

Repeating the numerical results of the submitted paper [?]

J. Ch. GILBERT[†]

July 26, 2023

This short note explains how the numerical results of the paper [?], submitted for possible publication in *Mathematical Programming Computation* (MPC), can be obtained with **Sdolab 0.4** from the data and solvers uploaded on the MPC site at the beginning of the refereeing process.

Table of contents

1	Introduction	1
1.1	Notation	1
1.2	Three versions of the same problem	2
2	Description of the uploaded files	3
2.1	The subdirectory <code>sdolab_0_4</code>	3
2.2	The subdirectory <code>test</code>	4
3	Getting the results	5
3.1	Select the test-case	5
3.2	Solve the test-cases	6
3.3	Results on the MPC computer	6
4	Brief description of Sdolab 0.4	10
	References	10

1 Introduction

The goal of the submitted paper [?] is *not* to present a new piece of software, namely **Sdolab 0.4**, but to highlight the interest of using a solver *in complex numbers* to solve a *complex number* semidefinite optimization (SDO) problem. When the algorithm is 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 a direct linear system solver is used, the speed-up is between 2 and 4, depending on the features of the original problem. **Sdolab 0.4** has been written for showing that this claim, motivated on several observations made in the submitted paper [?], is actually encountered in practice. The solver **Sdolab** has not been written for being competitive with other well established solvers developed for many years, although it solves a complex SDO problem sometimes faster than **SeDuMi 1.3**, which works on the real number equivalent problem. Some of the improvements that could help **Sdolab 0.4** be more efficient are listed in observation 6.1(3) in [?], but again, this is not the goal of the paper.

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

1.1 Notation

Before going further, let us recall, in a rather indigestible manner, part of the notation of the submitted paper [?], used below:

- $i := \sqrt{-1} \in \mathbb{C}$: the *pure imaginary number*;
- \mathbb{C}^m : \mathbb{C} -vector space (i.e., with scalars in \mathbb{C}) formed of the m -tuples of complex numbers (sometimes considered as $m \times 1$ matrices);
- $\mathbb{C}_{\mathbb{R}}^m$: \mathbb{R} -vector space (i.e., with scalars in \mathbb{R}) formed of the m -tuples of complex numbers;
- $\mathbb{C}^{m \times n}$: \mathbb{C} -vector space of the $m \times n$ complex matrices, equipped with the Hermitian scalar product

$$\langle \cdot, \cdot \rangle_{\mathbb{C}^{m \times n}} : (A, B) \in \mathbb{C}^{m \times n} \times \mathbb{C}^{m \times n} \mapsto \langle A, B \rangle_{\mathbb{C}^{m \times n}} := \text{tr } A^H B \in \mathbb{C}, \quad (1.1)$$

where $\text{tr } M = \sum_i M_{ii}$ is the trace of a square matrix M and A^H is the conjugate transpose of a matrix A ;

- $\mathbb{C}_{\mathbb{R}}^{m \times n}$: \mathbb{R} -vector space of $m \times n$ complex matrices, equipped with the scalar product

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

- \mathcal{H}^n : \mathbb{R} -vector space of complex *Hermitian* matrices of order n (i.e., those $A \in \mathbb{C}^{n \times n}$ such that $A^H = A$), equipped with the scalar product

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

recall that this is not a \mathbb{C} -vector space, since the identity matrix I is Hermitian, but not iI ;

- $i\mathcal{H}^n$: set of *skew-Hermitian* matrices (i.e., those $A \in \mathbb{C}^{n \times n}$ such that $A^H = -A$ or, equivalently, those $A \in \mathbb{C}^{n \times n}$ such that $A = iB$ for some $B \in \mathcal{H}^n$);
- \mathcal{S}^n : \mathbb{R} -vector space of real symmetric matrices of order n , equipped with the scalar product

$$\langle \cdot, \cdot \rangle_{\mathcal{S}^n} : (A, B) \in \mathcal{S}^n \times \mathcal{S}^n \mapsto \langle A, B \rangle_{\mathcal{S}^n} := \text{tr } AB \in \mathbb{R};$$

- $X \geq 0$ means that X is (or must be) positive semidefinite,

1.2 Three versions of the same problem

The above mentioned interior-point algorithm is tested on three versions of the complex SDO problem, which are linked to each other by some properties, namely problems (P) , (P') , and (\tilde{P}) , specified below. Actually, (P) and (P') are identical (same optimal values, same feasible sets, and same solutions), while (\tilde{P}) is a transformation of (P) into real numbers. Let us see this in detail.

From (3.1) and (3.5) in [?], the complex SDO problem (P) can be written

$$(P) \quad \begin{cases} \inf_{X \in \mathcal{H}^n} \langle C, X \rangle_{\mathcal{H}^n} \\ \langle A_k, X \rangle_{\mathcal{H}^n} = b_k, & \text{for } k \in K_r \\ \langle A_k, X \rangle_{\mathbb{C}^{n \times n}} = b_k, & \text{for } k \in K_c \\ X \geq 0, \end{cases} \quad (1.2)$$

where $C \in \mathcal{H}^n$, $A_k \in \mathcal{H}^n$ and $b_k \in \mathbb{R}$ for $k \in K_r := [1:m_r]$, and $A_k \in \mathbb{C}^{n \times n}$ and $b_k \in \mathbb{C}$ for $k \in K_c := [m_r+1:m_r+m_c]$. Since $\langle C, X \rangle_{\mathcal{H}^n}$ is a real number, the problem has a single objective.

The problem (P') is obtained by transforming the *complex* or *non Hermitian* constraints of (P) , i.e., those with indices in K_c , into *real* or *Hermitian* constraints. Using the operators $\mathcal{H} : \mathbb{C}^{n \times n} \rightarrow \mathcal{H}^n$ and $\mathcal{Z} : \mathbb{C}^{n \times n} \rightarrow i\mathcal{H}^n$ defined at $A \in \mathbb{C}^{n \times n}$ by

$$\mathcal{H}(A) := \frac{1}{2}(A + A^H) \quad \text{and} \quad \mathcal{Z}(A) := \frac{1}{2}(A - A^H),$$

one gets

$$(P') \quad \begin{cases} \inf \langle C, X \rangle_{\mathcal{H}^n} \\ \langle A_k, X \rangle_{\mathcal{H}^n} = b_k, \quad \forall k \in K_r \\ \langle \mathcal{H}(A_k), X \rangle_{\mathcal{H}^n} = \Re(b_k), \quad \forall k \in K_c \\ \langle i\mathcal{Z}(A_k), X \rangle_{\mathcal{H}^n} = \Im(b_k), \quad \forall k \in K_c \\ X \succeq 0. \end{cases} \quad (1.3)$$

This problem is clearly equivalent to (P) , in the sense that it has the same solution set and the same optimal value.

Problem (P') is still expressed in complex numbers, since C , A_k , $\mathcal{H}(A_k)$, $i\mathcal{Z}(A_k)$, and X are complex Hermitian matrices. To transform it into real numbers, [?] uses the map $\mathcal{J}_{\mathcal{H}^n} : \mathcal{H}^n \rightarrow \mathcal{S}^{2n}$, defined at $M \in \mathcal{H}^n$ by

$$\mathcal{J}_{\mathcal{H}^n}(M) = \frac{1}{\sqrt{2}} \begin{pmatrix} \Re(M) & -\Im(M) \\ \Im(M) & \Re(M) \end{pmatrix}.$$

The matrix $\mathcal{J}_{\mathcal{H}^n}(M)$ is symmetric since, when $M \in \mathcal{H}^n$, $\Re(M)$ is symmetric and $\Im(M)$ is skew-symmetric. One can show that $\mathcal{J}_{\mathcal{H}^n}$ is a vector space isometry since

$$\forall M, N \in \mathcal{H}^n : \quad \langle \mathcal{J}_{\mathcal{H}^n}(M), \mathcal{J}_{\mathcal{H}^n}(N) \rangle_{\mathcal{S}^{2n}} = \langle M, N \rangle_{\mathcal{H}^n}. \quad (1.4)$$

Therefore, it looks natural to take as real number version of problem (P) or (P') , the following problem

$$(\tilde{P}) \quad \begin{cases} \inf_{\tilde{X} \in \mathcal{S}^{2n}} \langle \mathcal{J}_{\mathcal{H}^n}(C), \tilde{X} \rangle_{\mathcal{S}^{2n}} \\ \langle \mathcal{J}_{\mathcal{H}^n}(A_k), \tilde{X} \rangle_{\mathcal{S}^{2n}} = b_k, \quad \forall k \in K_r \\ \langle \mathcal{J}_{\mathcal{H}^n}(\mathcal{H}(A_k)), \tilde{X} \rangle_{\mathcal{S}^{2n}} = \Re(b_k), \quad \forall k \in K_c \\ \langle \mathcal{J}_{\mathcal{H}^n}(i\mathcal{Z}(A_k)), \tilde{X} \rangle_{\mathcal{S}^{2n}} = \Im(b_k), \quad \forall k \in K_c \\ \tilde{X} \succeq 0, \end{cases} \quad (1.5)$$

where \tilde{X} plays the role of $\mathcal{J}_{\mathcal{H}^n}(X)$. Note however that there may be solutions to (\tilde{P}) that are not in the range space $\mathcal{R}(\mathcal{J}_{\mathcal{H}^n})$. Various properties linking (P) and (\tilde{P}) are presented in section 4.2 of the submitted paper [?].

2 Description of the uploaded files

The uploaded files are in a directory, which contains the following subdirectories

```
README.txt
gilbert-josz-2017-03-28.pdf
gilbert-josz-ref-2017-03-28.pdf
sdolab_0_4/
sedumi_1_3/
test/
testing_notes.pdf
```

The contents of the file `README.txt` invites to read `testing_notes.pdf`, which is the present notes. The file `gilbert-josz-2017-03-28.pdf` is the submitted paper [?] and the file `gilbert-josz-ref-2017-03-28.pdf` is a version of the submitted paper with additional proofs and notes to help the refereeing. The subdirectory `sdolab_0_4` contains `Sdolab 0.4`. The subdirectory `sedumi_1_3` contains the well known SDO solver `SeDuMi 1.3`, which is documented in [?, ?], and is used in [?] to compare its results with those of `Sdolab 0.4`. The subdirectory `test` is where to go to run a test-problem or to get the results of the submitted paper [?].

2.1 The subdirectory `sdolab_0_4`

The uploaded files are in a directory named `sdolab_0_4`. It contains the following files

```
QPLicense.txt
README.txt
dchdc.F
dchdc.mexa64
makemex.m
sdolab.m
sdolab_cls.m
sdolab_cpdist.m
sdolab_gw.m
sdolab_nt.m
sdolab_prelim.m
sdolab_solve.m
sdolab_solve_infeas_bigm1.m
sdolab_solve_infeas_bigm1m2.m
sdolab_values.m
sdolab_x0.m
zchdc.F
zchdc.mexa64
```

The files `sdolab_*` are those containing the Matlab code `Sdolab 0.4`. The files `dchdc.F` and `zchdc.F` contain Linpack and BLAS subroutines for the Cholesky factorization with pivoting of dense matrices, as well as `Matlab-Fortran` gateways. The object files `dchdc.mexa64` and `zchdc.mexa64` have been obtained from `dchdc.F` and `zchdc.F` by entering the command `makemex` of this directory in a `Matlab` window.

2.2 The subdirectory `test`

This is where to go to run a test-problem for `Sdolab 0.4`. The directory contains the following files

```
README.txt
main.m
matpower_case9_qcqp_c.txt
matpower_casewb2_qcqp_c.txt
matpower_casewb5_qcqp_c.txt
matpower_qcqp_c.m
mmnc.m
res-matpower-case9.txt
res-matpower-casewb2.txt
res-matpower-casewb5.txt
```

```

res-mmnc-0-10-100-100.txt
res-mmnc-0-100-100-100.txt
res-mmnc-0-20-100-100.txt
res-mmnc-0-200-100-100.txt
res-mmnc-0-50-100-100.txt
res-mmnc-0-500-100-100.txt
res-mmnc-10-0-100-100.txt
res-mmnc-100-0-100-100.txt
res-mmnc-20-0-100-100.txt
res-mmnc-200-0-100-100.txt
res-mmnc-50-0-100-100.txt
res-mmnc-500-0-100-100.txt

```

The file `README.txt` briefly explains how to run a test-problem. We give here more details on the contents of the directory. How to run a test-problem is described in section 3.

The main program to launch in **Matlab** to run a test-problem is

```
main.m
```

This script can solve the minimum matrix norm problem in complex number (see section 6.1.1 in the submitted paper [?]), which is coded in

```
mmnc.m
```

The script `main.m` can also solve a **Matpower** [?, ?] test-problem (see section 6.1.2 in the submitted paper [?]), which is coded in

```
matpower_qcqp.c.m
```

The files `matpower*_qcqp.c.txt` contains data of the 3 smallest **Matpower** test-problems `caseWB2`, `caseWB5`, and `case9`.

The script `main.m` solves the three versions (P) , (P') and (\tilde{P}) of the specified test-problem by **Sdolab** 0.4 and its form (\tilde{P}) by **SeDuMi** 1.3. The output is written in a file named `res` (this can be changed). The output `res` of `main`, running on the MPC computer, for each of the test-problems considered in the submitted paper [?] have been stored in the files named `res-*.txt`. More specifically, the files `res-mmnc-p1-p2-q-r.txt` correspond to the minimum matrix norm problem in complex number with parameters $p_1 = p_1$, $p_2 = p_2$, $q = q$, and $r = r$ and the files `res-matpower-*.txt` give the results on the **Matpower** test-problems.

3 Getting the results

It is now supposed that you are in the directory `test`. To get the results corresponding to a particular test-problem, you have to comment/uncomment lines in the file `main.m` (section 3.1), run `main` in a **Matlab** window (section 3.2) and retrieve the “results” from the generated output file `res` (section 3.3).

3.1 Select the test-case

To get the results corresponding to a particular test-problem, you first have to select the test-problem. This is done by commenting/uncommenting lines with numbers in [51:67] in the file `main.m`. For example, to solve the minimum matrix norm problem in complex number with parameter $p_1 = 10$, $p_2 = 0$, and $q = r = 100$, the lines numbered [51:67] must look as follows

```

49 % Test problems for the paper "Plea..." by Jean Charles Gilbert and Cédric Jozs
50 %
51 [A,b,c,x0,y0] = mmnc(10,0,100,100,fout);
52 [A,b,c,x0,y0] = mmnc(20,0,100,100,fout);
53 [A,b,c,x0,y0] = mmnc(50,0,100,100,fout);
54 [A,b,c,x0,y0] = mmnc(100,0,100,100,fout);
55 [A,b,c,x0,y0] = mmnc(200,0,100,100,fout);
56 [A,b,c,x0,y0] = mmnc(500,0,100,100,fout);
57 %
58 [A,b,c,x0,y0] = mmnc(0,10,100,100,fout);
59 [A,b,c,x0,y0] = mmnc(0,20,100,100,fout);
60 [A,b,c,x0,y0] = mmnc(0,50,100,100,fout);
61 [A,b,c,x0,y0] = mmnc(0,100,100,100,fout);
62 [A,b,c,x0,y0] = mmnc(0,200,100,100,fout);
63 [A,b,c,x0,y0] = mmnc(0,500,100,100,fout);
64 %
65 [A,b,c,x0,y0] = matpower_qcqp('caseWB2',fout); options.max_stepsize = 0.99;
66 [A,b,c,x0,y0] = matpower_qcqp('caseWB5',fout);
67 [A,b,c,x0,y0] = matpower_qcqp('case9',fout); options.eps = 1.e-8; options.max_stepsize = 0.95;

```

A possible other change is the name of the file on which the results are written. With

```

9 % Choose the output channel
10
11 fout = fopen('res','w');      % on the file named 'res'
12 % fout = 1;                  % on the screen

```

the results are written in the file `res`. With

```

9 % Choose the output channel
10
11 % fout = fopen('res','w');      % on the file named 'res'
12 fout = 1;                      % on the screen

```

the results are written on the screen.

Comments are given in `main.m` to explain what this script does.

Save the file `main.m` if modifications have been brought.

3.2 Solve the test-cases

To run Matlab on the MPC machine and getting the Matlab prompt ">>" enter (and wait a long time, possible entering some ^C; below, % is the Unix prompt)

```

% cd test
% /usr/local/MATLAB/R2014b/bin/matlab -nodesktop
>>

```

Then enter

```
>> main
```

By default, the results are written on the file `res`. For some problems, this is a long process. To follow what is written in the file `res`, you may want to enter the Unix command

```
% tail -1000f res
```

in another window.

3.3 Results on the MPC computer

If `main` is run in Matlab, as is, the output file `res` should be close to the file

```
res-mmnc-10-0-100-100.txt
```

Let us examine its contents and how to retrieve the numbers given in the tables 3.1 and 3.2.

As already said, `main` solves in sequence the problems (P) , (P') , and (\tilde{P}) by `Sdolab 0.4` and next (\tilde{P}) by `SeDuMi 1.3`. The lines 10 and 13 below

```
10 *** Problem in complex numbers *****
11
12 -----
13 SD0lab solver (version 0.4, March 2017)
14 ~~~~~
15 Dimensions of the problem
16 . Hermitian matrix size (n)          200
17 . nb of real constraints (mr)        11
18 . nb of complex constraints (mc)     0
```

inform that the problem (P) in complex numbers is considered and that this one is solved by `Sdolab 0.4`. The lines 16-18 give the dimensions of problem (P) , which are placed in the columns 2-4 in table 3.1. At the end of the run one finds

```
59 Solution found up to the required precision (1e-09)
..
66 nb of iterations          24
```

which informs that the problem has been solved up to the required precision in 24 iterations of the interior-point algorithm. Further one finds

```
68 Timing
69 . preliminaries          0.000000 sec ( 0.0%)
70 . scaling matrix W      1.428142 sec (11.6%)
71 . LS make               6.989424 sec (57.0%)
72 . LS solve              1.090746 sec ( 8.9%)
73 . total/iter            0.510925 sec
74 . total                 12.262199 sec
```

which provides the time spent in various part of the solver. One finds the time per iteration at line 73 (column 5 in table 3.1), the percentage of the time spent in computing the weighting matrix W (formula (5.3) in [?]) at line 70 (column 6 in table 3.1), the percentage of the time spent in forming the mixed real-complex linear system defining the NT direction (formula (5.25) in [?]) at line 71 (column 7 in table 3.1), and the percentage of the time spent in solving this linear system at line 72 (column 8 in table 3.1).

Next comes the solution of (P') by `Sdolab 0.4`, which is signaled by the lines

```
60294 *** Problem in complex numbers and Hermitian constraints *****
.....
60297 SD0lab solver (version 0.4, March 2017)
```

The information from that run that is reported in table 3.2 is contained in the lines

```
60350 nb of iterations          24
.....
60361 Ratio CPU time per iteration wrt sdolab in complex numbers = 9.68663e-01
60362 Ratio CPU time wrt sdolab in complex numbers              = 9.68663e-01
```

One finds the number of iterations of the run at line 60350 (column 4 in table 3.2), the ratio of the time spent per iteration by Sdolab 0.4 on (P') (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60361 (column 5 in table 3.2), and the ratio of the total time spent by Sdolab 0.4 on (P') (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60362 (column 6 in table 3.2). As quoted in observation 6.1(1) in the submitted paper [?], they are identical for the problems mmnc-p1-0-q-r because there is no complex (non-Hermitian) constraints ($p_2 = 0$).

Next comes the solution of (\tilde{P}) by Sdolab 0.4, which is signaled by the lines

```
60369 *** Problem in real numbers *****
.....
60372 SDolab solver (version 0.4, March 2017)
```

The information from that run that is reported in table 3.2 is contained in the lines

```
60429 nb of iterations          28
.....
60440 Ratio CPU time per iteration wrt sdolab in complex numbers    = 2.35401e+00
60441 Ratio CPU time wrt sdolab in complex numbers                  = 2.74635e+00
```

The meaning is similar to the one above: one finds the number of iterations of the run at line 60429 (column 7 in table 3.2), the ratio of the time spent per iteration by Sdolab 0.4 on (\tilde{P}) (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60440 (column 8 in table 3.2), and the ratio of the total time spent by Sdolab 0.4 on (\tilde{P}) (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60441 (column 9 in table 3.2). The last two numbers are now different because of the change in the number of iterations.

Finally, comes the solution of (\tilde{P}) by SeDuMi 1.3, which is signaled by the lines

```
60448 *** Problem in real numbers *****
60449
60450 SeDuMi 1.3 by AdvOL, 2005-2008 and Jos F. Sturm, 1998-2003.
```

The information from that run that is reported in table 3.2 is contained in the lines

```
60473 iter seconds digits          c*x          b*y
60474 16      17.3   9.8 -8.2335318428e+00 -8.2335318442e+00
.....
60487 Ratio CPU time per iteration wrt sdolab in complex numbers    = 2.14169e+00
60488 Ratio CPU time wrt sdolab in complex numbers                  = 1.42779e+00
```

The meaning is similar to the one above: one finds the number of iterations of the run, which is the first number at line 60474 (column 10 in table 3.2), the ratio of the time spent per iteration by SeDuMi 1.3 on (\tilde{P}) (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60487 (column 11 in table 3.2), and the ratio of the total time spent by SeDuMi 1.3 on (\tilde{P}) (the latter run) with respect to the one of Sdolab 0.4 on (P) at line 60488 (column 12 in table 3.2).

Tables 3.1 and table 3.2 below are the analogues of table 5.1 and 6.1 of the submitted paper [?], but now obtained on the MPC computer instead of on our computer. One can make the following remarks, insisting on the differences between corresponding tables.

1. The number of iterations required by any of the tested methods on any of the test-problems has not been affected by the change of computer, with one exception: problem case9 was not solvable in its versions (P') and (\tilde{P}) by Sdolab 0.4 on the MPC computer with the same precision as on our computer (failure of a Cholesky factorization close to the solution, which is essentially due to rounding errors), so that the required precision on the *duality*

Problems	dimensions			time/iter (sec)	NT direction		
					scaling	equation (5.25) in [?]	
	n	m_r	m_c		W	forming	solving
mmnc-10-0-100-100	200	11	0	0.511	11.6 %	57.0 %	8.9 %
mmnc-20-0-100-100	200	21	0	0.776	7.6 %	71.2 %	6.7 %
mmnc-50-0-100-100	200	51	0	1.695	3.5 %	83.4 %	6.4 %
mmnc-100-0-100-100	200	101	0	3.384	1.7 %	89.2 %	5.7 %
mmnc-200-0-100-100	200	201	0	7.539	0.8 %	92.9 %	4.8 %
mmnc-500-0-100-100	200	501	0	26.049	0.2 %	95.9 %	3.4 %
mmnc-0-10-100-100	200	1	10	0.508	11.6 %	58.7 %	6.4 %
mmnc-0-20-100-100	200	1	20	0.814	7.2 %	73.6 %	4.6 %
mmnc-0-50-100-100	200	1	50	1.844	3.2 %	87.5 %	3.0 %
mmnc-0-100-100-100	200	1	100	3.995	1.5 %	93.4 %	2.1 %
mmnc-0-200-100-100	200	1	200	9.831	0.6 %	95.5 %	1.6 %
mmnc-0-500-100-100	200	1	500	39.500	0.1 %	97.5 %	2.1 %
caseWB2	12	6	68	0.039	3.5 %	28.2 %	13.5 %
caseWB5	35	22	592	2.012	0.1 %	65.1 %	33.8 %
case9	57	36	1572	33.388	0.0 %	65.3 %	34.4 %

Table 3.1: Dimensions (n , m_r , m_c) of the test-problems in the form (P) in (1.2) 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 in [?], and problems **case*** denote the **Matpower** problems, described in section 6.1.2 in [?]. The first group of minimum matrix norm problems is made of problems having no complex constraints, hence being not sensitive to the transformation $(P)/(1.2) \rightsquigarrow (P')/(1.3)$.

		Sdolab 0.4							SeDuMi 1.3		
		(P) in (1.2)		(P') in (1.3)		(\tilde{P}) in (1.5)			(\tilde{P}) in (1.5)		
Problems	its	time/it	its	speed-down /it	tot	its	speed-down /it	tot	its	speed-down /it	tot
mmnc-10-0-100-100	24	0.511	24	0.97	0.97	28	2.35	2.75	16	2.14	1.43
mmnc-20-0-100-100	25	0.776	25	0.99	0.99	29	2.20	2.56	17	1.68	1.14
mmnc-50-0-100-100	26	1.695	26	0.95	0.95	31	2.00	2.38	17	1.16	0.76
mmnc-100-0-100-100	28	3.384	28	1.00	1.00	33	1.87	2.20	17	0.97	0.59
mmnc-200-0-100-100	31	7.539	31	1.00	1.00	36	1.73	2.00	17	0.83	0.45
mmnc-500-0-100-100	29	26.049	29	1.00	1.00	34	1.50	1.76	17	0.76	0.45
mmnc-0-10-100-100	25	0.508	25	1.48	1.48	29	3.30	3.82	17	2.47	1.68
mmnc-0-20-100-100	26	0.814	26	1.76	1.76	31	3.65	4.36	18	2.26	1.54
mmnc-0-50-100-100	29	1.844	29	1.81	1.81	34	3.47	4.07	16	1.79	0.99
mmnc-0-100-100-100	26	3.995	26	1.89	1.89	31	3.32	3.95	17	1.59	1.04
mmnc-0-200-100-100	27	9.831	27	1.91	1.91	31	3.05	3.50	17	1.45	0.91
mmnc-0-500-100-100	28	39.500	28	1.95	1.95	33	2.63	3.10	16	1.34	0.77
caseWB2	12	0.039	12	0.53	0.53	14	0.62	0.73	8	1.22	0.81
caseWB5	17	2.012	17	1.38	1.38	20	1.53	1.81	17	0.17	0.17
case9	19	33.388	19	1.36	1.36	22	1.49	1.65	18	0.17	0.16

Table 3.2: Computing time per iteration (“time/it”, in sec) and number of iterations (“its”) for the test-problems described in table 3.1 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.

$gap \hat{\mu}(z)$ was changed from $1.e-9$ to $1.e-8$ (hence from $5.e-10$ to $5.e-9$ for the problem versions (P') and (\tilde{P})). This explains why there is one less iteration to reach the tolerance on problem **case9** in table 3.2 (19 iterations) than in table 6.1 in [?] (20 iterations).

2. The MPC computer is slightly slower than the one used to get the results in the submitted paper [?], 18 % slower, on average, but the ratios of the CPU time spent in the selected three parts of the solver are similar (last three columns of table 3.1).
3. **SeDuMi 1.3** is slightly slower with respect to **Sdolab 0.4** on the MPC computer (last but one column of table 3.2), 35 % on average, but we have no explanation for this observation.

We have not quoted any other differences. It is important to mention that the observations 6.1 in [?] need not be revised by the change of computer.

4 Brief description of Sdolab 0.4

As mentioned in section 2.1, Sdolab 0.4 is implemented in the following files

```
dchdc.F
sdolab.m
sdolab_cls.m
sdolab_cpdist.m
sdolab_gw.m
sdolab_nt.m
sdolab_prelim.m
sdolab_solve.m
sdolab_solve_infeas_bigm1.m
sdolab_solve_infeas_bigm2.m
sdolab_values.m
sdolab_x0.m
zchdc.F
```

that can be found in the subdirectory `sdolab_0_4`. The entry point is `sdolab.m`. This procedure analyzes the problem data given on entry in `sdolab_prelim.m`, which prints an introductory message (provided `options.verb` is positive on entry). Next a solver is chosen, depending of the feasibility of the initial point (either given on entry of `sdolab` or computed by **Sdolab** in `sdolab_x0.m`). In the experiments related in the submitted paper [?], the given initial point is always feasible and close to the central path (and *on* the central path for the **Matpower** problems), so that the solution is computed by `sdolab_solve`.

The procedure `sdolab_solve` implements a feasible primal-dual interior-point algorithm, using the Nesterov-Todd (NT) direction. The weighting matrix W (formula (5.3) in [?]) is computed in `sdolab_gw.m` and the NT direction in `sdolab_nt.m`. The latter function uses `sdolab_cls.m` to compute the solution to the mixed real-complex linear system given by formula (5.25) in [?]. A unit stepsize is taken in the correction phases (since the iterates are sufficiently close to the central path) and a predetermined stepsize is computed in the prediction phases (with formula (5.8) in [?]). The proximity of the central path is computed in `sdolab_cpdist.m`. The procedures `dchdc.F` and `zchdc.F` are used to compute the Cholesky factorizations with pivoting for dense matrices (these contain and are based on the **Linpack** Fortran subroutines of the same name).

References

- [1] J.Ch. Gilbert, C. Jozs (2017). Plea for a semidefinite optimization solver in complex numbers. Submitted to *Mathematical Programming Computation*.
- [2] SEDUMI. Internet site <http://sedumi.ie.lehigh.edu>.
- [3] J.F. Sturm (1999). Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11, 625–653.
- [4] R.D. Zimmerman, C.E. Murillo-Sanchez (2016). MATPOWER 6.0 – User’s Manual. Technical report.
- [5] 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. [\[doi\]](#).