Plea for a semidefinite optimization solver in complex numbers

J. Ch. Gilbert [†] and C. Josz [‡]

March 28, 2017

Numerical optimization in complex numbers has drawn much less attention than in real numbers. A widespread opinion is that, since a complex number is a pair of real numbers, the best strategy to solve a complex optimization problem is to transform it into real numbers and to solve the latter by a real number solver. This paper defends another point of view and presents four arguments to convince the reader that skipping the transformation phase and using a complex number algorithm, if any, can sometimes have significant benefits. This is demonstrated for the particular case of a semidefinite optimization problem solved by a feasible corrector-predictor primal-dual interior-point method. In that case, the complex number formulation has the advantage (*i*) of having a smaller memory storage, (*ii*) of having a faster iteration, (*iii*) of requiring less iterations, and (*iv*) of having "smaller" feasible and solution sets. The computing time saving is rooted in the fact that some operations (like the matrix-matrix product) are much faster for complex operands than for their double size real counterparts, so that the conclusions of this paper could be valid for other problems in which these operations count a great deal in the computing time. The iteration number saving has its origin in the smaller (though complex) dimension of the problem in complex numbers. Numerical experiments show that all together these properties result in a code that can run up to four times faster. Finally, the feasible and solution sets are smaller since they are isometric to only a part of the feasible and solution sets of the problem in real numbers, which increases the chance of having a unique solution to the problem.

Keywords: Complex numbers, convergence, optimal power flow, corrector-predictor interior-point algorithm, $\mathbb{R}$-linear and $\mathbb{C}$-linear systems of equations, semidefinite optimization.

AMS MSC 2010: 65E05, 65K05, 90C22, 90C46, 90C51.

1 Introduction

Numerical linear algebra in complex numbers has drawn much less attention than in real numbers. For example, the two reference books, those by Golub and Van Loan [11] and by Higham [12], do not even quote the notion of Hermitian matrices in their respective index and discuss very little the numerical methods for complex matrices. The same observation can be made for numerical optimization in complex numbers (see the recent contributions [10, 43, 37, 44, 42, 19, 14] and the references therein for some exceptions). Actually, it is not uncommon to encounter computer scientists asserting that, since a complex number can be viewed as a pair of real numbers, the best strategy to solve a complex number problem is to transform it into real numbers and to run real number algorithms to solve the resulting problem; working directly in complex numbers would therefore be useless folklore. This paper defends another point of view and presents several arguments to convince the reader that developing algorithms in complex numbers can *sometimes* yield methods that are significantly faster than solving their transformed real number counterpart. This claim is often true when the computation involves matrices, which, in the real number version, have their size twice as large as in the complex formulation.

[†]INRIA Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France. E-mail: `Jean-Charles.Gilbert@inria.fr`.

[‡]LAAS, 7, avenue du Colonel Roche, BP 54200, 31031 Toulouse cedex 4, France. E-mail: `Cedric.Josz@gmail.com`.

This paper is not at such a high level of generality but demonstrates that a semidefinite optimization (SDO) problem, naturally or possibly defined in complex numbers, should most often also be solved by an SDO solver in complex numbers, if computing time prevails. We are even more specific, since this viewpoint is only theoretically and numerically validated for a primal-dual path-following interior-point algorithm, starting sufficiently close to the central path, whose iteration is a corrector-predictor step based on the Nesterov-Todd direction, and that uses a direct linear system solver; in that case, the speed-up is typically between two and four, depending on the structure of the problem, on the relative importance of the number of "complex" or "non-Hermitian" constraints (a term made more precise below). Such a claim in such a restrictive domain may sound of marginal relevance, but this acceleration should be observed in many other algorithms, provided some distinctive basic operations (such as those examined in section 5.4) form the bottleneck of the considered numerical method. In the case of the above described interior-point method, the most expensive and frequent basic operation for large problems is the product of matrices. Since, on paper, a speed-up of two is reachable for this operation, this benefit naturally and approximately extends to the whole algorithm when it solves sufficiently large problems. This speed-up can then be multiplied by two or so if all the constraints are non-Hermitian. This is a "theoretical" claim, based on the evaluation of the number of elementary operations. In practice, the block structure of the memory, the multiple cores of a particular machine, and the decisions taken by the compiler or interpreter may modify, up or down, this speed-up estimation. It is therefore necessary to validate this one experimentally. We have done so thanks to a home-made Matlab implementation of the above sketched interior-point algorithm for real/complex SDO problems; the piece of software Sdolab 0.4. A comparison with SeDuMi 1.3 [38, 35] is also presented and discussed.

This work has its source in and is motivated by our experience [18, 16] with the alternating current optimal power flow (OPF) problem, which is naturally defined in complex numbers [3, 2, 1]. One of the emblematic instance of the OPF problem consists in minimizing the Joule heating losses in a transmission electricity network, while satisfying the electricity demand by determining appropriately the active powers of the generators installed at given buses (or nodes) of the network. Structurally, one can view the problem as (or reduce to) a nonconvex quadratic optimization problem with nonconvex quadratic constraints in complex numbers [17; § 6]. Recalling that any polynomial optimization problem can be written that way [36, 5], one understands the potential difficulty of such a problem, which is actually NP-hard. In practice, approximate solutions can be obtained by rank relaxation (equivalent to a Lagrangian bi-dualisation) [41, 30, 22, 24, 25, 26], while more and more precise approximate solutions can be computed by memory-greedy moment-sos relaxations [32, 21, 23, 18, 29], as long as the computer accepts the resulting increase in the relaxed problem dimensions. These relaxations may be written like SDO problems in real or complex numbers. Some advantages of the latter formalism have been identified in [19]. As the present paper demonstrates it, having an efficient SDO solver in complex numbers would still increase the appeal of the formulation of the relaxations in complex numbers.

In the SDO context, the benefit of dealing directly with complex data, instead of transforming the SDO problem into real numbers, was already observed in [39; section 25.3.8.3] (see also the draft [40; section 3.5]), but the present paper considers complex SDO models that are more general and goes further in the analysis of the problem. The primal SDO model in [39] has constraints of the form $\langle A_k, X\rangle = b_k$, where the data A_k and the unknown X are Hermitian matrices, $\langle \cdot, \cdot \rangle$ is the standard scalar product on the vector space of Hermitian matrices, and the given scalar b_k is therefore a real number. In the model of this paper, which

occurs in the OPF problem briefly described above and in section 6.1.2, X is still searched as a positive semidefinite Hermitian matrix, but A_k may be an arbitrary complex square matrix (hence b_k is now a complex number). In the framework of this paper, constraints of the latter type are called *complex* or *non-Hermitian*. Of course, as we shall see in section 4.2.1, one can transform the latter model into the former, but an SDO solver like the one considered in this paper can run up to twice faster if this transformation is not made; see observation 6.1(1).

The benefits of solving an SDO problem in complex numbers with a complex interior-point SDO solver instead of its real number analogue can be summarized in four points:

- the approach requires less memory storage, since the matrices have smaller order (see observation 4.4),
- the presented corrector-predictor interior-point solver requires less iterations (observations 5.3 and 6.1(2)),
- each iteration is much faster in complex numbers (see section 5.4 and observations 6.1).

To this, one can add that

- the feasible and solution sets are "smaller" for the complex SDO problem, which increases the chance of having a unique solution (observation 4.8).

These findings alone are sufficient to wish more effort on the development of SDO solvers in complex numbers, which justifies the title of this paper. On the other hand, since it is shown below that the feasible and solution sets are nonempty and bounded simultaneously in the real and complex formulations, there is no clear advantage in using the real or complex formulation of the SDO problem with respect to the well-posedness of a primal, dual, or primal-dual interior-point method (observation 4.12).

The paper is organized as follows. In section 2, we recall some basic concepts in complex analysis. Section 3 presents a rather general form of the complex SDO problem and the way a dual problem can be obtained by Lagrangian duality. Section 4 describes in details how the complex primal and dual SDO problems can be transformed into SDO problems in real numbers, using vector space isometries and ring homomorphisms. Various properties linking the complex SDO problem and its real counterpart are established; in particular, it is shown that both problems have their feasible and solution sets simultaneously bounded (although they are not in a one to one correspondence; the real sets being somehow "larger"). Section 5 deals with algorithmic issues. The corrector-predictor algorithm for solving the complex SDO problem is introduced and it is shown that the generated iterates are not in correspondence with those generated by the same algorithm on the corresponding real version of the problem. In passing, an argument is given to show why the algorithm should generally require less iterations to converge when it solves the problem in complex numbers rather than the problem in real numbers. The computation of the complex NT direction is described with some details and it is shown that this computation is well defined under a regularity assumption (the surjectivity of the linear map used to define the affine constraint). This long section ends with a theoretical comparison of the computation efforts of some key operations occurring in complex and real numbers, which explains why an iteration of the complex corrector-predictor algorithm runs faster than in its real twin. Section 6 is dedicated to numerical experiments that highlight the advantages of dealing directly with the complex number SDO problem. The paper ends with the conclusion section 7.

2 Fragments of complex analysis

2.1 Complex numbers and vectors

We denote by $\mathbb{R}$ the set of real numbers and by $\mathbb{C}$ the set of complex numbers. The *pure imaginary number* is written $\mathrm{i} := \sqrt{-1}$. The *real part* of a complex number x is denoted by $\Re(x)$ and its *imaginary part* by $\Im(x)$; hence $x = \Re(x) + \mathrm{i}\Im(x)$. The *conjugate* of x is denoted by $\overline{x} := \Re(x) - \mathrm{i}\Im(x)$ and its *modulus* by $|x| = (\overline{x}x)^{1/2} = (\Re(x)^2 + \Im(x)^2)^{1/2} \in \mathbb{R}$.

We denote by $\mathbb{C}^m$ the $\mathbb{C}$-vector space (hence with scalars in $\mathbb{C}$) formed of the m-uples of complex numbers (sometimes considered as an $m \times 1$ matrix) and by $\mathbb{C}^m_{\mathbb{R}}$ the $\mathbb{R}$-vector space (hence with scalars in $\mathbb{R}$) formed of the same vectors. Recall that the dimension of the $\mathbb{C}$-vector space $\mathbb{C}^m$ is m, while the dimension of the $\mathbb{R}$-vector space $\mathbb{C}^m_{\mathbb{R}}$ is $2m$. The space $\mathbb{C}^m_{\mathbb{R}}$ is useful, since the primal SDO problem (section 3.1) is defined with an $\mathbb{R}$-linear map (i.e., a map that is linear when the scalars are taken in $\mathbb{R}$) whose values are complex vectors. We endow $\mathbb{C}^m_{\mathbb{R}}$ with the scalar product $\langle \cdot, \cdot \rangle_{\mathbb{C}^m_{\mathbb{R}}} : \mathbb{C}^m_{\mathbb{R}} \times \mathbb{C}^m_{\mathbb{R}} \to \mathbb{R}$, defined at $(u, v) \in \mathbb{C}^m_{\mathbb{R}} \times \mathbb{C}^m_{\mathbb{R}}$ by

$$\langle u, v \rangle_{\mathbb{C}^m_{\mathbb{R}}} = \Re(u^{\mathsf{H}} v) = \Re\left(\sum_{i=1}^{m} \overline{u}_i v_i \right) = \Re(u)^{\mathsf{T}} \Re(v) + \Im(u)^{\mathsf{T}} \Im(v), \tag{2.1}$$

where we have denoted $v^{\mathsf{H}} := \big(\overline{v}\big)^{\mathsf{T}}$ the conjugate transpose of the vector v. This choice of scalar product will have a direct impact on the form of the adjoint of an operator with values in $\mathbb{C}^m_{\mathbb{R}}$ (see section 3.2.2).

2.2 General complex matrices

The vector space $\mathbb{R}^{m \times n}$ of real $m \times n$ matrices is usually equipped with the scalar product $\langle A, B \rangle = \operatorname{tr} A^{\mathsf{T}} B$, where A^{T} is the transpose of A and $\operatorname{tr} A = \sum_i A_{ii}$ is its *trace*. Similarly, the vector space $\mathbb{C}^{m \times n}$ of the $m \times n$ complex matrices is equipped with the Hermitian scalar product

$$\langle \cdot, \cdot \rangle_{\mathbb{C}^{m \times n}} : (A, B) \in \mathbb{C}^{m \times n} \times \mathbb{C}^{m \times n} \mapsto \langle A, B \rangle_{\mathbb{C}^{m \times n}} := \operatorname{tr} A^{\mathsf{H}} B \in \mathbb{C}, \tag{2.2}$$

where A^{H} is the conjugate transpose of A. This one is *left-sesquilinear*, meaning that for $\alpha \in \mathbb{C}$: $\langle A, \alpha B \rangle = \alpha \langle A, B \rangle$ and $\langle \alpha A, B \rangle = \overline{\alpha} \langle A, B \rangle$. The associated norm is the Frobenius norm $\| \cdot \|_F : A \in \mathbb{C}^{m \times n} \mapsto \|A\|_F$, where

$$\|A\|_F := \langle A, A \rangle^{1/2}_{\mathbb{C}^{m \times n}} = \big(\operatorname{tr} A^{\mathsf{H}} A \big)^{1/2} = \left(\sum_{ij} |A_{ij}|^2 \right)^{1/2}.$$

The Hermitian scalar product (2.2) has the following properties: for A and $B \in \mathbb{C}^{m \times n}$ there hold

$$\langle B, A \rangle_{\mathbb{C}^{m \times n}} = \overline{\langle A, B \rangle_{\mathbb{C}^{m \times n}}} = \langle A^{\mathsf{H}}, B^{\mathsf{H}} \rangle_{\mathbb{C}^{n \times m}}. \tag{2.3}$$

At some places, we need to consider $\mathbb{C}^{m \times n}$ as an $\mathbb{R}$-linear space. It is then denoted by $\mathbb{C}^{m \times n}_{\mathbb{R}}$, which is equipped with the scalar product is defined by

$$\langle \cdot, \cdot \rangle_{\mathbb{C}^{m \times n}_{\mathbb{R}}} : (A, B) \in \mathbb{C}^{m \times n}_{\mathbb{R}} \times \mathbb{C}^{m \times n}_{\mathbb{R}} \mapsto \langle A, B \rangle_{\mathbb{C}^{m \times n}_{\mathbb{R}}} := \Re\big(\langle A, B \rangle_{\mathbb{C}^{m \times n}} \big) \in \mathbb{R}, \tag{2.4}$$

2.3 Hermitian matrices

A matrix $A \in \mathbb{C}^{n\times n}$ is *Hermitian* if $A^{\mathsf{H}} = A$. The set of complex Hermitian matrices is denoted by $\mathcal{H}^n$, while the set of real symmetric matrices is denoted by $\mathcal{S}^n$ and the set of real skew symmetric matrices by $\mathcal{Z}^n$. We also denote by $\mathcal{S}^n_+$ (resp. $\mathcal{S}^n_{++}$) the cone of $\mathcal{S}^n$ formed of the positive semidefinite (resp. definite) matrices. For a Hermitian matrix $A \in \mathcal{H}^n$, there hold $\Re(A) \in \mathcal{S}^n$ and $\Im(A) \in \mathcal{Z}^n$. The set $\mathcal{H}^n$ is an $\mathbb{R}$-vector space but not a $\mathbb{C}$-linear space, since the identity matrix I is Hermitian but not $\mathrm{i}I$; this observation will have some consequences below and it already implies that all the notions of convex analysis are naturally well defined in $\mathcal{H}^n$. The trace $\operatorname{tr} AB = \sum_{ij} \overline{A_{ij}} B_{ij}$ of the product of two Hermitian matrices A and B is a real number and the map

$$\langle \cdot, \cdot \rangle_{\mathcal{H}^n} : (A, B) \in \mathcal{H}^n \times \mathcal{H}^n \mapsto \langle A, B \rangle_{\mathcal{H}^n} := \operatorname{tr} AB \in \mathbb{R} \tag{2.5}$$

is a scalar product, making $\mathcal{H}^n$ a Euclidean space.

If $A \in \mathcal{H}^n$ and $v \in \mathbb{C}^n$, then $v^{\mathsf{H}} A v$ is a real number (since its conjugate transpose does not change its value). Therefore, for $A \in \mathcal{H}^n$, it makes sense to say that

$$\begin{aligned} A \text{ is positive semidefinite (notation } A \succcurlyeq 0) \quad &\Longleftrightarrow \quad v^{\mathsf{H}} A v \geqslant 0, \quad \forall v \in \mathbb{C}^n, \\ A \text{ is positive definite (notation } A \succ 0) \quad &\Longleftrightarrow \quad v^{\mathsf{H}} A v > 0, \quad \forall v \in \mathbb{C}^n \setminus \{0\}. \end{aligned}$$

The set of positive semidefinite (resp. positive definite) Hermitian matrices is denoted by $\mathcal{H}^n_+$ (resp. $\mathcal{H}^n_{++}$). The set $\mathcal{H}^n_+$ is a closed convex cone of $\mathcal{H}^n$; it is also self-dual, meaning that its dual cone $(\mathcal{H}^n_+)^+ := \{K \in \mathcal{H}^n : \langle K, H \rangle \geqslant 0 \text{ for all } H \in \mathcal{H}^n_+\}$ is equal to itself: $(\mathcal{H}^n_+)^+ = \mathcal{H}^n_+$.

A matrix A is said to be *skew-Hermitian* if $A^{\mathsf{H}} = -A$. Any matrix $A \in \mathbb{C}^{n\times n}$ can be uniquely written as the sum of a Hermitian matrix $\mathcal{H}(A)$ and a skew-Hermitian matrix $\mathcal{Z}(A)$:

$$A = \mathcal{H}(A) + \mathcal{Z}(A), \qquad \text{where } \mathcal{H}(A) := \tfrac{1}{2}(A + A^{\mathsf{H}}) \text{ and } \mathcal{Z}(A) := \tfrac{1}{2}(A - A^{\mathsf{H}}). \tag{2.6}$$

One can therefore write

$$A = \mathcal{H}(A) - \mathrm{i}\,(\mathrm{i}\mathcal{Z}(A))\,, \tag{2.7}$$

in which $\mathrm{i}\mathcal{Z}(A)$ is also Hermitian. Using this identity and the sesquilinearity of the scalar product $\langle \cdot, \cdot \rangle_{\mathbb{C}^{n\times n}}$, one gets for $A \in \mathbb{C}^{n\times n}$ and $X \in \mathcal{H}^n$:

$$\langle A, X \rangle_{\mathbb{C}^{n\times n}} = \underbrace{\langle \mathcal{H}(A), X \rangle_{\mathcal{H}^n}}_{\text{real number}} + \mathrm{i} \underbrace{\langle \mathrm{i}\mathcal{Z}(A), X \rangle_{\mathcal{H}^n}}_{\text{real number}}, \tag{2.8}$$

which is the decomposition of the complex number $\langle A, X \rangle_{\mathbb{C}^{n\times n}}$ in its real and imaginary parts.

3 The complex SDO problem

Some semidefinite optimization (SDO) problems like the OPF problem [3, 2, 1] are naturally defined in complex numbers, since they deal with the optimization of systems that are more conveniently described in complex numbers. This section introduces the complex (number) SDO problem in a rather general form (section 3.1), as well as its Lagrangian dual (section 3.2), and presents the various forms that can take a standard regularity assumption (proposition 3.2).

3.1 Primal problem

The primal form of the SDO problem consists in finding a Hermitian matrix $X \in \mathcal{H}^n$ minimizing a linear function on a set that is the intersection of the cone $\mathcal{H}^n_+$ of positive semidefinite Hermitian matrices and an affine subspace. It can therefore be written

$$(P) \quad \begin{cases} \inf_X \langle C, X \rangle_{\mathcal{H}^n} \\ \mathcal{A}(X) = b \\ X \succcurlyeq 0, \end{cases} \tag{3.1}$$

where $C \in \mathcal{H}^n$, $\mathcal{A}$ is an $\mathbb{R}$-linear map defined on $\mathcal{H}^n$ with values in

$$\mathbb{F} := \mathbb{F}_r \times \mathbb{F}_c, \qquad \text{where } \mathbb{F}_r := \mathbb{R}^{m_r} \text{ and } \mathbb{F}_c := \mathbb{C}^{m_c}_{\mathbb{R}}, \tag{3.2}$$

and $b \in \mathbb{F}$. Since $\langle C, X \rangle_{\mathcal{H}^n}$ is a real number, the problem has a single objective (to put it another way, the real-valued objective, which is linear on $\mathcal{H}^n$, is represented by the matrix $C \in \mathcal{H}^n$, using the Riesz-Fréchet representation theorem). The linear map $\mathcal{A}$ cannot be $\mathbb{C}$-linear since $\mathcal{H}^n$ is only an $\mathbb{R}$-linear space. Hence its arrival space $\mathbb{F}$ must also be considered as an $\mathbb{R}$-linear space, which is the reason why the complex space $\mathbb{C}^{m_c}_{\mathbb{R}}$, and not $\mathbb{C}^{m_c}$, defined in section 2.1 appears in the product space $\mathbb{F}$. The space $\mathbb{F}_r := \mathbb{R}^{m_r}$ is introduced to reflect the fact that some constraints are naturally expressed in real numbers. Of course a real number is just a particular complex number, so that one could have get rid of $\mathbb{F}_r$, but then the surjectivity of $\mathcal{A}$, considered as regularity assumption below (see proposition 3.2), could not be satisfied.

The feasible set of problem (P) is denoted by $\mathcal{F}_P$, its solution set by $\mathcal{S}_P$, and its optimal value by $\mathrm{val}(P)$. The notation is made shorter by introducing the index sets

$$K_r := [1 : m_r] \qquad \text{and} \qquad K_c := [m_r{+}1 : m_r{+}m_c],$$

so that $b \in \mathbb{F}$ can be decomposed in (b_{K_r}, b_{K_c}) with $b_{K_r} \in \mathbb{F}_r$ and $b_{K_c} \in \mathbb{F}_c$. We also introduce the partial $\mathbb{R}$-linear maps

$$(\mathcal{A}_r := \pi_r \circ \mathcal{A}) : \mathcal{H}^n \to \mathbb{F}_r, \qquad \text{and} \qquad (\mathcal{A}_c := \pi_c \circ \mathcal{A}) : \mathcal{H}^n \to \mathbb{F}_c, \tag{3.3}$$

where π_r and π_c are the canonical projectors $\pi_r : b = (b_{K_r}, b_{K_c}) \in \mathbb{F} \mapsto b_{K_r} \in \mathbb{F}_r$ and $\pi_c : b = (b_{K_r}, b_{K_c}) \in \mathbb{F} \mapsto b_{K_c} \in \mathbb{F}_c$. We equip $\mathbb{F}$ with the natural scalar product

$$\langle \cdot, \cdot \rangle_{\mathbb{F}} : (a, b) \in \mathbb{F}^2 \mapsto \langle a, b \rangle_{\mathbb{F}} = \langle a_{K_r}, a_{K_r} \rangle_{\mathbb{F}_r} + \langle a_{K_c}, a_{K_c} \rangle_{\mathbb{F}_c}, \tag{3.4}$$

where $a = (a_{K_r}, a_{K_c})$, $b = (b_{K_r}, b_{K_c})$, $\langle \cdot, \cdot \rangle_{\mathbb{F}_r}$ is the Euclidean scalar product of $\mathbb{F}_r := \mathbb{R}^{m_r}$, and $\langle \cdot, \cdot \rangle_{\mathbb{F}_c}$ is the scalar product of $\mathbb{F}_c := \mathbb{C}^{m_c}_{\mathbb{R}}$ defined in (2.1).

Let us now examine how the linear map $\mathcal{A}$ can be represented by matrices. By the Riesz-Fréchet representation theorem, for $k \in K_r$, the $\mathbb{R}$-linear map $X \in \mathcal{H}^n \mapsto [\mathcal{A}(X)]_k \in \mathbb{R}$ can be represented by a Hermitian matrix, say $A_k \in \mathcal{H}^n$, in the sense that

$$\forall\, X \in \mathcal{H}^n : \quad [\mathcal{A}(X)]_k = \langle A_k, X \rangle_{\mathcal{H}^n}.$$

Similarly, for $k \in K_c$, the $\mathbb{R}$-linear maps $X \in \mathcal{H}^n \mapsto \Re([\mathcal{A}(X)]_k) \in \mathbb{R}$ and $X \in \mathcal{H}^n \mapsto \Im([\mathcal{A}(X)]_k) \in \mathbb{R}$ can be represented by Hermitian matrices, say H^r_k and $H^i_k \in \mathcal{H}^n$:

$$\forall\, X \in \mathcal{H}^n : \quad \Re([\mathcal{A}(X)]_k) = \langle H^r_k, X \rangle_{\mathcal{H}^n} \quad \text{and} \quad \Im([\mathcal{A}(X)]_k) = \langle H^i_k, X \rangle_{\mathcal{H}^n}.$$

Hence, by the left-sesquilinearity of the scalar product $\langle\cdot,\cdot\rangle_{\mathbb{C}^{n\times n}}$, there holds

$$\forall\, X \in \mathcal{H}^n : \quad [\mathcal{A}(X)]_k = \langle H_k^r, X\rangle_{\mathbb{C}^{n\times n}} + \mathrm{i}\langle H_k^i, X\rangle_{\mathbb{C}^{n\times n}} = \langle H_k^r - \mathrm{i}H_k^i, X\rangle_{\mathbb{C}^{n\times n}}.$$

Let us introduce $A_k := H_k^r - \mathrm{i}H_k^i \in \mathbb{C}^{n\times n}$. Although A_k is formed from the Hermitian matrices H_k^r and H_k^i, the decomposition (2.7) shows that it has no particular structure, meaning that A_k is an arbitrary $n \times n$ complex matrix. In conclusion, we have shown that $\mathcal{A}$ has the following matrix representation

$$[\mathcal{A}(X)]_k = \begin{cases} \langle A_k, X\rangle_{\mathcal{H}^n} & \text{for } k \in K_r \\ \langle A_k, X\rangle_{\mathbb{C}^{n\times n}} & \text{for } k \in K_c, \end{cases} \tag{3.5}$$

where $A_k \in \mathcal{H}^n$ when $k \in K_r$ and $A_k \in \mathbb{C}^{n\times n}$ when $k \in K_c$.

3.2 Lagrangian duality

3.2.1 Dual problem

The Lagrange dual of the complex SDO problem (P) in (3.1) can be obtained like for the real SDO problem. Since problem (P) can be written

$$\inf_{X\succcurlyeq 0}\ \sup_{y\in\mathbb{F}} \langle C, X\rangle_{\mathcal{H}^n} - \langle y, \mathcal{A}(X) - b\rangle_{\mathbb{F}},$$

its Lagrange dual is the problem

$$\sup_{y\in\mathbb{F}}\ \inf_{X\succcurlyeq 0} \langle C, X\rangle_{\mathcal{H}^n} - \langle y, \mathcal{A}(X) - b\rangle_{\mathbb{F}} = \sup_{y\in\mathbb{F}} \left(\langle b, y\rangle_{\mathbb{F}} + \inf_{X\succcurlyeq 0} \langle C - \mathcal{A}^*(y), X\rangle_{\mathcal{H}^n} \right).$$

Now, using the self-duality of $\mathcal{H}_+^n$, one gets $\inf\{\langle C-\mathcal{A}^*(y), X\rangle_{\mathcal{H}^n} : X \succcurlyeq 0\} = -\mathcal{I}_{\mathcal{H}_+^n}(C-\mathcal{A}^*(y))$, where $\mathcal{I}_{\mathcal{H}_+^n}$ is the indicator function of $\mathcal{H}_+^n$ (that is $\mathcal{I}_{\mathcal{H}_+^n}(M) = 0$ if $M \in \mathcal{H}_+^n$ and $\mathcal{I}_{\mathcal{H}_+^n}(M) = +\infty$ if $M \notin \mathcal{H}_+^n$). Therefore, the Lagrange dual of (P) is the following problem in $(y, S) \in \mathbb{F} \times \mathcal{H}^n$:

$$(D) \quad \begin{cases} \sup_{(y,S)} \langle b, y\rangle_{\mathbb{F}} \\ \mathcal{A}^*(y) + S = C \\ S \succcurlyeq 0. \end{cases} \tag{3.6}$$

The structure of (D) is similar to the Lagrange dual obtained in the real formulation of the SDO problem. The feasible set of problem (D) is denoted by $\mathcal{F}_D$, its solution set by $\mathcal{S}_D$, and its optimal value by $\mathrm{val}(D)$.

3.2.2 Adjoint

To write a more specific version of the dual problem (3.6), with the representation matrices A_k of $\mathcal{A}$ given by (3.5), one must explicit how the adjoint operator $\mathcal{A}^* : \mathbb{F} \to \mathcal{H}^n$ of $\mathcal{A} : \mathcal{H}^n \to \mathbb{F}$ appearing in (D) can expressed in terms of these matrices. This adjoint depends on the scalar product equipping $\mathbb{F}$, which is chosen to be the one in (3.4). We denote by $\mathcal{A}_i^* : \mathbb{F}_i \to \mathcal{H}^n$, for $i \in \{r, c\}$, the adjoint map of the $\mathbb{R}$-linear map $\mathcal{A}_i : \mathcal{H}^n \to \mathbb{F}_i$ defined in (3.3).

Proposition 3.1 (adjoint of $\mathcal{A}$) *Suppose that $\mathcal{A} : \mathcal{H}^n \to \mathbb{F}$ is defined by (3.5) and*

that $\mathbb{F}$ is equipped with the scalar product in (3.4). Then, for $y = (y_{K_r}, y_{K_c}) \in \mathbb{F}_r \times \mathbb{F}_c = \mathbb{F}$, there holds:

$$\mathcal{A}^*(y) = \mathcal{A}_r^*(y_{K_r}) + \mathcal{A}_c^*(y_{K_c}), \tag{3.7a}$$

where

$$\mathcal{A}_r^*(y_{K_r}) = \textstyle\sum_{k\in K_r} y_k A_k, \tag{3.7b}$$

$$\mathcal{A}_c^*(y_{K_c}) = \mathcal{H}\left(\textstyle\sum_{l\in K_c} y_l A_l\right) = \frac{1}{2}\textstyle\sum_{l\in K_c}\left(y_l A_l + \overline{y}_l A_l^{\mathsf{H}}\right) \tag{3.7c}$$

$$= \textstyle\sum_{l\in K_c} \Re(y_l)\mathcal{H}(A_l) + \sum_{l\in K_c} \Im(y_l)(\mathrm{i}\mathcal{Z}(A_l)). \tag{3.7d}$$

Proof. The proofs of (3.7a) and (3.7b) are standard, see [9] for the details. The first identity in (3.7c) is obtained by

$$\begin{aligned}
\langle \mathcal{A}_c^*(y_{K_c}), X\rangle_{\mathcal{H}^n} &= \langle y_{K_c}, \mathcal{A}_c(X)\rangle_{\mathbb{F}_c} && [\text{definition of } \mathcal{A}_c^*]\\
&= \Re(\textstyle\sum_{l\in K_c} \overline{y}_l [\mathcal{A}(X)]_l) && [(2.1)]\\
&= \Re(\textstyle\sum_{l\in K_c} \overline{y}_l \langle A_l, X\rangle_{\mathbb{C}^{n\times n}}) && [(3.5)]\\
&= \Re(\langle \textstyle\sum_{l\in K_c} y_l A_l, X\rangle_{\mathbb{C}^{n\times n}}) && [(2.2)]\\
&= \langle \mathcal{H}(\textstyle\sum_{l\in K_c} y_l A_l), X\rangle_{\mathcal{H}^n} && [(2.8)].
\end{aligned}$$

The second identity in (3.7c) now comes from the definition (2.6) of the $\mathcal{H}$ operator. Finally, (3.7d) follows immediately from the last expression in (3.7c). □

Both y_l and its conjugate $\overline{y}_l$ appear in the second identity in (3.7c). It cannot be otherwise, because $\mathcal{A}^*$, like $\mathcal{A}$, is only $\mathbb{R}$-linear, while $\mathcal{L} : y \in \mathbb{C}^{m_c} \mapsto \sum_{l\in K_c} y_l A_l$ is $\mathbb{C}$-linear (since $\mathcal{L}(\mathrm{i}y) = \mathrm{i}\mathcal{L}(y)$). This fact will have an impact on the subsequent developments.

The surjectivity of the linear map $\mathcal{A}$ is a standard regularity assumption of a real SDO problem, which intervenes several times below. The next proposition makes explicit what this means in terms of the matrices A_k introduced by (3.5). In this proposition and below, for $\mathbb{K}_i$ being $\mathbb{R}$ or $\mathbb{C}$, we say that the complex matrices $A_1, \ldots, A_m$ are $\mathbb{K}_1 \times \cdots \times \mathbb{K}_m$*-linearly independent* if any $\alpha \in \mathbb{K}_1 \times \cdots \times \mathbb{K}_m$ satisfying $\sum_{i=1}^m \alpha_i A_i = 0$ vanishes. Of course, matrices that are $\mathbb{K}_1 \times \cdots \times \mathbb{K}_m$-linearly independent are also $\mathbb{K}_1' \times \cdots \times \mathbb{K}_m'$-linearly independent if $\mathbb{K}_i' \subseteq \mathbb{K}_i$ for all $i \in [1\,{:}\,m]$ (in particular, the $\mathbb{C}^m$-linear independence implies the $\mathbb{R}^m$-linear independence).

Proposition 3.2 (surjectivity of $\mathcal{A}$) *The following properties are equivalent:*

(i) the $\mathbb{R}$-linear operator $\mathcal{A} : \mathcal{H}^n \to \mathbb{F}$ is surjective,

(ii) any $y \in \mathbb{F}$ satisfying $\sum_{k\in K_r} y_k A_k + \mathcal{H}(\sum_{l\in K_c} y_l A_l) = 0$ vanishes,

(iii) $\{A_k\}_{k\in K_r} \cup \{A_l\}_{l\in K_c} \cup \{A_l^{\mathsf{H}}\}_{l\in K_c}$ are $(\mathbb{R}^{m_r} \times \mathbb{C}^{2m_c})$-linearly independent,

(iv) $\{A_k\}_{k\in K_r} \cup \{\mathcal{H}(A_l)\}_{l\in K_c} \cup \{\mathrm{i}\mathcal{Z}(A_l)\}_{l\in K_c}$ are $\mathbb{R}^{m_r+2m_c}$-linearly independent.

Proof. [$(i) \Leftrightarrow (ii)$] The surjectivity of the $\mathbb{R}$-linear map $\mathcal{A}$ is equivalent to the injectivity of its adjoint $\mathcal{A}^*$, which, by (3.7a), (3.7b), and (3.7c), is equivalent to (ii).

[$(ii) \Leftrightarrow (iii)$] Since the condition $\sum_{k\in K_r} y_k A_k + \mathcal{H}(\sum_{l\in K_c} y_l A_l) = 0$ reads $\sum_{k\in K_r}(2y_k)A_k + \sum_{l\in K_c} y_l A_l + \sum_{l\in K_c} \overline{y}_l A_l^{\mathsf{H}} = 0$, it is clear the $(iii) \Rightarrow (ii)$. Let us now show the converse, assuming that (ii) holds and that

$$\sum_{k\in K_r} y_k A_k + \sum_{l\in K_c} y_l A_l + \sum_{l\in K_c} y_l' A_l^{\mathsf{H}} = 0, \tag{3.8}$$

for some $(y_{K_r}, y_{K_c}, y'_{K_c}) \in \mathbb{R}^{m_r} \times \mathbb{C}^{m_c} \times \mathbb{C}^{m_c}$. Taking the conjugate transpose and adding, one gets

$$\sum_{k \in K_r} (2y_k) A_k + \sum_{l \in K_c} \left((y_l + \overline{y'_l}) A_l + \overline{(y_l + \overline{y'_l})} A_l^{\mathsf{H}} \right) = 0.$$

By (ii), $y_{K_r} = 0$ and $y_{K_c} + \overline{y'_{K_c}} = 0$. Then (3.8) yields

$$\sum_{l \in K_c} \left(y_l A_l - \overline{y_l} A_l^{\mathsf{H}} \right) = 0.$$

Multiplying by i, one gets

$$\sum_{l \in K_c} \left((\mathrm{i} y_l) A_l + \overline{(\mathrm{i} y_l)} A_l^{\mathsf{H}} \right) = 0.$$

By (ii) again, $y_{K_c} = 0$. Hence $y'_{K_c} = 0$ and $y = 0$.

[$(ii) \Leftrightarrow (iv)$] By (3.7d), the condition $\mathcal{A}^*(y) := \sum_{k \in K_r} y_k A_k + \mathcal{H}(\sum_{l \in K_c} y_l A_l) = 0$ can be written

$$\sum_{k \in K_r} y_k A_k + \sum_{l \in K_c} \mathfrak{R}(y_l) \mathcal{H}(A_l) + \sum_{l \in K_c} \mathfrak{I}(y_l) (\mathrm{i} \mathcal{Z}(A_l)) = 0.$$

Since $\mathfrak{R}(y_l)$ and $\mathfrak{I}(y_l)$ are arbitrary real numbers, the equivalence follows. □

The surjectivity of $\mathcal{A}$ implies a bound on the number of affine constraints: since the $\mathbb{R}$-dimension of $\mathcal{H}^n$ is $n + 2((n-1)n/2) = n^2$ (n real numbers on the diagonal and $(n-1)n/2$ complex numbers on the strictly lower triangular part) and the one of $\mathbb{F}$ is $m_r + 2m_c$, the inequality $m_r + 2m_c \leqslant n^2$ is a necessary condition of surjectivity of $\mathcal{A}$.

4 Complex and real SDO formulations

Since a complex number is a pair of real numbers, the complex SDO problem (3.1) can certainly be transformed into a real SDO problem. One of the goals of this paper is to compare the complex and real number SDO formulations and to highlight the advantages of the former when the problem is naturally raised in complex numbers. Before showing this, we have to be precise on the way such a complex number SDO problem is transformed into real numbers. This subject is examined in this section. Section 4.1 presents the operators that make possible to switch between the two formulations, as well as some of their properties. Section 4.2 describes a way of transforming the complex SDO problem into real numbers and presents a few properties linking the two formulations.

4.1 Switching between complex and real formulations

The transformation of the complex SDO problem of section 3.1 into a real SDO problem in section 4.2 is better done by using operators that establish a correspondence between complex objects (vector and matrices) and their real counterparts. This section introduces these operators. These have the remarkable properties of being vector space isometries or ring homomorphisms. The isometries of section 4.1.1 are more natural for transforming the complex SDO problem into real numbers and, as a result, for making correspondences between the feasible and solution sets, as well as the optimal values, which are identical. The homomorphisms of section 4.1.2 are more appropriate for establishing correspondences between the objects generated by the interior-point solvers in the complex and real spaces (section 5.1), in particular to compare their respective iterates (section 5.2).

4.1.1 Vector space isometries

One of the first questions that should be examined deals with the appropriate way of transforming into real numbers the condition $M \succcurlyeq 0$ appearing in (3.1) for a Hermitian matrix $M \in \mathcal{H}^n$. This condition can be written $v^{\mathsf{H}} M v \geqslant 0$ for all $v \in \mathbb{C}^n$ or

$$(\Re(v) - \mathrm{i}\Im(v))^{\mathsf{T}} (\Re(M) + \mathrm{i}\Im(M))(\Re(v) + \mathrm{i}\Im(v)) \geqslant 0, \qquad \text{for all } v \in \mathbb{C}^n. \tag{4.1}$$

Without surprise, this suggests us to transform the complex vector $v \in \mathbb{C}^n$ into the real vector $(\Re(v), \Im(v))$ made of its real and imaginary parts. The associated transformation operator is denoted by $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}} : \mathbb{C}^n_{\mathbb{R}} \to \mathbb{R}^{2n}$ and is defined at $v \in \mathbb{C}^n_{\mathbb{R}}$ by

$$\boxed{\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v) = \begin{pmatrix} \Re(v) \\ \Im(v) \end{pmatrix}.}$$

We prefer defining $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}$ on $\mathbb{C}^n_{\mathbb{R}}$ instead of $\mathbb{C}^n$ since it is an $\mathbb{R}$-linear map and, with the scalar product (2.1) on $\mathbb{C}^n_{\mathbb{R}}$ and the Euclidean scalar product on $\mathbb{R}^{2n}$, $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}$ is an isometry:

$$\forall\, v, w \in \mathbb{C}^n_{\mathbb{R}} : \qquad \langle \mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v), \mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(w) \rangle_{\mathbb{R}^{2n}} = \langle v, w \rangle_{\mathbb{C}^n_{\mathbb{R}}}.$$

Note that, even though the size of $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v) \in \mathbb{R}^{2n}$ is twice that of $v \in \mathbb{C}^n_{\mathbb{R}}$, the storage of these two vectors requires the same amount of memory.

A straightforward computation shows that (4.1) can now be written

$$\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v)^{\mathsf{T}} \begin{pmatrix} \Re(M) & -\Im(M) \\ \Im(M) & \Re(M) \end{pmatrix} \mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v) \geqslant 0, \qquad \text{for all } v \in \mathbb{C}^n.$$

It is therefore natural [39, 19] to introduce as matrix transformation operator, the $\mathbb{R}$-linear map $\mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}} : \mathbb{C}^{m\times n}_{\mathbb{R}} \to \mathbb{R}^{(2m)\times(2n)}$ defined at $M \in \mathbb{C}^{m\times n}_{\mathbb{R}}$ by

$$\boxed{\mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M) = \frac{1}{\sqrt{2}} \begin{pmatrix} \Re(M) & -\Im(M) \\ \Im(M) & \Re(M) \end{pmatrix}.}$$

Here and below, the index of the various operators $\mathcal{J}$ refers to its starting space. The presence of the factor $1/\sqrt{2}$ is motivated by the fact that $\mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}$ satisfies then the isometric-like identity

$$\forall\, M, N \in \mathbb{C}^{m\times n}_{\mathbb{R}} : \qquad \langle \mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M), \mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(N) \rangle_{\mathbb{R}^{(2m)\times(2n)}} = \langle M, N \rangle_{\mathbb{C}^{m\times n}_{\mathbb{R}}}.$$

Note also that

$$\forall M \in \mathbb{C}^{m\times n}_{\mathbb{R}} : \quad \mathcal{J}_{\mathbb{C}^{n\times m}_{\mathbb{R}}}(M^{\mathsf{H}}) = \mathcal{J}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)^{\mathsf{T}}. \tag{4.2}$$

We see that the restriction of $\mathcal{J}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}$ to $\mathcal{H}^n$ has values in $\mathcal{S}^{2n}$. This restriction is denoted by $\mathcal{J}_{\mathcal{H}^n} : \mathcal{H}^n \to \mathcal{S}^{2n}$:

$$\boxed{\mathcal{J}_{\mathcal{H}^n} = \mathcal{J}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}\big|_{\mathcal{H}^n}.}$$

It satisfies the isometry property

$$\forall\, M, N \in \mathcal{H}^n : \qquad \langle \mathcal{J}_{\mathcal{H}^n}(M), \mathcal{J}_{\mathcal{H}^n}(N) \rangle_{\mathcal{S}^{2n}} = \langle M, N \rangle_{\mathcal{H}^n}, \tag{4.3}$$

which shows that the adjoint $\mathcal{J}^*_{\mathcal{H}^n}$ of $\mathcal{J}_{\mathcal{H}^n}$ is a left inverse of $\mathcal{J}_{\mathcal{H}^n}$:

$$\mathcal{J}^*_{\mathcal{H}^n} \mathcal{J}_{\mathcal{H}^n} = I_{\mathcal{H}^n}. \tag{4.4}$$

In particular, $\jmath_{\mathcal{H}^n}$ is injective. We will see in (4.18) that to recover a solution to the complex SDO problem from one to its real transformation, it is useful to have an explicit formula for

$$\jmath^*_{\mathcal{H}^n}\begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix} = \frac{1}{\sqrt{2}}(S_{11}+S_{22}) + \frac{\mathrm{i}}{\sqrt{2}}(S_{21}-S_{12}), \tag{4.5}$$

in which we have assumed that S_{11}, $S_{22} \in \mathcal{S}^n$ and $S_{12}^{\mathsf{T}} = S_{21}$.

4.1.2 Ring homomorphisms

As observed in [19], the operator $\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}} := \sqrt{2}\,\jmath_{\mathbb{C}^{m\times n}_{\mathbb{R}}}$ having at $M \in \mathbb{C}^{m\times n}$ the value

$$\boxed{\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M) = \begin{pmatrix} \Re(M) & -\Im(M) \\ \Im(M) & \Re(M) \end{pmatrix}}$$

is a ring homomorphism, since it satisfies

$$\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M_1+M_2) = \hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M_1) + \hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M_2), \qquad \hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(I) = I,$$

and the property in point 1 of the next proposition. This last powerful property and its consequences will allow us to compare the iterates generated by an interior-point algorithm in the complex and real spaces. We give the straightforward proof of the proposition, to be comprehensive and because it plays a crucial role in the sequel.

Proposition 4.1 ($\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}$ is a ring homomorphism) 1) *For any $M \in \mathbb{C}^{m\times n}_{\mathbb{R}}$ and $N \in \mathbb{C}^{n\times p}_{\mathbb{R}}$, there holds $\hat{\jmath}_{\mathbb{C}^{m\times p}_{\mathbb{R}}}(MN) = \hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)\,\hat{\jmath}_{\mathbb{C}^{n\times p}_{\mathbb{R}}}(N)$.*

2) *If $M \in \mathbb{C}^{n\times n}$ is nonsingular, then $\hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(M)$ is nonsingular and $\hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(M^{-1}) = \hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(M)^{-1}$.*

Proof. 1) Let $M = M_{\mathrm{r}} + \mathrm{i}M_{\mathrm{i}}$ and $N = N_{\mathrm{r}} + \mathrm{i}N_{\mathrm{i}}$, with M_{r}, $M_{\mathrm{i}} \in \mathbb{R}^{m\times n}$ and N_{r}, $N_{\mathrm{i}} \in \mathbb{R}^{n\times p}$. There holds $MN = (M_{\mathrm{r}}N_{\mathrm{r}} - M_{\mathrm{i}}N_{\mathrm{i}}) + \mathrm{i}(M_{\mathrm{r}}N_{\mathrm{i}} + M_{\mathrm{i}}N_{\mathrm{r}})$ and therefore

$$\hat{\jmath}_{\mathbb{C}^{m\times p}_{\mathbb{R}}}(MN) = \begin{pmatrix} M_{\mathrm{r}}N_{\mathrm{r}} - M_{\mathrm{i}}N_{\mathrm{i}} & -M_{\mathrm{r}}N_{\mathrm{i}} - M_{\mathrm{i}}N_{\mathrm{r}} \\ M_{\mathrm{r}}N_{\mathrm{i}} + M_{\mathrm{i}}N_{\mathrm{r}} & M_{\mathrm{r}}N_{\mathrm{r}} - M_{\mathrm{i}}N_{\mathrm{i}} \end{pmatrix},$$

which has the same value as

$$\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)\,\hat{\jmath}_{\mathbb{C}^{n\times p}_{\mathbb{R}}}(N) = \begin{pmatrix} M_{\mathrm{r}} & -M_{\mathrm{i}} \\ M_{\mathrm{i}} & M_{\mathrm{r}} \end{pmatrix}\begin{pmatrix} N_{\mathrm{r}} & -N_{\mathrm{i}} \\ N_{\mathrm{i}} & N_{\mathrm{r}} \end{pmatrix} = \begin{pmatrix} M_{\mathrm{r}}N_{\mathrm{r}} - M_{\mathrm{i}}N_{\mathrm{i}} & -M_{\mathrm{r}}N_{\mathrm{i}} - M_{\mathrm{i}}N_{\mathrm{r}} \\ M_{\mathrm{r}}N_{\mathrm{i}} + M_{\mathrm{i}}N_{\mathrm{r}} & M_{\mathrm{r}}N_{\mathrm{r}} - M_{\mathrm{i}}N_{\mathrm{i}} \end{pmatrix}.$$

2) Applying point 1 to the identity $MM^{-1} = I$ yields $\hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(M)\,\hat{\jmath}_{\mathbb{C}^{n\times n}_{\mathbb{R}}}(M^{-1}) = I_{2n}$, hence the result. □

Note that

$$\jmath_{\mathbb{C}^n_{\mathbb{R}}}(v) = \hat{\jmath}_{\mathbb{C}^{n\times 1}_{\mathbb{R}}}(v)\,e^1, \tag{4.6}$$

where $e^1 := (1\ \ 0)^{\mathsf{T}}$ is the first basis vector of $\mathbb{R}^2$. Therefore, for $M \in \mathbb{C}^{m\times n}$ and $v \in \mathbb{C}^n$:

$$\jmath_{\mathbb{C}^m_{\mathbb{R}}}(Mv) = \Big(\hat{\jmath}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)\Big)\Big(\jmath_{\mathbb{C}^n_{\mathbb{R}}}(v)\Big), \tag{4.7}$$

since by (4.6) and point 1 of the previous proposition, there holds $\mathcal{J}_{\mathbb{C}^m_{\mathbb{R}}}(Mv) = \hat{\mathcal{J}}_{\mathbb{C}^{m\times 1}_{\mathbb{R}}}(Mv)\, e^1$ $= \hat{\mathcal{J}}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)\hat{\mathcal{J}}_{\mathbb{C}^{n\times 1}_{\mathbb{R}}}(v)\, e^1 = \hat{\mathcal{J}}_{\mathbb{C}^{m\times n}_{\mathbb{R}}}(M)\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v)$.

We denote by $\mathcal{R}(L)$ the range space of a linear operator L.

Proposition 4.2 ($\hat{\mathcal{J}}_{\mathcal{H}^n}$ on a psd matrix) *Let $M \in \mathcal{H}^n$. Then*
1) *$M \succcurlyeq 0$ if and only if $\hat{\mathcal{J}}_{\mathcal{H}^n}(M) \succcurlyeq 0$, or equivalently $\hat{\mathcal{J}}_{\mathcal{H}^n}(\mathcal{H}^n_+) = \mathcal{S}^{2n}_+ \cap \mathcal{R}(\hat{\mathcal{J}}_{\mathcal{H}^n})$,*
2) *$M \succ 0$ if and only if $\hat{\mathcal{J}}_{\mathcal{H}^n}(M) \succ 0$, or equivalently $\hat{\mathcal{J}}_{\mathcal{H}^n}(\mathcal{H}^n_{++}) = \mathcal{S}^{2n}_{++} \cap \mathcal{R}(\hat{\mathcal{J}}_{\mathcal{H}^n})$,*
3) *if $M \succcurlyeq 0$, then $\hat{\mathcal{J}}_{\mathcal{H}^n}(M^{1/2}) = \hat{\mathcal{J}}_{\mathcal{H}^n}(M)^{1/2}$.*

Proof. For a vector $v \in \mathbb{C}^n$, there hold

$$\begin{aligned} v^{\mathsf{H}} M v &= (e^1)^{\mathsf{T}} \hat{\mathcal{J}}_{\mathbb{C}^{1\times 1}_{\mathbb{R}}}(v^{\mathsf{H}} M v) e^1 \qquad [\text{definition of } \hat{\mathcal{J}}_{\mathbb{C}^{1\times 1}_{\mathbb{R}}}] \\ &= (e^1)^{\mathsf{T}} \hat{\mathcal{J}}_{\mathbb{C}^{n\times 1}_{\mathbb{R}}}(v)^{\mathsf{T}}\, \hat{\mathcal{J}}_{\mathcal{H}^n}(M)\, \hat{\mathcal{J}}_{\mathbb{C}^{n\times 1}_{\mathbb{R}}}(v) e^1 \qquad [\text{proposition 4.1(1) and (4.2)}] \\ &= \mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v)^{\mathsf{T}}\, \hat{\mathcal{J}}_{\mathcal{H}^n}(M)\, \mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v) \qquad [(4.6)]. \end{aligned}$$

Since $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v)$ can represent an arbitrary vector in $\mathbb{R}^{2n}$, and $v \neq 0$ if and only if $\mathcal{J}_{\mathbb{C}^n_{\mathbb{R}}}(v) \neq 0$, the first parts of points 1 and 2 follow.

The second parts of points 1 and 2 follow directly from the first parts. The equivalence of the two parts also results from the injectivity of $\hat{\mathcal{J}}_{\mathcal{H}^n}$.

Consider point 3. If $M \succcurlyeq 0$, it has a unique positive semi-definite square root in $\mathcal{H}^n_+$ [11; § 4.2.10], denoted $M^{1/2} \succcurlyeq 0$, and $\hat{\mathcal{J}}_{\mathcal{H}^n}(M^{1/2}) \succcurlyeq 0$ by point 2. Furthermore

$$\hat{\mathcal{J}}_{\mathcal{H}^n}(M^{1/2})\, \hat{\mathcal{J}}_{\mathcal{H}^n}(M^{1/2}) = \hat{\mathcal{J}}_{\mathcal{H}^n}(M),$$

by proposition 4.1(1). Hence $\hat{\mathcal{J}}_{\mathcal{H}^n}(M^{1/2}) = \hat{\mathcal{J}}_{\mathcal{H}^n}(M)^{1/2}$ (uniqueness of the square root). □

Actually, one can be more precise and show that the eigenvalues of $\hat{\mathcal{J}}_{\mathcal{H}^n}(M)$ are exactly those of M and that the dimension of the eigenspace associated with some eigenvalue is doubled when one considers $\hat{\mathcal{J}}_{\mathcal{H}^n}(M)$ instead of M [9].

Proposition 4.3 ($\hat{\mathcal{J}}^*_{\mathcal{H}^n}$ on a psd matrix) *$\hat{\mathcal{J}}^*_{\mathcal{H}^n}(\mathcal{S}^{2n}_+) = \mathcal{H}^n_+$ and $\hat{\mathcal{J}}^*_{\mathcal{H}^n}(\mathcal{S}^{2n}_{++}) = \mathcal{H}^n_{++}$.*

Proof. By proposition 4.2(1), (4.4), and the fact that $\mathcal{H}^n_+$ is a cone, there holds

$$\mathcal{H}^n_+ = \hat{\mathcal{J}}^*_{\mathcal{H}^n}(\mathcal{S}^{2n}_+ \cap \mathcal{R}(\hat{\mathcal{J}}_{\mathcal{H}^n})) = \hat{\mathcal{J}}^*_{\mathcal{H}^n}(\mathcal{S}^{2n}_+),$$

where the last equality comes from the fact that $\hat{\mathcal{J}}^*_{\mathcal{H}^n}(\mathcal{R}(\hat{\mathcal{J}}_{\mathcal{H}^n})^\perp) = \{0\}$. The second identity can be proven similarly, using proposition 4.2(2). □

Note that $\hat{\mathcal{J}}^*_{\mathcal{H}^n}(\tilde{M}) \succcurlyeq 0$ with $\tilde{M} \in \mathcal{S}^{2n}$ does not imply that $\tilde{M} \succcurlyeq 0$, since $\tilde{M}$ may have a large component in $\mathcal{R}(\hat{\mathcal{J}}_{\mathcal{H}^n})^\perp = \mathcal{N}(\hat{\mathcal{J}}^*_{\mathcal{H}^n})$ that prevents its positive semi-definiteness. For a counter-example with $n = 1$, observe using (4.5) that $\hat{\mathcal{J}}^*_{\mathcal{H}^n}(\tilde{M}) = 2 \succcurlyeq 0$ for the matrix

$$\tilde{M} = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \not\succcurlyeq 0.$$

4.2 Transformation into real numbers

4.2.1 Transformation of the primal problem into real numbers

Let us consider first the transformation of the primal problem (3.1) into real numbers. The complex constraints of this problem, those with index k in K_c, can be transformed into real constraints by using (2.8). This yields the following equivalent problem

$$(P') \quad \begin{cases} \inf \langle C, X \rangle_{\mathcal{H}^n} \\ \langle A_k, X \rangle_{\mathcal{H}^n} = b_k, \quad \forall\, k \in K_r \\ \langle \mathcal{H}(A_k), X \rangle_{\mathcal{H}^n} = \Re(b_k), \quad \forall\, k \in K_c \\ \langle \mathrm{i}\mathcal{Z}(A_k), X \rangle_{\mathcal{H}^n} = \Im(b_k), \quad \forall\, k \in K_c \\ X \succcurlyeq 0. \end{cases} \tag{4.8}$$

Since one can transform complex constraints into real ones, one can wonder whether it is useful to make more effort to deal with additional complex constraints (we have already said at the end of the paragraph containing (3.2) why it is not appropriate to discard the real affine constraints). Actually, we will see in the observation 6.1(1) that it is always faster to deal with problem (P) in (3.1) than with problem (P') above, which we feel as a sufficient justification for introducing complex constraints in (P).

Problem (4.8) is still expressed in complex numbers, since C, A_k, $\mathcal{H}(A_k)$, $\mathrm{i}\mathcal{Z}(A_k)$, and X are complex Hermitian matrices. Using the isometry property (4.3), it looks natural to take as real number version of the primal problem, the following problem in $\tilde{X} \in \mathcal{S}^{2n}$:

$$(\tilde{P}) \quad \begin{cases} \inf \langle \mathcal{J}_{\mathcal{H}^n}(C), \tilde{X} \rangle_{\mathcal{S}^{2n}} \\ \langle \mathcal{J}_{\mathcal{H}^n}(A_k), \tilde{X} \rangle_{\mathcal{S}^{2n}} = b_k, \quad \forall\, k \in K_r \\ \langle \mathcal{J}_{\mathcal{H}^n}(\mathcal{H}(A_k)), \tilde{X} \rangle_{\mathcal{S}^{2n}} = \Re(b_k), \quad \forall\, k \in K_c \\ \langle \mathcal{J}_{\mathcal{H}^n}(\mathrm{i}\mathcal{Z}(A_k)), \tilde{X} \rangle_{\mathcal{S}^{2n}} = \Im(b_k), \quad \forall\, k \in K_c \\ \tilde{X} \succcurlyeq 0, \end{cases} \tag{4.9}$$

where $\tilde{X}$ plays the role of $\mathcal{J}_{\mathcal{H}^n}(X)$. Now, as we shall see in proposition 4.6 and example 4.7, there is no guarantee that a solution to (4.9) is in the range space of $\mathcal{J}_{\mathcal{H}^n}$, although its image by $\mathcal{J}^*_{\mathcal{H}^n}$ is a solution to problem (P). Instead of using the isomorphism $\mathcal{J}_{\mathcal{H}^n}$ of section 4.1.1, one could have used the ring homomorphism $\hat{\mathcal{J}}_{\mathcal{H}^n}$ of section 4.1.2, but with the additional factor 2 in the right-hand side of the affine constraints.

Observation 4.4 The first advantage of dealing directly with the complex SDO problem (3.1), with (3.5), rather than with its real number transformation (4.9) can already be observed at this point: *the former requires less memory storage*. Indeed the unknown matrix $X \in \mathcal{H}^n$ takes twice less memory storage than its real number counterpart $\tilde{X} \in \mathcal{S}^{2n}$ ($2n^2$ real numbers against $4n^2$ if the full matrices are stored). The same observation holds for the matrices C and A_k (with $k \in K_r$), when they are transformed into $\mathcal{J}_{\mathcal{H}^n}(C)$ and $\mathcal{J}_{\mathcal{H}^n}(A_k)$. For the matrices A_k (with $k \in K_c$), the ratio is 4, since the $2n^2$ real numbers of A_k become the $8n^2$ real numbers of $\mathcal{J}_{\mathcal{H}^n}(\mathcal{H}(A_k))$ and $\mathcal{J}_{\mathcal{H}^n}(\mathrm{i}\mathcal{Z}(A_k))$, if the full matrices are stored. □

Problem (4.9) can be written compactly as a standard primal SDO problem in real numbers as follows

$$(\tilde{P}) \quad \begin{cases} \inf \langle \tilde{C}, \tilde{X} \rangle_{\mathcal{S}^{2n}} \\ \tilde{\mathcal{A}}(\tilde{X}) = \tilde{b} \\ \tilde{X} \succcurlyeq 0, \end{cases} \tag{4.10}$$

where $\tilde{C} \coloneqq \jmath_{\mathcal{H}^n}(C)$, while the map $\tilde{\mathcal{A}} : \mathcal{S}^{2n} \to \mathbb{R}^{m_r+2m_c}$ and the vector $\tilde{b} \in \mathbb{R}^{m_r+2m_c}$, whose meanings are easily guessed from (4.9), are now defined with additional objects that will be useful below. The feasible set of problem $(\tilde{P})$ is denoted by $\mathcal{F}_{\tilde{P}}$, its solution set by $\mathcal{S}_{\tilde{P}}$, and its optimal value by $\mathrm{val}(\tilde{P})$.

The form of the right-hand side $\tilde{b}$ of the affine constraints in (4.9) suggests us to introduce the $\mathbb{R}$-linear bijection

$$\jmath_{\mathbb{F}} \coloneqq (a_{K_r}, a_{K_c}) \in \mathbb{F} \coloneqq \mathbb{R}^{m_r} \times \mathbb{C}^{m_c} \mapsto \big(a_{K_r}, \Re(a_{K_c}), \Im(a_{K_c})\big) \in \mathbb{R}^{m_r+2m_c}, \tag{4.11}$$

since then

$$\tilde{b} = \jmath_{\mathbb{F}}(b). \tag{4.12}$$

Note that, with the scalar product (3.4) defined on $\mathbb{F}$ and the Euclidean scalar product on $\mathbb{R}^{m_r+2m_c}$, there holds

$$\langle \jmath_{\mathbb{F}}(a), \jmath_{\mathbb{F}}(b) \rangle_{\mathbb{R}^{m_r+2m_c}} = \langle a, b \rangle_{\mathbb{F}}, \tag{4.13}$$

so that $\jmath_{\mathbb{F}}$ is also an isometry. It is also nonsingular and satisfies

$$\jmath_{\mathbb{F}}^*(\tilde{a}) = \jmath_{\mathbb{F}}^{-1}(\tilde{a}) = (\tilde{a}_{K_r}, \tilde{a}_{K_c} + \mathrm{i}\tilde{a}'_{K_c}), \tag{4.14}$$

where we have assumed that $\tilde{a}$ reads $((\tilde{a}_k)_{k\in K_r}, (\tilde{a}_k)_{k\in K_c}, (\tilde{a}'_k)_{k\in K_c}) \in \mathbb{R}^{m_r+2m_c}$. With the new operator $\jmath_{\mathbb{F}}$, the map $\tilde{\mathcal{A}} : \mathcal{S}^{2n} \to \mathbb{R}^{m_r+2m_c}$ in $(\tilde{P})$ reads simply

$$\tilde{\mathcal{A}} \coloneqq \jmath_{\mathbb{F}} \circ \mathcal{A} \circ \jmath_{\mathcal{H}^n}^*. \tag{4.15}$$

From (4.4), one deduces that

$$\tilde{\mathcal{A}} \circ \jmath_{\mathcal{H}^n} \coloneqq \jmath_{\mathbb{F}} \circ \mathcal{A}. \tag{4.16}$$

Proposition 4.5 (surjectivity of $\tilde{\mathcal{A}}$) *The map $\mathcal{A}$ introduced in problem (3.1) is surjective if and only if the map $\tilde{\mathcal{A}}$ in problem (4.10) is surjective.*

Proof. Straightforward from (4.15) and the properties of $\jmath_{\mathcal{H}^n}$ and $\jmath_{\mathbb{F}}$. □

In the next proposition, we highlight some of the links between the feasible and solution sets of (P) and $(\tilde{P})$. In particular it is shown how $\jmath_{\mathcal{H}^n}^*$ can be used to recover a solution to (P) from a solution to its real number counterpart $(\tilde{P})$; this will be useful in the experiments. The proposition does not assume either that the feasible or solution sets of (P) or $(\tilde{P})$ are nonempty, or even that the optimal values are finite.

Proposition 4.6 (feasible and solution sets of (P) and $(\tilde{P})$) *There hold*

$$\jmath_{\mathcal{H}^n}(\mathcal{F}_P) = \mathcal{F}_{\tilde{P}} \cap \mathcal{R}(\jmath_{\mathcal{H}^n}) \quad \text{and} \quad \jmath_{\mathcal{H}^n}^*(\mathcal{F}_{\tilde{P}}) = \mathcal{F}_P, \tag{4.17}$$

$$\jmath_{\mathcal{H}^n}(\mathcal{S}_P) = \mathcal{S}_{\tilde{P}} \cap \mathcal{R}(\jmath_{\mathcal{H}^n}) \quad \text{and} \quad \jmath_{\mathcal{H}^n}^*(\mathcal{S}_{\tilde{P}}) = \mathcal{S}_P, \tag{4.18}$$

and $\mathrm{val}(P) = \mathrm{val}(\tilde{P})$.

Proof. Let us just prove (4.18). The proofs of the other claims are given in [9].

- $[\jmath_{\mathcal{H}^n}(\mathcal{S}_P) \subseteq \mathcal{S}_{\tilde{P}} \cap \mathcal{R}(\jmath_{\mathcal{H}^n})]$ Let $X \in \mathcal{S}_P$ and set $\tilde{X} := \jmath_{\mathcal{H}^n}(X)$. By (4.17), $\tilde{X} \in \mathcal{F}_{\tilde{P}}$. Now, $\tilde{X} \in \mathcal{S}_{\tilde{P}}$ because for any $\tilde{X}' \in \mathcal{F}_{\tilde{P}}$, there holds

$$\begin{aligned} \langle \tilde{C}, \tilde{X}' \rangle_{\mathcal{S}^{2n}} &= \langle C, \jmath^*_{\mathcal{H}^n}(\tilde{X}') \rangle_{\mathcal{H}^n} \qquad [\tilde{C} = \jmath_{\mathcal{H}^n}(C)] \\ &\geqslant \langle C, X \rangle_{\mathcal{H}^n} \qquad [X \in \mathcal{S}_P \text{ and } \jmath^*_{\mathcal{H}^n}(\tilde{X}') \in \mathcal{F}_P] \\ &= \langle \tilde{C}, \tilde{X} \rangle_{\mathcal{S}^{2n}} \qquad [(4.3)]. \end{aligned}$$

- $[\jmath^*_{\mathcal{H}^n}(\mathcal{S}_{\tilde{P}}) \subseteq \mathcal{S}_P]$ Let $\tilde{X} \in \mathcal{S}_{\tilde{P}}$ and set $X := \jmath^*_{\mathcal{H}^n}(\tilde{X})$. By (4.17), $X \in \mathcal{F}_P$. Now, $X \in \mathcal{S}_P$ because for any $X' \in \mathcal{F}_P$, there holds

$$\begin{aligned} \langle C, X' \rangle_{\mathcal{H}^n} &= \langle \tilde{C}, \jmath_{\mathcal{H}^n}(X') \rangle_{\mathcal{S}^{2n}} \qquad [(4.3)] \\ &\geqslant \langle \tilde{C}, \tilde{X} \rangle_{\mathcal{H}^n} \qquad [\tilde{X} \in \mathcal{S}_{\tilde{P}} \text{ and } \jmath_{\mathcal{H}^n}(X') \in \mathcal{F}_{\tilde{P}}] \\ &= \langle C, X \rangle_{\mathcal{H}^n} \qquad [\tilde{C} = \jmath_{\mathcal{H}^n}(C) \text{ and } \jmath^*_{\mathcal{H}^n}(\tilde{X}) = X]. \end{aligned}$$

- $[\jmath^*_{\mathcal{H}^n}(\mathcal{S}_{\tilde{P}}) \supseteq \mathcal{S}_P]$ Just apply $\jmath^*_{\mathcal{H}^n}$ to both sizes of the first proven inclusion $\jmath_{\mathcal{H}^n}(\mathcal{S}_P) \subseteq \mathcal{S}_{\tilde{P}}$ and use (4.4).
- $[\jmath_{\mathcal{H}^n}(\mathcal{S}_P) \supseteq \mathcal{S}_{\tilde{P}} \cap \mathcal{R}(\jmath_{\mathcal{H}^n})]$ Let $\tilde{X} := \jmath_{\mathcal{H}^n}(X) \in \mathcal{S}_{\tilde{P}}$ for some $X \in \mathcal{H}^n$. Then $X = \jmath^*_{\mathcal{H}^n}(\tilde{X}) \in \mathcal{S}_P$ by (4.4) and the second proven inclusion. Therefore $\tilde{X} = \jmath_{\mathcal{H}^n}(X) \in \jmath_{\mathcal{H}^n}(\mathcal{S}_P)$. □

The next example shows that $\mathcal{F}_{\tilde{P}}$ and $\mathcal{S}_{\tilde{P}}$ may contain points that are not in $\mathcal{R}(\jmath_{\mathcal{H}^n})$. Therefore, the intersections with $\mathcal{R}(\jmath_{\mathcal{H}^n})$ in the right-hand sides of the first identities in (4.17) and (4.18) cannot be removed.

Example 4.7 (a simple complex SDO problem) Consider one of the simplest complex SDO problems (3.1), in which $n = 2$, $m_r = 1$, $m_c = 0$,

$$C = I_2, \quad A = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix} \in \mathcal{H}^2, \quad \text{and} \quad b = -2.$$

The problem can be rewritten in terms of the elements of $X \in \mathcal{H}^2$ as follows

$$\begin{cases} \inf \ X_{11} + X_{22} \\ \mathfrak{I}(X_{12}) = 1 \\ X_{11} \geqslant 0, \quad X_{22} \geqslant 0, \quad X_{11} X_{22} \geqslant \mathfrak{R}(X_{12})^2 + 1. \end{cases} \tag{4.19}$$

This problem has the optimal value 2 and for unique solution

$$X^* = \begin{pmatrix} 1 & \mathrm{i} \\ -\mathrm{i} & 1 \end{pmatrix}.$$

The transformation of the problem into real numbers is the problem in $\tilde{X} \in \mathcal{S}^4$, obtained thanks to isomorphism $\jmath_{\mathcal{H}^n}$ like in (4.9):

$$\begin{cases} \inf \ \frac{1}{\sqrt{2}} (\tilde{X}_{11} + \tilde{X}_{22} + \tilde{X}_{33} + \tilde{X}_{44}) \\ -\tilde{X}_{14} + \tilde{X}_{23} + \tilde{X}_{32} - \tilde{X}_{41} = 2\sqrt{2} \\ \tilde{X} \succcurlyeq 0. \end{cases} \tag{4.20}$$

We know from proposition 4.6 that

$$\jmath_{\mathcal{H}^n}(X^*) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

is a solution to $(\tilde{P})$. We claim that $\jmath_{\mathcal{H}^n}(X^*) + tD$ is also a solution to $(\tilde{P})$ if $t \in [0,1]$ and

$$D = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & -1 \\ 1 & 0 & 0 & -1 \\ 0 & -1 & -1 & 0 \end{pmatrix} \in \mathcal{R}(\jmath_{\mathcal{H}^n})^\perp = \mathcal{N}(\jmath^*_{\mathcal{H}^n}),$$

see formula (4.5). Indeed, since $(\tilde{P})$ is a convex problem, it suffices to show that $\jmath_{\mathcal{H}^n}(X^*)+D$ is in $\mathcal{S}_{\tilde{P}}$. Since $D \in \mathcal{N}(\jmath^*_{\mathcal{H}^n})$, there holds $\langle \jmath_{\mathcal{H}^n}(C), \jmath_{\mathcal{H}^n}(X^*) + D\rangle = \langle \jmath_{\mathcal{H}^n}(C), \jmath_{\mathcal{H}^n}(X^*)\rangle = \langle C, X^*\rangle$, so that the objective of $(\tilde{P})$ takes at $\jmath_{\mathcal{H}^n}(X^*) + D$ its optimal value. Let us now show that $\jmath_{\mathcal{H}^n}(X^*) + D$ is feasible for $(\tilde{P})$. The affine constraint of $(\tilde{P})$ is satisfied by $\jmath_{\mathcal{H}^n}(X^*) + D$, since $\langle \jmath_{\mathcal{H}^n}(A), \jmath_{\mathcal{H}^n}(X^*) + D\rangle = \langle A, X^*\rangle = b$. It remains to show that

$$\jmath_{\mathcal{H}^n}(X^*) + D = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix} \text{ is positive semi-definite.}$$

This property is a consequence of that fact that the matrix is symmetric and has only two eigenvalues, 0 and $4/\sqrt{2}$, which are nonegative (othonormal eigenvectors are $(1,-1,0,0)/\sqrt{2}$ $(1,1,-1,1)/2$, $(0,0,1,1)/\sqrt{2}$, and $(1,1,1,-1)/2$).

Therefore, for this example, $\jmath_{\mathcal{H}^n}(\mathcal{S}_P) \neq \mathcal{S}_{\tilde{P}}$. □

Observation 4.8 The first identity in (4.17) (resp. (4.18)) shows that the feasible (resp. solution) set of the SDO problem in complex numbers is likely to be "smaller" than the one of its real number counterpart. As highlighted by counter-example 4.7, the chance of having a unique solution in the complex number formulation is therefore more important. This may have an impact on theoretical considerations and on the "stability" of the numerical method chosen to solve the problem. □

We now raise the question to know whether $\mathcal{S}_{\tilde{P}}$ may be unbounded while $\mathcal{S}_P$ is bounded (the converse cannot hold by the second identities in (4.17) and (4.18)). This question is of interest for interior-point methods, since an unbounded solution set makes the solution difficult to compute by these algorithms. It would be unfortunate that a path-following interior-point method could be well defined for solving problem (P) but not for solving problem $(\tilde{P})$. The next proposition shows that this situation does not occur.

Proposition 4.9 (bounded solution sets for (P) and $(\tilde{P})$) *There hold*

$$\mathcal{F}_P \text{ is bounded} \iff \mathcal{F}_{\tilde{P}} \text{ is bounded,} \tag{4.21}$$

$$\mathcal{S}_P \text{ is bounded} \iff \mathcal{S}_{\tilde{P}} \text{ is bounded.} \tag{4.22}$$

PROOF. [(4.21)] [⇒] Let X be feasible for (P). Hence $\jmath_{\mathcal{H}^n}(X)$ is feasible for $(\tilde{P})$ by (4.17). Since $\mathcal{F}_{\tilde{P}}$ is a closed convex set, its boundedness will be proven if we show that $\tilde{D} = 0$ when $\jmath_{\mathcal{H}^n}(X) + t\tilde{D} \in \mathcal{F}_{\tilde{P}}$ for $\tilde{D} \in \mathcal{S}^{2n}$ and all $t \geqslant 0$ [33; theorem 8.4]. From the condition $\jmath_{\mathcal{H}^n}(X) + t\tilde{D} \succcurlyeq 0$ for all $t \geqslant 0$, we get

$$\tilde{D} \succcurlyeq 0.$$

On the other hand $\mathcal{J}^*_{\mathcal{H}^n}(\tilde{D}) = 0$ since $\mathcal{J}^*_{\mathcal{H}^n}(\mathcal{J}_{\mathcal{H}^n}(X) + t\tilde{D}) = X + t\mathcal{J}^*_{\mathcal{H}^n}(\tilde{D})$ is feasible for (P) for all $t \geqslant 0$, by (4.17), and $\mathcal{F}_P$ is bounded. Hence, by (4.5), $\tilde{D}$ is of the form

$$\tilde{D} = \begin{pmatrix} \tilde{D}_{11} & \tilde{D}_{12} \\ \tilde{D}_{12} & -\tilde{D}_{11} \end{pmatrix},$$

where the blocks $\tilde{D}_{ij} \in \mathcal{S}^n$. Since both $\tilde{D}_{11}$ and $-\tilde{D}_{11}$ must be positive semidefinite to preserve $\tilde{D} \succcurlyeq 0$, there must hold $\tilde{D}_{11} = 0$. Next $\tilde{D}_{12} = 0$ to preserve $\tilde{D} \succcurlyeq 0$. We have shown that $\tilde{D} = 0$ and therefore the boundedness of $\mathcal{F}_{\tilde{P}}$.

[(4.21)] [$\Leftarrow$] Since $\mathcal{F}_P = \mathcal{J}^*_{\mathcal{H}^n}(\mathcal{F}_{\tilde{P}})$ by (4.17), the boundedness of $\mathcal{F}_{\tilde{P}}$ implies that of $\mathcal{F}_P$.

[(4.22)] The proof is similar, using (4.18) instead of (4.17), see [9] for the details. □

4.2.2 Transformation of the dual problem into real numbers

The real Lagrange dual SDO problem reads

$$\begin{cases} \sup \langle \mathcal{J}_{\mathbb{F}}(b), \tilde{y} \rangle_{\mathbb{R}^{m_r+2m_c}} \\ \sum_{k\in K_r} \tilde{y}_k \mathcal{J}_{\mathcal{H}^n}(A_k) + \sum_{l\in K_c} \tilde{y}_l \mathcal{J}_{\mathcal{H}^n}(\mathcal{H}(A_l)) + \sum_{l\in K_c} \tilde{y}'_l \mathcal{J}_{\mathcal{H}^n}(\mathrm{i}\mathcal{Z}(A_l)) + \tilde{S} = \mathcal{J}_{\mathcal{H}^n}(C) \\ \tilde{S} \succcurlyeq 0. \end{cases} \tag{4.23}$$

This one can be obtained by writing the Lagrange dual of the real SDO problem (4.9)–(4.10) or by translating the complex Lagrange dual (3.6) into real numbers (commutative diagram). In the latter case, $(\tilde{y}, \tilde{S})$ in (4.23) plays the role of $(\mathcal{J}_{\mathbb{F}}(y), \mathcal{J}_{\mathcal{H}^n}(S))$, the objective is obtained by using the isometry identity (4.13), and the affine constraint is the result of applying $\mathcal{J}_{\mathcal{H}^n}$ to both sides of the affine constraint of (3.6). Using the isometry

$$\mathcal{J}_{\mathbb{F}\times\mathcal{H}^n} : (y, S) \in \mathbb{F} \times \mathcal{H}^n \mapsto (\mathcal{J}_{\mathbb{F}}(y), \mathcal{J}_{\mathcal{H}^n}(S)) \in \mathbb{R}^{m_r+2m_c} \times \mathcal{S}^{2n}, \tag{4.24}$$

one can show results similar to propositions 4.6 and 4.9 (see [9] for the proofs). All these yield observation 4.12 below.

Proposition 4.10 (feasible and solution sets of (D) and $(\tilde{D})$) *There hold* $\mathcal{J}_{\mathbb{F}\times\mathcal{H}^n}(\mathcal{F}_D) = \mathcal{F}_{\tilde{D}} \cap \mathcal{R}(\mathcal{J}_{\mathbb{F}\times\mathcal{H}^n})$, $\mathcal{J}^*_{\mathbb{F}\times\mathcal{H}^n}(\mathcal{F}_{\tilde{D}}) = \mathcal{F}_D$, $\mathcal{J}_{\mathbb{F}\times\mathcal{H}^n}(\mathcal{S}_D) = \mathcal{S}_{\tilde{D}} \cap \mathcal{R}(\mathcal{J}_{\mathbb{F}\times\mathcal{H}^n})$, $\mathcal{J}^*_{\mathbb{F}\times\mathcal{H}^n}(\mathcal{S}_{\tilde{D}}) = \mathcal{S}_D$, *and* $\mathrm{val}(D) = \mathrm{val}(\tilde{D})$.

Proposition 4.11 (bounded solution sets for (D) and $(\tilde{D})$) *$\mathcal{F}_D$ is bounded if and only if $\mathcal{F}_{\tilde{D}}$ is bounded. $\mathcal{S}_D$ is bounded if and only if $\mathcal{S}_{\tilde{D}}$ is bounded.*

Observation 4.12 There is no advantage in using the real or complex formulation of the SDO problem with respect to the well-posedness of a primal, dual, or primal-dual interior-point method, since the primal and dual solution sets are nonempty and bounded simultaneously in the two formulations. □

5 A feasible corrector-predictor algorithm

As already said, the goal of this plea is to highlight the advantages of solving a complex SDO problem directly with a complex SDO solver, not to present a competitive complex SDO piece of software. With that objective in mind, we have implemented in Matlab a simplified version of a corrector-predictor path-following interior-point algorithm, using the primal-dual Nesterov-Todd (NT) direction, in which the iterates always satisfy the affine constraint (sometimes qualified as *feasible algorithm*) and remain close to the central path. This algorithm has the advantage of being easily implemented [6]. It is introduced in section 5.1. Section 5.2 shows why the convergence of the algorithm is not deducible from the one of its real version applied to the associated real SDO problem. Section 5.3 gives the details on the computation of the NT directions computed by the algorithm *in complex numbers*. The comparison of the computational effort required by the complex and real algorithms is the subject of section 5.4.

We denote the *primal-dual strictly feasible sets* by

$$\begin{aligned}\mathcal{F}^s &:= \{(X,y,S) \in \mathcal{H}^n_{++} \times \mathbb{F} \times \mathcal{H}^n_{++} : \mathcal{A}(X) = b,\ \mathcal{A}^*(y) + S = C\},\\ \tilde{\mathcal{F}}^s &:= \{(\tilde{X},\tilde{y},\tilde{S}) \in \mathcal{S}^{2n}_{++} \times \mathbb{R}^{m_r+2m_c} \times \mathcal{S}^{2n}_{++} : \tilde{\mathcal{A}}(\tilde{X}) = \tilde{b},\ \tilde{\mathcal{A}}^*(\tilde{y}) + \tilde{S} = \tilde{C}\}.\end{aligned}$$

In all this section, we assume that

$$\mathcal{F}^s \neq \varnothing \text{ and } \mathcal{A} \text{ is surjective} \tag{5.1}$$

or, equivalently, that $\tilde{\mathcal{F}}^s \neq \varnothing$ and $\tilde{\mathcal{A}}$ is surjective (see propositions 4.2, 4.3, 4.5, 4.6, and 4.10).

5.1 Description of the algorithm

This section presents an algorithm that is a faithful adaptation to complex numbers of a corrector-predictor algorithm for real SDO problems discussed and analyzed in details by de Klerk [4; § 7.6]. This one is descending from the homonymous algorithm proposed and studied by Roos, Terlaky, and Vial [34; § 7.7.4] for linear optimization, itself inherited from the contributions of Mehrotra, Mizuno, Todd, and Ye [27, 28]. The algorithm is *primal-dual*, which means that the iterates are triplets $z := (X,y,S)$ made of the primal $X \in \mathcal{H}^n$ and the dual $(y,S) \in \mathbb{F} \times \mathcal{H}^n$ variables. It is also *feasible* in the sense that the iterates belong to $\mathcal{F}^s$. The iterates are forced to follow the *central path* $\mathcal{C}$, which is the image of the map $\mu \in \mathbb{R}_{++} \mapsto z_\mu := (X_\mu, y_\mu, S_\mu) \in \mathcal{H}^n_{++} \times \mathbb{F} \times \mathcal{H}^n_{++}$, where z_μ is the unique solution $z = (X,y,S)$ to the system

$$\begin{cases} \mathcal{A}^*(y) + S = C \\ \mathcal{A}(X) = b \\ XS = \mu I. \end{cases}$$

Uniqueness is indeed guaranteed when the regularity assumption (5.1) holds.

The proximity of the central path is measured as follows. First, one observes that the central point z_μ that is the closest to a given primal-dual triplet $z = (X,y,S)$ is, *in some sense*, the one with the value of μ set to the positive quantity

$$\hat{\mu}_{\mathcal{H}^n}(z) := \frac{\langle X, S\rangle_{\mathcal{H}^n}}{n}, \tag{5.2}$$

Next, one notes that, when $X \succ 0$ and $S \succ 0$, there is a unique matrix W in $\mathcal{H}^n_{++}$ satisfying $WSW = X$, which is given by

$$W := X^{1/2}(X^{1/2}SX^{1/2})^{-1/2}X^{1/2} = S^{-1/2}(S^{1/2}XS^{1/2})^{1/2}S^{-1/2}. \tag{5.3}$$

The matrix square roots in (5.3) make sense since they only act on Hermitian matrices. Since $W^{-1/2}XSW^{1/2}$ is Hermitian, one can take its square root and define the following scaled variable

$$V \equiv V(z) := \left(W^{-1/2}XSW^{1/2}\right)^{1/2} = W^{-1/2}XW^{-1/2} = W^{1/2}SW^{1/2}. \tag{5.4}$$

Observe now that a point $z \in \mathcal{F}^s$ is on the central path $\mathcal{C}$ if and only if $\hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\, V(z) = \hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\, V(z)^{-1}$. Therefore, it looks now natural to define the *proximity measure* of the central path as the map (no square on the norm, despite the factor 1/2, see [15] and [4; § 7.1])

$$\delta : z \in \mathcal{F}^s \mapsto \delta(z) := \frac{1}{2}\left\|\hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\, V(z) - \hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\, V(z)^{-1}\right\|_F \in \mathbb{R}. \tag{5.5}$$

To this proximity measure, one associates the following neighborhoods of the central path, depending on the parameter $\theta \in \mathbb{R}_+$:

$$\mathcal{V}(\theta) := \{z \in \mathcal{F}^s : \delta(z) \leqslant \theta\}. \tag{5.6}$$

The iterates are actually maintained in the neighvorhood $\mathcal{V}(1/3)$ of the central path. Each iterate is computed from the previous one by moving along NT directions. The NT direction $d = (d_X, d_y, d_S) \in \mathcal{H}^n \times \mathbb{F} \times \mathcal{H}^n$ at a point $z \in \mathcal{F}^s$ is the unique solution to (see [31, 20; 1997] and [4; § 7.1])

$$\mathcal{A}^*(d_y) + d_S = 0, \tag{5.7a}$$
$$\mathcal{A}(d_X) = 0, \tag{5.7b}$$
$$d_X + W d_S W = \mu S^{-1} - X. \tag{5.7c}$$

The uniqueness of the NT direction is ensured when the regularity assumption (5.1) holds. When μ is set to $\hat{\mu}_{\mathcal{H}^n}(z)$ in (5.7), the resulting direction is said to be a *corrector direction* and is denoted by d^c (the superscript c is standard and stands for *centering*); and when μ is set to 0, the resulting direction is said to be a *predictor direction* and is denoted by d^a (the superscript a is also standard and comes from *affine scaling*). Each iteration of the algorithm is composed of a correction phase followed by a prediction phase (or vice versa). The goal of the corrector phase is to find an intermediate iterate $z' := z + d^c$ close enough to the central path so that the stepsize along the prediction direction that follows can be taken sufficiently large, while maintaining the next iterate in the neighborhood $\mathcal{V}(1/3)$. The goal of the predictor phase is to decrease as much as possible the positive value of $\hat{\mu}_{\mathcal{H}^n}$ (it vanishes at a solution) by taking a large stepsize

$$\alpha := \frac{2}{1 + \sqrt{1 + \frac{13}{2\hat{\mu}_{\mathcal{H}^n}(z')}\|W^{-1/2}d_X^a d_S^a W^{1/2} + W^{1/2}d_S^a d_X^a W^{-1/2}\|_F}}. \tag{5.8}$$

along d^a to get the next iterate $z_+ := z' + \alpha d^a$ still in $\mathcal{V}(1/3)$ (W and d^a must be computed at z'). Here is a more schematic description.

Algorithm 5.1 (feasible corrector-predictor) A tolerance $\varepsilon > 0$ on $\hat{\mu}_{\mathcal{H}^n}(z)$ is given to determine when stopping the iterations. One iteration of the algorithm, from $z \in$

$\mathcal{V}(1/3)$ to $z_+ \in \mathcal{V}(1/3)$, is formed of the following three phases.

1. *Stopping test.* If $\hat{\mu}_{\mathcal{H}^n}(z) \leqslant \varepsilon$, stop.
2. *Correction phase.* Compute the corrector NT direction d^c, as the solution to (5.7) with $\mu = \hat{\mu}_{\mathcal{H}^n}(z)$. Compute the intermediate iterate
$$z' := z + d^c.$$
3. *Prediction phase.* Compute the predictor NT direction d^a at z', as the solution to (5.7) with $z = z'$ and $\mu = 0$, and the stepsize $\alpha > 0$ given by (5.8). Take as next iterate
$$z_+ := z' + \alpha d^a.$$

5.2 Non corresponding iterates

In view of the nice correspondences between the feasible and solution sets of problems (P) and $(\tilde{P})$ established in section 4.2, one could reasonably think that the iterates generated by the corrector-predictor algorithm 5.1, applied to problem (P), are in correspondence with those generated by the real version of this algorithm, applied to the real transformed SDO problem $(\tilde{P})$. The goal of this section is to invalidate this opinion. The first indication is that some natural linear map, introduced in section 5.2.1, allows us to establish a correspondence between many objects generated during the particular examined iteration (sections 5.2.2, 5.2.3, 5.2.4, and 5.2.6), but not for the stepsizes in the predictor phase, which is larger in the complex space (section 5.2.5). As a result, the complex number algorithm should converge faster than its real analogue (observation 5.3). The second indication will be seen in section 6, which demonstrates that the here suspected increase of iterations of the real algorithm actually occurs in the experiments (observation 6.1(2)).

5.2.1 A possible correspondence

Let us assume that the current iterate $z := (X, y, S) \in \mathcal{F}^s$ generated by algorithm 5.1 for solving the complex SDO problem (P) in (3.1) is in correspondence with the iterate $\tilde{z} := (\tilde{X}, \tilde{y}, \tilde{S}) \in \tilde{\mathcal{F}}^s$ generated by the real version of this algorithm for solving the real version problem $(\tilde{P})$ in (4.10) of (P), through the following rules

$$\tilde{X} = \jmath_{\mathcal{H}^n}(X), \qquad \tilde{y} = \jmath_{\mathbb{F}}(y), \qquad \text{and} \qquad \tilde{S} = \jmath_{\mathcal{H}^n}(S). \tag{5.9}$$

This correspondence is suggested by the fact that $\tilde{X}$ plays the role of $\jmath_{\mathcal{H}^n}(X)$ in (4.9) and $(\tilde{y}, \tilde{S})$ plays the role of $\jmath_{\mathbb{F}\times\mathcal{H}^n}(y, S)$ in (4.23), and could be imposed for the first iterate in order to compare the behavior of the algorithms in the complex and real spaces. Propositions 4.6 and 4.10 reinforce the logic of this correspondence choice. In agreement with (5.9), we introduce the global correspondence map

$$\jmath : z = (X, y, S) \in \mathcal{H}^n \times \mathbb{F} \times \mathcal{H}^n \mapsto \jmath(z) := (\jmath_{\mathcal{H}^n}(X), \jmath_{\mathbb{F}}(y), \jmath_{\mathcal{H}^n}(S)). \tag{5.10}$$

We now ask whether the correspondence (5.9) is still satisfied after one iteration of the corrector-predictor algorithm on the complex and real transformed problems.

To fix the notation, let us specify how the standard real version of algorithm 5.1 proceeds on the real transformed problem (4.10). The variable update is done by

$$\tilde{X}_+ := \tilde{X} + \tilde{\alpha}\tilde{d}_{\tilde{X}}, \qquad \tilde{y}_+ := \tilde{y} + \tilde{\alpha}\tilde{d}_{\tilde{y}}, \qquad \text{and} \qquad \tilde{S}_+ := \tilde{S} + \tilde{\alpha}\tilde{d}_{\tilde{S}}.$$

where $\tilde{d}$ is the solution to

$$\tilde{\mathcal{A}}^*(\tilde{d}_{\tilde{y}}) + \tilde{d}_{\tilde{S}} = 0, \tag{5.11a}$$

$$\tilde{\mathcal{A}}(\tilde{d}_{\tilde{X}}) = 0, \tag{5.11b}$$

$$\tilde{d}_{\tilde{X}} + \tilde{W}\tilde{d}_{\tilde{S}}\tilde{W} = \tilde{\mu}\tilde{S}^{-1} - \tilde{X}. \tag{5.11c}$$

The scaling positive definite matrix $\tilde{W} \in \mathcal{S}^{2n}$ is defined by the analog of (5.3), namely

$$\tilde{W} := \tilde{X}^{1/2}(\tilde{X}^{1/2}\tilde{S}\tilde{X}^{1/2})^{-1/2}\tilde{X}^{1/2} = \tilde{S}^{-1/2}(\tilde{S}^{1/2}\tilde{X}\tilde{S}^{1/2})^{1/2}\tilde{S}^{-1/2} \in \mathcal{S}^{2n} \tag{5.12}$$

and $\tilde{\mu} \in \mathbb{R}_{++}$ is a parameter. In the correction phase at $\tilde{z}$, $\tilde{d}$ is denoted by $\tilde{d}^c$, μ is set to $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z}) := \langle \tilde{X}, \tilde{S}\rangle_{\mathcal{S}^{2n}}/(2n)$, and $\tilde{\alpha} = 1$. The denominator $2n$ in the definition of $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})$ is justified by the size of the matrices in $\mathcal{S}^{2n}$. In the prediction phase at $\tilde{z}' := \tilde{z} + \tilde{d}^c$, $\tilde{d}$ is denoted by $\tilde{d}^a$, μ is set to 0, and the stepsize $\tilde{\alpha} > 0$ is given by the analog of (5.8), namely

$$\tilde{\alpha} := \frac{2}{1 + \sqrt{1 + \frac{13}{2\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z}')}\|\tilde{W}^{-1/2}\tilde{d}^a_{\tilde{X}}\tilde{d}^a_{\tilde{S}}\tilde{W}^{1/2} + \tilde{W}^{1/2}\tilde{d}^a_{\tilde{S}}\tilde{d}^a_{\tilde{X}}\tilde{W}^{-1/2}\|_F}}. \tag{5.13}$$

Since it is desirable to have $\tilde{z} + \tilde{d}^c = \mathcal{J}(z + d^c)$ and $\tilde{z}' + \tilde{\alpha}\tilde{d}^a = \mathcal{J}_{\mathcal{H}^n}(z' + \alpha d^a)$, and since $\mathcal{J}(z + d^c) = \tilde{z} + \mathcal{J}(d^c)$ and $\mathcal{J}(z' + \alpha d^a) = \mathcal{J}(z') + \alpha\mathcal{J}(d^a)$, it is natural to wonder whether

$$\tilde{d} = \mathcal{J}(d) \qquad \text{and} \qquad \tilde{\alpha} = \alpha, \tag{5.14}$$

where $\tilde{d}$ and d stand for the prediction or correction directions.

5.2.2 Correspondence between $\tilde{W}$ and W

By the formulas (5.12) and (5.3) of $\tilde{W}$ and W, and by propositions 4.1 and 4.2(3), one gets

$$\tilde{W} = \hat{\mathcal{J}}_{\mathcal{H}^n}(X)^{1/2}\left(\hat{\mathcal{J}}_{\mathcal{H}^n}(X)^{1/2}\hat{\mathcal{J}}_{\mathcal{H}^n}(S)\hat{\mathcal{J}}_{\mathcal{H}^n}(X)^{1/2}\right)^{-1/2}\hat{\mathcal{J}}_{\mathcal{H}^n}(X)^{1/2} = \hat{\mathcal{J}}_{\mathcal{H}^n}(W). \tag{5.15}$$

5.2.3 Correspondence between $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})$ and $\hat{\mu}_{\mathcal{H}^n}(z)$

With the definitions of $\hat{\mu}_{\mathcal{H}^n}(z)$ in (5.2) and $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})$, and with (4.3), we have

$$\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z}) = \frac{\langle \tilde{X}, \tilde{S}\rangle_{\mathcal{S}^{2n}}}{2n} = \frac{\langle \mathcal{J}_{\mathcal{H}^n}(X), \mathcal{J}_{\mathcal{H}^n}(S)\rangle_{\mathcal{S}^{2n}}}{2n} = \frac{\langle X, S\rangle_{\mathcal{H}^n}}{2n} = \frac{1}{2}\hat{\mu}_{\mathcal{H}^n}(z). \tag{5.16}$$

5.2.4 Correspondence between the NT directions $\tilde{d}$ and d

Let us now show that $\mathcal{J}(d)$ is indeed the solution $\tilde{d}$ to the NT system in the real space (5.11), when $\tilde{\mu}$ and μ are set for a correction or prediction phase, provided $\tilde{z} = \mathcal{J}(z)$. Since, by the regularity assumption (5.1), the system (5.11) has a unique solution, it suffices to show that it is satisfied with $\tilde{d} = \mathcal{J}(d)$.

- [(5.11a)] From (4.15), it results that $\tilde{\mathcal{A}}^* = \mathcal{J}_{\mathcal{H}^n} \circ \mathcal{A}^* \circ \mathcal{J}_{\mathbb{F}}^*$ and, from (4.14), that $\mathcal{J}_{\mathcal{H}^n} \circ \mathcal{A}^* = \tilde{\mathcal{A}}^* \circ \mathcal{J}_{\mathbb{F}}$. Then, applying $\mathcal{J}_{\mathcal{H}^n}$ to equation (5.7a) yields indeed $\tilde{\mathcal{A}}^*(\mathcal{J}_{\mathbb{F}}(d_y)) + \mathcal{J}_{\mathcal{H}^n}(d_S) = 0$, which shows that (5.11a) is verified with $\tilde{d} = \mathcal{J}(d)$.
- [(5.11b)] From (4.15) and (4.4), it results that $\mathcal{J}_{\mathbb{F}} \circ \mathcal{A} = \tilde{\mathcal{A}} \circ \mathcal{J}_{\mathcal{H}^n}$. Then, applying $\mathcal{J}_{\mathbb{F}}$ to equation (5.7b) yields indeed $\tilde{\mathcal{A}}(\mathcal{J}_{\mathcal{H}^n}(d_X)) = 0$, which shows that (5.11b) is verified with $\tilde{d} = \mathcal{J}(d)$.
- [(5.11c)] By applying the ring homomorphism $\hat{\mathcal{J}}_{\mathcal{H}^n} := \sqrt{2}\mathcal{J}_{\mathcal{H}^n}$ to both sides of equation (5.7c) and using proposition 4.1, one gets

$$\hat{\mathcal{J}}_{\mathcal{H}^n}(d_X) + \hat{\mathcal{J}}_{\mathcal{H}^n}(W)\hat{\mathcal{J}}_{\mathcal{H}^n}(d_S)\hat{\mathcal{J}}_{\mathcal{H}^n}(W) = \mu\hat{\mathcal{J}}_{\mathcal{H}^n}(S)^{-1} - \hat{\mathcal{J}}_{\mathcal{H}^n}(X).$$

  Therefore, if we assume that $\tilde{z} = \mathcal{J}(z)$, use (5.15) and (5.9), and divide by $\sqrt{2}$, we get

$$\mathcal{J}_{\mathcal{H}^n}(d_X) + \tilde{W}\mathcal{J}_{\mathcal{H}^n}(d_S)\tilde{W} = \frac{\mu}{2}\tilde{S}^{-1} - \tilde{X}.$$

  Therefore (5.11c) is verified with $\tilde{d} = \mathcal{J}(d)$, whether μ is set to $\hat{\mu}_{\mathcal{H}^n}(z)$ (correction direction, use (5.16) in that case) or to zero (prediction direction).

Up to here, we have shown that $\tilde{z}' = \mathcal{J}(z')$ and $\tilde{d}^a = \mathcal{J}(d^a)$. Therefore, whatever are the values of $\tilde{\alpha}$ et α, we can already claim that the generated sequence of iterates in the real space is in $\mathcal{R}(\mathcal{J})$ if the first iterate is in that space. For a future reference, we express this observation in a proposition.

Proposition 5.2 (real iterates in $\mathcal{R}(\mathcal{J})$) *The sequence generated by the real version of the interior-point algorithm 5.1, described in section 5.2.1, is in the space $\mathcal{R}(\mathcal{J})$ if the first iterate is in $\mathcal{R}(\mathcal{J})$.*

5.2.5 Correspondence between the stepsizes $\tilde{\alpha}$ and α

To see whether $\tilde{z}_+ = \mathcal{J}(z_+)$, we still have to compare the stepsizes $\tilde{\alpha}$ and α. We have seen in the previous section that $\tilde{d}^a = \mathcal{J}(d^a)$, since $\tilde{z}' = \mathcal{J}(z')$. Then, using proposition 4.1 and (5.15), we get

$$\begin{aligned}
\tilde{W}^{-1/2}\tilde{d}_{\tilde{X}}^a\tilde{d}_{\tilde{S}}^a\tilde{W}^{1/2} &= \frac{1}{2}\hat{\mathcal{J}}_{\mathcal{H}^n}(W^{-1/2})\hat{\mathcal{J}}_{\mathcal{H}^n}(d_X^a)\hat{\mathcal{J}}_{\mathcal{H}^n}(d_S^a)\hat{\mathcal{J}}_{\mathcal{H}^n}(W^{1/2}) \\
&= \frac{1}{2}\hat{\mathcal{J}}_{\mathcal{H}^n}(W^{-1/2}d_X^a d_S^a W^{1/2}) \\
&= \frac{1}{\sqrt{2}}\mathcal{J}_{\mathcal{H}^n}(W^{-1/2}d_X^a d_S^a W^{1/2}).
\end{aligned}$$

Similarly, $\tilde{W}^{1/2}\tilde{d}_{\tilde{S}}^a\tilde{d}_{\tilde{X}}^a\tilde{W}^{-1/2} = (1/\sqrt{2})\,\mathcal{J}_{\mathcal{H}^n}(W^{1/2}d_S^a d_X^a W^{-1/2})$. Therefore, since $\mathcal{J}_{\mathcal{H}^n}$ is an isometry, one gets

$$\|\tilde{W}^{-1/2}\tilde{d}_{\tilde{X}}^a\tilde{d}_{\tilde{S}}^a\tilde{W}^{1/2} + \tilde{W}^{1/2}\tilde{d}_{\tilde{S}}^a\tilde{d}_{\tilde{X}}^a\tilde{W}^{-1/2}\|_F = \frac{1}{\sqrt{2}}\|W^{-1/2}d_X^a d_S^a W^{1/2} + W^{1/2}d_S^a d_X^a W^{-1/2}\|_F.$$

Using (5.16), one term in the denominator of the stepsize formula (5.13) reads

$$\begin{aligned}
&\left(\frac{13}{2\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})}\|\tilde{W}^{-1/2}\tilde{d}_{\tilde{X}}^a\tilde{d}_{\tilde{S}}^a\tilde{W}^{1/2} + \tilde{W}^{1/2}\tilde{d}_{\tilde{S}}^a\tilde{d}_{\tilde{X}}^a\tilde{W}^{-1/2}\|_F\right) \\
&\quad = \sqrt{2}\left(\frac{13}{2\hat{\mu}_{\mathcal{H}^n}(z)}\|W^{-1/2}d_X^a d_S^a W^{1/2} + W^{1/2}d_S^a d_X^a W^{-1/2}\|_F\right).
\end{aligned}$$

By the factor $\sqrt{2}$ in the right-hand side, the formulas (5.13) and (5.8) of the stepsizes $\tilde{\alpha}$ and α yield the inequality

$$\tilde{\alpha} < \alpha. \tag{5.17}$$

Hence the stepsizes taken in the prediction phases of the corrector-predictor algorithm when it solves the complex and associated real SDO problems are not identical and, as a result, $\tilde{z}_+$ and z_+ differ.

5.2.6 Correspondence between the proximity measures to the central paths

One may believe that the stepsize formulas (5.8) and (5.13) are not appropriate and that there may exist other formulas that would guarantee the equality of the stepsizes in the prediction phases. The goal of this section is to show that this is unlikely, because the stepsize inequality (5.17) can also be explained by the fact that the neighborhoods of the central path are larger for the complex problem than for the real transformed problem, in a sense that is now specified.

One can introduce a scaled variable $\tilde{V} \equiv \tilde{V}(\tilde{z})$ in the real space as we did by (5.4) for the scaled variable $V \equiv V(z)$ in the complex space. The two variables are linked as follows:

$$\begin{aligned}
\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})^{-1/2}\,\tilde{V} &= \hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})^{-1/2}\,\tilde{W}^{-1/2}\tilde{X}\tilde{W}^{-1/2} && \text{[definition of } \tilde{V}\text{]} \\
&= \hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\,\hat{\jmath}_{\mathcal{H}^n}(W)^{-1/2}\hat{\jmath}_{\mathcal{H}^n}(X)\hat{\jmath}_{\mathcal{H}^n}(W)^{-1/2} && \text{[(5.16), (5.15), (5.9)]} \\
&= \hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\,\hat{\jmath}_{\mathcal{H}^n}\left(W^{-1/2}XW^{-1/2}\right) && \text{[proposition 4.1]} \qquad (5.18) \\
&= \sqrt{2}\,\jmath_{\mathcal{H}^n}\left(\hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\,V\right). && (5.19)
\end{aligned}$$

Furthermore, from (5.18) and proposition 4.1:

$$\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})^{1/2}\,\tilde{V}^{-1} = \hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\,\hat{\jmath}_{\mathcal{H}^n}(V^{-1}) = \sqrt{2}\,\jmath_{\mathcal{H}^n}\left(\hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\,V^{-1}\right). \tag{5.20}$$

One can now introduce a proximity measure $\tilde{\delta}$ in the real space, as we did for δ in the complex space by (5.5). The two proximity measures are linked as follows:

$$\begin{aligned}
\tilde{\delta}(\tilde{z}) &= \frac{1}{2}\left\|\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})^{-1/2}\,\tilde{V} - \hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})^{1/2}\,\tilde{V}^{-1}\right\|_F && \text{[definition of } \tilde{\delta}\text{, like in (5.5)]} \\
&= \frac{1}{2}\left\|\sqrt{2}\,\jmath_{\mathcal{H}^n}\left(\hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\,V\right) - \sqrt{2}\,\jmath_{\mathcal{H}^n}\left(\hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\,V^{-1}\right)\right\|_F && \text{[(5.19) and (5.20)]} \\
&= \frac{\sqrt{2}}{2}\left\|\hat{\mu}_{\mathcal{H}^n}(z)^{-1/2}\,V - \hat{\mu}_{\mathcal{H}^n}(z)^{1/2}\,V^{-1}\right\|_F && \text{[}\jmath_{\mathcal{H}^n}\text{ is an isometry]} \\
&= \sqrt{2}\,\delta(z) && \text{[definition of } \delta\text{]}.
\end{aligned}$$

Therefore

$$\delta(z) \leqslant \theta \quad \Longleftrightarrow \quad \tilde{\delta}(\tilde{z}) \leqslant \sqrt{2}\,\theta. \tag{5.21}$$

5.2.7 Conclusion

Using the global correspondence map $\jmath$ defined in (5.10), the equivalence (5.21), and a size $\theta > 0$, the neighborhoods of the central paths $\mathcal{V}(\theta)$ given by (5.6) and its real analogue $\tilde{V}(\theta) := \{\tilde{z} \in \tilde{\mathcal{F}}^s : \tilde{\delta}(\tilde{z}) \leqslant \theta\}$ correspond to each other by

$$\jmath(\mathcal{V}(\theta)) = \{\jmath(z) : z \in \mathcal{F}^s,\ \delta(z) \leqslant \theta\} = \{\tilde{z} \in \tilde{\mathcal{F}}^s \cap \mathcal{R}(\jmath) : \tilde{\delta}(\tilde{z}) \leqslant \sqrt{2}\,\theta\} = \tilde{\mathcal{V}}(\sqrt{2}\,\theta) \cap \mathcal{R}(\jmath),$$

where we have used propositions 4.6 and 4.10. Therefore

$$\mathcal{J}(\mathcal{V}(\theta)) \supset \tilde{\mathcal{V}}(\theta) \cap \mathcal{R}(\mathcal{J}), \tag{5.22}$$

meaning that the transformed neighborhood $\mathcal{J}(\mathcal{V}(\theta))$ is strictly larger than the neighborhood $\tilde{\mathcal{V}}(\theta) \cap \mathcal{R}(\mathcal{J})$ with the same size θ (for example $\theta = 1/3$ like in algoritm 5.1). Now, we have seen in proposition 5.2 that the generated sequence of iterates in the real space is in $\mathcal{R}(\mathcal{J})$ when the first iterate is in that space. Therefore, the intersection with $\mathcal{R}(\mathcal{J})$ in (5.22) does not prevent us to claim that the reason why the stepsize $\tilde{\alpha}$ in the real space is smaller than the one α in the complex space is intimately linked to the sizes of the neighborhoods and not to the structure of the formulas (5.8) and (5.13). Incidentally, this also means that the convergence of the complex interior-point SDO algorithm 5.1 requires a specific analysis, which is presented in [8].

Observation 5.3 The inequality (5.17) on the stepsizes and the inclusion (5.22) on the neighborhoods suggest us that the considered corrector-predictor algorithm should converge more rapidly on the complex problem than on the real transformed problem. Indeed, the speed of convergence only depends on the reductions of $\hat{\mu}_{\mathcal{H}^n}(z)$ and $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})$ which are given at each iteration (during the predictor phase actually) by

$$\hat{\mu}_{\mathcal{H}^n}(z_+) = (1-\alpha)\hat{\mu}_{\mathcal{H}^n}(z) \qquad \text{and} \qquad \hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z}_+) = (1-\tilde{\alpha})\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z}).$$

Because $\alpha > \tilde{\alpha}$, the reduction of $\hat{\mu}_{\mathcal{H}^n}(z)$ towards zero for the complex problem is more important than that of $\hat{\mu}_{\mathcal{S}^{2n}}(\tilde{z})$ for its associated real problem, when the first real iterate $\tilde{z}_1$ is in $\mathcal{R}(\mathcal{J})$. This observation has been claimed with the conditional because it is based on the examination of a single iteration and that the non-corresponding iterates finally takes non-corresponding paths towards the solution, which are difficult to compare. □

5.3 Computing the NT direction

A standard way of solving (5.7) consists in applying $\mathcal{A}$ to both sides of (5.7c) in order to eliminate d_X using (5.7b). This yields $\mathcal{A}(W d_S W) = \mathcal{A}(\mu S^{-1} - X)$. Next d_S is removed from this last equation by using (5.7a) to obtain a system in only d_y:

$$\mathcal{A}(W\mathcal{A}^*(d_y)W) = \mathcal{A}(X - \mu S^{-1}). \tag{5.23}$$

This linear system is generally solved by a direct method after computing and storing the matrix of the system. Let us look at the structure of that matrix when $\mathcal{A}$ is the $\mathbb{R}$-linear complex map specified by (3.5). Using the definitions of $\mathcal{A}_r$ and $\mathcal{A}_c$ in (3.3), as well as (3.7a), we see that this system can be decomposed into

$$\mathcal{A}_r(W\mathcal{A}_r^*((d_y)_{\mathbb{F}_r})W) + \mathcal{A}_r(W\mathcal{A}_c^*((d_y)_{\mathbb{F}_c})W) = \mathcal{A}_r(X - \mu S^{-1}), \tag{5.24a}$$

$$\mathcal{A}_c(W\mathcal{A}_r^*((d_y)_{\mathbb{F}_r})W) + \mathcal{A}_c(W\mathcal{A}_c^*((d_y)_{\mathbb{F}_c})W) = \mathcal{A}_c(X - \mu S^{-1}). \tag{5.24b}$$

In matrix form, this becomes

$$\begin{pmatrix} \boxed{L^{11}} & L^{12} & L^{13} \\ L^{21} & \boxed{\boxed{L^{22}}} & \boxed{\boxed{L^{23}}} \end{pmatrix} \begin{pmatrix} (d_y)_{K_r} \\ (d_y)_{K_c} \\ (\overline{d_y})_{K_c} \end{pmatrix} = \begin{pmatrix} \mathcal{A}_r(X - \mu S^{-1}) \\ \mathcal{A}_c(X - \mu S^{-1}) \end{pmatrix}, \tag{5.25}$$

where a simple box surrounds the *square real* submatrix $L^{11} \in \mathbb{R}^{m_r \times m_r}$ and double boxes surround the *square complex* submatrices L^{22} and $L^{23} \in \mathbb{C}^{m_c \times m_c}$. The other submatrices

have complex elements: $L^{12} \in \mathbb{C}^{m_r \times m_c}$, $L^{13} \in \mathbb{C}^{m_r \times m_c}$, and $L^{21} \in \mathbb{C}^{m_c \times m_r}$. For section 5.4, it is useful to make explicit the value of the elements of these matrices:

$$L^{11}_{kk'} = \langle G^{\mathsf{H}} A_k G, G^{\mathsf{H}} A_{k'} G \rangle_{\mathcal{H}^n} \in \mathbb{R}, \qquad L^{21}_{lk} = \langle G^{\mathsf{H}} A_l G, G^{\mathsf{H}} A_k G \rangle_{\mathbb{C}^{n\times n}} \in \mathbb{C}, \tag{5.26a}$$

$$L^{12}_{kl} = \tfrac{1}{2} \langle G^{\mathsf{H}} A_k G, G^{\mathsf{H}} A_l G \rangle_{\mathbb{C}^{n\times n}} \in \mathbb{C}, \qquad L^{13}_{kl} = \tfrac{1}{2} \langle G^{\mathsf{H}} A_k G, G^{\mathsf{H}} A_l^{\mathsf{H}} G \rangle_{\mathbb{C}^{n\times n}} \in \mathbb{C}, \tag{5.26b}$$

$$L^{22}_{ll'} = \tfrac{1}{2} \langle G^{\mathsf{H}} A_l G, G^{\mathsf{H}} A_{l'} G \rangle_{\mathbb{C}^{n\times n}} \in \mathbb{C}, \qquad L^{23}_{ll'} = \tfrac{1}{2} \langle G^{\mathsf{H}} A_l G, G^{\mathsf{H}} A_{l'}^{\mathsf{H}} G \rangle_{\mathbb{C}^{n\times n}} \in \mathbb{C}, \tag{5.26c}$$

where the indices $(k, k') \in K_r \times K_r$, $(l, l') \in K_c \times K_c$, and we have used the factorization $W = GG^{\mathsf{H}}$. See [9] for the details.

The next proposition gives properties of the matrices L^{ij}, in particular, conditions are given to ensure the nonsingularity of the matrix L^{11}; see [9] for a proof.

Proposition 5.4 (properties of the matrices L^{ij}) 1) *The real square matrix L^{11} is symmetric and positive semidefinite. If the Hermitian matrices $\{A_k\}_{k\in K_r}$ are $\mathbb{R}$-linearly independent in $\mathcal{H}^n$, then L^{11} is positive definite.*
2) *The complex square matrix L^{22} is Hermitian and positive semidefinite. If the matrices $\{A_l\}_{l\in K_c}$ are $\mathbb{C}$-linearly independent in $\mathbb{C}^{n\times n}$, then L^{22} is positive definite.*
3) *The complex square matrix L^{23} is symmetric and $(L^{13})^{\mathsf{T}} = \frac{1}{2} L^{21} = (L^{12})^{\mathsf{H}}$.*

The difficulty in solving (5.25) resides in the presence of real and complex unknown vectors. Of course, a possibility would be to transform the problem into a linear system in real variables, but this would result in a waste of computing time, as this will be apparent after reading section 5.4. For this reason, we prefer a mixed real-complex resolution procedure. When L^{11} is nonsingular, one can get an expression of $(d_y)_{K_r}$ from the first block row in (5.25), by solving a real square linear system, which gives

$$(d_y)_{K_r} = -(L^{11})^{-1} \left(L^{12} (d_y)_{K_c} + L^{13} (\overline{d_y})_{K_c} \right) + (L^{11})^{-1} \left(\mathcal{A}_r (X - \mu S^{-1}) \right).$$

Substituting this vector in the second block row yields the system

$$Mv + N\overline{v} = p, \tag{5.27a}$$

where $v := (d_y)_{K_c} \in \mathbb{C}^{m_c}$, $p := \mathcal{A}_c(X - \mu S^{-1}) - L^{21}(L^{11})^{-1}(\mathcal{A}_r(X - \mu S^{-1})) \in \mathbb{C}^{m_c}$,

$$M := L^{22} - L^{21}(L^{11})^{-1} L^{12} \in \mathbb{C}^{m_c \times m_c}, \quad \text{and} \quad N := L^{23} - L^{21}(L^{11})^{-1} L^{13} \in \mathbb{C}^{m_c \times m_c}. \tag{5.27b}$$

For a computational efficiency reason, however, we prefer transforming it into a $\mathbb{C}$-linear complex system of the same dimension in the unknown variable v, hence, without the presence of its conjugate $\overline{v}$. This is possible when M is nonsingular (for necessary and sufficient conditions ensuring the possibility of doing this rewriting in the general case, see [7]), which occurs when some regularity conditions are satisfied (see proposition 5.5 below). Then, one can write $v = M^{-1}(p - N\overline{v})$ as an affine function of $\overline{v}$, take its conjugate $\overline{v} = \overline{M}^{-1}(\overline{p} - \overline{N}v)$, and substitute this last value in (5.27a), to get the following linear system in v:

$$\overline{M^s} v = p - N\overline{M}^{-1}\overline{p}, \tag{5.28a}$$

where

$$M^s := \overline{M} - \overline{N} M^{-1} N. \tag{5.28b}$$

The reason for the notation M^s comes from the block elimination that has been realized, making M^s the Schur complement of M in some larger matrix (see [9]). The next proposition shows that the surjectivity of $\mathcal{A}$ is a sufficient condition to have a system (5.28) that is well defined and has a unique solution (see [9] for a proof).

Proposition 5.5 (properties of the systems (5.27) and (5.28)) 1) *If the matrices* $\{A_k\}_{k\in K_r} \cup \{A_l\}_{l\in K_c}$ *are* $(\mathbb{R}^{m_r} \times \mathbb{C}^{m_c})$*-linearly independent in* $\mathbb{C}^{n\times n}$*, then the matrices* M *and* N *are well defined by (5.27b),* M *is Hermitian positive definite, and* N *is symmetric.*

2) *If* $\mathcal{A}$ *is surjective, then the matrix* M^s *is well defined by (5.28b), Hermitian, and positive definite.*

5.4 Computational efforts in the real and complex algorithms

We are now ready to compare the number of *elementary operations*, i.e., the additions and products of real numbers, required to perform an iteration of the interior-point algorithm 5.1 in real and complex numbers. The comparison is restricted to the time consuming operations, those requiring a *finite* number of elementary operations depending on the *cube* of the dimension of the involved objects. To do so, it is necessary to start with the comparison of the elementary operations in complex and real numbers.

Let us denote by $\mathtt{a}_\mathbb{R}$ and $\mathtt{a}_\mathbb{C}$ the computing time for the addition (or subtraction) of two real and complex numbers, and by $\mathtt{p}_\mathbb{R}$ and $\mathtt{p}_\mathbb{C}$ the computing time for the product of two real and complex numbers. If we assume that, for a, b, c, and $d \in \mathbb{R}$, the addition and product of complex numbers are computed by straightforward implementations of the rules $(a+\mathrm{i}b)+(c+\mathrm{i}d) = (a+c)+\mathrm{i}(b+d)$ and $(a+\mathrm{i}b)(c+\mathrm{i}d) = (ac-bd)+\mathrm{i}(ad+bc)$, there hold

$$\mathtt{a}_\mathbb{C} = 2\,\mathtt{a}_\mathbb{R} \qquad \text{and} \qquad \mathtt{p}_\mathbb{C} = 2\,\mathtt{a}_\mathbb{R} + 4\,\mathtt{p}_\mathbb{R}. \tag{5.29}$$

To get an idea of the expensive operations in algorithm 5.1, table 5.1 provides the CPU time percentages spent in the main parts of an iteration in a home-made piece of software in complex numbers, called Sdolab 0.4. These occur in the computation of the NT direction: for computing the weighting matrix W in (5.3) (third last column in the table), the construction of the normal equation (5.25) (second last column) and the solution computation of this one (last column). These percentages are listed for the test-problems considered in section 6.1 (one row per test-problem). We see that the largest part of the computing time is spent in forming the normal equation; this is actually due to the matrix products in (5.26). The second most time consuming operation consists in solving the system (5.25), using in particular a Cholesky factorization to solve (5.28a). We analyze next these two operations.

Let us start by the multiplication of matrices that is present in (5.26) for instance. This is actually the most time consuming part in our implementation as soon as the dimensions of the problem increase. The product of two $n \times n$ matrices requires the computation of n^2 elements, each of which is a scalar products of two vectors of size n. Since the latter requires n products and $n-1$ additions, the total computing time is $n^2(n\,\mathtt{p}+(n-1)\,\mathtt{a}) \simeq n^3(\mathtt{p}+\mathtt{a})$. Using (5.29), this results in the following estimations.

- The computing time of the product of two $n \times n$ complex (Hermitian) matrices can be estimated at

$$\mathtt{mmp}_\mathbb{C}(n) \simeq n^3(\mathtt{a}_\mathbb{C} + \mathtt{p}_\mathbb{C}) = 4n^3(\mathtt{a}_\mathbb{R} + \mathtt{p}_\mathbb{R}). \tag{5.30}$$

					NT direction		
	dimensions			time/iter	scaling	equation (5.25)	
Problems	n	m_r	m_c	(sec)	W	forming	solving
`mmnc-10-0-100-100`	200	11	0	0.414	11.7 %	57.8 %	6.7 %
`mmnc-20-0-100-100`	200	21	0	0.643	7.1 %	72.6 %	5.8 %
`mmnc-50-0-100-100`	200	51	0	1.433	3.4 %	84.7 %	5.0 %
`mmnc-100-0-100-100`	200	101	0	2.964	1.7 %	90.0 %	4.6 %
`mmnc-200-0-100-100`	200	201	0	6.735	0.8 %	92.8 %	4.3 %
`mmnc-500-0-100-100`	200	501	0	23.010	0.2 %	94.9 %	4.4 %
`mmnc-0-10-100-100`	200	1	10	0.418	10.7 %	59.5 %	6.3 %
`mmnc-0-20-100-100`	200	1	20	0.683	6.7 %	74.0 %	4.3 %
`mmnc-0-50-100-100`	200	1	50	1.584	2.9 %	87.2 %	2.6 %
`mmnc-0-100-100-100`	200	1	100	3.504	1.4 %	92.9 %	1.8 %
`mmnc-0-200-100-100`	200	1	200	8.632	0.5 %	95.7 %	1.6 %
`mmnc-0-500-100-100`	200	1	500	34.669	0.1 %	97.4 %	1.9 %
`caseWB2`	12	6	68	0.030	2.8 %	18.1 %	12.1 %
`caseWB5`	35	22	592	1.713	0.1 %	64.4 %	34.0 %
`case9`	57	36	1572	27.968	0.0 %	65.4 %	34.1 %

Table 5.1: Dimensions (n, m_r, m_c) of the test-problems in the form (P) in (3.1) and percentages of the CPU time spent in various parts of an iteration of Sdolab 0.4 in complex numbers for the various test-problems (results obtained on a 1 core machine). Problem `mmnc-`p_1`-`p_2`-`q`-`r denotes the minimum matrix norm problem, described in section 6.1.1, and problems `case*` denote the Matpower problems, described in section 6.1.2. The first group of minimum matrix norm problems is made of problems having no complex constraints, hence being not sensitive to the transformation (P)/(3.1) $\rightsquigarrow$ (P')/(4.8).

- When the same matrices are transformed into real numbers, using the operator $\mathcal{J}_{\mathcal{H}^n}$ defined before (4.3), their size is doubled, so that the computing time of their product can be estimated at

$$\mathtt{mmp}_{\mathbb{R}}(n) \simeq (2n)^3(\mathtt{a}_{\mathbb{R}} + \mathtt{p}_{\mathbb{R}}) = 8n^3(\mathtt{a}_{\mathbb{R}} + \mathtt{p}_{\mathbb{R}}). \tag{5.31}$$

Therefore, one gets the ratio

$$\mathtt{mmp}_{\mathbb{R}}(n)/\mathtt{mmp}_{\mathbb{C}}(n) \simeq 2, \tag{5.32}$$

indicating that the computation is approximately twice faster in complex numbers.

In practice, the ratio $\mathtt{mmp}_{\mathbb{R}}(n)/\mathtt{mmp}_{\mathbb{C}}(n)$ may differ from the one given by this simple estimation, because the product of two matrices may depend on the ability of the function realizing this task to take advantage of the block structure of the memory, the possible parallelization realized by the compiler or interpreter, and the number of cores of a particular machine. Numerical experiment is therefore welcome. The one related in figure 5.1 shows that

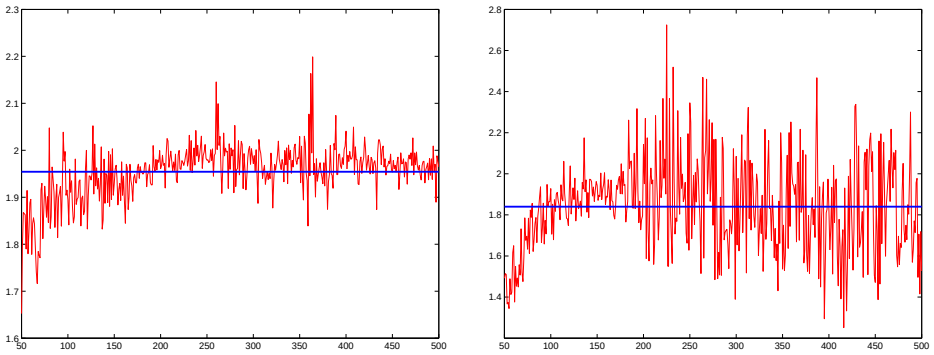

Figure 5.1: Ratio $\mathtt{mmp}_{\mathbb{R}}(n)/\mathtt{mmp}_{\mathbb{C}}(n)$ averaged on 10 runs for the product of matrices of respective order $2n$ and n, for $n \in [50:500]$. Left: one core (mean value 1.95). Right: 4 cores (mean value 1.84).

a mean ratio around 1.95 is obtained for a single core, which is very close to the "theoretical" estimate (5.32).

Similar estimations can be obtained for other operations [9]. For example, for a particular Cholesky factorization (there are many possibilities [13]), one can estimate the ratio of the computing time in real ($\mathtt{cf}_{\mathbb{R}}(n)$) and complex ($\mathtt{cf}_{\mathbb{C}}(n)$) numbers by $\mathtt{cf}_{\mathbb{R}}(n)/\mathtt{cf}_{\mathbb{C}}(n) \simeq (4\mathtt{a}_{\mathbb{R}} + 8\mathtt{p}_{\mathbb{R}})/(2\mathtt{a}_{\mathbb{R}} + 3\mathtt{p}_{\mathbb{R}}) \geqslant 2$, hence a ratio around 2.46 if $\mathtt{p}_{\mathbb{R}} \simeq 1.5\,\mathtt{a}_{\mathbb{R}}$.

6 Experimenting with a home-made SDO solver

The motivation of this paper comes essentially from the observation that the solution to a complex SDO problem can be solved more efficiently if it is not transformed into real numbers. This transformation is done by the SDO solvers we are aware of. For this reason and to be able to master the comparisons between the real and complex versions, we have developed our own solver, called Sdolab 0.4, which is a toy implementing the standard corrector-predictor algorithm described in section 5.1 (see [6] for an elementary description of the preliminary real version of the code). The solver can only consider dense data, so that this data has been forced to have that representation. Before presenting the numerical results in section 6.2, we start by describing the test problems.

6.1 Test problem description

6.1.1 Minimum matrix norm problem

The ℓ_2 norm of a complex vector v is denoted by $\|v\|_2 = (\sum_i |v_i|^2)^{1/2}$, where $|v_i|$ is the module of $v_i \in \mathbb{C}$, and the ℓ_2 norm of a complex matrix B is denoted by $\|B\|_2 := \sup\{\|Mv\|_2 : \|v\|_2 \leqslant 1\}$.

The *minimum matrix norm* problem in complex numbers consists in minimizing the largest singular value of an affine combination $z \in \mathbb{R}^{p_1} \times \mathbb{C}^{p_2}_{\mathbb{R}} \mapsto \mathcal{B}(z) := B_0 + \sum_{i=1}^{p} z_i B_i$ of matrices $B_i \in \mathbb{C}^{q\times r}$. Hence the problem reads

$$\inf_{z\in\mathbb{R}^{p_1}\times\mathbb{C}^{p_2}_{\mathbb{R}}} \|\mathcal{B}(z)\|_2 \,. \tag{6.1}$$

See [41] for the real version of the problem and [40; section 4.6] for complex matrices and real coefficients ($p_2 = 0$). The problem is nonsmooth, but can be expressed as the complex SDO problem

$$\begin{cases} \max_{(t,z)\in\mathbb{R}\times(\mathbb{R}^{p_1}\times\mathbb{C}^{p_2}_{\mathbb{R}})} -t \\ \begin{pmatrix} t\,I_q & \mathcal{B}(z) \\ \mathcal{B}(z)^{\mathsf{H}} & t\,I_r \end{pmatrix} \succcurlyeq 0. \end{cases} \tag{6.2}$$

Problem (6.2) can be viewed as a complex SDO problem, written in the standard dual form (3.6), with dimension $n = q + r$, $m_r = p_1 + 1$, and $m_c = p_2$, and with a linear map $\mathcal{A}$ that is surjective if and only if the matrices $\{B_k\}_{k=1}^{p_1+p_2}$ are $\mathbb{R}^{p_1} \times \mathbb{C}^{p_2}$-linearly independent [9].

6.1.2 Three small OPF problems

As already said in the introduction, one of the emblematic instance of the so-called Optimal Power Flow (OPF) problem, in alternating current, consists in minimizing the production cost in an electricity transmission network, while satisfying the electricity demand, by determining appropriately the powers of the generators at given buses [3, 2, 1]. Structurally, the problem can be reduced to a nonconvex *all quadratic optimization problem* (also called QCQP for *Quadratically Constrained Quadratic optimization Problem*), meaning that both the objective and the constraints of the optimization problem are nonconvex quadratic functions.

This all quadratic representation of an OPF instance can be obtained by the Matlab function `qcqp_opf` [17], which is available in Matpower 6.0 [45] (a Matlab package for simulating power systems). This function has several output formats; one of them is the following problem in $x \in \mathbb{C}^n$:

$$\begin{cases} \inf_x \; x^{\mathsf{H}} C x + c \\ x^{\mathsf{H}} A_k x = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ x^{\mathsf{H}} B_k x \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p], \end{cases} \tag{6.3}$$

where $C \in \mathcal{H}^n$, $c \in \mathbb{R}$, and for $k \geqslant 1$, A_k, $B_k \in \mathcal{H}^n$ or $\mathbb{C}^{n\times n}$, and a_k, $b_k \in \mathbb{R}$ or $\mathbb{C}$ (a complex inequality has to be understood as two real inequalities on the real and imaginary parts).

Problem (6.3) can be relaxed in a standard primal SDO problem of the form (3.1), by rank relaxation. This is achieved by first considering instead the following relaxed problem in $X_0 \in \mathcal{H}^n_+$:

$$\begin{cases} \inf_{X_0} \langle C, X_0\rangle + c \\ \langle A_k^{\mathsf{H}}, X_0\rangle_{\mathbb{C}^{n\times n}} = a_k, \quad \text{for } k \in [1\,{:}\,m] \\ \langle B_k^{\mathsf{H}}, X_0\rangle_{\mathbb{C}^{n\times n}} \leqslant b_k, \quad \text{for } k \in [1\,{:}\,p], \end{cases} \tag{6.4}$$

The objective and the equality constraints are in the proper format. The kth inequality is then transformed into the equality $\langle B_k^{\mathsf{H}}, X_0\rangle_{\mathbb{C}^{n\times n}} + z_k = b_k$, using an unknown slack variable

$z_k \in \mathbb{R}$ or $\mathbb{C}$, whose real and complex parts must be nonnegative. The new unknown X is then a matrix of larger size containing X_0 and the real/imaginary parts of the z_k's on its diagonal, with additional linear constraints zeroing the elements of X that do not correspond to X_0 or the z_k's [9].

Numerical tests have been undertaken with the test-problems named `caseWB2`, `caseWB5`, and `case9`, with respectively 2, 5, and 9 buses (the resulting dimensions n, m_r, and m_c of the associated relaxed problems are given in table 5.1), for which a central point z_μ with parameter $\mu = 1$ has been computed and has been used as a starting point [9].

6.2 Numerical results

The numerical results are gathered in table 6.1 (for the meaning of its contents, see the

	Sdolab 0.4								SeDuMi 1.3		
	(P) in (3.1)		(P') in (4.8)			($\tilde{P}$) in (4.9)			($\tilde{P}$) in (4.9)		
				speed-down			speed-down			speed-down	
Problems	its	time/it	its	/it	tot	its	/it	tot	its	/it	tot
`mmnc-10-0-100-100`	24	0.414	24	0.95	0.95	28	2.29	**2.67**	16	1.85	1.23
`mmnc-20-0-100-100`	25	0.643	25	0.98	0.98	29	2.19	**2.53**	17	1.38	0.94
`mmnc-50-0-100-100`	26	1.433	26	0.99	0.99	31	1.96	**2.34**	17	0.88	0.58
`mmnc-100-0-100-100`	28	2.964	28	1.00	1.00	33	1.85	**2.18**	17	0.68	0.42
`mmnc-200-0-100-100`	31	6.735	31	0.98	0.98	36	1.73	**2.01**	17	0.56	0.31
`mmnc-500-0-100-100`	29	23.010	29	0.99	0.99	34	1.55	**1.81**	17	0.53	0.31
`mmnc-0-10-100-100`	25	0.418	25	1.50	1.50	29	3.29	**3.82**	17	1.99	1.36
`mmnc-0-20-100-100`	26	0.683	26	1.65	1.65	31	3.36	**4.01**	18	1.64	1.13
`mmnc-0-50-100-100`	29	1.584	29	1.80	1.80	34	3.35	**3.93**	16	1.24	0.69
`mmnc-0-100-100-100`	26	3.504	26	1.89	1.89	31	3.41	**4.06**	17	1.09	0.71
`mmnc-0-200-100-100`	27	8.632	27	1.94	1.94	31	3.11	**3.57**	17	1.02	0.64
`mmnc-0-500-100-100`	28	34.669	28	1.93	1.93	33	2.69	**3.17**	16	1.03	0.59
`caseWB2`	12	0.030	12	0.37	0.37	14	0.44	**0.51**	8	1.16	0.77
`caseWB5`	17	1.713	17	1.32	1.32	20	1.46	**1.72**	17	0.12	0.12
`case9`	20	27.968	20	1.35	1.35	23	1.45	**1.66**	19	0.11	0.11

Table 6.1: Computing time per iteration ("time/it", in sec) and number of iterations ("its") for the test-problems described in table 5.1 and for two solvers: Sdolab 0.4 and SeDuMi 1.3. Sdolab is run on problem (P) in complex numbers and complex constraints, on (P') in complex numbers and real constraints, and on $(\tilde{P})$ in real numbers. SeDuMi 1.3 is run on problem $(\tilde{P})$ in real numbers. For each problem, the "speed-down" is the ratio of the computing time of the considered solver/problem-formulation with respect to the one of Sdolab on (P); it is given when comparing the computing time per iteration ("/it") or the total computing time ("tot"), the latter taking into account the difference in the number of iterations. Therefore, the column in bold gives the speed-up that is obtained by running an SDO solver in complex numbers rather than in real numbers; we see that this one can reach the value 4.

caption). They have been realized in Matlab, version 7.14.0.739 (R2012a), on a 2.8 GHz Intel Core I7 (a single core is imposed by a Matlab directive, in order to avoid uncontrolled parallelism), with 16 Gbytes of memory at 1600 MHz and OS X 10.9.5. The stopping test is $\hat{\mu}(z) \leqslant 10^{-9}$ for all the problems in the form (P) or (P') and $\hat{\mu}(z) \leqslant 5\,10^{-10}$ when the problems are in the form $(\tilde{P})$ to take into account the identity (5.16). One can make the

following observations.

Observations 6.1 1) Consider first the impact of the transformation of the constraints with complex right-hand sides of problem (P) in (3.1) into the constraints with real right-hand sides of problem (P') in (4.8). This is reflected by the columns under the labels (P) and (P') in table 6.1.

A first observation is that, for the considered experiments, the number of iterations is not affected by the constraint transformation. As a result, the ratios in the columns "/it" and "tot" are identical.

Another observation is that this transformation does not affect the computing time needed to solve the problems `mmnc-*-0-*-*` (in the first group), since these have only real constraints, which implies that problems (P) and (P') are identical (the speed-down differs sometimes from 1 only because of the random behavior of the computing time). For the other problems, we see that this transformation is clearly not recommended, since it can increase the computing time per iteration significantly: the speed-down is often much larger than 1, ranging approximately in the interval $[1, 2]$. This speed-down can be explained by the increase in the number of constraints that the transformation induces and the fact that there is a double matrix product $G^{\mathsf{H}} A_k G$ to make for each constraint (see (5.26)). For the problems `mmnc-0-`p_2`-*-*`, we have seen in section 6.1.1 that $m_r = 1$ and $m_c = p_2$, so that the number of constraints is $p_2 + 1$ for (P) and $2p_2 + 1$ for (P'). Since these products intervene in a larger and larger part of the computing time when p_2 increases (see the last but one column in table 5.1), the speed-down is closer and closer to $(2p_2+1)/(p_2+1) \simeq 2$ when p_2 increases. This is what can be observed for the problems `mmnc-0-`p_2`-*-*` in the second column below the label (P') in table 6.1. This observation justifies the interest of having a piece of software that is able to deal with non-Hermitian constraints (i.e., with non-Hermitian matrices A_k), since this avoids the need of making the transformation of (P) in (3.1) into (P') in (4.8).

2) Consider now the impact of the transformation of the complex problem (P) in (3.1) into the real number problem $(\tilde{P})$ in (4.9), hence making the additional transformation of (P') in (4.8) into $(\tilde{P})$. This is reflected by the comparison of the columns labeled (P), (P'), and $(\tilde{P})$ under Sdolab 0.4 in table 6.1.

The first observation is that the number of iterations is increased by this transformation into real numbers, which is consistent with the conclusion of the analysis of section 5.2, according to which the stepsizes of the prediction phases are smaller when solving $(\tilde{P})$ than when solving (P). Since the speed of convergence increases when the stepsizes increase, the converges is slowed down on $(\tilde{P})$. Even though the change in the number of iterations is not important, it has a significant effect on the speed-down as this can be seen on the difference between columns "/it" and "tot" under the label $(\tilde{P})$.

A second observation is that the computing time per iteration (column "/it") is significantly increased by this transformation into real numbers. This is due in part by the approximate ratio 2 on the matrix operation cost quoted in section 5.4, which is revealed on the problems `mmnc-0-`p_2`-*-*` for which the second factor that follows does not intervene. This is also due to the increases in the number of double matrix products $G^{\mathsf{H}} A_k G$ mentioned in point 1 above.

The cumulated effect of the three factors identified above (increase in the number of double matrix products $G^{\mathsf{H}} A_k G$, in the number of iterations, and in the matrix operation cost) is summarized by the "tot" column under the label $(\tilde{P})$ in table 6.1, which contains bold

ratios. We see that the benefit of working directly in complex numbers can go up to divide the computing time by 4.

3) The comparison with SeDuMi 1.3 [38, 35], whose performance is revealed by the last three columns in table 6.1, shows that much progress can still be obtained on the proprotype solver Sdolab 0.4. The better performance of SeDuMi can be explained by the following facts: (*i*) SeDuMi works on sparse data, while Sdolab imposes dense data (this makes a big difference, in particular for the Matpower problems), (*ii*) the computing time per iteration is much smaller for SeDuMi than for Sdolab when both solvers work on the same problems ($\tilde{P}$) (but not always when Sdolab works directly on (P), which SeDuMi cannot do), (*iii*) SeDuMi is written in Matlab, but calls many functions written in C, while Sdolab is entirely written in Matlab (except for the use of the Cholesky factorization subroutines `dchdc` and `zchdc` from Linpack, which are written in `Fortran`), and (*iv*) finally the implemented algorithms are different, which may explain why the number of iterations is much less with SeDuMi (but the starting points are not the same, since there is no way to give one to SeDuMi). □

7 Conclusion

This paper should be considered as a plea for the development of an efficient complex-number self-dual conic optimization solver for problems that are naturally written in complex numbers; Sdolab 0.4 is just a toy or a preliminary version developed to certify the usefulness of such a solver. The numerical experiments have demonstrated that such a solver can be 2...4 times faster than a real number solver tackling the problem by first transforming it into real numbers, particularly when there are constraints in complex numbers (i.e., defined by non-Hermitian matrices). Several observations, some based on more abstract considerations, have been highlighted to explain that speed-up, which can be summarized as follows: (*i*) matrix-matrix product are approximately twice more time consuming for the double size matrices that result from a real-number transformation of the complex-number problem, and these products represent the bottleneck of the considered algorithm, (*ii*) the number of matrix-matrix products may be doubled when the affine constraints are expressed with non-Hermitian matrices, and (*iii*) the number of iterations of the considered interior-point algorithm increases when it deals with the real-number representation of the complex-number problem.

References

[1] 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]. 2, 5, 29

[2] 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]. 2, 5, 29

[3] M.J. Carpentier (1962). Contribution à l'étude du dispatching économique. *Bulletin de la Société Française des Électriciens*, 8(3), 431–447. 2, 5, 29

[4] E. de Klerk (2002). *Aspects of Semidefinite Programming - Interior Point Algorithms and Selected Applications*. Kluwer Academic Publishers, Dordrecht. [doi]. 18, 19

[5] C. Ferrier (1998). Hilbert's 17th problem and best dual bounds in quadratic minimization. *Cybernetics and Systems Analysis*, 34, 696–709. [doi]. 2

[6] J.Ch. Gilbert (2016). *Step by step design of an interior point solver in self-dual conic 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]. 18, 28

[7] J.Ch. Gilbert (2017). On the equivalence between R-linear and C-linear systems of equations in complex numbers. Research report (to appear), INRIA-Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France. 25

[8] J.Ch. Gilbert (2017). Convergence of a feasible predictor-corrector interior-point algorithm for the semidefinite optimization problem in complex numbers. Research report (to appear), INRIA-Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France. 24

[9] J.Ch. Gilbert, C. Josz (2017). Plea for a semidefinite optimization solver in complex numbers – The full report. Research report, INRIA-Paris, 2 rue Simone Iff, CS 42112, 75589 Paris Cedex 12, France. 8, 12, 14, 17, 25, 26, 28, 29, 30

[10] M.X. Goemans, D.P. Williamson (2004). Approximation algorithms for MAX-3-CUT and other problems via complex semidefinite programming. *Journal of Computer and System Sciences*, 2, 442–470. [doi]. 1

[11] G.H. Golub, C.F. Van Loan (1996). *Matrix Computations* (third edition). The Johns Hopkins University Press, Baltimore, Maryland. 1, 12

[12] N.J. Higham (2002). *Accuracy and Stability of Numerical Algorithms* (second edition). SIAM Publication, Philadelphia. 1

[13] N.J. Higham (2009). *Cholesky factorization*, volume 1, pages 251–254. Wiley, New York. 28

[14] W. Huang, K. Gallivan, X. Zhang (2016). Solving PhaseLift by low-rank Riemannian optimization methods for complex semidefinite constraints. Technical report. [Optimization Online]. 1

[15] J. Jiang (1998). A long step primal-dual path following method for semidefinite programming. *Operations Research Letters*, 23(1-2), 53–62. [doi]. 19

[16] C. Josz (2016). *Application of Polynomial Optimization to Electricity Transmission Networks*. PhD Thesis, University Paris VI. 2

[17] C. Josz, S. Fliscounakis, J. Maeght, P. Panciatici (2016). AC power flow data in MATPOWER and QCQP format: iTesla, RTE snapshots, and PEGASE. Technical report. arXiv:1603.01533. 2, 29

[18] 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]. 2

[19] C. Josz, D.K. Molzahn (2015). Moment/sum-of-squares hierarchy for complex polynomial optimization. *SIAM Journal on Optimization* (in revision). arXiv:1508.02068. 1, 2, 10, 11

[20] 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. 19

[21] J.B. Lasserre (2001). Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11, 796–817. [doi]. 2

[22] J.B. Lasserre (2001). Convergent LMI relaxations for nonconvex quadratic programs. Technical report. Graduate seminar at MIT, fall 2001. 2

[23] J.B. Lasserre (2015). *An Introduction to Polynomial and Semi-Algebraic Optimization*. Cambridge Texts in Applied Mathematics. Cambridge University Press. 2

[24] J. Lavaei, S.H. Low (2012). Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 27(1), 92–107. [doi]. 2

[25] S.H. Low (2014). Convex relaxation of optimal power flow – Part I: formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1), 15–27. [doi]. 2

[26] S.H. Low (2014). Convex relaxation of optimal power flow – Part II: exactness. *IEEE Transactions on Control of Network Systems*, 1(2), 177–189. [doi]. 2

[27] S. Mehrotra (1992). On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2, 575–601. 18

[28] 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]. 18

[29] D.K. Molzahn, I.A. Hiskens (2013). Moment-based relaxation of the optimal power flow problem. Technical report, University of Michigan. 2

[30] Y. Nesterov (2000). Squared functional systems and optimization problems. In J.B.G. Frenk, C. Roos, T. Terlaky, S. Zhang (editors), *High Performance Optimization*, pages 405–440. Kluwer Academic Publishers. 2

[31] 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]. 19

[32] P.A. Parrilo (2000). *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD Thesis, California Institute of Technology. 2

[33] R.T. Rockafellar (1970). *Convex Analysis*. Princeton Mathematics Ser. 28. Princeton University Press, Princeton, New Jersey. 16

[34] C. Roos, T. Terlaky, J.-Ph. Vial (1997). *Theory and Algorithms for Linear Optimization – An Interior Point Approach*. John Wiley & Sons, Chichester. 18

[35] SeDuMi. Internet site http://sedumi.ie.lehigh.edu. 2, 32

[36] N.Z. Shor (1987). Quadratic optimization problems. *Soviet Journal of Computer and System Sciences*, 25, 1–11. Translated from *Izvestiya Akademii Nauk SSSR Tekhnicheskaya Kibernetika*, 1987, no. 25:1, 128-139. 2

[37] L. Sorber, M. Van Barel, L. De Lathauwer (2012). Unconstrained optimization of real functions in complex variables. *SIAM Journal on Optimization*, 22(3), 879–898. 1

[38] J.F. Sturm (1999). Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11, 625–653. 2, 32

[39] 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]. 2, 10

[40] K.-C. Toh, R.H. Tütüncü, M.J. Todd (2006). On the implementation and usage of SDPT3 - a Matlab software package for semidefinite-quadratic-linear programming, version 4.0. Technical report. User's guide of SDPT3. 2, 29

[41] L. Vandenberghe, S. Boyd (1996). Semidefinite programming. *SIAM Review*, 38, 49–95. [doi]. 2, 29

[42] I. Waldspurger, A. d'Aspremont, S. Mallat (2015). Phase recovery, maxcut and complex semidefinite programming. *Mathematical Programming*, 149(1-2), 47–81. [doi]. 1

[43] J. Watrous (2009). Semidefinite programs for completely bounded norms. *Theory of Computing*, 5(11), 217–238. [doi]. 1

[44] J. Watrous (2013). Simpler semidefinite programs for completely bounded norms. *Chicago Journal of Theoretical Computer Science*. Article 8. [doi]. 1

[45] 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]. 29