in \mathbb{E}$ (this is certainly the case when $x \bullet s = 0$).

When A is surjective, the central path is the (image of the) map $\mu \in \mathbb{R}_{++} \mapsto z(\mu) \in \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$, where $z = z(\mu)$ is the unique solution to the perturbed optimality conditions (see question 1.3)

$$\begin{cases} A^*(y) + s = c, \quad s \in K^s \\ A(x) = b, \quad x \in K^s \\ x \bullet s = \mu e. \end{cases} \tag{1.11}$$

When $\mathcal{F}^s \neq \varnothing$, the central path converges to a primal-dual solution, when $\mu \downarrow 0$. The goal of the *path-following* algorithms that we shall consider is to follow the central path to reach a solution asymptotically.

1.2 The NT direction

1.2.1 Overview

Given a strictly feasible point $z = (x, y, s) \in \mathcal{F}^s \subset \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$, the goal of one iteration of the primal-dual path-following interior-point (IP) algorithm we consider is to make a displacement d in $\mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ towards a central point $z(\mu)$, for some $\mu \geqslant 0$, that is "closer" to the solution than the current iterate z. This displacement $d = (d_x, d_y, d_s) \in \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ is a Newton step on the perturbed system (1.11) without its conic conditions $x \in K^s$ and $s \in K^s$ (these will be enforced at each iteration), namely

$$\begin{cases} A^*(y) + s = c \\ A(x) = b \\ x \bullet s = \mu e. \end{cases} \tag{1.12}$$

When $z \in \mathcal{F}$ is not on the central path, the point $x \bullet s$ may not be in $\mathbb{E}$, since its matrix components may not be symmetric matrices. In that case, one should work with more equations than unknowns in (1.12), which we wish to avoid. To tell it otherwise, consider the linearization of the system (1.12) at $z = (x, y, s)$, which is the system in $d = (d_x, d_y, d_s)$:

$$\begin{cases} A^*(d_y) + d_s = c - A^*(y) - s \\ A(d_x) = b - A(x) \\ d_x \bullet s + x \bullet d_s = \mu e - x \bullet s. \end{cases}$$

If d_s is in $\mathbb{E}$ by the first equation (since A^* takes its values in $\mathbb{E}$), there are examples [23] showing that d_x might not be in $\mathbb{E}$ (because $x \bullet d_s \bullet s^{-1}$ in the third equation is not necessarily in $\mathbb{E}$, by lack of symmetry of its matrix components). For this reason a "symmetrization" operation must be introduced in addition to the linearization of the system, in order to keep $x + d_x$ in $\mathbb{E}$. This can be done using a large number of methods, leading to many different IP directions. The one we consider in this project is the *Nesterov-Todd* (NT) direction (called that way because of [28, 29, 39; 1997-1998], but the direction might have been proposed independently and earlier in the later published paper [38; 1999]). It is often implemented and it has the following features:

- it preserves the primal and dual displacements d_x and d_s in $\mathbb{E}$,
- it is uniquely defined for any strictly feasible iterate z,
- it is scale invariant,
- it is less computation demanding than many other directions.

Another efficient direction is the so-called HKM direction (obtained independently in [13, 18], see also [32]). It is also implemented in SDPT3, for instance.

1.2.2 Derivation of the NT direction

The NT direction d at $z = (x, y, s) \in K^s \times \mathbb{R}^m \times K^s$ can be shortly presented as the result of three operations [7].

1. *Scaling.* A *weight* $w \in K^s$ is introduced, which is the unique element in K^s that satisfies the identity (see question 1.4)

$$w \bullet s \bullet w = x. \tag{1.13}$$

It is given by the formulas

$$w := x^{1/2} \bullet \left(x^{1/2} \bullet s \bullet x^{1/2}\right)^{-1/2} \bullet x^{1/2} = s^{-1/2} \bullet \left(s^{1/2} \bullet x \bullet s^{1/2}\right)^{1/2} \bullet s^{-1/2}. \tag{1.14}$$

Let us also introduce the scaled variable

$$v := w^{-1/2} \bullet x \bullet w^{-1/2} = w^{1/2} \bullet s \bullet w^{1/2} \in K^s. \tag{1.15}$$

Note that the vector component of w and v simply read

$$w^l = (x^l)^{1/2} \cdot (s^l)^{-1/2} \qquad \text{and} \qquad v^l = (x^l)^{1/2} \cdot (s^l)^{1/2}. \tag{1.16}$$

2. *Symmetrization.* If $z = (x, y, s)$ is the current iterate, one would like to find a displacement $d = (d_x, d_y, d_s)$ such that the third equation in (1.12) is satisfied at $z + d$, namely

$$(x + d_x) \bullet (s + d_s) = \mu e,$$

for some $\mu \geqslant 0$ to be defined. After a left-Hadamard-multiplication by $w^{-1/2}$ and a right-Hadamard-multiplication by $w^{1/2}$, this equation takes the equivalent form

$$(v + \tilde{d}_x) \bullet (v + \tilde{d}_s) = \mu e,$$

where

$$\tilde{d}_x = w^{-1/2} \bullet d_x \bullet w^{-1/2} \qquad \text{and} \qquad \tilde{d}_s = w^{1/2} \bullet d_s \bullet w^{1/2}.$$

We now *symmetrize* the lhs of this equation as follows

$$\frac{1}{2}\left[(v + \tilde{d}_x) \bullet (v + \tilde{d}_s) + (v + \tilde{d}_s) \bullet (v + \tilde{d}_x)\right] = \mu e.$$

3. *Pseudo-linearization* (the term "pseudo" is used because the weight w, which depends on the current iterate, is not linearized). The linearization consists in dropping the Hadamard products $\tilde{d}_x \bullet \tilde{d}_s$ and $\tilde{d}_s \bullet \tilde{d}_x$ from the last identity, which then becomes

$$\frac{1}{2}\left[v \bullet (\tilde{d}_x + \tilde{d}_s) + (\tilde{d}_x + \tilde{d}_s) \bullet v\right] = \mu e - v \bullet v.$$

This is a *Lyapunov equation* in the unknown $\tilde{d}_x + \tilde{d}_s$. Since $v \in K^s$, it has a unique solution, which is

$$\tilde{d}_x + \tilde{d}_s = \mu v^{-1} - v.$$

Left and right-Hadamard-multiplication of the two sides of this last equation by $w^{1/2}$ yield the symmetric pseudo-linearization at z of the third equation in (1.12):

$$d_x + w \bullet d_s \bullet w = \mu s^{-1} - x.$$

Therefore, when $z \in \mathcal{F}^s$, the NT direction is obtained by solving the following linear system in $d = (d_x, d_y, d_s) \in \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$:

$$A^*(d_y) + d_s = 0 \tag{1.17a}$$

$$A(d_x) = 0 \tag{1.17b}$$

$$d_x + w \bullet d_s \bullet w = \mu s^{-1} - x, \tag{1.17c}$$

in which the value of μ still needs to be specified. We will see in section 1.2.4, that when A is surjective, the system (1.17) has a unique solution, whatever is the right-hand side (in particular, whatever is $\mu \in \mathbb{R}$).

1.2.3 Computation of the weight w

The weight w given by (1.14) intervenes in the computation of the NT direction and must be computed explicitly.

Each matrix components $w^{s,j}$ of w can be computed using two Cholesky factorizations and one singular value decompositions (SVD) [39; § 4.1]. The Cholesky factorizations are those of the matrix components in $x \in K^s$ and $s \in K^s$:

$$\boxed{x^{s,j} = L_{x^{s,j}} L_{x^{s,j}}^{\mathsf{T}}} \qquad \text{and} \qquad \boxed{s^{s,j} = L_{s^{s,j}} L_{s^{s,j}}^{\mathsf{T}},}$$

where $L_{x^{s,j}}$ and $L_{s^{s,j}}$ are lower triangular nonsingular matrices of order n_j. The SVD is the one of $L_{s^{s,j}}^{\mathsf{T}} L_{x^{s,j}}$:

$$\boxed{L_{s^{s,j}}^{\mathsf{T}} L_{x^{s,j}} = U_j \Sigma_j V_j^{\mathsf{T}}.}$$

where U_j and V_j are orthogonal matrices and Σ_j is the diagonal matrix formed of the positive singular values of the nonsingular matrix $L_{s^{s,j}}^{\mathsf{T}} L_{x^{s,j}}$. Define

$$Q_j := L_{x^{s,j}}^{-1} (x^{s,j})^{1/2},$$

which is an orthogonal matrix. It needs not be computed. Now, there holds

$$\begin{aligned}
(x^{s,j})^{1/2} s^{s,j} (x^{s,j})^{1/2} &= (x^{s,j})^{1/2} \underbrace{L_{x^{s,j}}^{-\mathsf{T}} L_{x^{s,j}}^{\mathsf{T}}}_{I^{n_j}} \underbrace{L_{s^{s,j}} L_{s^{s,j}}^{\mathsf{T}}}_{s^{s,j}} \underbrace{L_{x^{s,j}} L_{x^{s,j}}^{-1}}_{I^{n_j}} (x^{s,j})^{1/2} \\
&= Q_j^{\mathsf{T}} \underbrace{L_{x^{s,j}}^{\mathsf{T}} L_{s^{s,j}}}_{V_j \Sigma_j U_j^{\mathsf{T}}} \underbrace{L_{s^{s,j}}^{\mathsf{T}} L_{x^{s,j}}}_{U_j \Sigma_j V_j^{\mathsf{T}}} Q_j \\
&= Q_j^{\mathsf{T}} V_j \Sigma_j^2 V_j^{\mathsf{T}} Q_j.
\end{aligned}$$

From this identity and the orthogonality of $Q_j^{\mathsf{T}} V_j$, it results

$$((x^{s,j})^{1/2} s^{s,j} (x^{s,j})^{1/2})^{-1/2} = Q_j^{\mathsf{T}} V_j \Sigma_j^{-1} V_j^{\mathsf{T}} Q_j.$$

From the first identity in (1.13), we get

$$w^{s,j} = \underbrace{(x^{s,j})^{1/2} Q_j^{\mathsf{T}}}_{L_{x^{s,j}}} V_j \Sigma_j^{-1} V_j^{\mathsf{T}} \underbrace{Q_j (x^{s,j})^{1/2}}_{L_{x^{s,j}}^{\mathsf{T}}} = L_{x^{s,j}} V_j \Sigma_j^{-1} V_j^{\mathsf{T}} L_{x^{s,j}}^{\mathsf{T}}.$$

Finally

$$\boxed{w^{s,j} = G_j G_j^{\mathsf{T}}, \qquad \text{where } G_j := L_{x^{s,j}} V_j \Sigma_j^{-1/2}.} \tag{1.18}$$

The vector component w^l of w is easily computed by (1.16).

The matrices G_j and their transpose intervene below in (1.23), for which it is useful to introduce the vectors in $\mathbb{E}$:

$$g := (G_1, \ldots, G_s, (w^l)^{1/2}) \qquad \text{and} \qquad \tilde{g} := (G_1^{\mathsf{T}}, \ldots, G_s^{\mathsf{T}}, (w^l)^{1/2}), \tag{1.19}$$

allowing us to write

$$w = g \bullet \tilde{g}. \tag{1.20}$$

1.2.4 Computation of the NT direction

A standard way of solving the system (1.17) defining the NT direction d consists in eliminating first d_x using the second and third equation, to get

$$A(w \bullet d_s \bullet w) = A(\mu s^{-1} - x)$$

and eliminating next d_s using the first equation:

$$A(w \bullet A^*(d_y) \bullet w) = A(x - \mu s^{-1}) \tag{1.21}$$

Once d_y is known, d_s can be computed by the first equation in (1.17) and then d_x by the third equation in (1.17). So let us concentrate on ways of computing d_y by (1.21).

We assume that A is surjective, which implies that (1.21) is a linear system in d_y with a positive definite linear operator, hence having a unique solution. Here are two ways of computing d_y by (1.21).

- A first way of computing the solution d_y to (1.21) is to write this equation as the linear system in d_y:

$$\boxed{\mathcal{M}(d_y) = A(x - \mu s^{-1}),} \tag{1.22}$$

  where $\mathcal{M} : d_y \in \mathbb{R}^m \mapsto A(w \bullet A^*(d_y) \bullet w) \in \mathbb{R}^m$ is a symmetric positive definite (hence nonsingular) linear map and to form the "compact" (when m is small) $m \times m$ coefficient matrix $\mathcal{M}$. For i and $j \in [1\,{:}\,m]$, there holds

$$\begin{aligned} \mathcal{M}_{kl} &= [A(w \bullet A^*(e^l) \bullet w)]_k \\ &= \langle a_k, w \bullet a_l \bullet w \rangle_{\mathbb{E}} \qquad [(1.6) \text{ and } (1.7)] \\ &= \langle a_k, g \bullet \tilde{g} \bullet a_l \bullet g \bullet \tilde{g} \rangle_{\mathbb{E}} \qquad [(1.20)] \\ &= \langle \tilde{g} \bullet a_k \bullet g, \tilde{g} \bullet a_l \bullet g \rangle_{\mathbb{E}}. \end{aligned} \tag{1.23}$$

  Then, one takes the Cholesky factorization $\mathcal{M} = L_{\mathcal{M}} L_{\mathcal{M}}^{\mathsf{T}}$ and solve (1.21).

- Another way of computing d_y is to observe that (1.21) or ...

1.3 Implementation

1.3.1 Data representation

We propose to follow the apparent data structure used in SeDuMi.

- An element $x \in \mathbb{E}$ is represented by a large *vector* of dimension $n_{\mathbb{F}}$. This large vector is formed of the matrices $x^{s,j} \in \mathcal{S}^{n_j}$ represented by a vector containing its $n_j \times n_j$ elements (not only its symmetric part having $n_j(n_j+1)/2$ elements) and the vector $x^l \in \mathbb{R}^{n_l}$.

  This representation has the advantage of making many computations easy. For instance the scalar product $\langle x, s \rangle$, for x and $s \in \mathbb{E}$, is obtained in Matlab by `x'*s` if `x`

is the $n_{\mathbb{F}}$ vector representing x and `s` is the $n_{\mathbb{F}}$ vector representing s. This makes the design of the piece of software easier and also makes it more efficient computationally (since there is no loop).

The above representation of the elements of $\mathbb{E}$ makes it necessary to often switch between the vector and matrix representations of matrices. This can be done with the Matlab functions `mat` and `vec` (see below), probably inexpensively if the result is not safeguarded.

- The linear operator $A : \mathbb{E} \to \mathbb{R}^m$ introduced in the primal and dual problems in (1.5) is first represented by the vectors $a_i \in \mathbb{E}$, for $i \in [1:m]$, using (1.6). Then each vector $a_i \in \mathbb{E}$ is represented as an $n_{\mathbb{F}}$ vector, as described in the first point, whose transpose forms the ith row of a matrix `A`.

  With these representations of A and x, the value $A(x)$ is obtained in Matlab by `A*x` and the value of $A^*(y)$ is obtained by `A'*y`, which are elegant expressions, easy to program and debug, and computationally efficient.

1.3.2 Calling statement

We propose to give to the sdco solver, the following Matlab function structure

```
[x,y,s,info] = sdco (A,b,c,K,x0,y0,options)
```

where

- `A` is an $m \times n_{\mathbb{F}}$ representation of the linear operator A, as described in section 1.3.1;
- `b` is an m real vector, representing the right hand side of the equality constraint in (P);
- `c` is an $n_{\mathbb{F}}$ vector representing the vector $c \in \mathbb{E}$, which determines the linear objective in (P);
- `K` is a Matlab structure describing the structure of the vectors $x \in \mathbb{E}$:
  - `K.s` is the Matlab vector `[n1; n2; ...; ns]`, providing the orders `nj` $= n_j$, for $j \in [1:s]$, of the matrices $x^{s,j}$;
  - `K.l` is the length n_l of the vector x^l;
- `x0` is an $n_{\mathbb{F}}$ vector representing the vector $x_0 \in \mathbb{E}$, which is the initial guess of the primal solution x;
- `y0` is an m-vector, which is the initial guess of the dual solution $y \in \mathbb{R}^m$; s_0 is then deduced form y_0 by $s_0 := c - A^*(y_0)$;
- `options` is a structure of possible options, which will be described in other session; this one can be used to select some features of the algorithm, like the threshold to decide optimality;
- `x` is an $n_{\mathbb{F}}$ vector representing a vector $x \in \mathbb{E}$, which gives the primal solution x if any;
- `y` is an m vector, giving the dual solution y if any;
- `s` is an $n_{\mathbb{F}}$ vector, giving the dual solution s if any; it is linked to y by $s = c - A^*(y)$;

- `info` is a structure providing information of the run (success, failure, primal/dual infeasibility, primal/dual unboundedness, duality gap, *etc*).

1.3.3 Matlab functions

Here are a few useful Matlab functions.

- `L = chol(A,'lower')` computes the Cholesky factorization of $A = LL^T$, where L is lower triangular (only the lower triangular part of A is used).
- `[U,S,V] = svd(A)` computes the singular value decomposition (SVD) of a real $n_{\mathbb{F}} \times n_{\mathbb{F}}$ matrix $A = USV^T$, where U and V are orthogonal matrices and S is diagonal with nonnegative entries (the singular values of A).
- `M = mat(v)` converts an n^2 vector `v` into an $n \times n$ matrix `M`, such that the columns of `M` are stacked below each other in `v`. In some version of Matlab, this function does not exist but can be simulated using `reshape`. The reverse operation is realized by `vec`.
- `[Q,R] = qr(A,0)` computes the QR factorization of a real $m \times n$ matrix $A = QR$, where Q is orthogonal and R is upper triangular. The second zero argument is useful when $m > n$ (our case), in which case, only the first n column of Q and the first n rows of R are computed.
- `v = vec(M)` converts an $m \times n$ matrix `M` into an mn vector `v`, stacking the columns of `M` below each other in `v`. In some version of Matlab, this function does not exist but can be simulated using `reshape`. The reverse operation is realized by `mat`.

It is useful to introduce a Matlab function computing the Hadamard product of two vectors x and s in $\mathbb{E}$:

```
[hdot] = sdco_hdot (x,s,K)
```

Using this function and introducing a matrix B whose kth column is $\tilde{g} \bullet a_k \bullet g$, the expression (1.23) of $\mathcal{M}$ simply reads $B^T B$.

1.3.4 Test case 1a: an easy SDO problem of dimension 3

Consider the semidefinite optimization (SDO) problem in $y \in \mathbb{R}^3$, written in standard dual form

$$\begin{cases} \sup\ e^T y \\ \begin{pmatrix} 1-y_1 & -y_3 & -y_2 \\ -y_3 & 1-y_2 & 0 \\ -y_2 & 0 & 1-y_3 \end{pmatrix} \succcurlyeq 0, \end{cases}$$

where $e \in \mathbb{R}^3$ is the vector of all ones. It is easily verified that $x_0 = I^3$ and $(y_0, s_0) = (0, I^3)$ are strictly feasible for the primal and dual problems respectively, so that these problems have a solution without duality gap. There holds $b^T y_0 = 0 < 3 = \langle c, x_0 \rangle$, so that $z_0 := (x_0, y_0, s_0)$ is not a primal-dual solution.

1.3.5 Test case 1b: a simple LO problem of dimension 2

Consider the following linear optimization (LO) problem in only 2 variables and a single affine constraint, which reads

$$\begin{cases} \inf\ x_1 + x_2 \\ x_1 + 2x_2 = 1 \\ x \geqslant 0. \end{cases}$$

The initial primal and dual variables are

$$x_0 := \begin{pmatrix} 1/3 \\ 1/3 \end{pmatrix} \qquad \text{and} \qquad y_0 := 0,$$

which are strictly feasible points.

1.3.6 Test case 1c: two SDO problems and one LO problem in parallel

The goal of this problem is to test to capacity of the developed solver to deal simultaneously with two positive semidefinite matrices and one nonnegative vector as primal variables. The problem consists in solving in parallel two copies of the semidefinite optimization problems of test case 1a, each one starting from a different matrix, and one addition linear optimization (LO) problem.

Denote by c the gradient of the objective of the primal expression of test case 1a and by $A(x) = b$ its affine constraint. Then the primal expression of test case 1c is the following problem in $x = (x^{s,1}, x^{s,2}, x^l) \in \mathcal{S}^3 \times \mathcal{S}^3 \times \mathbb{R}^2$

$$\begin{cases} \inf \langle c, x^{s,1} \rangle + \langle c, x^{s,2} \rangle + (e^2)^{\mathsf{T}} x^l \\ A(x^{s,1}) = b \\ A(x^{s,2}) = b \\ x^l_1 + 2x^l_2 = 1 \\ x^{s,1} \succcurlyeq 0, \quad x^{s,2} \succcurlyeq 0, \quad x^l \geqslant 0, \end{cases}$$

The initial primal and dual variables are

$$x_0^{s,1} := I^3, \quad x_0^{s,1} := \begin{pmatrix} 1 & 0 & 1/4 \\ 0 & 1/2 & 0 \\ 1/4 & 0 & 1 \end{pmatrix}, \quad x_0^l := \begin{pmatrix} \frac{5-\sqrt{17}}{2} \\ \frac{-3+\sqrt{17}}{4} \end{pmatrix}, \quad \text{and} \quad y_0 := -\tfrac{1+\sqrt{17}}{4},$$

which are strictly feasible points.

Notes

There are various Matlab SDCO solvers. Having a look at SeDuMi [37, 35] is certainly a good idea, since sdco may be viewed as a reduced feature version of that solver. Another source of inspiration is SDPT3 [40].

Questions

When a question starts with a number within braces, the number gives an *indication* on the difficulty of the question, *with respect to the chapter* (some questions will be easier to answer when subsequent chapters will be known). This indication ranges from 1 to 5: 1 for easy-or-classical and short arguments, 2 for easy-or-classical arguments, 3 for arguments requiring specific knowledge, 4 for more difficult arguments, 5 for very difficult arguments or when a advanced computer program must be written to answer the question.

1.1. {1} *Adjoint of A.* Show that the adjoint of the linear map $A : \mathbb{E} \to \mathbb{R}^m$ defined by (1.6) is given by (1.7) when $\mathbb{R}^m$ is equipped with the Euclidean scalar product.

1.2. {2} *Towards a square optimality system.* Show the equivalence (1.9) when x and s are in the cone K.

1.3. {2} *Unique central point.* Show that (1.11) has a unique solution $z = (x, y, s) \in \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$, provided A is surjective.

1.4. {2} *Weight w.* Show that, when x and $s \in K^s$, the identity (1.13) has a unique solution $w \in K^s$, which is given by (1.14).

1.5. {5} *On test cases* 1*a,* 1*b, and* 1*c.* Consider test cases 1*a*, 1*b*, and 1*c*. Let $z_0 := (x_0, y_0, s_0)$ and d_0 be the NT direction computed at z_0 for some μ.

1) {1} Is z_0 on the central path?
2) {5} Compute the NT direction for $\mu = 0$ by a computer program.
3) {1} Is $z_1 = z_0 + d_0$ a primal-dual solution to the problem if $\mu = 0$?

2 A predictor-corrector algorithm

2.1 Algorithmic techniques

2.1.1 The closest central point

In a path-following algorithm, the first question that arises deals with the determination of the central point $z(\mu)$ to which the current iterate $z := (x, y, s)$ is the closest. Indeed, in such an algorithm, the next iterate aims a point on the central path $\mathscr{C}$ and gets close to it by a Newton step, provided this central point is not too far from z. If we want the iteration to make a progress towards the solution it is necessary that this aimed central point be closer to the solution than the central point that is the closest to z, hence the necessity to determine the latter.

Finding the closest central point is not a well defined concept, actually, since the central path is not a convex set, making the projection on $\mathscr{C}$ not well defined. To overcome the difficulty, we look instead to the closest central point in a transformed space in which the central path becomes a half-line. The transformation is

$$\tau : z = (x, y, s) \in \mathcal{F} \mapsto x \bullet s \in \mathbb{F}.$$

This transformation is certainly not a bijection, but it turns out to be useful, which is enough to make it appropriate. Since $\mathscr{C} := \{z \in \mathcal{F} : x \bullet s \in \mathbb{R}_{++}I\}$, it follows that $\tau(\mathscr{C}) = \mathbb{R}_{++}I$. Then, by the convexity and closedness of $\overline{\tau(\mathscr{C})}$, it makes sense to look for the point in $\overline{\tau(\mathscr{C})}$ that is the closest to $\tau(z)$. That problem simply reads

$$\min_{\mu \geqslant 0} \|x \bullet s - \mu e\|_{\mathbb{F}}^2. \tag{2.1}$$

It has a unique solution (see question 2.1), which is

$$\overline{\mu}(z) := \frac{\langle x, s \rangle_{\mathbb{E}}}{n_c}, \tag{2.2a}$$

where n_c is given by (1.4). The *closest central point* is therefore "considered" to be the one on the central path with the parameter $\mu = \overline{\mu}(z)$. This is not irrelevant, since when $z \in \mathscr{C}$, it is clear that the closest central point to z in the preceding sense is z itself; reassuring.

2.1.2 Neighborhood of the central path

Experimentation has convinced the algorithmicians that it is not good to let the iterates going too far from the central path $\mathscr{C}$ or, more correctly, too close to the boundary of the feasible set $\mathcal{F}$ (it is not the same thing actually, since the "central" path may be close to the boundary of $\mathcal{F}$ [8]). When z is close to that boundary, the stepsize along the Newton direction may be very small, so much as to prevent any progress to the solution: the iterates get stuck on the boundary of $\mathcal{F}$. For this reason, the iterates are maintained in some neighborhood of the central path, often the one that we now define. Recall that the central path is an analytic concept (not a geometric one, depending only on the feasible set), so that this is a second best solution.

Observe first that for a point $z \in \mathcal{F}^s$:

$$x \bullet s = \mu I \iff v \bullet v = \mu I \iff \mu^{1/2} v^{-1} = \mu^{-1/2} v, \tag{2.3}$$

where $v := w^{-1/2} \bullet x \bullet w^{-1/2} = w^{1/2} \bullet s \bullet w^{1/2} \in K^s$ was already defined in (1.15) and $w \in K^s$ was already defined in (1.14) as the unique solution to (1.13). Then one defines a *proximity measure* of the central point $z(\mu)$ by ([15] and [7; § 7.1]):

$$\delta_\mu : z \in \mathcal{F}^s \mapsto \delta_\mu(z) := \frac{1}{2} \left\| \mu^{1/2} v^{-1} - \mu^{-1/2} v \right\|_{\mathbb{E}} \in \mathbb{R}. \tag{2.4}$$

Observe now that for a point $z \in \mathcal{F}^s$:

$$z \in \mathscr{C} \iff x \bullet s = \overline{\mu}(z) I. \tag{2.5}$$

Therefore, it makes sense to define a *proximity measure* of the central path by

$$\delta : z \in \mathcal{F}^s \mapsto \delta(z) := \delta_{\overline{\mu}(z)}(z). \tag{2.6}$$

Accordingly, for $\theta \in [0, 1[$, one introduces the following neighborhood of the central path

$$\mathcal{V}(\theta) := \{ z \in \mathcal{F}^s : \delta(z) \leqslant \theta \}.$$

We note that, since $w = g \bullet \tilde{g}$ by (1.20), there holds

$$\delta(z) = \frac{1}{2} \left\| \tilde{g} \bullet \left(\overline{\mu}(z)^{1/2} x^{-1} - \overline{\mu}(z)^{-1/2} s \right) \bullet g \right\|_{\mathbb{E}}, \tag{2.7}$$

so that the distance can be computed efficiently.

2.1.3 A predictor-corrector algorithm

The predictor-corrector algorithms are the most often implemented interior-point methods. We essentially give the description of the *Mizuno-Todd-Ye predictor-corrector method* [26, 7], whose iteration is composed of two phases: a corrector phase, followed by a predictor phase (in reverse order, in a way).

- The *corrector phase* starts with an iterate $z \in \mathcal{V}(1/3)$. Its goal is to compute an intermediate iterate z', close enough to the central path, so that the predictor step that follows will compute a subsequent iterate z_+ again in the neighborhood $\mathcal{V}(1/3)$. This goal is achieved by making a displacement along the NT direction with $\mu = \overline{\mu}(z)$, say d^c (known as the *centering direction*), and a unit stepsize. The intermediate iterate is then
$$z' := z + d^c.$$

- The *predictor phase*, which follows the corrector step, starts with $z' \in \mathcal{F}^s$ and consists in making a displacement along the NT direction with $\mu = 0$, say d^a (known as the *affine-scaling direction*). The stepsize $\alpha > 0$ along that direction is given by the formula (no need of linesearch, provided a condition given below is fulfilled)
$$\alpha = \frac{2}{1 + [1 + 13\|\tilde{d}_x^a \bullet \tilde{d}_s^a + \tilde{d}_s^a \bullet \tilde{d}_x^a\|_{\mathbb{E}}/(2\overline{\mu}(z'))]^{1/2}}, \tag{2.8a}$$
where
$$\tilde{d}_x^a = w^{-1/2} \bullet d_x^a \bullet w^{-1/2} \qquad \text{and} \qquad \tilde{d}_s^a = w^{1/2} \bullet d_s^a \bullet w^{1/2}.$$
Note that the matrix w is computed at z', not at z (hence by (1.14) with (x, s) changed into (x', s')). It can be shown that
$$\|\tilde{d}_x^a \bullet \tilde{d}_s^a + \tilde{d}_s^a \bullet \tilde{d}_x^a\|_{\mathbb{E}} = \|g^{-1} \bullet d_x^a \bullet d_s^a \bullet g + \tilde{g} \bullet d_s^a \bullet d_x^a \bullet \tilde{g}^{-1}\|_{\mathbb{E}}, \tag{2.8b}$$
where g and $\tilde{g}$ are defined by (1.19), so that the computation of this norm can be done efficiently.

  Note that for the vector parts of d_x^a and d_s^a, there holds
$$\tilde{d}_{x^l}^a \cdot \tilde{d}_{s^l}^a = \tilde{d}_{s^l}^a \cdot \tilde{d}_{x^l}^a = d_{x^l}^a \cdot d_{s^l}^a,$$
so that the weight w does not intervene.

  The stepsize (2.8) is small enough to ensure that the next iterate
$$z_+ := z' + \alpha d^a$$
is in the neighborhood $\mathcal{V}(1/3)$ and large enough to ensure the polynomiality of the algorithm (see below).

Algorithm 2.1.1 (predictor-corrector) A tolerance $\varepsilon > 0$ on $\overline{\mu}(z)$ is given to determine when stopping the iterations. The iteration from z to z_+ starts at some $z \in \mathcal{V}(1/3)$.

1. *Stopping test.* If $\overline{\mu}(z) \leqslant \varepsilon$, stop.

2. *Corrector phase.* Compute the NT direction at z with $\mu = \overline{\mu}(z)$, denote it d^c (centering direction). Take as next iterate

$$z' := z + d^c.$$

3. *Predictor phase.* Compute the NT direction at z' with $\mu = 0$, denote it d^a (affine-scaling direction), and the stepsize α given by (2.8). Set

$$z_+ := z' + \alpha d^a.$$

It can be shown [7] that α given by (2.8) satisfies

$$\alpha \geqslant \frac{2}{1 + [1 + 13n_c/2]^{1/2}},$$

which implies the polynomiality of the algorithm, with the iteration complexity

$$K := \left\lceil \left(1 + \sqrt{1 + \frac{13n_c}{2}}\right) \log \frac{\overline{\mu}(z_0)}{\varepsilon} \right\rceil. \tag{2.9}$$

This means that if the algorithm starts with $z_0 \in \mathcal{V}(1/3)$, it ends up with z_K satisfying $\overline{\mu}(z_K) \leqslant \varepsilon$, where the number K of iterations is given by (2.9).

2.2 Implementation

2.2.1 Recommendations

- It is important to check numerically that the following formula is verified after each step αd from a point $z \in \mathcal{F}^s$, where $\alpha > 0$ and d is the NT direction computed for a certain μ (see question 2.2):

$$\overline{\mu}(z + \alpha d) = (1 - \alpha)\overline{\mu}(z) + \alpha\mu. \tag{2.10}$$

  In particular, $\overline{\mu}(z') = \overline{\mu}(z)$ and $\overline{\mu}(z_+) = (1 - \alpha)\overline{\mu}(z)$, showing that the progress to the solution is made by the prediction step. Observe indeed that z_* is a solution if and only if $z_* \in \mathcal{F}^s$ and $\overline{\mu}(z_*) = 0$, so that the goal of the algorithm is to force the decrease of $\overline{\mu}(z)$.

  Note finally that if $\mu = \overline{\mu}(z)$, then (2.10) shows that $\overline{\mu}(z + \alpha d)$ is the constant $\overline{\mu}(z)$, whatever is $\alpha \geqslant 0$.

- The value of α given by (2.8) may be very close to one, or even equal to one. This usually yields vectors x and/or s that are no longer in the cone K. In this case, the algorithm gets stuck in a Cholesky factorization. Limiting this computed value to

$$\texttt{options.max\_stepsize} = 0.999$$

  or so, may help the algorithm to go further and compute a more precise solution.

2.2.2 Test cases 2: minimum matrix norm

We consider the *minimum matrix norm* problem which reads

$$\min_{v\in\mathbb{R}^p} \|\mathcal{B}(v)\|_2\,, \tag{2.11}$$

where $\mathcal{B}(v) := B_0 + \sum_{i=1}^p v_i B_i$, the matrices $B_i \in \mathbb{R}^{q\times r}$ and $\|\cdot\|_2$ denotes the ℓ_2-norm. This problem is convex, but nonsmooth. It can be expressed as an SDO problem as follows.

Problem (2.11) can be rewritten $\min_{t,v}\{t : \|\mathcal{B}(v)\|_2^2 \leqslant t^2\}$. Using the Schur complement technique, it can be verified that this last problem can also be written as the following SDO problem in dual form [41]:

$$\begin{cases} \max_{(t,v)\in\mathbb{R}\times\mathbb{R}^p} -t \\ \begin{pmatrix} tI^q & \mathcal{B}(v) \\ \mathcal{B}(v)^\mathsf{T} & tI^r \end{pmatrix} \succcurlyeq 0. \end{cases} \tag{2.12}$$

This problem has dimension $n = q+r$ and $m = p+1$. One can also easily find a strictly feasible primal-dual point (question 2.4).

Notes

The proximity measure δ defined by (2.6) was introduced by Jiang [15], extending to semidefinite optimization a similar measure introduced by Jansen et al. [14] in linear optimization.

Questions

2.1. {1} *Formula of* $\overline{\mu}(z)$. Show that the solution to problem (2.1) is given by (2.2).

2.2. {2} *Evolution of* $\overline{\mu}(z)$ *along the NT direction.* Show formula (2.10).

2.3. {5} *On test cases* 1*a,* 1*b, and* 1*c.* Using your solver, find a solution to test cases 1*a*, 1*b*, and 1*c*.

2.4. {5} *On test case* 2.

1) {1} Show that, in standard notation, $z_0 = (x_0, y_0, s_0)$, with

$$x_0 = \tfrac{1}{q+r}\, I^{q+r}, \quad y_0 = (t_0, v_0) = (\|B_0\|_2 + 1, 0), \quad \text{and} \quad s_0 = \begin{pmatrix} t_0 I^q & B_0 \\ B_0^\mathsf{T} & t_0 I^r \end{pmatrix}$$

is a strictly feasible primal-dual point for problem (2.12).

2) {5} Using your solver, find a solution to problem (2.11), equivalent to problem (2.12), with, say $p = 2$, $q = 2$, $r = 2$, and random data for the matrices $B_i \in \mathbb{R}^{q\times r}$, with $i \in [0\,{:}\,2]$.

3 Finding an appropriate starting point

The predictor-corrector algorithm that was presented in section 2.1.3 assumes that the first iterate is strictly feasible and is in some neighborhood $\mathcal{V}(\theta)$ of the central path. In this section, we consider a method to get such a point, provided a strictly feasible primal-dual point $z_0 = (x_0, y_0, s_0)$ is known. This idea is to minimize a primal-dual merit function that has a central point as unique minimizer, starting the minimization process at z_0. The NT direction can be used to force the decrease of that merit function.

3.1 Getting a feasible point close to the central path

We use on $\mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ the scalar product inherited from those on $\mathbb{E}$ and $\mathbb{R}^m$, which at $z = (x, y, s)$ and $z' = (x', y', s')$ is denoted by and takes the value

$$\langle z, z' \rangle = \langle x, x' \rangle_{\mathbb{E}} + y^{\mathsf{T}} y' + \langle s, s' \rangle_{\mathbb{E}}.$$

The associated norm is denoted by $\| \cdot \|$.

3.1.1 A primal-dual merit function

We investigate the simple idea that consists in minimizing *approximately* on $\mathcal{F}^s$ a function

$$\varphi_\mu : \mathbb{E} \times \mathbb{R}^m \times \mathbb{E} \to \mathbb{R} \cup \{+\infty\},$$

whose unique minimizer in $\mathcal{F}^s$ is the central point $z(\mu)$, that is the unique solution to the perturbed optimality system (1.11). There are several possibilities for φ_μ, including a function defined only on $\mathbb{E}$ (see the indication given for solving exercise 1.3). We prefer, however, using a function defined on $\mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ that can be minimized using the NT direction, which has already been implemented.

With that objective in mind, we follow [7; § 7.1] by defining the function φ_μ at $z \in \mathbb{E} \times \mathbb{R}^m \times \mathbb{E}$ by

$$\varphi_\mu(z) := \langle x, s \rangle_{\mathbb{E}} + \mu \, \psi(x \bullet s), \tag{3.1}$$

where $\mu > 0$ is a fixed parameter and $\psi : \mathbb{F} \to \mathbb{R} \cup \{+\infty\}$ is defined at $v \in \mathbb{F}$ by

$$\psi(v) = \sum_{j=1}^{s} \operatorname{ld} v^{s,j} - \sum_{i=1}^{n_l} \log v_i^l.$$

This expression makes use of the *self-concordant barrier* [27, 33] $\mathrm{ld} : \mathbb{R}^{n\times n} \to \mathbb{R}\cup\{+\infty\}$ of the cone of positive definite matrices that are not necessarily symmetric, defined at $M \in \mathbb{R}^{n\times n}$ by

$$\mathrm{ld}(M) = \begin{cases} -\log\det M & \text{if } \det M > 0 \\ +\infty & \text{otherwise.} \end{cases} \tag{3.2}$$

It can be shown that, when $\mathcal{F}^s \neq \varnothing$ and A is surjective, φ_μ is strictly convex on $\mathcal{F}^s$ and has the central point $z(\mu)$ as unique minimizer on $\mathcal{F}^s$ (see exercise 3.1). This has for consequence that generating iterates in $\mathcal{F}^s$ that minimize φ_μ will eventually provide a point in $\mathcal{V}(\theta)$ for some prescribed $\theta > 0$.

3.1.2 Use of the NT direction

Now, the question arises to see whether φ_μ can be minimized using the NT direction. The next result shows that the NT direction d defined at $z \in \mathcal{F}^s$, with the parameter $\mu > 0$, is a descent direction of φ_μ. It also provides with (3.3) the value of the directional derivative of φ_μ along the NT direction.

Proposition 3.1.1 (decrease of the primal-dual barrier along the NT direction) *Let d be the NT direction at a point $z \in \mathcal{F}^s$ for some $\mu > 0$. Then*

$$\langle \nabla\varphi_\mu(z), d\rangle = -4\,\mu\,\delta_\mu(z)^2, \tag{3.3}$$

where δ_μ is defined by (2.4). Furthermore, the following properties are equivalent:

(i) $z \neq z(\mu)$,
(ii) $d \neq 0$,
(iii) $\langle \nabla\varphi_\mu(z), d\rangle < 0$.

Proof. Using point 1 of exercise 3.1, one gets

$$\langle \nabla\varphi_\mu(z), d\rangle = \langle s - \mu x^{-1}, d_x\rangle_{\mathbb{E}} + \langle x - \mu s^{-1}, d_s\rangle_{\mathbb{E}}.$$

Using the scaling vector w defined by (1.14) and the scaled directions $\tilde{d}_x := w^{-1/2} \bullet d_x \bullet w^{-1/2}$ and $\tilde{d}_s := w^{1/2} \bullet d_s \bullet w^{1/2}$, one gets

$$\langle \nabla\varphi_\mu(z), d\rangle = \langle s - \mu x^{-1}, w^{1/2} \bullet \tilde{d}_x \bullet w^{1/2}\rangle_{\mathbb{E}} + \langle x - \mu s^{-1}, w^{-1/2} \bullet \tilde{d}_s \bullet w^{-1/2}\rangle_{\mathbb{E}}.$$

It is known, from (1.15), that $v := w^{-1/2} \bullet x \bullet w^{-1/2} = w^{1/2} \bullet s \bullet w^{1/2}$ and, from (1.17c), that $\tilde{d}_x + \tilde{d}_s = \mu v^{-1} - v$, so that

$$\begin{aligned} \langle \nabla\varphi_\mu(z), d\rangle &= \langle v - \mu v^{-1}, \tilde{d}_x + \tilde{d}_s\rangle_{\mathbb{E}} \\ &= -\|v - \mu v^{-1}\|_{\mathbb{E}}^2 \\ &= -\mu\|\mu^{-1/2} v - \mu^{1/2} v^{-1}\|_{\mathbb{E}}^2 \\ &= -4\mu\delta_\mu(z)^2 \qquad [(2.4)], \end{aligned}$$

which is (3.3). The next equivalences follow easily. □

Note that if μ is set to $\overline{\mu}(z)$, if d is the NT direction computed for that μ, and if a stepsize $\alpha > 0$ is taken along d to force the decrease of φ_μ, then $\overline{\mu}(z + \alpha d) = \overline{\mu}(z)$ by (2.10), so that an NT direction at $z + \alpha d$ with $\mu = \overline{\mu}(z + \alpha d)$ will also be a descent direction of the *same* merit function φ_μ.

Proposition 3.1.2 (distance to the central path after a NT step) *Let d be the NT direction at a point $z \in \mathcal{F}^s$ for $\mu := \overline{\mu}(z)$. Then the distance δ to the central path satisfies the inequality*

$$\delta(z + d) \leqslant \frac{\delta(z)^2}{\sqrt{2(1 - \delta(z)^2)}}. \tag{3.4}$$

Proof. Adapt the proof of [7; lemma 7.4]. □

3.1.3 An algorithm

A consequence of the fact that $z(\mu)$ uniquely minimizes φ_μ (exercise 3.1) and proposition 3.1.1 is that, to get closer to the central path, it makes sense to minimize the function φ_μ, with $\mu = \overline{\mu}(z)$, along the NT direction, using a linesearch technique. Furthermore, from (3.4) in proposition 3.1.2,

$$\delta(z) \leqslant \sqrt{\frac{2\tau^2}{1 + 2\tau^2}} \qquad \Longrightarrow \qquad \delta(z + d) \leqslant \tau\delta(z),$$

so that, choosing $\tau < 1$, the distance decreases linearly without the need to make linesearch, as soon as $\delta(z) \leqslant \sqrt{2\tau^2/(1 + 2\tau^2)}$, which is $< \sqrt{2/3}$. Inequality (3.4) also tells us that the distance $\delta(z)$ converges to zero quadratically.

This discussion leads to the following algorithm, which compute a point in the neighborhood $\mathcal{V}(\theta)$ of the central path, starting from a point some given $z \in \mathcal{F}^s$.

Algorithm 3.1.3 (getting a point in $\mathcal{V}(\theta)$) The algorithm uses three parameters: the scalar $\theta > 0$ specifying the neighborhood to reach, a desired linear speed of convergence $\tau < 1$ close to 1, and a linesearch parameter $\omega > 0$ close to zero. Typically $\theta = 1/3$, $\tau \simeq 0.99$, and $\omega \simeq 10^{-4}$. Let $z \in \mathcal{F}^s$.

Repeat until $z \in \mathcal{V}(\theta)$.

1. *NT direction.* Compute the NT direction at z with $\mu := \overline{\mu}(z)$, denote it d^c.

2. *Stepsize* α. If $\delta(z) \leqslant \sqrt{2\tau^2/(1+2\tau^2)}$, take $\alpha = 1$. Otherwise, find the largest α of the form 2^{-i} with $i \in \mathbb{N}$ such that

$$\varphi_\mu(z + \alpha d^c) \leqslant \varphi_\mu(z) - 4\,\omega\,\mu\,\alpha\,\delta(z)^2. \tag{3.5}$$

3. *Next iterate*. Set the next iterate z to $z + \alpha d^c$.

The previous algorithm hides a serious difficulty, which may occur during the line-search in (3.5), which evaluates the quality of the trial stepsize α. By definition, the function $\varphi_\mu(z) = +\infty$ when $z \notin K^s$. This fact may not be seen by simply evaluating $x^{s,j}s^{s,j}$ (resp. $x_i^l s_i^l$), which may be positive definite (resp. positive) even when $x^{s,j} \not\succ 0$ or $s^{s,j} \not\succ 0$ (resp. when both $x_i^l < 0$ and $s_i^l < 0$). Therefore, the correct evaluation of φ_μ requires to check that, for all $j \in [1:s]$ and all $i \in [1:n_l]$: $x^{s,j} \succ 0$ and $s^{s,j} \succ 0$ (either by the computation of the minimum eigenvalues of $x^{s,j}$ and $s^{s,j}$ or by doing their Cholesky factorizations) and that $x_i^l > 0$ and $s_i^l > 0$.

3.2 Implementation

3.2.1 Test case 3: global minimization of a univariate polynomial

We consider the problem of finding the *global minimum of a univariate real polynomial* $p \in \mathbb{R}[x]$, in the variable $x \in \mathbb{R}$, which reads

$$\min_{x\in\mathbb{R}} \left(p(x) := \sum_{\alpha\in[0:d]} p_\alpha\, x^\alpha \right), \tag{3.6}$$

where $p_\alpha \in \mathbb{R}$ and $d \in \mathbb{N}$ is the degree $\deg p$ of the polynomial p. Problem (3.6) can be rewritten as an SDO problem.

First, we write (3.6) as a problem with an infinite number of constraints:

$$\begin{cases} \max_{s\in\mathbb{R}}\ s \\ p(x) \geqslant s, \quad \forall\, x \in \mathbb{R}. \end{cases}$$

This infinite number of constraints "disappears" if we trivially express them in terms of membership to the cone $\mathcal{P}$ of univariate nonnegative polynomials:

$$\begin{cases} \max_{s\in\mathbb{R}}\ s \\ p - s \in \mathcal{P}. \end{cases} \tag{3.7}$$

It turns out that membership to $\mathcal{P}$ has an LMI representation (this is no longer true for multivariate polynomials, but there are bypasses [30, 19, 21, 2, 3, 22]). That fact is based on the following two properties: a univariate nonnegative polynomial can be written as a *sum of squares* (SOS) of polynomials (proposition 3.2.1, which is no longer

true for multivariate polynomials) and an SOS polynomial can be represented thanks to a positive semidefinite matrix (proposition 3.2.2, which is still true for multivariate polynomials).

Proposition 3.2.1 (nonnegative univariate polynomial) *A univariate polynomial $p \in \mathbb{R}[x]$ is nonnegative on $\mathbb{R}$ if and only if it is of even degree, say $2m$, and reads $p = q^2 + r^2$, with q, $r \in \mathbb{R}[x]$, $\deg q = m$, and $\deg r \leqslant m-1$.*

Proof. The given conditions are clearly sufficient. Let us show that they are necessary.

Being nonnegative on $\mathbb{R}$ the polynomial is necessary of even degree, say $2m$, and the leading coefficient is positive. There is therefore no loss of generality in supposing that this leading coefficient is 1. Then the polynomial can be decomposed in m factors of the form

$$(x-a)^2 + b^2.$$

It is indeed the form of $(x-r)(x-\bar{r})$ when r and $\bar{r}$ are complex conjugate roots $a \pm ib$. On the other hand, any real root has an even multiplicity (otherwise the polynomial would have positive and negative values around the root) and each double root is of the form above with $b = 0$.

The m factors of the form $q^2 + r^2$ are then multiplied successively, by using the following formula

$$(q_j^2 + r_j^2)(q^2 + r^2) = (q_j q + r_j r)^2 + (q_j r - r_j q)^2 =: q_{j+1}^2 + r_{j+1}^2.$$

By induction, we see that $\deg q_j = j$ and $\deg r_j \leqslant j-1$. It is indeed the case for $j = 1$ since $\deg q_1 = 1$ and $\deg r_1 = 0$. Now, be r vanishing or not, $\deg q_{j+1} = \deg q_j + 1 = j+1$. Finally, if $r = 0$, $\deg r_{j+1} = \deg r_j + 1 \leqslant j$ and, if $r \neq 0$, $\deg r_{j+1} \leqslant \max(\deg q_j, \deg r_j + 1) \leqslant j$. □

Proposition 3.2.2 (caracterization of SOS polynomials) *A polynomial $p \in \mathbb{R}[x]$ of degree $\leqslant 2m$ is a sum of r squares of polynomials if and only if there exists a matrix $S \succcurlyeq 0$ of order $m+1$ and of rank $\leqslant r$ such that $p(x) = v_m(x)^\mathsf{T} S v_m(x)$, where $v_m(x)$ is the vector of monomials $(1 \; x \; x^2 \; \ldots \; x^m)^\mathsf{T}$.*

Proof. [⇒] If $p \in \mathbb{R}[x]$ is of degree $\leqslant 2m$ and reads $\sum_{i=1}^r \sigma_i^2$, where $\sigma_i \in \mathbb{R}[x]$, the degrees $\deg \sigma_i \leqslant m$. Therefore, on can find vectors $s_i \in \mathbb{R}^{m+1}$ such that

$$p(x) = \sum_{i=1}^r (s_i^\mathsf{T} v_m(x))^2 = \sum_{i=1}^r v_m(x)^\mathsf{T} s_i s_i^\mathsf{T} v_m(x) = v_m(x)^\mathsf{T} S v_m(x),$$

where $S := \sum_{i=1}^r s_i s_i^\mathsf{T} \succcurlyeq 0$ is of rank $\leqslant r$.

[⇐] Conversely, suppose that $p(x) = v_m(x)^\mathsf{T} S v_m(x)$, with $S \succcurlyeq 0$ of rank $\leqslant r$. The spectral decomposition of $S = \sum_{i=1}^{r} s_i s_i^\mathsf{T}$ allows us to write

$$p(x) = \sum_{i=1}^{r} v_m(x)^\mathsf{T} s_i s_i^\mathsf{T} v_m(x) = \sum_{i=1}^{r} (s_i^\mathsf{T} v_m(x))^2,$$

showing that p is the sum of the squares of at most r polynomials. □

Using the above propositions and setting $s = -t$, problem 3.7 can be written

$$\begin{cases} \min_{(S,t)\in\mathcal{S}^{m+1}\times\mathbb{R}} t \\ p(x) + t = v_m(x)^\mathsf{T} S v_m(x), \qquad \forall\, x \in \mathbb{R} \\ S \succcurlyeq 0. \end{cases} \tag{3.8}$$

Knowing p, the first constraint in (3.8) is an affine constraint on the unknown $t \in \mathbb{R}$ and the unknown real coefficients of S, if we impose equality between the coefficients of the same monomials in both sides of the equality. Therefore, problem (3.8) is almost in the primal form of an SDO problem. The difference is the free variable t, which is only constrained by the affine constraint in (3.8), not by a nontrivial cone.

At least two techniques could be considered to solve this problem, which is not in the standard form of the primal SDO problem.

1. Introduce additional *free variables* $x^f \in \mathbb{R}^p$ in the SDCO model (1.5) (by *free variables*, we mean variables that are forced to satisfy the affine constraint but not to belong to an additional cone).
2. Write $t = u - v$ and impose $u \geqslant 0$, $v \geqslant 0$. The standard SDCO model (1.5) can be used with $s = 1$, $n_1 = m + 1$, $n_l = 2$,

$$x = \begin{pmatrix} S \\ u \\ v \end{pmatrix}, \qquad c = \begin{pmatrix} 0_{m+1} \\ 1 \\ -1 \end{pmatrix}, \tag{3.9}$$

where O_{m+1} is the zero matrix of order $m + 1$, and A is the map linking linearly the cofficients of $p(x) + u - v$ and those of $v_m(x)^\mathsf{T} S v_m(x)$.

We suggest using the second approach, which has the advantage of allowing us to use the so far developped code without modification. Actually, some authors [40] decompose each free variable into the difference of two nonnegative variables, as we did it for t above, since their numeerical treatment seems to introduce difficulties.

Notes

Proposition 3.1.1 is taken from [7; p. 117] and proposition 3.1.2 from [7; lemma 7.4]. The proof of proposition 3.2.1 is taken from [31; VI 44].

Questions

3.1. {3} *Computing a central point by primal-dual minimization.* Suppose that $\mathcal{F}^s \neq \varnothing$ and that A is surjective. Let φ_μ be given by (3.1) for some $\mu > 0$ and ld be given by (3.2). Show that

1) {3} for z and $d \in \mathbb{F} \times \mathbb{R}^m \times \mathbb{F}$, there hold

$$\nabla\varphi_\mu(z) = \begin{pmatrix} s - \mu x^{-1} \\ 0 \\ x - \mu s^{-1} \end{pmatrix} \quad \text{and} \quad \nabla^2\varphi_\mu(z)\, d = \begin{pmatrix} \mu\, x^{-1} \bullet d_x \bullet x^{-1} + d_s \\ 0 \\ d_x + \mu\, s^{-1} \bullet d_x \bullet s^{-1} \end{pmatrix},$$

2) {3} φ_μ is strictly convex on $\mathcal{F}^s$,

3) {3} the central point $z(\mu)$ is the unique minimizer of the problem

$$\inf_{z \in \mathcal{F}^s} \varphi_\mu(z). \tag{3.10}$$

3.2. {5} *Approaching the central path.* Implement algorithm 3.1.3 and try it on the test case 2 defined in section 2.2.2, with various values of the parameters p, q, r, and random data for the matrices $B_i \in \mathbb{R}^{q\times r}$, with $i \in [0\,{:}\,p]$. The data of this test case can be recovered by entering

```
[A,b,c,K,x0,y0] = testcase_2 (p,q,r);
```

where `p`, `q`, `r` are the parameters (p, q, r).

3.3. *Getting the global minimizer of a univariate polynomial.* Write the dual of problem (3.8) and show that this one yields the global minimizer of p, provided this one is unique.

4 Dealing with infeasibility with the big M approach

4.1 Starting from an infeasible point

In all this section, we assume that an initial $x_0 \in \mathbb{E}$ is known, which satisfies the equality constraint

$$A(x_0) = b. \tag{4.1}$$

Having a pair (y_0, s_0) satisfying the affine constraint of the dual problem makes no difficulty, since y_0 can be taken arbitrarily and s_0 can be set to $c - A^*(y_0)$. A way of getting a point $x_0 \in \mathbb{E}$ satisfying (4.1) is described in section 4.1.1.

We then discuss the so-called *big M methods* for solving the SDCO problem from a point $z_0 = (x_0, y_0, s_0)$ satisfying (4.1) and $A^*(y_0) + s_0 = c$ but not the conditions $x_0 \in K^s$ and $s_0 \in K^s$ that would make z_0 strictly feasible and for which the algorithm of chapter 3 could be used. The case when $x_0 \in K^s$ and $s_0 \notin K^s$ is considered in section 4.1.2, the case when $x_0 \notin K^s$ and $s_0 \in K^s$ is considered in section 4.1.3, and the case when $x_0 \notin K^s$ and $s_0 \notin K^s$ is considered in section 4.1.4.

The contents of this section is adapted from [41; § 6].

4.1.1 Getting primal affine feasibility

We assume below that $A : \mathbb{E} \to \mathbb{R}^m$ has the following natural extension from $\mathbb{E}$ to $\mathbb{F}$, denoted

$$A_{\mathbb{F}} : \mathbb{F} \to \mathbb{R}^m.$$

This one is constructed as follows. First, we take the expression (1.6) of A, with vectors $a_i \in \mathbb{E}$, for $i \in [1\,{:}\,m]$. Then, for $x \in \mathbb{F}$, each component $[A_{\mathbb{F}}(x)]_i$ of $A_{\mathbb{F}}(x)$ is defined to be $[A_{\mathbb{F}}(x)]_i = \langle a_i, x\rangle_{\mathbb{F}}$. Obviously, the restriction of $A_{\mathbb{F}}$ to $\mathbb{E}$ is A, which is what we mean by "extension".

Now, if $x_0 \in \mathbb{E}$ satisfying (4.1) is not given by the user of the solver, this one should compute such a point or claim that (4.1) has no solution $x_0 \in \mathbb{E}$. The solver can proceed as follows.

- The first step is to check whether $b \in \mathcal{R}(A_{\mathbb{F}})$. This can be done using linear algebra techniques. If this is not the case, the primal problem is infeasible and there is no reason to go further.
- Otherwise, we claim that (4.1) has a solution (in $\mathbb{E}$). This one can be obtained by first computing $x \in \mathbb{F}$ satisfying the affine constraint

$$A_{\mathbb{F}}(x) = b \tag{4.2a}$$

and then symmetrizing it by taking

$$x_0^l := x^l \qquad \text{and} \qquad x_0^{s,j} = \frac{1}{2}\left(x^{s,j} + (x^{s,j})^{\mathsf{T}}\right), \quad \forall\, j \in [1:s]. \tag{4.2b}$$

It is asked in question 4.1 to verify that $x_0 \in \mathbb{E}$ obtained in that manner is indeed a solution to (4.1).

4.1.2 Starting from a strictly feasible primal point

We assume in this section that a strictly feasible primal point $x_0 \in \mathcal{F}_P^s$ is known, meaning that

$$A(x_0) = b \qquad \text{and} \qquad x_0 \in K^s, \tag{4.3}$$

but that $s_0 \notin K^s$.

Generally speaking, the *big M method* to solve the SDCO problem when a strictly feasible point is not available, using the technique of chapter 3, consists in introducing a modified SDCO problem that has the same solutions as the original problem, provided a parameter M in the modified SDCO problem is "large enough" (hence the words "big M"), and for which it is easy to find a strictly feasible point. The technique has therefore an effect that is similar to an exact penalty method.

In case (4.3) holds, the modified SDCO problem is obtained from a modification of the dual problem, for which a strictly feasible point is not known. The dual problem becomes the problem in $(y, \eta, s) \in \mathbb{R}^m \times \mathbb{R} \times \mathbb{E}$ (the modified parts are in red):

$$\begin{cases} \sup\ b^{\mathsf{T}} y - M_1 \eta \\ A^*(y) - \eta e + s = c \\ s \in K \\ \eta \geqslant 0. \end{cases} \tag{4.4}$$

The new parameter η is used to realize feasibility easily (see below). This nonnegative parameter is then forced to be as small as possible by taking M_1 "sufficiently" large (the rational is that, when $\eta = 0$, the original dual problem (D) is recovered). We claim that

- problem (4.4) can be cast as a standard dual SDCO problem, for which a strictly feasible point is easy to determine,
- its primal-dual solutions are the same as the original problem, provided M_1 is chosen "sufficiently large" and the dual problem is feasible.

Let us see this.

One can cast problem (4.4) as the standard dual SDCO problem with $\tilde{\mathbb{E}} := \mathbb{E} \times \mathbb{R}$, $\tilde{K} = K \times \mathbb{R}_+$, and the dual variable $(\tilde{y}, \tilde{s}) \in \mathbb{R}^{m+1} \times \tilde{\mathbb{E}}$ that solves

$$\begin{cases} \sup\ \tilde{b}^{\mathsf{T}} \tilde{y} \\ \tilde{A}^*(\tilde{y}) + \tilde{s} = \tilde{c} \\ \tilde{s} \in \tilde{K}. \end{cases} \tag{4.5}$$

This problem is identical to problem (4.4) provided the vectors $\tilde{b}$ and $\tilde{y} \in \mathbb{R}^{m+1}$, the adjoint linear map $\tilde{A}^* : \mathbb{R}^{m+1} \to \tilde{\mathbb{E}}$, and the vectors $\tilde{s}$ and $\tilde{c} \in \tilde{\mathbb{E}}$ are defined by

$$\tilde{b} = \begin{pmatrix} b \\ -M_1 \end{pmatrix}, \quad \tilde{y} = \begin{pmatrix} y \\ \eta \end{pmatrix}, \quad \tilde{A}^*(\tilde{y}) = \begin{pmatrix} A^*(y) - \eta e \\ -\eta \end{pmatrix}, \quad \tilde{s} = \begin{pmatrix} s \\ \sigma \end{pmatrix}, \quad \text{and} \quad \tilde{c} = \begin{pmatrix} c \\ 0 \end{pmatrix}.$$

It is easy to get a strictly feasible point for this model by taking

$$\boxed{y_0 \text{ arbitrary}, \quad \eta_0 = \sigma_0 > [\min_K(c - A^*(y_0))]^-, \quad \text{and} \quad s_0 = c - A^*(y_0) + \eta_0 e,}$$

where $\min_K$ has been defined by (1.3) and $\alpha^- := \max(0, -\alpha)$.

We know that (4.5) is the dual of

$$\begin{cases} \inf \langle \tilde{c}, \tilde{x} \rangle_{\tilde{\mathbb{E}}} \\ \tilde{A}(\tilde{x}) = \tilde{b} \\ \tilde{x} \in \tilde{K}, \end{cases} \tag{4.6}$$

where $\tilde{x} = (x, \xi_1) \in \mathbb{E} \times \mathbb{R}$. Now, $\tilde{A} : \tilde{\mathbb{E}} \to \mathbb{R}^{m+1}$ is defined at $\tilde{x}$ by

$$\tilde{A}(\tilde{x}) = \begin{pmatrix} A(x) \\ -\langle e, x \rangle_{\mathbb{E}} - \xi_1 \end{pmatrix}.$$

Therefore, problem (4.6) also reads

$$\begin{cases} \inf \langle c, x \rangle_{\mathbb{E}} \\ A(x) = b \\ \langle e, x \rangle_{\mathbb{E}} + \xi_1 = M_1 \\ x \in K \\ \xi_1 \geqslant 0. \end{cases} \tag{4.7}$$

A strictly feasible point $(x_0, \xi_{1,0})$ for (4.6)-(4.7) can be

$$\boxed{\text{the known } x_0 \qquad \text{and} \qquad \xi_{1,0} = M_1 - \langle e, x_0 \rangle_{\mathbb{E}},}$$

provided $M_1 > \langle e, x_0 \rangle_{\mathbb{E}}$.

The proposed approach consists in solving (4.7)-(4.4), or equivalently (4.6)-(4.5) in standard form, from a strictly feasible primal-dual pair hoping that at the primal-dual solution $(x, \xi_1, y, \eta, s, \sigma)$ the scalar η vanishes, in which case problem (P) is actually solved. If $\eta \neq 0$, one increases M_1 and solve problem (4.7)-(4.4) again. The process is stopped when $\eta = 0$ is found or when M_1 is considered as too large. The latter case may occur if (D) is infeasible.

Algorithm 4.1.1 (getting a solution from a strictly feasible primal point) Let $x \in \mathcal{F}_P^s$ and let $M_1 > \langle e, x\rangle_{\mathbb{E}}$. One iteration of the algorithm is as follows.

1. *Solve* (4.7)-(4.4) from a primal-dual strictly feasible point to get $(x, \xi_1, y, \eta, s, \sigma)$.
2. *Successful stopping test.* If $\eta = 0$, stop and declare that (x, y, s) is a primal-dual solution to (P).
3. *Failure stopping test.* If M_1 is too large, stop and declare that it is likely that *the dual problem* (D) *is infeasible.*
4. *Increase* M_1.

4.1.3 Starting from a strictly feasible dual point

We assume in this section that a strictly feasible dual pair $(y_0, s_0) \in \mathcal{F}_D^s$ is known, meaning that

$$A^*(y_0) + s_0 = c \qquad \text{and} \qquad s_0 \in K^s,$$

but that $x_0 \notin K^s$, and we determine a modified SDCO problem such that it is easy to maintain (y_0, s_0) strictly feasible and to determine a strictly feasible primal point x_0.

Consider the following two equivalent modified primal problems in $(x, \xi_2) \in \mathbb{E} \times \mathbb{R}$:

$$\left\{\begin{array}{l} \inf \langle c, x\rangle_{\mathbb{E}} + M_2\xi_2 \\ A(x) = b \\ x + \xi_2 e \in K \\ \xi_2 \geqslant 0 \end{array}\right. \quad \text{or} \quad \left\{\begin{array}{l} \inf \langle c, x\rangle_{\mathbb{E}} + (M_2 - \langle c, e\rangle_{\mathbb{E}})\,\xi_2 \\ A(x - \xi_2 e) = b \\ x \in K \\ \xi_2 \geqslant 0. \end{array}\right. \tag{4.8}$$

The problem in the left-hand side in (4.8) shows that one tries to have an as small as possible perturbation of x in K, namely $x + \xi_e e$, by forcing the nonnegative scalar ξ_2 to be as small as possible thanks to the positive number M_2 in the objective, which is taken "sufficiently" large. This is indeed desirable, since if $\xi_2 = 0$, the original primal problem (P) is recovered. The equivalent problem in the right-hand side, obtained by the redefinition $x + \xi_2 e \curvearrowright x$, is in a form closer to the standard primal SDCO problem. We claim that

- this problem can be cast as the standard primal SDCO problem, for which a strictly feasible point is easy to define,
- its primal-dual solutions are the same as the original problem, provided M_2 is chosen "sufficiently large" and the primal problem is feasible.

Let us see this.

One can cast the problem as the right-hand side of (4.8) in the standard primal SDCO problem in the variable $\tilde{x} \in \tilde{\mathbb{E}} := \mathbb{E} \times \mathbb{R}$:

$$\begin{cases} \inf \langle \tilde{c}, \tilde{x} \rangle_{\tilde{\mathbb{E}}} \\ \tilde{A}(\tilde{x}) = b \\ \tilde{x} \in \tilde{K}. \end{cases} \tag{4.9}$$

This problem is identical to problem (4.8) provided the vectors $\tilde{c}$ and $\tilde{x} \in \tilde{\mathbb{E}}$, the linear map $\tilde{A} : \tilde{\mathbb{E}} \to \mathbb{R}^m$, and the cone $\tilde{K}$ are defined by

$$\tilde{c} = \begin{pmatrix} c \\ M_2 - \langle c, e \rangle_{\mathbb{E}} \end{pmatrix}, \quad \tilde{x} = \begin{pmatrix} x \\ \xi_2 \end{pmatrix}, \quad \tilde{A}(\tilde{x}) = A(x - \xi_2 e), \quad \text{and} \quad \tilde{K} := K \times \mathbb{R}_+.$$

A strictly feasible point for (4.9) is given by

$$\tilde{x}_0 := \begin{pmatrix} x_0 + \xi_{2,0} e \\ \xi_{2,0} \end{pmatrix},$$

where

$$\boxed{\begin{array}{c} x_0 \in \mathbb{E} \text{ is an arbitrary solution to } A(x_0) = b, \\ \xi_{2,0} > [\min_K(x_0)]^-. \end{array}}$$

We know that (4.9) has for dual the following problem in $(y, \tilde{s}) \in \mathbb{R}^m \times \tilde{\mathbb{E}}$:

$$\begin{cases} \sup \ b^{\mathsf{T}} y \\ \tilde{A}^*(y) + \tilde{s} = \tilde{c} \\ \tilde{s} \in \tilde{K}. \end{cases} \tag{4.10}$$

Now, $\tilde{A}^* : \mathbb{R}^m \to \tilde{\mathbb{E}}$ is defined at $\tilde{x}$ by

$$\tilde{A}^*(y) = \begin{pmatrix} A^*(y) \\ -A(e)^{\mathsf{T}} y \end{pmatrix}.$$

Writing

$$\tilde{s} := \begin{pmatrix} s \\ \sigma_2 \end{pmatrix},$$

we see that problem (4.10) also reads

$$\begin{cases} \sup \ b^{\mathsf{T}} y \\ A^*(y) + s = c \\ -A(e)^{\mathsf{T}} y + \sigma_2 = M_2 - \langle c, e \rangle_{\mathbb{E}} \\ s \in K \\ \sigma_2 \geqslant 0. \end{cases}$$

Eliminating σ_2, the dual problem (4.10) becomes the following problem in $(y, s) \in \mathbb{R}^m \times \mathbb{E}$:

$$\begin{cases} \sup \ b^{\mathsf{T}} y \\ A^*(y) + s = c \\ s \in K \\ \langle e, s \rangle_{\mathbb{E}} \leqslant M_2. \end{cases} \tag{4.11}$$

It can be shown that the variable ξ_2 in (4.8) is a dual variable associated with the constraint $\langle e, s\rangle_{\mathbb{E}} \leqslant M_2$ in (4.11). The given strictly feasible dual pair for (y_0, s_0) for (D) is still a strictly feasible pair for the modified dual problems (4.10) provided

$$\boxed{M_2 > \langle e, s_0\rangle_{\mathbb{E}}.}$$

Note that this setting ensures that $M_2 > 0$ (since $s_0 \in K^s$), as desired.

The proposed approach consists in solving (4.8)-(4.11) from a strictly feasible primal-dual pair (instead of the original problem (P)) hoping that at the solution (x, ξ_2, y, s) the scalar ξ_2 vanishes, in which case problem (P) is actually solved. If $\xi_2 \neq 0$, the penalty parameter M_2 is increased and problem (4.8) is solved again. The process is stopped when a solution with $\xi_2 = 0$ is found or when M_2 is considered as too large, which may occur if (P) is infeasible.

Algorithm 4.1.2 (getting a solution from a strictly feasible dual point) Let $(y, s) \in \mathcal{F}_D^s$ and let $M_2 > \langle e, s\rangle_{\mathbb{E}}$. One iteration of the algorithm is as follows.

1. *Solve* (4.8)-(4.11) from a primal-dual strictly feasible point to get (x, ξ_2, y, s).
2. *Successful stopping test.* If $\xi_2 = 0$, then stop and declare that (x, y, s) is a primal-dual solution to (P).
3. *Failure stopping test.* If M_2 is too large, stop and declare that it is likely that *the primal is infeasible.*
4. *Increase* M_2.

4.1.4 Starting without a strictly feasible point

If no strictly feasible primal and/or dual point is known, then one considers the modified primal problem, obtained by combining the methods of sections 4.1.2 and 4.1.3:

$$\left\{\begin{array}{l} \inf \langle c, x\rangle_{\mathbb{E}} + M_2\xi_2 \\ A(x) = b \\ \langle e, x\rangle_{\mathbb{E}} + \xi_1 = M_1 \\ x + \xi_2 e \in K \\ \xi_1 \geqslant 0 \\ \xi_2 \geqslant 0 \end{array}\right. \quad \text{or} \quad \left\{\begin{array}{l} \inf \langle c, x\rangle_{\mathbb{E}} + (M_2 - \langle c, e\rangle_{\mathbb{E}})\xi_2 \\ A(x - \xi_2 e) = b \\ \langle e, x\rangle_{\mathbb{E}} + \xi_1 - n_c\xi_2 = M_1 \\ x \in K \\ \xi_1 \geqslant 0 \\ \xi_2 \geqslant 0, \end{array}\right. \tag{4.12}$$

in which both M_1 and M_2 are taken "sufficiently" large (initial values are given below). The problem in the left-hand side in (4.12) satisfies the affine constraint $A(x) = b$ (a constraint that can be easily verified by the technique proposed in section 4.1.1) without $x \in K$, but ensures that $x + \xi_2 e$ is in K, where ξ_2 will be small when M_2 is taken large. Its second affine constraint $\langle e, x\rangle_{\mathbb{E}} + \xi_1 = M_1$ is directly inspired from the one appearing in (4.7) and will allow us to find easily a strictly feasible point for the associated dual problem. The problem in the left-hand side of (4.12) is not in standard primal form

because of the constraint $x + \xi_2 e \in K$, which is the reason why it is rewritten as the one in the right-hand side, which is obtained after the change of variable $x + \xi_2 e \curvearrowright x$.

Problem in the right-hand side of (4.12) can be cast as the standard primal SDCO problem in the variable $\tilde{x} \in \tilde{K} := K \times \mathbb{R}^2_+ \subset \tilde{\mathbb{E}} := \mathbb{E} \times \mathbb{R}^2$:

$$\begin{cases} \inf \langle \tilde{c}, \tilde{x} \rangle_{\tilde{\mathbb{E}}} \\ \tilde{A}(\tilde{x}) = \tilde{b} \\ \tilde{x} \in \tilde{K}, \end{cases} \tag{4.13}$$

where the vectors $\tilde{c}$ and $\tilde{x} \in \tilde{\mathbb{E}}$, the linear map $\tilde{A} : \tilde{\mathbb{E}} \to \mathbb{R}^{m+1}$, and the vector $\tilde{b} \in \mathbb{R}^{m+1}$ are defined by

$$\tilde{c} = \begin{pmatrix} c \\ 0 \\ M_2 - \langle c, e \rangle_{\mathbb{E}} \end{pmatrix}, \ \tilde{x} = \begin{pmatrix} x \\ \xi_1 \\ \xi_2 \end{pmatrix}, \ \tilde{A}(\tilde{x}) = \begin{pmatrix} A(x - \xi_2 e) \\ -\langle e, x \rangle_{\mathbb{E}} - \xi_1 + n_c \xi_2 \end{pmatrix}, \text{ and } \tilde{b} = \begin{pmatrix} b \\ -M_1 \end{pmatrix}.$$

The opposite of the second constraint in (4.12) adopted in the definitions above is motivated by the desire to have $-M_1$ as the last component of $\tilde{b}$, which will result in requiring $M_1 > 0$, which goes in the same sense as the condition $M_2 > 0$ required above. A strictly feasible point for the modified primal problems (4.12) or (4.13) is

$$\tilde{x}_0 := \begin{pmatrix} x_0 + \xi_{2,0} e \\ \xi_{1,0} \\ \xi_{2,0} \end{pmatrix},$$

where one takes in order:

$$\boxed{\begin{array}{c} x_0 \in \mathbb{E} \text{ is an arbitrary solution to } A(x_0) = b, \\ \xi_{2,0} > [\min_K(x_0)]^-, \\ M_1 > [\langle e, x_0 \rangle_{\mathbb{E}}]^+, \\ \xi_{1,0} := M_1 - \langle e, x_0 \rangle_{\mathbb{E}}, \end{array}} \tag{4.14}$$

where $\min_K$ has been defined by (1.3), $\alpha^+ = \max(0, \alpha)$, and $\alpha^- = \max(0, -\alpha)$.

We know that the dual of (4.13) reads

$$\begin{cases} \sup \tilde{b}^{\mathsf{T}} \tilde{y} \\ \tilde{A}^*(\tilde{y}) + \tilde{s} = \tilde{c} \\ \tilde{s} \in \tilde{K}. \end{cases} \tag{4.15}$$

where $\tilde{A}^* : \mathbb{R}^{m+4n+3} \to \tilde{\mathbb{E}}$ is defined at $\tilde{y} = (y, \eta) \in \mathbb{R}^m \times \mathbb{R}$ by

$$\tilde{A}^*(\tilde{y}) = \begin{pmatrix} A^*(y) - \eta e \\ -\eta \\ -\langle e, A^*(y) \rangle_{\mathbb{E}} + n_c \eta \end{pmatrix}.$$

Therefore, with $\tilde{s} = (s, \sigma_1, \sigma_2)$, problem (4.15) reads

$$\begin{cases} \sup\ b^{\mathsf{T}} y - M_1 \eta \\ A^*(y) - \eta e + s = c \\ -\eta + \sigma_1 = 0 \\ -\langle e, A^*(y)\rangle_{\mathbb{E}} + n_c \eta + \sigma_2 = M_2 - \langle c, e\rangle_{\mathbb{E}} \\ s \in K \\ \sigma_1 \geqslant 0 \\ \sigma_2 \geqslant 0 \end{cases}$$

or

$$\begin{cases} \sup\ b^{\mathsf{T}} y - M_1 \eta \\ A^*(y) - \eta e + s = c \\ -\eta + \sigma_1 = 0 \\ \langle e, s\rangle_{\mathbb{E}} + \sigma_2 = M_2 \\ s \in K \\ \sigma_1 \geqslant 0 \\ \sigma_2 \geqslant 0. \end{cases} \tag{4.16}$$

Recall that the parameter M_1 fixed by (4.14) is positive, which is a desired property, since one would like to have $\eta = 0$ in the above problem. A strictly feasible point for the modified dual problems (4.15) or (4.16) is $\tilde{y}_0 = (y_0, \eta_0)$ and $\tilde{s}_0 = (s_0, \sigma_{1,0}, \sigma_{2,0})$, where one sets successively

$$\boxed{\begin{array}{c} y_0 \text{ arbitrary}, \\ \eta_0 = \sigma_{1,0} > [\min_K(c - A^*(y_0))]^-, \\ s_0 := c - A^*(y_0) + \eta_0 e, \\ M_2 > [\langle e, s_0\rangle_{\mathbb{E}}]^+, \\ \sigma_{2,0} := M_2 - \langle e, s_0\rangle_{\mathbb{E}}, \end{array}} \tag{4.17}$$

where $\min_K$ has been defined by (1.3), $\alpha^- = \max(0, -\alpha)$, and $\alpha^+ = \max(0, \alpha)$. The setting of M_2 above also ensures its positivity, which was a desired property of problems (4.12).

The goal of the following algorithm is to solve a sequence of primal-dual problems (4.13)-(4.15) or (4.12)-(4.16) by increasing M_1 and M_2 at each loop in order to have $\eta = \xi_2 = 0$, which guarantees that the original primal-dual SDCO problem has been solved. The form of the problems (4.12) and (4.16) clearly indicates that one must increase M_1 if $\eta > 0$ and one must increase M_2 when $\xi_2 > 0$.

Algorithm 4.1.3 (getting a solution from a linear feasible point) One loop of the algorithm is as follows.

1. *Solve* (4.13)-(4.15) or (4.12)-(4.16) from a primal-dual strictly feasible point to get $(x, \xi_1, \xi_2, y, \eta, s, \sigma_1, \sigma_2)$.

2. *Successful stopping test.* If $\xi_2 = 0$ and $\eta = 0$, stop and declare that (x, y, s) is a primal-dual solution to (P).
3. *Failure on dual infeasibility.* If M_1 is too large, stop and declare that it is likely that *the dual problem is infeasible.*
4. *Failure on primal infeasibility.* If M_2 is too large, stop and declare that it is likely that *the primal problem is infeasible.*
5. *Increase* M_1 if $\eta > 0$.
6. *Increase* M_2 if $\xi_2 > 0$.

4.2 Implementation

4.2.1 Recommendations

- Because of the introduction of modified SDCO problems that must be solved by the developed solver, this one has to deal with SDCO problems of various dimensions and data in the same run. It is therefore recommended to have a Matlab function that can find a solution of an SDCO problem when a strictly feasible primal-dual starting point is given (and when the problem has a solution). It is therefore recommended to write this function with the code that has been developed so far in the previous chapter 3. I could have the form

```
[x,y,s] = sdco_solve (A,b,c,K,x0,y0,s0)
```

- To develop the solver, a good idea is to test it on the easy test cases 1 and 2, in which the initial values of x and y has been discarded. Once the solver works correctly on these test cases, you may want to test it on the more special problems of sections 4.2.2, 4.2.3, and 4.2.4.
- The linear operator $\tilde{A}$ does not depend on M_1 and M_2 and can be set outside the loop described in algorithm 4.1.3.
- Check whether the linear constraints $\tilde{A}(\tilde{x}_0) = \tilde{b}$ and $\tilde{A}^*(\tilde{y}) + \tilde{s} = \tilde{c}$ are satisfied at the beginning of *each cycle*, after the setting of $\tilde{A}$, $\tilde{x}_0$, and $\tilde{b}$.

4.2.2 Test case 4a

Consider the problem [41; p. 65]:

$$\left\{ \begin{array}{l} \sup\ y_1 \\ \begin{pmatrix} 0 & y_1 & 0 \\ y_1 & y_2 & 0 \\ 0 & 0 & y_1 + 1 \end{pmatrix} \succcurlyeq 0. \end{array} \right.$$

4.2.3 Test case 4b

Consider the primal SDO problem written in standard form with the following data:

$$C = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad A_1 = \begin{pmatrix} -1 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} -1 \\ 0 \end{pmatrix}.$$

4.2.4 Test case 4c

Consider the primal SDO problem written in standard form with the following data:

$$C = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} 0 \\ 2 \end{pmatrix}.$$

Questions

4.1. {1} *Getting an affine feasible primal point.* Justify the correctness of the procedure (4.2) to get a vector $x_0 \in \mathbb{E}$ satisfying (4.1).

4.2. {5} *Infeasible algorithm.* Implement algorithm 4.1.3 and solve/diagnose the test cases of sections 4.2.2, 4.2.3, and 4.2.4.

5 Shor relaxation of some small OPF problems

5.1 Relaxation of an OPF problem

One of the emblematic instance of the so-called Optimal Power Flow (OPF) problem, in alternating current, consists in minimizing the Joule heating losses in an electricity transmission network, while satisfying the electricity demand, by determining appropriately the powers of the generators installed at given buses (or nodes of the network) [6, 5, 4]. This optimization problem is NP hard, implying that there is (presently) no algorithm to find the (global) solution in polynomial time (i.e., in a reasonable time when the network size is like the one in most countries or union of countries), while there are several good reasons for being only interested in the global solutions.

5.1.1 QCQP formulation

Structurally, one can view (or reduce) the problem to a nonconvex quadratic optimization problem with nonconvex quadratic constraints (sometimes abbreviated by the acronym QCQP for Quadratically Constrained Quadratic Programming). Recalling that any polynomial optimization problem can be written that way [36, 9, 24], one understands the potential difficulty of such a problem.

A QCQP representation of an OPF instance can be obtained by the Matlab function `qcqp_opf` [16], which should be available soon in Matpower [43] (a Matlab package for simulating power systems). This function writes a specified OPF instance as a QCQP problem in complex numbers:

$$\left\{\begin{array}{ll} \inf_z \ z^{\mathsf{H}} C z & \\ z^{\mathsf{H}} A_k z = a_k, & \text{for } k \in [1\,{:}\,m] \\ z^{\mathsf{H}} B_k z \leqslant b_k, & \text{for } k \in [1\,{:}\,p], \end{array}\right. \tag{5.1}$$

where $z \in \mathbb{C}^n$ is a complex vector, C, A_k, and $B_k \in \mathbb{C}^{n\times n}$ are Hermitian matrices[1], a and b are real vectors, and the exponent $\cdot^{\mathsf{H}}$ is used to denote the conjugate transpose. In this representation, n is the number of buses (or nodes of the network).

To solve (5.1) (more precisely a relaxation of it) by an SDCO solver in real numbers, the first task is to rewrite the problem in real numbers (see the discussion in [12] on the interest in having an all-complex approach). For $M \in \mathbb{C}^{n\times n}$, we write $M = \Re(M) + \mathrm{i}\Im(M)$, where $\Re(M)$ and $\Im(M) \in \mathbb{R}^{n\times n}$, and $\mathrm{i} \in \mathbb{C}$ the pure imaginary number

[1] Recall that $M \in \mathbb{C}^{n\times n}$ is Hermitian if $M^{\mathsf{H}} = M$.

($\mathrm{i}^2 = -1$). Similarly, a vector $z \in \mathbb{C}^n$ is decomposed in $z = \Re(z) + \mathrm{i}\Im(z)$, with $\Re(z)$ and $\Im(z) \in \mathbb{R}^n$. It is easy to see that

$$M \text{ is Hermitian} \quad \Longleftrightarrow \quad \begin{cases} \Re(M) \text{ is symmetric,} \\ \Im(M) \text{ is skew symmetric.} \end{cases}$$

With this notation and a Hermitian matric M, there holds

$$z^{\mathsf{H}} M z = \tilde{z}^{\mathsf{T}} \tilde{M} \tilde{z},$$

where

$$\tilde{z} := \begin{pmatrix} \Re(z) \\ \Im(z) \end{pmatrix} \in \mathbb{R}^{2n} \qquad \text{and} \qquad \tilde{M} := \begin{pmatrix} \Re(M) & -\Im(M) \\ \Im(M) & \Re(M) \end{pmatrix} \in \mathcal{S}^{2n}.$$

Note indeed that $\tilde{M}$ is symmetric. Then, problem (5.1) becomes the QCQP in real numbers

$$\begin{cases} \inf_{\tilde{z}} \ \tilde{z}^{\mathsf{T}} \tilde{C} \tilde{z} \\ \tilde{z}^{\mathsf{T}} \tilde{A}_k \tilde{z} = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ \tilde{z}^{\mathsf{T}} \tilde{B}_k \tilde{z} \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p]. \end{cases} \tag{5.2}$$

The dimension of this problem is twice the one of (5.1).

5.1.2 Shor relaxation

The second task is to define an SDO relaxation of (5.2). Recalling that $\langle \cdot, \cdot \rangle_{\mathcal{S}^{2n}}$ denotes the scalar product on the space of symmetric matrices (see section 1.1.1), problem (5.2) reads

$$\begin{cases} \inf_{\tilde{z}} \ \langle \tilde{C}, \tilde{z}\tilde{z}^{\mathsf{T}} \rangle_{\mathcal{S}^{2n}} \\ \langle \tilde{A}_k, \tilde{z}\tilde{z}^{\mathsf{T}} \rangle_{\mathcal{S}^{2n}} = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ \langle \tilde{B}_k, \tilde{z}\tilde{z}^{\mathsf{T}} \rangle_{\mathcal{S}^{2n}} \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p] \end{cases}$$

or

$$\begin{cases} \inf_{\tilde{X}} \ \langle \tilde{C}, \tilde{X} \rangle \\ \langle \tilde{A}_k, \tilde{X} \rangle = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ \langle \tilde{B}_k, \tilde{X} \rangle \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p] \\ \tilde{X} \succcurlyeq 0 \\ \operatorname{rank}(\tilde{X}) \leqslant 1. \end{cases}$$

The rank constraint of this formulation is very annoying (it takes integer values, hence is discontinuous). The Shor relaxation of the problem drops that constraint and therefore reads

$$\begin{cases} \inf_{\tilde{X}} \ \langle \tilde{C}, \tilde{X} \rangle \\ \langle \tilde{A}_k, \tilde{X} \rangle = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ \langle \tilde{B}_k, \tilde{X} \rangle \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p] \\ \tilde{X} \succcurlyeq 0. \end{cases}$$

We still have to write the relaxed problem in the standard primal SDCO form in (1.5), which looks like the closest to the previous one. This form can be obtained by

introducing p *slack variables* $v_k \in \mathbb{R}$, for $k \in [1:p]$, and by writing the problem as follows

$$\begin{cases} \inf_{(\tilde{X},v)} \langle \tilde{C}, \tilde{X} \rangle \\ \langle \tilde{A}_k, \tilde{X} \rangle = a_k, \quad \text{for } k \in [1:m] \\ \langle \tilde{B}_k, \tilde{X} \rangle + v_k = b_k, \quad \text{for } k \in [1:p] \\ \tilde{X} \succcurlyeq 0 \\ v \geqslant 0, \end{cases} \tag{5.3}$$

which is in the primal form (1.5), in the unknown $(\tilde{X}, v)$.

5.2 Test cases

The real SDO relaxation of the OPF problem, namely the standard primal SDCO form deduced in (5.3), is available by

```
[A,b,c,K,x0,y0] = testcase_5 (testcase);
```

where `testcase` is a string giving the name of a Matpower test case (see the table below for a short list), `A`, `b`, and `c` stand for the data A, b, and c of the standard primal SDCO problem in (1.5), while `x0` and `y0` are empty variables (no initial primal-dual point is provided).

Table 5.1 lists the Matpower test cases that have been implemented and gives some

Problem name	n	m	p	$v_{\texttt{mips}}$	v_{Shor}
`caseWB2`	2	2	8	–	8.9082
`caseWB5`	5	6	18	13.780	9.9095
`case9`	9	12	30	373.83	373.83

Table 5.1. Some Matpower test cases.

of their features:

- the name of the problem (to put between simple quotes as argument `testcase` of the function `testcase_5` above),
- the dimensions n, m, p of (5.1) (the number n is also the number of buses of the network),
- the optimal value $v_{\texttt{mips}}$ obtained by the Matlab local interior-point solver `mips` ("–" means that the solver fails),
- and the optimal value v_{Shor} obtained by the Shor relaxation (5.3).

The optimal value of the problem is necessarily in the interval $[v_{\text{Shor}}, v_{\texttt{mips}}]$. Therefore the optimal value is obtained by the local solver `mips` in `case9`, but the situation is uncertain in `caseWB5`.

The Shor relaxation of a QCQP can be viewed as the Lasserre [19] (or moment-sos) relaxation of degree one [20]. For the OPF problem, when the degree one relaxation of

its QCQP formulation is inexact (i.e., it does not provide the solution to the original QCQP), it is often the case that the degree 2 moment-sos relaxation is exact [17].

References

[1] M.F. Anjos, J.B. Lasserre (2012). *Handbook on Semidefinite, Conic and Polynomial Optimization*. Springer. [doi]. 1

[2] M.F. Anjos, J.B. Lasserre (2012). *Introduction to semidefinite, conic and polynomial optimization*, volume 166 of *International Series in Operations Research & Management Science*, chapter 1. Springer. 28

[3] G. Blekherman, P.A. Parrilo, R.R. Thomas (2013). *Semidefinite Optimization and Convex Algebraic Geometry*. MOS-SIAM Series on Optimization. SIAM and MPS, Philadelphia. [doi]. 28

[4] F. Capitanescu (2016). Critical review of recent advances and further developments needed in AC optimal power flow. *Electric Power Systems Research*, 136, 57–68. [doi]. 43

[5] F. Capitanescu, J.L. Martinez Ramos, P. Panciatici, D. Kirschen, A. Marano Marcolini, L. Platbrood, L. Wehenkel (2011). State-of-the-art, challenges, and future trends in security-constrained optimal power flow. *Electric Power Systems Research*, 81(8), 1731–1741. [doi]. 43

[6] M.J. Carpentier (1962). Contribution à l'étude du dispatching économique. *Bulletin de la Société Française des Électriciens*, 8(3), 431–447. 43

[7] E. de Klerk (2002). *Aspects of Semidefinite Programming - Interior Point Algorithms and Selected Applications*. Kluwer Academic Publishers, Dordrecht. [doi]. 10, 20, 22, 25, 27, 30

[8] A. Deza, E. Nematollahi, R. Peyghami, T. Terlaky (2006). The central path visits all the vertices of the Klee-Minty cube. *Optimization Methods and Software*, 21, 849–863. [doi]. 20

[9] C. Ferrier (1998). Hilbert's 17th problem and best dual bounds in quadratic minimization. *Cybernetics and Systems Analysis*, 34, 696–709. [doi]. 43

[10] J.Ch. Gilbert (2016). *Fragments d'Optimisation Différentiable – Théorie et Algorithmes*. Syllabus de cours à l'ENSTA, Paris. [internet]. 9

[11] J.Ch. Gilbert (2016). *Step by step design of an interior point solver in semidefinite optimization – Application to the Shor relaxation of some small OPF problems*. Lecture notes of the Master-2 "Optimization" at Paris-Saclay University, Paris, France. [doi]. 1

[12] J.Ch. Gilbert, C. Josz (2016). Plea for a semidefinite optimization solver in complex numbers. Research report (to appear), INRIA-Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France. 43

[13] C. Helmberg, F. Rendl, R.J. Vanderbei, H. Wolkowicz (1996). An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2), 342–361. [doi]. 10

[14] B. Janssen (1994). Primal-dual algorithms for linear programming based on the logarithmic barrier method. *Journal of Optimization Theory and Applications*, 83, 1–26. 23

[15] J. Jiang (1998). A long step primal-dual path following method for semidefinite programming. *Operations Research Letters*, 23(1-2), 53–62. [doi]. 20, 23

[16] C. Josz, S. Fliscounakis, J. Maeght, P. Panciatici (2015). The `Matlab` function `qcqp_opf`. Piece of software, Réseau de Transport d'Electricité, France. Personal communication. 43

[17] C. Josz, J. Maeght, P. Panciatici, J.Ch. Gilbert (2015). Application of the moment-SOS approach to global optimization of the OPF problem. *IEEE Transactions on Power Systems*, 30(1), 463–470. [doi]. 46

[18] M. Kojima, S. Shindoh, S. Hara (1997). Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM Journal on Optimization*, 7, 86–125. 10

[19] J.B. Lasserre (2001). Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11, 796–817. [doi]. 28, 45

[20] J.B. Lasserre (2001). Convergent LMI relaxations for nonconvex quadratic programs. Technical report. Graduate seminar at MIT, fall 2001. http://www.mit.edu/~6.454/www_fall_2001/cmcaram/lasserre_1.pdf. 45
[21] J.B. Lasserre (2010). *Moments Positive Polynomials and Their Applications*. Imperial College Press Optimization Series 1. Imperial College Press. 28
[22] J.B. Lasserre (2015). *An Introduction to Polynomial and Semi-Algebraic Optimization*. Cambridge Texts in Applied Mathematics. Cambridge University Press. 28
[23] C.-J. Lin, R. Saigal (1995). A predictor-corrector method for semi-definite linear programming. Technical Report 95-20, Department of Industrial & Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109-2117, USA. 10
[24] R. Madani, G. Fazelnia, J. Lavaei (2014). Rank-2 matrix solution for semidefinite relaxations of arbitrary polynomial optimization problems. Technical report. 43
[25] H.D. Mittelmann (2012). *The state-of-the-art in conic optimization software*, volume 166 of *International Series in Operations Research & Management Science*, chapter 23. Springer. 1
[26] S. Mizuno, M.J. Todd, Y. Ye (1993). On adaptive-step primal-dual interior-point algorithms for linear programming. *Mathematics of Operations Research*, 18(4), 964–981. [doi]. 20
[27] Y.E. Nesterov, A.S. Nemirovskii (1994). *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM Studies in Applied Mathematics 13. SIAM, Philadelphia, PA, USA. 26
[28] Y.E. Nesterov, M.J. Todd (1997). Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research*, 22(1), 1–42. [doi]. 10
[29] Y.E. Nesterov, M.J. Todd (1998). Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8(2), 324–364. [doi]. 10
[30] P.A. Parrilo (2000). On a decomposition for multivariable forms via LMI methods. In *Proceedings of the American Control Conference*. 28
[31] G. Pólya, G. Szegö (1976). *Problems and Theorems in Analysis II*. Springer-Verlag, Berlin. 30
[32] F. Rendl, R. Sotirov, H. Wolkowicz (2002). A simplified/improved HKM direction for certain classes of semidefinite programming. Technical report. 10
[33] J. Renegar (2001). *A Mathematical View of Interior-Point Methods in Convex Optimization*. MPS-SIAM Series on Optimization 3. SIAM. 26
[34] S.M. Robinson (1976). Stability theory for systems of inequalities, part II: differentiable nonlinear systems. *SIAM Journal on Numerical Analysis*, 13, 497–513. 9
[35] SeDuMi. [internet]. 16
[36] N.Z. Shor (1987). Class of global minimum bounds of polynomial functions. *Kibernetica*, 6, 9–11. [English translation: *Cybernetics*, 23 (1987) 731-734]. 43
[37] J.F. Sturm (1999). Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11, 625–653. 16
[38] J.F. Sturm, S. Zhang (1999). Symmetric primal-dual path-following algorithms for semidefinite programming. *Applied Numerical Mathematics*, 29(3), 301–315. [doi]. 10
[39] M.J. Todd, K.-C. Toh, R.H. Tütüncü (1998). On the Nesterov-Todd direction in semidefinite programming. *SIAM Journal on Optimization*, 8, 769–796. [doi]. 10, 12
[40] K.-C. Toh, M.J. Todd, R.H. Tütüncü (2012). On the implementation and usage of SDPT3 - a Matlab software package for semidefinite-quadratic-linear programming, version 4.0. In M.F. Anjos, J.B. Lasserre (editors), *Handbook on Semidefinite, Conic and Polynomial Optimization*, International Series in Operations Research and Management Science 166, pages 715–754. Springer. [doi]. 1, 16, 30
[41] L. Vandenberghe, S. Boyd (1996). Semidefinite programming. *SIAM Review*, 38, 49–95. [doi]. 23, 33, 41
[42] H. Wolkowicz, R. Saigal, L. Vandenberghe (editors) (2000). *Handbook of Semidefinite Programming – Theory, Algorithms, and Applications*, volume 27 of *International Series in Operations Research & Management Science*. Kluwer Academic Publishers. 1
[43] R.D. Zimmerman, C.E. Murillo-Sánchez, R.J. Thomas (2011). MATPOWER steady-state operations, planning and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1), 12–19. [doi]. 43

Index