Implementation of a block-decomposition algorithm for solving large-scale conic semidefinite programming problems

Renato D. C. Monteiro* Camilo Ortiz† Benar F. Svaiter‡

May 12, 2011 (Revised: October 8, 2012 and July 11, 2013)

Abstract

In this paper, we consider block-decomposition first-order methods for solving large-scale conic semidefinite programming problems given in standard form. Several ingredients are introduced to speed-up the method in its pure form such as: an aggressive choice of stepsize for performing the extragradient step; use of scaled inner products; dynamic update of the scaled inner product for properly balancing the primal and dual relative residuals; and proper choices of the initial primal and dual iterates, as well as the initial parameter for the scaled inner product. Finally, we present computational results showing that our method outperforms the two most competitive codes for large-scale conic semidefinite programs, namely: the boundary-point method introduced by Povh et al. and the Newton-CG augmented Lagrangian method by Zhao et al.

1 Introduction

Let $\mathcal{X}$ and $\mathcal{Y}$ be finite dimensional inner product spaces, with inner products and associated norms denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$, respectively. The conic programming problem is

$$\min\{\langle c,x\rangle \ : \ \mathcal{A}x=b,\ x\in K\}, \tag{1}$$

where $\mathcal{A}:\mathcal{X}\to\mathcal{Y}$ is a linear map, $b\in\mathcal{Y}$, $c\in\mathcal{X}$ and $K\subset\mathcal{X}$ is a closed convex cone. The corresponding dual problem is

$$\max\{\langle b,y\rangle \ : \ c-\mathcal{A}^*y\in K^*\}, \tag{2}$$

where $\mathcal{A}^*$ denotes the adjoint of $\mathcal{A}$ and K^* is the dual cone of K defined as

$$K^*:=\{v\in\mathcal{X}:\langle x,v\rangle\geq 0,\ \forall x\in K\}. \tag{3}$$

Let $\mathbb{R}^n$ denote the n dimensional Euclidean space and $\mathbb{R}^n_+$ denote the cone of nonnegative vectors in $\mathbb{R}^n$. Also, let $\mathcal{S}^n$ denote the linear space of all $n\times n$ symmetric matrices and $\mathcal{S}^n_+$ denote the cone of $n\times n$ symmetric positive semidefinite matrices. In this paper, we report our computational experience with a first-order block-decomposition (BD) method for solving large-scale conic semidefinite programming problems (1), where

$$\mathcal{X}=\mathbb{R}^{n_u+n_l}\times\mathcal{S}^{n_s},\qquad \mathcal{Y}=\mathbb{R}^m,\qquad K=\mathbb{R}^{n_u}\times\mathbb{R}^{n_l}_+\times\mathcal{S}^{n_s}_+, \tag{4}$$

and the inner products in $\mathcal{X}$ and $\mathcal{Y}$ are the standard Euclidean/Frobenius inner products. Iteration-complexity bounds for this method have been studied in [10] (see also [5]). In particular, paper [10] derives the iteration-complexity of this method by using the fact that it is a special case of the hybrid proximal

*School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0205 (email: monteiro@isye.gatech.edu). The work of this author was partially supported by NSF Grants CCF-0808863, CMMI-0900094 and CMMI-1300221, and ONR Grant ONR N00014-11-1-0062.

†School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0205 (email: camiort@gatech.edu).

‡IMPA, Estrada Dona Castorina 110, 22460-320 Rio de Janeiro, Brazil (email:benar@impa.br). The work of this author was partially supported by CNPq grants no. 474944/2010-7, 303583/2008-8 and FAPERJ grant E-26/110.821/2008.

extragradient (HPE) method (referred to here as the HPE framework) introduced in [17, 18] by Solodov and Svaiter, and whose complexity is derived in [13, 12]. Moreover, as will be seen later on, the use of the HPE framework to analyze BD methods results in some crucial ideas towards improving their practical performance.

Though the BD methods described in [5, 10] are simple and have nice convergence properties, their implementation in its pure form is far from being efficient. This paper introduces several ingredients to the BD method of [10] to obtain a highly efficient algorithm for solving (1). The first ingredient is the use of an aggressive choice of stepsize based on a certain error criterion for performing the extragradient step. The second important idea is the implementation of the method with $\mathcal{Y}$ endowed with a scaled inner product. The third idea is to allow the scaled inner product in the $\mathcal{Y}$ space to dynamically change as the algorithm progresses, with the aim of properly balancing the sizes of the primal and dual relative residuals so as to make them go to zero according to the same order of magnitude. The fourth idea is proper choices of the initial primal and dual iterates, as well as the initial parameter for the scaled inner product.

Recently, augmented Lagrangian approaches have been proposed to solve the dual formulation (2) with $\mathcal{X}$, $\mathcal{Y}$ and K as in (4) for the case when m, n_u and n_l are large (up to a few millions) and n_s is moderate (up to a few thousands). In [9, 15], a boundary-point method for solving (1) is proposed which can be viewed as variants of the alternating direction method of multipliers introduced in [7, 8] applied to (2). In [22], an inexact augmented Lagrangian method is proposed which solves a reformulation of the augmented Lagrangian subproblem involving only the y variable via a semismooth Newton approach combined with the conjugate gradient method. Moreover, [9, 15] and [22] report numerical results indicating that their methods are currently the best alternatives for solving large-scale conic programming problems of the form (1) and (4). In this paper, we present computational results showing that our method is faster and more robust than the ones in [9, 15] and [22] in a larger percentage of conic programming instances.

It should be noted that another highly efficient variant of the BP method, which performs a more aggressive Lagrange multiplier update, has been studied and implemented by Wen et al. in [21]. The resulting package, namely SDPAD, contains in fact a number of codes designed to solve different classes of graph-related SDP relaxations. Moreover, each code is written in such a way as to exploit the special structure of each SDP class, without requiring the input to be given in standard form. Since SDPAD is not a general-purpose package (in the sense that it accepts any standard form conic SDP as input) such as the ones mentioned in the previous paragraph, we have not included it in our present computational experiments. However, in a follow-up paper [11], we have compared SDPAD with a specialized version of our BD method for solving different classes of graph-related SDP relaxations, and have found that the latter one outperforms the first one in all problem classes.

This paper is organized as follows. Section 2 presents an adaptive block-decomposition HPE framework in the context of a block-structured monotone inclusion problem. This framework is similar to the one presented in [10], but makes an aggressive choice of extragradient stepsize. An instance of this framework for solving the conic programming problem (1) in which the $\mathcal{Y}$ space is endowed with a scaled inner product is described in Section 3. Section 4 describes in detail all the ingredients needed to speed-up the pure form of the adaptive block-decomposition HPE method, and presents numerical results demonstrating the efficiency of the resulting algorithm for solving many large instances of (1) and (4).

1.1 Notation

The norm of a linear operator $\mathcal{A} : \mathcal{X} \to \mathcal{Y}$ is defined as

$$\|\mathcal{A}\| := \sup_{\|x\| \le 1} \|\mathcal{A}x\|.$$

The norm of the pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is defined as

$$\|(x, y)\| := \sqrt{\|x\|^2 + \|y\|^2}.$$

Let a closed convex set $C \subset \mathcal{X}$ be given. The projection operator $\Pi_C : \mathcal{X} \to C$ onto C and the distance function $\mathrm{dist}_C : \mathcal{X} \to \mathbb{R}_+$ with respect to C are defined as

$$\Pi_C(x) := \arg\min_{\tilde{x} \in C} \{\|x - \tilde{x}\|\}, \qquad \mathrm{dist}_C(x) := \min_{\tilde{x} \in C} \{\|x - \tilde{x}\|\}, \qquad \forall x \in \mathcal{X}. \tag{5}$$

Finally, the indicator function $\delta_C : \mathcal{X} \to \bar{\mathbb{R}}$ of C is defined as

$$\delta_C(x) := \begin{cases} 0, & x \in C, \\ \infty, & x \notin C, \end{cases}$$

and the normal cone operator $N_C : \mathcal{X} \rightrightarrows \mathcal{X}$ for C is the point-to-set map given by

$$N_C(x) := \begin{cases} \emptyset, & x \notin C, \\ \{w \in \mathcal{X} : \langle \tilde{x} - x, w \rangle \leq 0 \, , \forall \tilde{x} \in C\}, & x \in C. \end{cases}$$

Clearly, the normal cone operator N_C of C can be expressed in terms of its indicator function as $N_C = \partial \delta_C$ (see Subsection 2.1 for the definition of the subdifferential of a map).

2 An adaptive block-decomposition HPE framework

In this section, we discuss an adaptive block-decomposition HPE (A-BD-HPE) framework which is an extension of the BD-HPE framework introduced in [10]. This framework is analyzed in the context of a block-structured monotone inclusion problem similar to the one in Section 3 of [10], but with the addition of an adaptive (and aggressive) stepsize choice for performing the extragradient step. This section is divided into two subsections. The first one reviews some basic definitions and facts about ε-subdifferentials of functions and ε-enlargements of monotone operators. The second one presents the A-BD-HPE framework.

2.1 The ε-subdifferential and ε-enlargement of monotone operators

Let $\mathcal{Z}$ denote a finite dimensional inner product space. A point-to-set operator $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ is a relation $T \subset \mathcal{Z} \times \mathcal{Z}$ and

$$T(z) := \{v \in \mathcal{Z} : (z, v) \in T\} .$$

Alternatively, one can consider T as a multi-valued function of $\mathcal{Z}$ into the family $\wp(\mathcal{Z}) = 2^{(\mathcal{Z})}$ of subsets of $\mathcal{Z}$. Regardless of the approach, it is usual to identify T with its graph defined as

$$Gr(T) := \{(z, v) \in \mathcal{Z} \times \mathcal{Z} : v \in T(z)\} .$$

The domain of T , denoted by $\operatorname{Dom} T$, is defined as

$$\operatorname{Dom} T := \{z \in \mathcal{Z} : T(z) \neq \emptyset\} .$$

An operator $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ is *affine* if its graph is an affine manifold. Moreover, $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ is monotone if

$$\langle v - \tilde{v}, z - \tilde{z} \rangle \geq 0, \qquad \forall (z, v), (\tilde{z}, \tilde{v}) \in Gr(T),$$

and T is maximal monotone if it is monotone and maximal in the family of monotone operators with respect to the partial order of inclusion, i.e., $S : \mathcal{Z} \rightrightarrows \mathcal{Z}$ monotone and $Gr(S) \supset Gr(T)$ implies that $S = T$.

In [2], Burachik, Iusem and Svaiter introduced the ε-enlargement of maximal monotone operators. In [12] this concept was extended to a generic point-to-set operator in $\mathcal{Z}$ as follows. Given $T : \mathcal{Z} \rightrightarrows \mathcal{Z}$ and a scalar ε, define $T^\varepsilon : \mathcal{Z} \rightrightarrows \mathcal{Z}$ as

$$T^\varepsilon(z) := \{v \in \mathcal{Z} : \langle z - \tilde{z}, v - \tilde{v} \rangle \geq -\varepsilon, \forall \tilde{z} \in \mathcal{Z}, \forall \tilde{v} \in T(\tilde{z})\}, \quad \forall z \in \mathcal{Z}.$$

The following result, whose proof can be found in Lemma 3.3 in [12], gives the characterization of the ε-enlargement of the normal cone of a closed convex cone.

Lemma 2.1 (Lemma 3.3 in [12])**.** *If K is a nonempty closed convex cone and K^* is its dual cone defined in* (3)*, then for every $x \in K$, we have*

$$-q \in (N_K)^\varepsilon(x) \iff q \in K^*, \ \langle x, q \rangle \leq \varepsilon.$$

For a scalar $\varepsilon \geq 0$, the ε-subdifferential of a function $f : \mathcal{X} \to \bar{\mathbb{R}}$ is the operator $\partial_\varepsilon f : \mathcal{X} \rightrightarrows \mathcal{X}$ defined as

$$\partial_\varepsilon f(x) := \{w \in \mathcal{X} : f(\tilde{x}) \geq f(x) + \langle \tilde{x} - x, w\rangle - \varepsilon, \forall \tilde{x} \in \mathcal{X}\}, \quad \forall x \in \mathcal{X}.$$

When $\varepsilon = 0$, the operator $\partial_\varepsilon f$ is simply denoted by ∂f and is referred to as the subdifferential of f. The operator ∂f is trivially monotone if f is proper. If f is a proper lower semi-continuous convex function, then ∂f is maximal monotone [16].

Finally, we refer the reader to [3, 19] for further discussion on the ε-enlargement of a maximal monotone operator.

2.2 The A-BD-HPE framework

In this subsection, we discuss the A-BD-HPE framework which is an extension of the BD-HPE framework introduced in [10].

This framework is analyzed in the context of a block-structured monotone inclusion problem similar to the one in Section 3 of [10]. In what follows we give the details of this block-structured monotone inclusion problem.

Let $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$ be arbitrary inner products in $\mathcal{X}$ and $\mathcal{Y}$, respectively, and denote their induced norms by $\|\cdot\|_{\mathcal{X}}$ and $\|\cdot\|_{\mathcal{Y}}$, respectively. We endow the product space $\mathcal{X} \times \mathcal{Y}$ with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{X},\mathcal{Y}}$ defined as

$$\langle (x, y), (x', y') \rangle_{\mathcal{X},\mathcal{Y}} := \langle x, x' \rangle_{\mathcal{X}} + \langle y, y' \rangle_{\mathcal{Y}}, \qquad \forall (x, y), (x', y') \in \mathcal{X} \times \mathcal{Y},$$

and denote its induced norm by $\|\cdot\|_{\mathcal{X},\mathcal{Y}}$. Consider the block-structured monotone inclusion problem of finding $(x, y) \in \mathcal{X} \times \mathcal{Y}$ such that

$$0 \in [F + (C \otimes D)](x, y), \tag{6}$$

where $C : \mathcal{X} \rightrightarrows \mathcal{X}$, $D : \mathcal{Y} \rightrightarrows \mathcal{Y}$ and the operator $C \otimes D : \mathcal{X} \times \mathcal{Y} \rightrightarrows \mathcal{X} \times \mathcal{Y}$ is defined as

$$(C \otimes D)(x, y) = C(x) \times D(y), \quad \forall (x, y) \in \mathcal{X} \times \mathcal{Y}.$$

We make the following assumptions regarding (6):

A.1 C and D are maximal monotone operators (with respect to $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$, respectively);

A.2 $F : \operatorname{Dom} F \subseteq \mathcal{X} \times \mathcal{Y} \to \mathcal{X} \times \mathcal{Y}$ is a continuous map such that $\operatorname{Dom} F \supseteq \mathcal{X} \times \operatorname{cl}(\operatorname{Dom} D)$;

A.3 F is monotone on $\operatorname{Dom} C \times \operatorname{Dom} D$ (with respect to $\langle (\cdot, \cdot), (\cdot, \cdot) \rangle_{\mathcal{X},\mathcal{Y}}$);

A.4 there exists $L_{yx} > 0$ such that

$$\|F_y(x', y) - F_y(x, y)\|_{\mathcal{Y}} \leq L_{yx} \|x' - x\|_{\mathcal{X}}, \quad \forall y \in \operatorname{Dom} D, \forall x, x' \in \mathcal{X}.$$

Assumption A.1 implies that $C \otimes D$ is maximal monotone. Hence, in view of Assumption A.2 above and Proposition A.1 of [13], it follows that the operator $F + C \otimes D$ in (6) is maximal monotone.

We now state an extension of the BD-HPE framework of [10] which uses an adaptive rule for aggressively choosing the extragradient stepsize.

A-BD-HPE Framework: An adaptive block-decomposition HPE framework for (6)

0) Let $x_0 \in \mathcal{X}$, $y_0 \in \mathcal{Y}$, $\sigma \in (0,1]$, $\sigma_y \in [0,1]$ and $\sigma_x, \tilde{\sigma}_y \in [0,1)$ be given and set $k = 1$;

1) choose $\tilde{\lambda}_k > 0$ such that

$$\sigma_k := \lambda_{\max}\left(\begin{bmatrix} \sigma_y^2 & \tilde{\lambda}_k \tilde{\sigma}_y L_{yx} \\ \tilde{\lambda}_k \tilde{\sigma}_y L_{yx} & \sigma_x^2 + \tilde{\lambda}_k^2 L_{yx}^2 \end{bmatrix}\right)^{1/2} \leq \sigma; \tag{7}$$

2) compute $\tilde{y}_k, \tilde{d}_k \in \mathcal{Y}$ and $\epsilon_k^y \geq 0$ such that

$$\tilde{d}_k \in D^{\epsilon_k^y}(\tilde{y}_k), \quad \|\tilde{\lambda}_k[F_y(x_{k-1}, \tilde{y}_k) + \tilde{d}_k] + \tilde{y}_k - y_{k-1}\|_{\mathcal{Y}}^2 + 2\tilde{\lambda}_k \epsilon_k^y \leq \sigma_y^2 \|\tilde{y}_k - y_{k-1}\|_{\mathcal{Y}}^2, \tag{8}$$

$$\|\tilde{\lambda}_k[F_y(x_{k-1}, \tilde{y}_k) + \tilde{d}_k] + \tilde{y}_k - y_{k-1}\|_{\mathcal{Y}} \leq \tilde{\sigma}_y \|\tilde{y}_k - y_{k-1}\|_{\mathcal{Y}}; \tag{9}$$

compute $\tilde{x}_k, \tilde{c}_k \in \mathcal{X}$ and $\epsilon_k^x \geq 0$ such that

$$\tilde{c}_k \in C^{\epsilon_k^x}(\tilde{x}_k), \quad \|\tilde{\lambda}_k[F_x(\tilde{x}_k, \tilde{y}_k) + \tilde{a}_k] + \tilde{x}_k - x_{k-1}\|_{\mathcal{X}}^2 + 2\tilde{\lambda}_k \epsilon_k^x \leq \sigma_x^2 \|\tilde{x}_k - x_{k-1}\|_{\mathcal{X}}^2; \tag{10}$$

3) choose λ_k to be the largest $\lambda > 0$ such that

$$\|\lambda[F(\tilde{x}_k, \tilde{y}_k) + (\tilde{c}_k, \tilde{d}_k)] + (\tilde{x}_k, \tilde{y}_k) - (x_{k-1}, y_{k-1})\|_{\mathcal{X},\mathcal{Y}}^2 + 2\lambda(\epsilon_k^x + \epsilon_k^y) \leq \sigma^2 \|(\tilde{x}_k, \tilde{y}_k) - (x_{k-1}, y_{k-1})\|_{\mathcal{X},\mathcal{Y}}^2; \tag{11}$$

4) set

$$(x_k, y_k) = (x_{k-1}, y_{k-1}) - \lambda_k[F(\tilde{x}_k, \tilde{y}_k) + (\tilde{c}_k, \tilde{d}_k)], \tag{12}$$

and $k \leftarrow k + 1$, and go to step 1.

The A-BD-HPE framework is a more aggressive version of the BD-HPE framework studied in [10]. In contrast to the A-BD-HPE framework which chooses the extragradient stepsize as the largest scalar satisfying (11), the BD-HPE framework in [10] performs the extragradient step with $\lambda_k = \tilde{\lambda}_k$. The following result shows that $\tilde{\lambda}_k$ satisfies (11) and, as a consequence, guarantees the well-definedness of the adaptive extragradient stepsize λ_k in step 3 of the A-BD-HPE framework.

Proposition 2.2. *Consider the sequences $\{(x_k, y_k)\}, \{(\tilde{x}_k, \tilde{y}_k)\}, \{(\tilde{c}_k, \tilde{d}_k)\}$, $\{\tilde{\lambda}_k\}$ and $\{(\epsilon_k^x, \epsilon_k^y)\}$ generated by the A-BD-HPE framework. Then, for every $k \in \mathbb{N}$,*

$$F(\tilde{x}_k, \tilde{y}_k) + (\tilde{c}_k, \tilde{d}_k) \in [F + (C \otimes D)^{\epsilon_k^x + \epsilon_k^y}](\tilde{x}_k, \tilde{y}_k) \subset [F + (C \otimes D)]^{\epsilon_k^x + \epsilon_k^y}(\tilde{x}_k, \tilde{y}_k) \tag{13}$$

and $\lambda = \tilde{\lambda}_k$ satisfies (11). As a consequence $\lambda_k \geq \tilde{\lambda}_k$.

Proof. This result follows from Proposition 3.1 in [10]. □

In view of (11), (12) and (13), it follows that the A-BD-HPE framework is a special case of the HPE framework for solving (6). This observation allows us to obtain complexity results for the A-BD-HPE framework using the general complexity results derived in [12]. In what follows, we state two convergence results whose proofs are analogous to those of Theorems 3.2 and 3.3 of [10], but use Proposition 2.2 above instead of Proposition 3.1 in [10].

Theorem 2.3. *Assume that $\sigma < 1$ and consider the sequences $\{(\tilde{x}_k, \tilde{y}_k)\}, \{(\tilde{c}_k, \tilde{d}_k)\}$, $\{\lambda_k\}$ and $\{(\epsilon_k^x, \epsilon_k^y)\}$ generated by the A-BD-HPE framework and let d_0 denote the distance of the initial point $(x_0, y_0) \in \mathcal{X} \times \mathcal{Y}$ to the solution set of (6) with respect to $\|(\cdot,\cdot)\|_{\mathcal{X},\mathcal{Y}}$. Then, for every $\alpha \in \mathbb{R}$ and $k \in \mathbb{N}$,*

$$(\tilde{c}_k, \tilde{d}_k) \in C^{\epsilon_k^x}(\tilde{x}_k) \times D^{\epsilon_k^y}(\tilde{y}_k),$$

and there exists $i \leq k$ such that

$$\|F(\tilde{x}_i, \tilde{y}_i) + (\tilde{c}_i, \tilde{d}_i)\|_{\mathcal{X},\mathcal{Y}} \leq d_0 \sqrt{\frac{(1+\sigma)}{(1-\sigma)} \left(\frac{\lambda_i^{\alpha-2}}{\sum_{j=1}^k \lambda_j^\alpha} \right)}, \quad \epsilon_i^x + \epsilon_i^y \leq \frac{d_0^2 \sigma^2}{2(1-\sigma^2)} \left(\frac{\lambda_i^{\alpha-1}}{\sum_{j=1}^k \lambda_j^\alpha} \right).$$

Theorem 2.4. *Assume that F is affine and consider the sequences $\{(\tilde{x}_k, \tilde{y}_k)\}, \{(\tilde{c}_k, \tilde{d}_k)\}$, $\{\lambda_k\}$ and $\{(\epsilon_k^x, \epsilon_k^y)\}$ generated by the A-BD-HPE framework and define for every $k \in \mathbb{N}$:*

$$(\tilde{x}_k^a, \tilde{y}_k^a) = \frac{1}{\Lambda_k} \sum_{i=1}^k \lambda_i (\tilde{x}_i, \tilde{y}_i), \quad (\tilde{c}_k^a, \tilde{d}_k^a) = \frac{1}{\Lambda_k} \sum_{i=1}^k \lambda_i (\tilde{c}_k, \tilde{d}_k), \tag{14}$$

and

$$\epsilon_k^{x,a} := \frac{1}{\Lambda_k} \sum_{i=1}^k \lambda_i (\epsilon_i^x + \langle \tilde{x}_i - \tilde{x}_k^a, \tilde{c}_i \rangle) \geq 0, \qquad \epsilon_k^{y,a} := \frac{1}{\Lambda_k} \sum_{i=1}^k \lambda_i (\epsilon_i^y + \langle \tilde{y}_i - \tilde{y}_k^a, \tilde{d}_i \rangle) \geq 0,$$

where $\Lambda_k = \sum_{i=1}^k \lambda_i$. Let d_0 denote the distance of the initial point $(x_0, y_0) \in \mathcal{X} \times \mathcal{Y}$ to the solution set of (6) with respect to $\|(\cdot,\cdot)\|_{\mathcal{X},\mathcal{Y}}$ and $\sigma_{xy} = \max\{\sigma_x, \sigma_y\}$. Then, for every $k \in \mathbb{N}$,

$$(\tilde{c}_k^a, \tilde{d}_k^a) \in C^{\epsilon_k^{x,a}}(\tilde{x}_k^a) \times D^{\epsilon_k^{y,a}}(\tilde{y}_k^a), \quad \left\| F(\tilde{x}_k^a, \tilde{y}_k^a) + (\tilde{c}_k^a, \tilde{d}_k^a) \right\|_{\mathcal{X},\mathcal{Y}} \leq \frac{2d_0}{\Lambda_k}, \quad \epsilon_k^{x,a} + \epsilon_k^{y,a} \leq \frac{2d_0^2}{\Lambda_k}(1 + \bar{\eta}_k),$$

where

$$\bar{\eta}_k := \frac{2\sqrt{2}\sigma}{1 - \sigma_{xy}} \left(1 + \frac{1}{(1-\sigma_y)^2} \right)^{1/2}.$$

Observe that the convergence rate bounds described in Theorems 2.3 (with $\alpha = 1$) and 2.4 suggest that the rate of convergence of the A-BD-HPE framework becomes better the larger the stepsize λ_k is chosen (under the condition that (11) is satisfied so as to guarantee that Theorems 2.3 and 2.4 still apply). In fact, the choice of λ_k at step 3 of the A-BD-HPE framework is motivated by this observation.

3 A scaled A-BD method for conic programming

In this section, we introduce an instance of the A-BD-HPE framework applied to (1) in which $\langle \cdot, \cdot \rangle_{\mathcal{X}} = \langle \cdot, \cdot \rangle$ and $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$ is a scaled inner product constructed by means of the original inner product $\langle \cdot, \cdot \rangle$ in $\mathcal{Y}$. (Recall that the original inner products in $\mathcal{X}$ and $\mathcal{Y}$ are both being denoted by $\langle \cdot, \cdot \rangle$.) Even though there is nothing new in this section from the theoretical point of view, we will use the convergence results of Section 2 to give a plausible argument showing that an appropriate choice of scaled inner product in the $\mathcal{Y}$ space may lead to a substantial speed-up of the BD method relative to its unscaled version.

We consider problem (1) with the following assumptions:

C.1 $\mathcal{A} : \mathcal{X} \to \mathcal{Y}$ is a surjective linear map and $b \in \mathcal{Y}$;

C.2 there exists $x^* \in \mathcal{X}$ satisfying the inclusion

$$0 \in c + \partial \delta_K(x) + N_{\mathcal{M}}(x), \tag{15}$$

where $\mathcal{M} := \{x \in \mathcal{X} : \mathcal{A}(x) = b\}$.

We now make a few observations about the above assumptions. First, any x^* as in C.2 is an optimal solution of (1). Second, observe that a sufficient condition for (15) to hold is that (1) has an optimal solution and satisfies the Slater condition, i.e. $\mathcal{A}\hat{x} = b$ for some $\hat{x} \in \mathrm{ri}(K)$. Third, (15) is equivalent to the existence of $y^* \in \mathcal{Y}$ such that the pair (x^*, y^*) satisfies the inclusion

$$c + \partial \delta_K(x) - \mathcal{A}^* y \ni 0, \quad \mathcal{A}x - b = 0. \tag{16}$$

Fourth, the set of solutions of the above inclusion is exactly $\mathcal{X}^* \times \mathcal{Y}^*$, where $\mathcal{X}^*$ and $\mathcal{Y}^*$ denote the set of optimal solutions of (1) and (2), respectively. Fifth, observe that (16) can be easily put into the form (6) and that Assumptions A.1-A.4 all hold when $\mathcal{X}$ and $\mathcal{Y}$ are both endowed with the original inner product $\langle \cdot, \cdot \rangle$. Hence, one can apply any instance of the A-BD-HPE framework to solve (16).

However, from the computational point of view, it is more efficient to introduce a scaled inner product in the $\mathcal{Y}$ space and work with a scaled version of (16). More specifically, given a self adjoint positive definite linear mapping $\mathcal{U} : \mathcal{Y} \to \mathcal{Y}$, endow $\mathcal{X}$ and $\mathcal{Y}$ with the inner products defined as

$$\langle \cdot, \cdot \rangle_{\mathcal{X}} := \langle \cdot, \cdot \rangle, \qquad \langle \cdot, \cdot \rangle_{\mathcal{Y}} := \langle \cdot, \mathcal{U} \cdot \rangle, \tag{17}$$

respectively, and define F, C and D as

$$F(x,y) := \begin{pmatrix} c - \mathcal{A}^* y \\ \mathcal{U}^{-1}\left(\mathcal{A}x - b\right) \end{pmatrix}, \quad C(x) = \partial \delta_K(x) = N_K(x), \quad D(y) = 0, \tag{18}$$

where the normal cone $N_K(x)$ is with respect to $\langle \cdot, \cdot \rangle_{\mathcal{X}} := \langle \cdot, \cdot \rangle$.

The following proposition can be easily shown.

Proposition 3.1. *F, C and D defined in* (18) *and the above inner products defined in* (17) *satisfy assumptions A.1-A.4 of Section 2 with $L_{yx} = \left\| \mathcal{U}^{-1/2} \mathcal{A} \right\|$.*

As a consequence of the above proposition, any instance of the A-BD-HPE framework of Section 2 with F, C and D as in (18) and the inner products $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$ defined as in (17) will satisfy the global convergence rate properties described in Theorems 2.3 and 2.4. Below we describe such an instance.

Algorithm 1 : Scaled adaptive block-decomposition (SA-BD) method for solving (1) .

---

0) Let $x_0 \in \mathcal{X}$, $y_0 \in \mathcal{Y}$, $0 < \sigma \leq 1$ and $\theta > 0$ be given, and set $k = 1$ and

$$\tilde{\lambda} = \frac{\sigma}{\|\mathcal{U}^{-1/2}\mathcal{A}\|}; \tag{19}$$

1) compute

$$\tilde{y}_k = y_{k-1} - \tilde{\lambda}\mathcal{U}^{-1}(\mathcal{A}x_{k-1} - b), \qquad \tilde{x}_k = \Pi_K\left[x_{k-1} - \tilde{\lambda}\left(c - \mathcal{A}^*\tilde{y}_k\right)\right]; \tag{20}$$

2) define

$$\tilde{v}_k = \begin{pmatrix} (x_{k-1} - \tilde{x}_k)/\tilde{\lambda} \\ \mathcal{U}^{-1}\left(\mathcal{A}\tilde{x}_k - b\right) \end{pmatrix}, \tag{21}$$

choose λ_k to be the largest $\lambda > 0$ such that

$$\left\| \lambda \tilde{v}_k + \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix} - \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} \right\|_{\mathcal{X},\mathcal{Y}} \leq \sigma \left\| \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix} - \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} \right\|_{\mathcal{X},\mathcal{Y}};$$

3) set $(x_k, y_k) = (x_{k-1}, y_{k-1}) - \lambda_k \tilde{v}_k$ and $k \leftarrow k + 1$, and go to step 1.

Proposition 3.2. *Let $\sigma_x = \sigma_y = \tilde{\sigma}_y = 0$ and define the inner products $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{Y}}$, and operators F, C and D according to* (17) *and* (18)*. Consider the sequences $\{(x_k, y_k)\}$ and $\{(\tilde{x}_k, \tilde{y}_k)\}$ generated by Algorithm 1 and, for every $k \in \mathbb{N}$, define*

$$\tilde{\lambda}_k = \tilde{\lambda}, \quad \epsilon_k^x = \epsilon_k^y = 0, \quad \tilde{d}_k = 0, \quad \tilde{c}_k = -\tilde{z}_k, \tag{22}$$

where

$$\tilde{z}_k := c - \mathcal{A}^*\tilde{y}_k - \frac{1}{\tilde{\lambda}}(x_{k-1} - \tilde{x}_k). \tag{23}$$

Then the following statements hold for every $k \in \mathbb{N}$:

a) $\tilde{\lambda}_k$ *satisfies* (7) *with* $L_{yx} = \left\|\mathcal{U}^{-1/2}\mathcal{A}\right\|$;

b) $\tilde{x}_k \in K$ *and* $\tilde{z}_k \in -N_K(\tilde{x}_k)$, *or equivalently,* $\tilde{x}_k \in K$, $\tilde{z}_k \in K^*$ and $\langle \tilde{x}_k, \tilde{z}_k\rangle = 0$;

c) $\tilde{\lambda}_k$, y_{k-1}, x_{k-1}, *and the triples* $(\tilde{y}_k, \tilde{d}_k, \epsilon_k^y)$ *and* $(\tilde{x}_k, \tilde{c}_k, \epsilon_k^x)$ *satisfy* (8), (9) *and* (10).

As a consequence, Algorithm 1 is a special instance of the A-BD-HPE framework.

Proof. a) This statement follows straightforwardly from (19).

b) Define $w_k := x_{k-1} - \tilde{\lambda}\,(c - \mathcal{A}^*\tilde{y}_k)$ and note that $\tilde{x}_k = \Pi_K(w_k) \in K$ in view of (20). This together with (23) then imply that

$$\tilde{\lambda}\tilde{z}_k = \tilde{\lambda}(c - \mathcal{A}^*\tilde{y}_k) - (x_{k-1} - \Pi_K(w_k)) = -\left(w_k - \Pi_K(w_k)\right) \in -N_K\left(\tilde{x}_k\right), \tag{24}$$

where the inclusion follows from the well-known fact that $x - \Pi_K(x) \in N_K(\Pi_K(x))$ for all $x \in \mathcal{X}$. Statement b) now follows from the above observations and the fact that $N_K\left(\tilde{x}_k\right)$ is a cone. The equivalent statement of b) follows from Lemma 2.1 with $q = \tilde{z}_k$, $x = \tilde{x}_k$ and $\varepsilon = 0$.

c) It follows from (22), (23), the definition of F in (18), and the definition of $\tilde{y}_k$ in (20) that

$$\tilde{\lambda}_k\left[F_y(x_{k-1}, \tilde{y}_k) + \tilde{d}_k\right] + \tilde{y}_k - y_{k-1} = 0, \qquad \tilde{\lambda}_k\left[F_x(\tilde{x}_k, \tilde{y}_k) + \tilde{c}_k\right] + \tilde{x}_k - x_{k-1} = 0.$$

Clearly, the fact that $\sigma_x = \sigma_y = \tilde{\sigma}_y = \epsilon_k^x = \epsilon_k^y = 0$ and the two identities above imply that all the inequalities in (8), (9) and (10) are satisfied. The inclusions in (8) and (10) hold from the definitions of D, ϵ_k^y, $\tilde{d}_k$, C, ϵ_k^x and $\tilde{c}_k$ in (18) and (22), and the inclusion in (24). Hence, statement c) follows.

Finally, the definitions of F in (18), $\tilde{v}_k$ in (21), and $\tilde{c}_k$ and $\tilde{d}_k$ in (22), imply that

$$\tilde{v}_k = F(\tilde{x}_k, \tilde{y}_k) + (\tilde{c}_k, \tilde{d}_k).$$

This observation together with the fact that $\epsilon_k^x = \epsilon_k^y = 0$ then imply that steps 2 and 3 of Algorithm 1 are equivalent to steps 3 and 4 of the A-BD-HPE framework. Therefore, the conclusion of the proposition follows. □

We now specialize the convergence rate results of Section 2, namely Theorems 2.3 and 2.4, to the context of Algorithm 1. First, define for every non-singular linear mapping $\mathcal{B} : \mathcal{Y} \to \mathcal{Y}$ and $y \in \mathcal{Y} \setminus \{0\}$ the following quantity

$$\xi(\mathcal{B}, y) := \frac{\|\mathcal{B}y\|}{\|\mathcal{B}\|\|y\|} \in \left[\frac{1}{\kappa(\mathcal{B})}, 1\right], \tag{25}$$

where $\kappa(\mathcal{B}) := \|\mathcal{B}^{-1}\|\|\mathcal{B}\|$ denotes the condition number of $\mathcal{B}$. Note that for any scalar $\theta \in \mathbb{R} \setminus \{0\}$ we have that $\xi(\theta\mathcal{B}, y) = \xi(\mathcal{B}, y)$ and $\xi(\theta\mathcal{I}, y) = 1$.

Theorem 3.3. *Consider the sequences* $\{(x_k, y_k)\}$ *and* $\{(\tilde{x}_k, \tilde{y}_k)\}$ *generated by Algorithm 1, the sequence* $\{\tilde{z}_k\}$ *defined as in* (23), *the sequences* $\{(\tilde{x}_k^a, \tilde{y}_k^a)\}$ *and* $\{\tilde{c}_k^a\}$ *defined as in* (14), *and the sequence* $\{\tilde{z}_k^a\}$ *defined by* $\tilde{z}_k^a = -\tilde{c}_k^a$ *for every* $k \in \mathbb{N}$. *Let* $\mathcal{X}^*$ *and* $\mathcal{Y}^*$ *denote the set of optimal solutions of* (1) *and* (2), *respectively, and* $(x^*, y^*) \in \mathcal{X}^* \times \mathcal{Y}^*$ *be such that*

$$d_{0,x} := \min\{\|x_0 - x\| : x \in \mathcal{X}^*\} = \|x_0 - x^*\|, \quad d_{0,y} := \min\{\|y_0 - y\| : y \in \mathcal{Y}^*\} = \|y_0 - y^*\|. \tag{26}$$

Then, for every $k \in \mathbb{N}$, *the following statements hold:*

a) $\tilde{x}_k \in K$, $\tilde{z}_k \in K^*$, $\langle \tilde{x}_k, \tilde{z}_k\rangle = 0$, *and if* $\sigma < 1$, *there exists* $i \le k$ *such that*

$$\|\mathcal{A}^*\tilde{y}_i + \tilde{z}_i - c\|^2 + \|\mathcal{U}^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|^2 \le \left(\frac{1+\sigma}{1-\sigma}\right)\|\mathcal{U}^{-1/2}\mathcal{A}\|^2\frac{d_{0,x}^2 + \xi_0(\mathcal{U})\|\mathcal{U}\|d_{0,y}^2}{k\sigma^2}, \tag{27}$$

where $\xi_0(\mathcal{U}) := [\xi(\mathcal{U}^{1/2}, y_0 - y^*)]^2$.

b) $\tilde{x}_k^a \in K$, $\tilde{z}_k^a \in K^*$ *and*

$$\|\mathcal{A}^*\hat{y}_k^a + \tilde{z}_k^a - c\|^2 + \|\mathcal{U}^{-1/2}(\mathcal{A}\tilde{x}_k^a - b)\|^2 \leq \|\mathcal{U}^{-1/2}\mathcal{A}\|^2 \frac{d_{0,x}^2 + \xi_0(\mathcal{U})\|\mathcal{U}\| d_{0,y}^2}{(k\sigma)^2},$$

$$\langle \tilde{x}_k^a, \tilde{z}_k^a \rangle \leq (2 + 8\sigma) \left\|\mathcal{U}^{-1/2}\mathcal{A}\right\| \frac{d_{0,x}^2 + \xi_0(\mathcal{U})\|\mathcal{U}\| d_{0,y}^2}{k\sigma}.$$

Proof. We first prove statement a). Let $k \in \mathbb{N}$ be given. First note that from Proposition 3.2(d) we have $\tilde{z}_k \in K^*$ and $\langle \tilde{x}_k, \tilde{z}_k \rangle = 0$. Observe also that $\mathcal{X}^* \times \mathcal{Y}^*$ is the set of solutions of the inclusion problem (6) and (18) (see the fourth observation after (15)). Let d_0 denote the distance of (x_0, y_0) to $\mathcal{X}^* \times \mathcal{Y}^*$ with respect to the scaled norm $\|\cdot\|_{\mathcal{X},\mathcal{Y}}$, and observe that the definitions of (x^*, y^*), $d_{0,x}$, $d_{0,y}$ and $\xi_0(\mathcal{U})$ imply

$$d_0^2 \leq \|x_0 - x^*\|_{\mathcal{X}}^2 + \|y_0 - y^*\|_{\mathcal{Y}}^2 = \|x_0 - x^*\|^2 + \|\mathcal{U}^{1/2}(y_0 - y^*)\| = d_{0,x}^2 + \xi_0(\mathcal{U})\|\mathcal{U}\| d_{0,y}^2.$$

Moreover, by Proposition 3.2 and Theorem 2.3 with $\alpha = 1$, we conclude that if $\sigma < 1$, there exists $i \leq k$ such that

$$\begin{aligned} \|c - \mathcal{A}^*\tilde{y}_i + \tilde{c}_i\|_{\mathcal{X}}^2 + \|\mathcal{U}^{-1}\left(\mathcal{A}\tilde{x}_i - b\right) + \tilde{d}_i\|_{\mathcal{Y}}^2 &\leq \left(\tfrac{1+\sigma}{1-\sigma}\right) \frac{d_0^2}{\lambda_i \sum_{j=1}^k \lambda_j} \leq \left(\tfrac{1+\sigma}{1-\sigma}\right) \frac{d_0^2}{\tilde{\lambda}^2 k} \\ &= \left(\tfrac{1+\sigma}{1-\sigma}\right) \|\mathcal{U}^{-1/2}\mathcal{A}\|^2 \frac{d_0^2}{k\sigma^2}, \end{aligned}$$

where the second inequality follows from Proposition 2.2 with $\tilde{\lambda}_k = \tilde{\lambda}$, and the last equality follows from the definition of $\tilde{\lambda}$ in (19). Also, in view of (17) and the definitions of $\tilde{d}_i$ and $\tilde{c}_i$ in (22), we have

$$\|c - \mathcal{A}^*\tilde{y}_i + \tilde{c}_i\|_{\mathcal{X}}^2 = \|\mathcal{A}^*\tilde{y}_i + \tilde{z}_i - c\|^2, \qquad \|\mathcal{U}^{-1}\left(\mathcal{A}\tilde{x}_i - b\right) + \tilde{d}_i\|_{\mathcal{Y}}^2 = \|\mathcal{U}^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|^2.$$

Then, combining the last four relations we obtain (27), and hence statement a) follows.

Statement b) can be proved in a similar way using Theorem 2.4 instead of Theorem 2.3. □

We now make several remarks about Theorem 3.3. For the sake of simplicity, we will focus our discussion on the point-wise convergence rate bound (27). Define the self-adjoint positive definite linear mapping $\mathcal{U}_0 : \mathcal{Y} \to \mathcal{Y}$ as

$$\mathcal{U}_0 := \mathcal{A}\mathcal{A}^*.$$

First, the term $\|\mathcal{U}^{-1/2}\mathcal{A}\|$ in the right hand side of (27) is minimized over the class $\mathcal{C}(\mathcal{A})$ consisting of all self-adjoint positive definite linear mappings $\mathcal{U} : \mathcal{Y} \to \mathcal{Y}$ satisfying $\|\mathcal{U}^{-1/2}\mathcal{A}\|^2 = \|\mathcal{A}\|^2/\|\mathcal{U}\|$, or equivalently,

$$\|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|^2 = \|\mathcal{U}_0\|/\|\mathcal{U}\|. \tag{28}$$

Note that any positive multiple of the identity operator $\mathcal{I}$ or the operator $\mathcal{U}_0$ belongs to $\mathcal{C}(\mathcal{A})$. In view of this remark, we will assume from now on that $\mathcal{U} \in \mathcal{C}(\mathcal{A})$, and within this class we will consider the subclass $\mathcal{C}_\theta(\mathcal{A})$ consisting of the operators $\mathcal{U} \in \mathcal{C}(\mathcal{A})$ such that $\|\mathcal{U}\| = \theta$, where $\theta > 0$ is some pre-specified scalar. Second, recall that the definition of the term $\xi_0(\mathcal{U})$ implies that $\xi_0(\mathcal{U}) \in [1/\kappa(\mathcal{U}), 1]$ and that $\xi_0(\theta\mathcal{I}) = 1$. Hence, $\mathcal{U} = \theta\mathcal{I}$ maximizes $\xi_0(\mathcal{U})$ over $\mathcal{C}_\theta(\mathcal{A})$. Also, it is interesting to observe that the best possible value $\xi_0(\mathcal{U})$ might take over $\mathcal{C}(\mathcal{A})$, namely $1/\kappa(\mathcal{U})$, achieves its minimum value when $\mathcal{U}$ is a positive multiple of $\mathcal{U}_0$. Indeed, in view of (28) and the definition of $\kappa(\cdot)$, we have

$$\kappa(\mathcal{U}) = \|\mathcal{U}\|\|\mathcal{U}^{-1}\| = \|\mathcal{U}\|\|\mathcal{U}^{-1/2}\|^2 \leq \|\mathcal{U}\|\|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|^2\|\mathcal{U}_0^{-1/2}\|^2 = \|\mathcal{U}\|\frac{\|\mathcal{U}_0\|}{\|\mathcal{U}\|}\|\mathcal{U}_0^{-1}\| = \kappa(\mathcal{U}_0), \quad \forall \mathcal{U} \in \mathcal{C}(\mathcal{A}).$$

Third, if you view the vector $u_0 := (y_0 - y^*)/\|y_0 - y^*\|$ as being uniformly distributed on the unit sphere, then Lemma A.1 and the definition of $\xi_0(\mathcal{U})$ implies that the expected value of $\xi_0(\mathcal{U})$ with respect to u_0 is $\mathrm{tr}(\mathcal{U})/(m\|\mathcal{U}\|)$, or in words, the average of the eigenvalues of $\mathcal{U}$ divided by the maximum eigenvalue of $\mathcal{U}$. Hence, if $\mathcal{U}_0$ is such that $\mathrm{tr}(\mathcal{U}_0)/(m\|\mathcal{U}_0\|)$ is of the same order of magnitude as $1/\kappa(\mathcal{U}_0)$ and $\mathcal{U}_0$ is ill-conditioned, then the choice of $\mathcal{U} = \theta\mathcal{U}_0/\|\mathcal{U}_0\|$ from the class $\mathcal{C}_\theta(\mathcal{A})$ for Algorithm 1 will be nearly optimal in

the sense of minimizing $\xi_0(\mathcal{U})$. Note that the latter condition happens when $\mathcal{U}_0$ is ill-conditioned and most of the eigenvalues of $\mathcal{U}_0$ are relatively close to its minimum eigenvalue.

We will now interpret the bound (27) from a geometrical point of view. Define the primal and dual manifolds as

$$\mathcal{M}_p := \{x : \mathcal{A}x = b\}, \qquad \mathcal{M}_d := \{c - \mathcal{A}^* y : y \in \mathcal{Y}\},$$

and define

$$\hat{\xi}_i(\mathcal{U}) \;:=\; \left(\frac{1}{\xi(\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}, \mathcal{U}_0^{-1/2}(\mathcal{A}\tilde{x}_i - b))}\right)^2. \tag{29}$$

For every $\mathcal{U} \in \mathcal{C}_\theta(\mathcal{A})$, we can easily see that (27), (28), the definition of $\mathrm{dist}_C(\cdot)$ in (5), and the fact that $\|\mathcal{U}\| = \theta$ and $\|\mathcal{U}^{-1/2}\mathcal{A}\| = \|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|$, imply

$$[\mathrm{dist}_{\mathcal{M}_d}(\tilde{z}_i)]^2 \leq \|\mathcal{A}^*\tilde{y}_i + \tilde{z}_i - c\|^2 \leq \left(\frac{1+\sigma}{1-\sigma}\right)\frac{\|\mathcal{U}_0\|}{k\sigma^2}\left(\frac{d_{0,x}^2}{\theta} + \xi_0(\mathcal{U})d_{0,y}^2\right). \tag{30}$$

We will now bound the distance $\mathrm{dist}_{\mathcal{M}_p}(\tilde{x}_i)$. First, it is easy to see that

$$\mathrm{dist}_{\mathcal{M}_p}(\tilde{x}_i) = \|\mathcal{U}_0^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|.$$

Hence, for every $\mathcal{U} \in \mathcal{C}_\theta(\mathcal{A})$, we have that (27), (25), (29), and the fact that $\|\mathcal{U}\| = \theta$ and $\|\mathcal{U}^{-1/2}\mathcal{A}\| = \|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|$, imply

$$\begin{aligned}
[\mathrm{dist}_{\mathcal{M}_p}(\tilde{x}_i)]^2 &= \|\mathcal{U}_0^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|^2 = \left(\frac{\|\mathcal{U}^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|}{\xi(\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}, \mathcal{U}_0^{-1/2}(\mathcal{A}\tilde{x}_i - b))\|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|}\right)^2 \\
&= \frac{\hat{\xi}_i(\mathcal{U})}{\|\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}\|^2}\|\mathcal{U}^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|^2 \leq \left(\frac{1+\sigma}{1-\sigma}\right)\frac{\hat{\xi}_i(\mathcal{U})}{k\sigma^2}\left(d_{0,x}^2 + \xi_0(\mathcal{U})\theta d_{0,y}^2\right).
\end{aligned} \tag{31}$$

Note that the definition of the term $\hat{\xi}_i(\mathcal{U})$ implies that $\hat{\xi}_i(\mathcal{U}) \in [1, (\kappa(\mathcal{U}^{-1/2}\mathcal{U}_0^{1/2}))^2]$ and that $\xi_0(\theta\mathcal{U}_0) = 1$. Hence, the choice of $\mathcal{U} = \theta\mathcal{U}_0/\|\mathcal{U}_0\|$ minimizes $\hat{\xi}_i(\mathcal{U})$ over $\mathcal{C}_\theta(\mathcal{A})$ which, in view of the observations about the term $\xi_0(\mathcal{U})$ above, can lead to nearly optimal bounds for the distances to the primal and dual manifolds in (31) and (30), respectively.

The convergence rate bounds in (30) and (31), not only highlight the benefits obtained by $\xi_0(\mathcal{U}) \leq 1$ for an ill-conditioned $\mathcal{U}$, but also suggest how the magnitude of $\|\mathcal{U}\| = \theta$ affects the size of the primal and dual residuals. More specifically, viewing all the quantities in (30) and (31), with the exception of θ, as constants, and noting that

$$[\mathrm{dist}_{\mathcal{M}_p}(\tilde{x}_i)]^2 = \frac{\|\mathcal{U}_0^{-1/2}(\mathcal{A}\tilde{x}_i - b)\|^2\|\mathcal{U}_0^{1/2}\|^2}{\|\mathcal{A}\tilde{x}_i - b\|^2}\frac{\|\mathcal{A}\tilde{x}_i - b\|^2}{\|\mathcal{U}_0^{1/2}\|^2} = \frac{\hat{\xi}_i(\mathcal{I})\|\mathcal{A}\tilde{x}_i - b\|^2}{\|\mathcal{U}_0\|} \geq \frac{\|\mathcal{A}\tilde{x}_i - b\|^2}{\|\mathcal{U}_0\|},$$

we can see that the primal and dual residuals are

$$\|\mathcal{A}\tilde{x}_i - b\|^2 = \mathcal{O}\left(\max\left\{1, \theta^{1/2}\right\}\right), \qquad \|\mathcal{A}^*\tilde{y}_i + \tilde{z}_i - c\|^2 = \mathcal{O}\left(\max\left\{1, \theta^{-1/2}\right\}\right),$$

respectively. Hence, as $\theta \to 0$, the dual residual can become significantly larger than the primal one while, as $\theta \to \infty$, the primal residual can become significantly larger than the dual one. In fact, we have observed in our computational experiments that these residuals behave exactly as just described. In Section 4, we will use $\mathcal{U} = \theta\mathcal{U}_0$ and a dynamic choice of the scaling parameter θ in our implementation of Algorithm 1 so as to empirically balance the primal and dual residuals and as a consequence improve the practical performance of the method.

4 Implementation details and numerical results

In this section, we describe all the ingredients needed to speed-up the implementation of Algorithm 1, and present numerical results demonstrating the efficiency of the resulting method for solving many large instances of (1) and (4). More specifically, we describe two important ingredients, namely: i) convenient choice of initial primal and dual iterates, and initial parameter for the scaled inner product (17) on the space $\mathcal{X}$, and; ii) dynamic change of the scaled inner product in the $\mathcal{X}$ space as the algorithm progresses, with the aim of properly balancing the sizes of the primal and dual relative residuals. This section also contains five subsections reporting computational results which compare our method with the ones in [9, 15] and [22] for various types of conic semidefinite programming problems.

For every $k \in \mathbb{N}$, define the primal and dual relative residuals as

$$\epsilon_{P,k} := \frac{\|\mathcal{A}\tilde{x}_k - b\|}{1 + \|b\|}, \quad \epsilon_{D.k} := \frac{\|\mathcal{A}^*\tilde{y}_k + \tilde{z}_k - c\|}{1 + \|c\|}, \tag{32}$$

where $\{\tilde{x}_k\}$ and $\{\tilde{y}_k\}$ are the sequence generated by Algorithm 1, and $\{\tilde{z}_k\}$ is given by (23). In our implementation, we used the stopping criterion

$$\max\{\epsilon_{P,k}, \epsilon_{D,k}\} \leq \bar{\epsilon}, \tag{33}$$

where $\bar{\epsilon} > 0$ is a given tolerance. We observe that the complementarity measure is $\langle \tilde{x}_k, \tilde{z}_k \rangle = 0$ for every $k \in \mathbb{N}$, in view of Theorem 3.3(a). We note that the two methods we compare our code to also use the stopping criterion (33) and satisfy the later complementarity property.

Our implementation chooses the initial iterates x_0 and y_0 as

$$x_0 = 0, \qquad y_0 = \arg\min \|\mathcal{A}^* y - c\| = \mathcal{U}_0^{-1}\mathcal{A}c. \tag{34}$$

Another possibility would be to choose x_0 as the vector with minimum norm lying in the manifold $\{x \in \mathcal{X} : \mathcal{A}x = b\}$. However, the computational results reported in this paper are based on the choice of the initial iterates given by (34).

Our benchmark is based on an implementation of Algorithm 1 in which $\sigma = 0.99$ and the operator $\mathcal{U}$ is chosen as

$$\mathcal{U} = \theta\mathcal{U}_0, \tag{35}$$

where θ is dynamically updated whenever a specified number of iterations is performed. In the next two paragraphs we discuss how to initialize θ and the scheme for dynamically updating it.

First we discuss how to initialize θ. Note that the choice (35) of $\mathcal{U}$ implies that $\left\|\mathcal{U}^{-1/2}\mathcal{A}\right\| = \theta^{-1/2}$, and hence

$$\tilde{\lambda} = \sigma\theta^{1/2},$$

in view of (19). This observation together with (32), (23) and (34) imply that the initial relative residuals $\epsilon_{P,1}$ and $\epsilon_{D,1}$ as a function of θ are given by

$$\epsilon_{P,1} = \epsilon_{P,1}(\theta) := \frac{\|\mathcal{A}\tilde{x}_1(\theta) - b\|}{1 + \|b\|}, \qquad \epsilon_{D,1} = \epsilon_{D,1}(\theta) := \frac{\|(x_0 - \tilde{x}_1(\theta))/\tilde{\lambda}\|}{1 + \|c\|} = \frac{\left\|\sigma^{-1}\theta^{-1/2}\tilde{x}_1(\theta)\right\|}{1 + \|c\|}, \tag{36}$$

where

$$\begin{aligned} \tilde{x}_1 = \tilde{x}_1(\theta) &:= \Pi_K\left[x_0 - \tilde{\lambda}\left(c - \mathcal{A}^*\tilde{y}_1\right)\right] = \Pi_K\left[x_0 - \tilde{\lambda}\left(c - \mathcal{A}^*(y_0 - \tilde{\lambda}\mathcal{U}^{-1}(\mathcal{A}x_0 - b))\right)\right] \\ &= \sigma\theta^{1/2}\Pi_K\left[-c + \mathcal{A}^*\left(y_0 + \sigma\theta^{1/2}\mathcal{U}^{-1}b\right)\right]. \end{aligned}$$

Using the definition of y_0 in (34), we easily see that $\|\mathcal{A}^* y_0 - c\| \leq \|c\|$, and hence , as $\theta \to 0$, we have from (36) that

$$\epsilon_{P,1}(\theta) \to \frac{\|b\|}{1 + \|b\|} < 1, \qquad \epsilon_{D,1}(\theta) \to \frac{\|\Pi_K\left[\mathcal{A}^* y_0 - c\right]\|}{1 + \|c\|} \leq \frac{\|\mathcal{A}^* y_0 - c\|}{1 + \|c\|} \leq \frac{\|c\|}{1 + \|c\|} < 1,$$

As a consequence, we can always choose an initial θ so as to enforce $\max\{\epsilon_{P,1}, \epsilon_{D,1}\}$ to be $\mathcal{O}(1)$. In fact, in our implementation we use the following procedure. Given a constant $\rho \geq 1$, we check whether $\max\{\epsilon_{P,1}(1), \epsilon_{D,1}(1)\} \leq \rho$. If so, we set the initial θ to be 1, otherwise we successively divide the current value of θ by 2 until $\max\{\epsilon_{P,1}(\theta), \epsilon_{D,1}(\theta)\} \leq \rho$ is satisfied, and use this value as the initial θ. The motivation behind this initial choice of θ is to guarantee that the initial primal and dual relative residuals $\epsilon_{P,k}$ and $\epsilon_{D,k}$ are not too large at the first iteration of Algorithm 1.

Even though, the convergence rate bounds of Theorem 3.3 are guaranteed for a fixed value of θ, we have used in our computational results the heuristic of changing θ every time a specified number $\bar{k}$ of iterations have been performed. The motivation for dynamically changing θ, is that our preliminary computational experiments have suggested us that the performance of the method is improved as $\epsilon_{P,k}$ and $\epsilon_{D,k}$ are of the same order of magnitude. More specifically, if θ_k denotes the dynamic value of θ at the kth iteration of the algorithm, we use the following rule for updating θ_k,

$$\theta_k = \begin{cases} \theta_{k-1}, & k \not\equiv 0 \bmod \bar{k} \text{ or } \gamma^{-1} \leq \epsilon_{P,k-1}/\epsilon_{D,k-1} \leq \gamma \\ \theta_{k-1} \cdot \tau, & k \equiv 0 \bmod \bar{k} \text{ and } \epsilon_{P,k-1}/\epsilon_{D,k-1} > \gamma \\ \theta_{k-1}/\tau, & k \equiv 0 \bmod \bar{k} \text{ and } \epsilon_{D,k-1}/\epsilon_{P,k-1} > \gamma \end{cases}, \qquad \forall k \geq 2,$$

for some pre-specified integer $\bar{k} \geq 1$, and scalars $\gamma > 1$ and $0 < \tau < 1$. In our computational experiments, we have used $\bar{k} = 5$, $\gamma = 1.5$ and $\tau = 0.9$. Note that this update rule is motivated by the last observation in Section 3. In summary, the update rule changes the value of θ at most a single time in the right direction, so as to balance the sizes of the primal and dual relative residuals based on the information provided by their values at the previous iteration. We should emphasize that convergence rate bounds for Algorithm 1 endowed with this updating rule are not available, but we have observed that this variant of Algorithm 1 performs extremely well.

In our computational experiments, we will refer to the variant of Algorithm 1 described above as the *dynamically scaled adaptive block-decomposition* (DSA-BD) method for solving (1). This variant was implemented for spaces $\mathcal{X}$ and $\mathcal{Y}$, and cone K given as in (4). Hence, our code is able to solve conic programming problems given in standard form (i.e., as in (1)) with n_u unrestricted scalar variables, n_l nonnegative scalar variables and an $n_s \times n_s$ positive semidefinite symmetric matrix variable. The inner products (before scaling) used in $\mathcal{X}$ and $\mathcal{Y}$ are the standard ones, namely: the scalar inner product in $\mathcal{Y}$ and the following inner product in $\mathcal{X}$

$$\langle x, \tilde{x} \rangle := x_v^T \tilde{x}_v + X_s \bullet \tilde{X}_s,$$

for every $x = (x_v, X_s) \in \mathbb{R}^{n_u+n_l} \times \mathcal{S}^{n_s}$ and $\tilde{x} = (\tilde{x}_v, \tilde{X}_s) \in \mathbb{R}^{n_u+n_l} \times \mathcal{S}^{n_s}$, where $X \bullet Y := Tr(X^T Y)$ for every $X, Y \in \mathcal{S}^{n_s}$.

We present a computational benchmark of our algorithm (DSA-BD) compared to the semismooth Newton-CG augmented Lagrangian (SDPNAL) method in [22] and the boundary-point (BP) method in [9, 15]. We implemented the DSA-BD method in MATLAB using the SDPT3 data structures described in [20], but without exploiting any possible block sparsity on the semidefinite variable X_s. All the tests were made using a server with 2 Xeon X5460 processors at 3.16GHz and 32GB RAM.

Various large-scale SDP problems are solved to obtain this benchmark, ranging from purely random to SDP relaxations of combinatorial optimization problems such as the frequency assignment problem (FAP), the binary integer quadratic (BIQ) problem, the quadratic assignment problem (QAP) and the maximum stable set problem (Lovász θ-function and θ_+-function). In the following subsections, we describe in detail the problems included in our computational tests but before that, we make some general remarks about how the results are reported on the several tables given below. In Tables 1 and 11, we compare our method to BP and SDPNAL methods while, in Tables 3, 5, 6, 9 and 13, we compare it against SDPNAL only due to the fact that the current version of the BP method available to us only accepts conic optimization SDP problems without nonnegative scalar variables. In some of these tables, we report computational results for the same problem using two different tolerances. They are listed in two different rows of the table to the right of the name and size of the instance. We mark the time and the residual for a method in red, and also with an asterisk (*), whenever the instance cannot be solved to the required accuracy, with the convention that the time and residual reported are the ones obtained at the last iteration of the method. Also, the time marked in blue in a row is the best one among the times listed in that row under the convention that when a method cannot solve the instance, the corresponding time is assumed to be ∞.

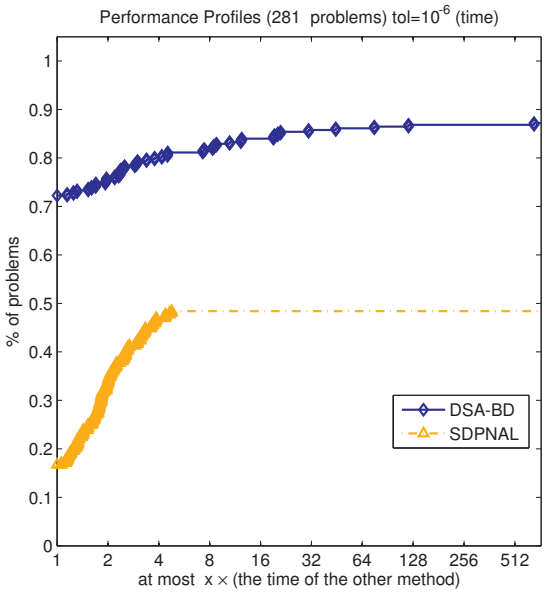

Figure 1: Performance profiles of DSA-BD and SDPNAL for solving 281 conic semidefinite programming problems with accuracy $\bar{\epsilon} = 10^{-6}$.

Observe that the final relative residuals obtained by BP and DSA-BD are very close to the desired accuracy when the latter is achieved. On the other hand, the ones obtained by SDPNAL can be noticeably smaller than the desired accuracy when the latter is achieved. This is due to the fact that SDPNAL is a second-order method and therefore it performs much fewer (and computationally more expensive) iterations than the other two methods. As a result, SDPNAL improves the relative residuals in a single iteration substantially more than the other two methods.

In Tables 2, 4, 7, 8, 10, 12 and 14, we report more detailed computational results obtained by our method DSA-BD. We do not report the violations to the conditions $\tilde{x} \in K$, $\tilde{z} \in K^*$, $\langle \tilde{x}, \tilde{z} \rangle = 0$, since they are satisfied up to machine precision at every iteration of the DSA-BD algorithm applied to all the instances in our benchmark.

Finally, we recall the following definition of a performance profile. For a given instance, a method A is said to be at most x times slower than method B, if the time taken by method A is at most x times the the time taken by method B. A point (x, y) is in the performance profile curve of a method if it can solve exactly $(100y)\%$ of all the tested instances x times slower than any other competing method. Figure 1 plots the performance profiles (see [6]) of DSA-BD and SDPNAL methods based on all instances used in our benchmark. Note that the curve for SDPNAL becomes flat for $x \geq 6$ at a y value equal to about 0.5. This is due to the fact that SDPNAL fails to solve about 50% of the instances, although it is faster than DSA-BD on 18% of the instances. Other performance profiles based on instances belonging to the same class of conic programming problems will be reported in the subsequent subsections.

4.1 Random SDPs

This subsection compares the performance of our method DSA-BD with that of BP and SDPNAL on a collection of random sparse SDP problems. These instances were also used in [9] to report the performance of BP introduced in [15].

Table 1 compares the three methods on a collection of random sparse SDP instances using the tolerance $\bar{\epsilon} = 10^{-6}$. Table 2 gives more detailed computational results on these instances obtained by our method DSA-BD, such as the objective values, number of iterations, and the primal and dual relative residuals.

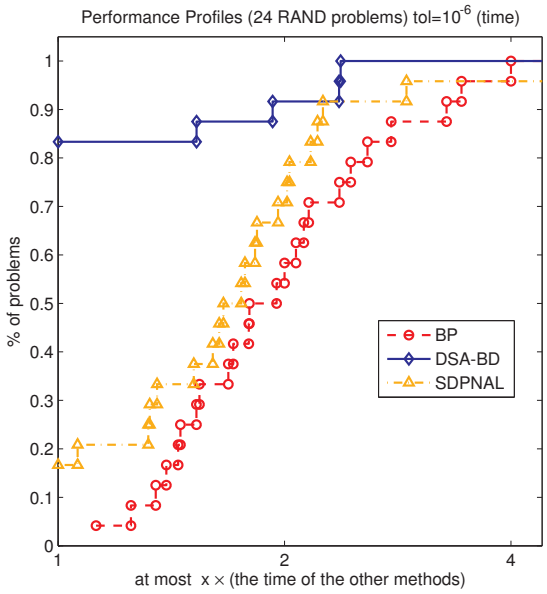

Figure 2: Performance profiles of DSA-BD, SDPNAL and BP for solving 24 random SDP problems with accuracy $\bar{\epsilon} = 10^{-6}$.

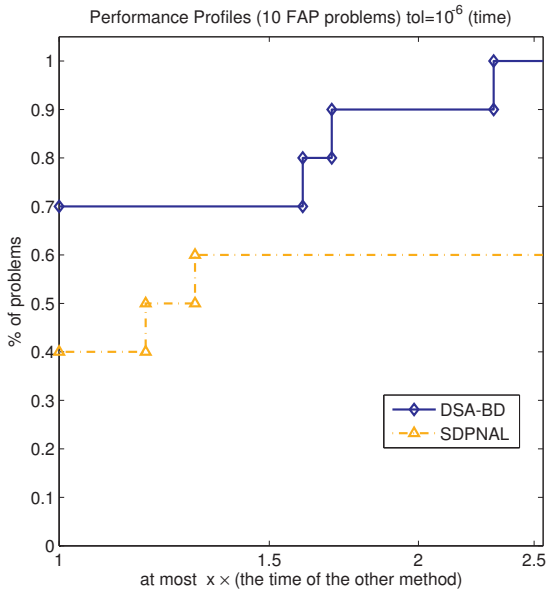

Figure 3: Performance profiles of DSA-BD and SDPNAL for solving 10 SDP relaxations of FAPs with accuracy $\bar{\epsilon} = 10^{-6}$.

Figure 2 plots the performance profiles of the three methods based on these random sparse SDP instances only.

Note that DSA-BD finds a solution with an accuracy of at least 10^{-6} faster than BP and SDPNAL do in most of the random sparse SDP instances tested. In particular, DSA-BD is the fastest method on the larger instances.

4.2 Frequency assignment problems

This subsection compares the performance of our method DSA-BD with that of SDPNAL on a collection of SDP relaxations of FAPs.

The SDP relaxation of the FAP can be described as follows (see for example Subsection 2.4 in [4]). Given a network represented by a graph G and an edge-weight matrix W, the frequency assignment problem on G can be formulated as a k-cut problem

$$\begin{array}{ll} \max & \left[\left(\frac{k-1}{2k}\right) L(G,W) - \frac{1}{2}\mathrm{Diag}(We)\right] \bullet X \\ \text{s.t.} & -E^{ij} \bullet X \le 2/(k-1) \quad \forall\ (i,j) \\ & -E^{ij} \bullet X = 2/(k-1) \quad \forall\ (i,j) \in U \subseteq E \\ & \mathrm{diag}(X) = e,\ X \succeq 0,\ \mathrm{rank}(X) = k, \end{array}$$

where $k > 1$ is an integer, $L(G,W) := \mathrm{Diag}(We) - W$ is the Laplacian matrix, $E^{ij} = e_i e_j^T + e_j e_i^T$ with $e_i \in \mathbb{R}^n$ the vector with all zeros except in the ith position and $e \in \mathbb{R}^n$ is the vector with all ones. An SDP relaxation of the problem above is obtained by dropping the rank restriction and the inequality constraint

for the non-edges to obtain the following formulation

$$\begin{array}{ll} \max & \left[\left(\dfrac{k-1}{2k}\right) L(G,W) - \dfrac{1}{2}\mathrm{Diag}(We)\right] \bullet X \\ \text{s.t.} & -E^{ij} \bullet X \le 2/(k-1) \quad \forall\ (i,j) \in E \setminus U \\ & -E^{ij} \bullet X = 2/(k-1) \quad \forall\ (i,j) \in U \subseteq E \\ & \mathrm{diag}(X) = e,\ X \succeq 0. \end{array}$$

Table 3 compares the two methods on a collection of SDP relaxations of FAPs using the tolerances $\bar{\epsilon} = 10^{-5}, 10^{-6}$. In this table, computational results for each instance are reported in two rows, the first one for $\bar{\epsilon} = 10^{-5}$, and the second one for $\bar{\epsilon} = 10^{-6}$. Table 4 gives more detailed computational results on these instances obtained by our method DSA-BD, such as the objective values, number of iterations, and the primal and dual relative residuals. Figure 3 plots the performance profiles of both methods based on these SDP relaxations of FAPs.

Note that our method performs better than SDPNAL on large SDP relaxations of FAPs (i.e., `fap25` and `fap36`).

4.3 Binary integer quadratic problems

This subsection compares the performance of our method DSA-BD with that of SDPNAL on a collection of SDP relaxations of BIQ problems.

The SDP relaxation of the BIQ problem can be described as follows (see for example Section 7 in [22]). Given an $n \times n$ symmetric matrix Q the BIQ problem can be formulated as

$$\min\left\{x^T Q x : x \in \{0,1\}^n\right\}.$$

By representing the binary set $\{0,1\}^n$ as $\left\{x \in \mathbb{R}^n | x_i^2 - x_i = 0\right\}$, we obtain an SDP relaxation as follows

$$\begin{array}{ll} \min & Q \bullet X \\ \text{s.t.} & \mathrm{diag}(X) - x = 0,\ \alpha = 1, \\ & \begin{bmatrix} X & x \\ x^T & \alpha \end{bmatrix} \succeq 0,\ X \ge 0,\ x \ge 0. \end{array}$$

Tables 5 and 6 compare the two methods on a collection of SDP relaxations of BIQ problems using the tolerance $\bar{\epsilon} = 10^{-6}$. Tables 7 and 8 give more detailed computational results on these instances obtained by our method DSA-BD, such as the objective values, number of iterations, and the primal and dual relative residuals. Figure 4 plots the performance profiles of both methods based on these SDP relaxations of BIQ problems.

Note that SDPNAL takes more time than DSA-BD to find a solution with an accuracy of at least 10^{-6} in almost all of the SDP relaxations of BIQ problems tested, and it fails to compute such a solution on more than half of these instances. On the other hand, our method DSA-BD was able to find a solution with an accuracy of at least 10^{-6} for all of the SDP relaxations of BIQ problems tested.

4.4 Quadratic assignment problems

This subsection compares the performance of our method DSA-BD with that of SDPNAL on a collection of SDP relaxations of QAPs.

Given the set Π of $n \times n$ permutation matrices and $A, B \in \mathbb{R}^{n \times n}$, the quadratic assignment problem can be formulated as

$$\min\left\{\langle X, AXB\rangle : X \in \Pi\right\}.$$

For a matrix $X = [x_1, \ldots, x_n] \in \mathbb{R}^{n \times n}$, we will identify it with the n^2-vector $x = (x_1; \ldots; x_n)$. For a matrix $Y \in \mathbb{R}^{n^2 \times n^2}$, we let Y^{ij} be the $n \times n$ block corresponding to $x_i x_j^T$ in the matrix xx^T. In [14], it is shown

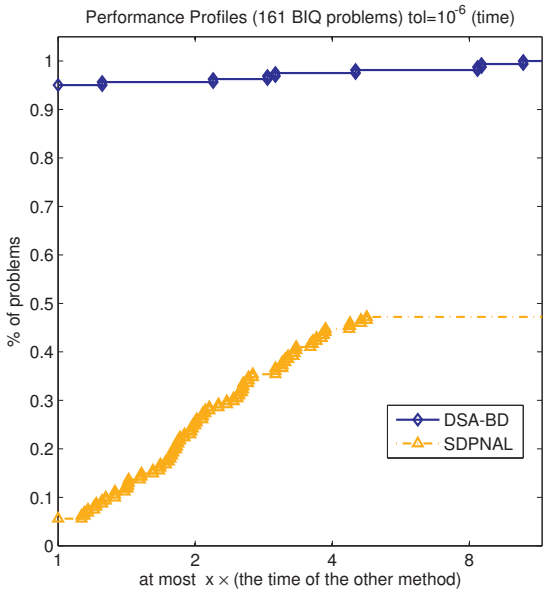

Figure 4: Performance profiles of DSA-BD and SDPNAL for solving 161 SDP relaxations of BIQ problems with accuracy $\bar{\epsilon} = 10^{-6}$.

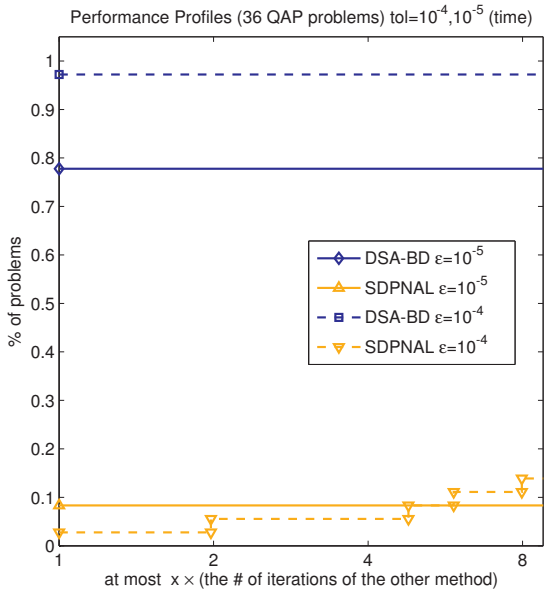

Figure 5: Performance profiles of DSA-BD and SDPNAL for solving 36 SDP relaxations of QAPs with accuracies $\bar{\epsilon} = 10^{-4}, 10^{-5}$.

that an SDP relaxation of the QAP is

$$\begin{aligned} \max \quad & \langle B \otimes A, Y \rangle \\ \text{s.t.} \quad & \sum_{i=1}^{n} Y^{ii} = I, \ \langle I, Y^{ij} \rangle = \delta_{ij} \ \forall \ 1 \le i \le j \le n, \\ & \langle E, Y^{ij} \rangle = 1 \ , \forall 1 \le i \le j \le n, \\ & Y \succeq 0, \ Y \ge 0, \end{aligned}$$

where $E \in \mathbb{R}^{n \times n}$ is the matrix of ones, and $\delta_{ij} = 1$ if $i = j$, and 0 otherwise.

Table 9 compares the two methods on a collection of SDP relaxations of QAPs using the tolerances $\bar{\epsilon} = 10^{-4}, 10^{-5}$. In this table, computational results for each instance are reported in two rows, the first one for $\bar{\epsilon} = 10^{-4}$, and the second one for $\bar{\epsilon} = 10^{-5}$. Table 10 gives more detailed computational results on these instances obtained by our method DSA-BD, such as the objective values, number of iterations, and the primal and dual relative residuals.

Figure 5 plots the performance profiles of both methods based on these SDP relaxations of QAPs. Note that SDPNAL fails to find a solution with an accuracy of at least 10^{-4} for almost all of the SDP relaxations of QAPs tested. On the other hand, our method DSA-BD was able to find a solution with an accuracy of at least 10^{-5} for almost all of the SDP relaxations of QAPs tested. Observe also that a flat line in this figure means that the corresponding method is faster for some instances, but fails to solve the rest of them. For example, SDPNAL is the fastest method for solving $\sim 8\%$ of the instances with an accuracy of $\bar{\epsilon} = 10^{-5}$, but fails to obtain a solution for the other $\sim 92\%$ of the instances.

4.5 SDPs arising from relaxation of maximum stable set problems

This subsection compares the performance of our method DSA-BD with that of BP and SDPNAL on a collection of SDPs corresponding to θ-functions and θ_+-functions of graph stable set problems.

The SDPs for θ-functions and θ_+-functions of graph stable set problems can be described as follows. Given a graph G with n nodes and an edge set E, the SDP relaxations $\theta(G)$ and $\theta_+(G)$ of the maximum

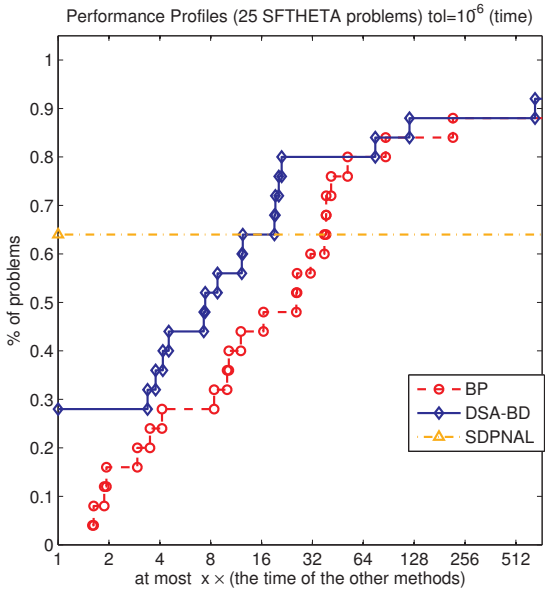

Figure 6: Performance profiles of DSA-BD, SDPNAL and BP for solving 25 $\theta(G)$ problems with accuracy $\bar{\epsilon} = 10^{-6}$.

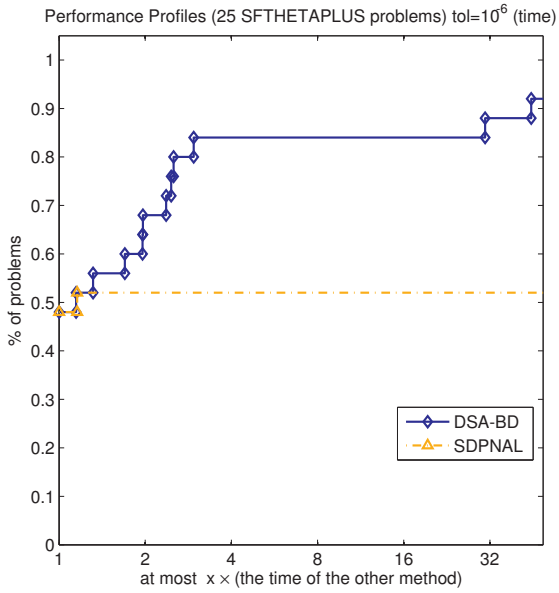

Figure 7: Performance profiles of DSA-BD and SDPNAL for solving 25 $\theta_+(G)$ problems with accuracy $\bar{\epsilon} = 10^{-6}$.

stable set problem are defined as

$$\begin{aligned} \theta(G) &:= \max\{C \bullet X,\ X_{ij} = 0,\ (i,j) \in E,\ I \bullet X = 1,\ X \succeq 0\}, \\ \theta_+(G) &:= \max\{C \bullet X,\ X_{ij} = 0,\ (i,j) \in E,\ I \bullet X = 1,\ X \succeq 0,,\ X \geq 0\}, \end{aligned}$$

where $C = ee^T$, $X \in \mathcal{S}^n$ and $e \in \mathbb{R}^n$ is the vector with all ones.

Tables 11 and 13 compare the three methods on a collection of $\theta(G)$ and $\theta_+(G)$ problems using the tolerance $\bar{\epsilon} = 10^{-6}$. Tables 12 and 14 give more detailed computational results on these instances obtained by our method DSA-BD, such as the objective values, number of iterations, and the primal and dual relative residuals. Figures 6 and 7 plot the performance profiles of the three methods based on these $\theta(G)$ and $\theta_+(G)$ problems.

Note that even though SDPNAL is faster than BP and DSA-BD on more than 60% of the $\theta(G)$ instances, BP and DSA-BD are more robust, as they are able to solve almost all of the $\theta(G)$ instances to an accuracy of at least 10^{-6}, while SDPNAL fails to do so in more than 35% of them. Also, BP takes more time than DSA-BD to find a solution with an accuracy of at least 10^{-6} in almost all of the $\theta(G)$ instances tested.

Note also that our method DSA-BD was able to find a solution with an accuracy of at least 10^{-6} for almost all of the $\theta_+(G)$ instances tested, while SDPNAL fails to do so for almost half of them.

References

[1] A. Bottcher and S. Grudsky. The norm of the product of a large matrix and a random vector. *Electronic Journal of Probability [electronic only]*, 8:Paper No. 7, 29 p., electronic only–Paper No. 7, 29 p., electronic only, 2003. URL: `http://eudml.org/doc/124759`.

[2] R. S. Burachik, A. N. Iusem, and B. F. Svaiter. Enlargement of monotone operators with applications to variational inequalities. *Set-Valued Analysis*, 5:159–180, 1997. 10.1023/A:1008615624787. URL: `http://dx.doi.org/10.1023/A:1008615624787`.

[3] R. S. Sandra Burachik and B. F. Svaiter. Maximal monotone operators, convex functions and a special family of enlargements. *Set-Valued Analysis*, 10:297–316, 2002. 10.1023/A:1020639314056. URL: http://dx.doi.org/10.1023/A:1020639314056.

[4] S. Burer, R. D. C. Monteiro, and Y. Zhang. A computational study of a gradient-based log-barrier algorithm for a class of large-scale SDPs. *Mathematical Programming*, 95:359–379, 2003. 10.1007/s10107-002-0353-7. URL: http://dx.doi.org/10.1007/s10107-002-0353-7.

[5] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011. 10.1007/s10851-010-0251-1. URL: http://dx.doi.org/10.1007/s10851-010-0251-1.

[6] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles, January 2002. URL: http://dx.doi.org/http://dx.doi.org/doi:10.1007/s101070100263, doi:http://dx.doi.org/doi:10.1007/s101070100263.

[7] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers Mathematics with Applications*, 2(1):17 – 40, 1976. URL: http://www.sciencedirect.com/science/article/pii/0898122176900031, doi:10.1016/0898-1221(76)90003-1.

[8] R. Glowinski and A. Marrocco. Sur l'approximation par éléments finis et la résolution par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *RAIRO Anal. Numér.*, 2:41 – 76, 1975.

[9] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. *SIAM J. on Optimization*, 20(1):336–356, April 2009. URL: http://dx.doi.org/10.1137/070704575, doi:10.1137/070704575.

[10] R. Monteiro and B. Svaiter. Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers. *SIAM Journal on Optimization*, 23(1):475–507, 2013. URL: http://epubs.siam.org/doi/abs/10.1137/110849468, arXiv:http://epubs.siam.org/doi/pdf/10.1137/110849468, doi:10.1137/110849468.

[11] R. D. C. Monteiro, C. Ortiz, and B. F. Svaiter. A first-order block-decomposition method for solving two-easy-block structured semidefinite programs. *Optimization-online preprint 3544*, pages 1–33, 2012. URL: http://www.optimization-online.org/DB_HTML/2012/07/3544.html.

[12] R. D. C. Monteiro and B. F. Svaiter. On the complexity of the hybrid proximal extragradient method for the iterates and the ergodic mean. *SIAM Journal on Optimization*, 20(6):2755–2787, 2010. URL: http://link.aip.org/link/?SJE/20/2755/1, doi:10.1137/090753127.

[13] R. D. C. Monteiro and B. F. Svaiter. Complexity of variants of Tseng's modified F-B splitting and korpelevich's methods for hemivariational inequalities with applications to saddle-point and convex optimization problems. *SIAM Journal on Optimization*, 21(4):1688–1720, 2011. URL: http://link.aip.org/link/?SJE/21/1688/1, doi:10.1137/100801652.

[14] J. Povh and F. Rendl. Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optimization*, 6(3):231 – 241, 2009. URL: http://www.sciencedirect.com/science/article/pii/S1572528609000036, doi:DOI:10.1016/j.disopt.2009.01.002.

[15] J. Povh, F. Rendl, and A. Wiegele. A boundary point method to solve semidefinite programs. *Computing*, 78:277–286, 2006. 10.1007/s00607-006-0182-2. URL: http://dx.doi.org/10.1007/s00607-006-0182-2.

[16] R. T. Rockafellar. On the maximal monotonicity of subdifferential mappings. *Pacific J. Math.*, 33:209–216, 1970.

[17] M. V. Solodov and B. F. Svaiter. A hybrid approximate extragradient – proximal point algorithm using the enlargement of a maximal monotone operator. *SetValued Analysis*, 7(4):323–345, 1999.

[18] M. V. Solodov and B. F. Svaiter. A hybrid projection-proximal point algorithm. *Journal of Convex Analysis*, 6(1):59–70, 1999.

[19] B. F. Svaiter. A family of enlargements of maximal monotone operators. *Set-Valued Analysis*, 8:311–328, 2000. 10.1023/A:1026555124541. URL: `http://dx.doi.org/10.1023/A:1026555124541`.

[20] K. C. Toh, M. J. Todd, and R. H. Tütüncü. Sdpt3 - a matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[21] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2:203–230, 2010. 10.1007/s12532-010-0017-1. URL: `http://dx.doi.org/10.1007/s12532-010-0017-1`.

[22] X.-Y. Zhao, D. Sun, and K.-C. Toh. A Newton-CG augmented lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4):1737–1765, 2010. URL: `http://link.aip.org/link/?SJE/20/1737/1`, `doi:10.1137/080718206`.

A Technical results

Lemma A.1 (Theorem 2.2 in [1])**.** *Given a self adjoint positive definite linear mapping $\mathcal{U} : \mathcal{Y} \to \mathcal{Y}$ and a random vector $y \in \mathcal{Y}$ uniformly distributed on a ball, we have that*

$$\mathbb{E}\left(\frac{\|\mathcal{U}^{1/2}y\|^2}{\|\mathcal{U}\|\,\|y\|^2}\right) = \frac{1}{\sigma_m}\frac{\sum\sigma_i}{m} \leq 1$$

where $\sigma_1 \leq \cdots \leq \sigma_m$ are the eigenvalues values of $\mathcal{U}$.

B Tables

Table 1: Comparison of the methods on random SDP problems

Problem		$\max\{\epsilon_P, \epsilon_D\}$			Time		
Instance	$n_s \vert m$	BP	DSA-BD	SDPNAL	BP	DSA-BD	SDPNAL
RAND-0.3k10k	300|10000	1.0 -6	9.4 -7	8.5 -7	29	27	14
RAND-0.4k15k	400|15000	1.0 -6	9.8 -7	8.0 -7	61	52	22
RAND-0.6k20k	600|20000	9.8 -7	1.0 -6	5.5 -7	144	55	36
RAND-0.5k20k	500|20000	9.8 -7	9.7 -7	7.5 -7	110	76	32
RAND-0.3k20k	300|20000	9.5 -7	9.8 -7	6.6 -7	26	18	39
RAND-0.3k25k	300|25000	9.8 -7	1.0 -6	6.8 -7	73	65	69
RAND-0.4k30k	400|30000	9.4 -7	9.7 -7	8.4 -7	37	24	54
RAND-0.5k30k	500|30000	9.5 -7	9.6 -7	7.1 -7	52	29	53
RAND-0.4k40k	400|40000	9.2 -7	9.4 -7	8.6 -6*	115	92	134*
RAND-0.5k40k	500|40000	1.0 -6	9.6 -7	9.4 -7	53	31	57
RAND-0.6k40k	600|40000	9.5 -7	9.8 -7	7.2 -7	87	41	68
RAND-0.7k50k	700|50000	9.8 -7	9.7 -7	7.7 -7	140	65	88
RAND-0.6k50k	600|50000	9.6 -7	9.6 -7	6.1 -7	82	42	122
RAND-0.5k50k	500|50000	9.8 -7	9.6 -7	9.5 -7	89	66	100
RAND-0.6k60k	600|60000	9.5 -7	9.2 -7	9.2 -7	96	57	101
RAND-0.8k70k	800|70000	1.0 -6	9.8 -7	7.2 -7	196	80	131
RAND-0.7k70k	700|70000	9.6 -7	9.9 -7	6.3 -7	126	63	127
RAND-0.7k90k	700|90000	9.8 -7	9.5 -7	8.3 -7	202	145	192
RAND-0.9k100k	900|100000	1.0 -6	1.0 -6	6.3 -7	263	102	200
RAND-0.8k100k	800|100000	9.8 -7	9.6 -7	6.8 -7	203	113	250
RAND-1.0k100k	1000|100000	9.7 -7	9.7 -7	7.1 -7	450	137	240
RAND-0.8k110k	800|110000	9.9 -7	9.8 -7	3.8 -7	252	165	265
RAND-0.9k140k	900|140000	9.3 -7	9.6 -7	9.2 -7	384	264	348
RAND-1.0k150k	1000|150000	9.9 -7	9.7 -7	8.4 -7	459	194	394

Table 2: DSA-BD results on random SDP problems

Instance	$n_s \mid m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
RAND-0.3k10k	300|10000	1.6597390 +2	1.6597440 +2	185	9.4 -7	4.0 -9	27
RAND-0.4k15k	400|15000	-6.5500010 +2	-6.5500030 +2	202	9.8 -7	5.1 -9	52
RAND-0.6k20k	600|20000	1.0452686 +3	1.0452662 +3	194	8.6 -7	1.0 -6	55
RAND-0.5k20k	500|20000	3.2800420 +2	3.2800230 +2	206	8.6 -7	9.7 -7	76
RAND-0.3k20k	300|20000	7.6135160 +2	7.6135210 +2	170	9.8 -7	9.3 -9	18
RAND-0.3k25k	300|25000	7.3838100 +1	7.3838400 +1	264	1.0 -6	5.2 -9	65
RAND-0.4k30k	400|30000	1.0721414 +3	1.0721394 +3	168	9.7 -7	8.9 -9	24
RAND-0.5k30k	500|30000	1.1076268 +3	1.1076267 +3	215	9.6 -7	5.5 -9	29
RAND-0.4k40k	400|40000	8.0576960 +2	8.0576860 +2	196	9.4 -7	6.0 -9	92
RAND-0.5k40k	500|40000	8.1661180 +2	8.1661080 +2	183	9.6 -7	5.0 -9	31
RAND-0.6k40k	600|40000	3.0661780 +2	3.0661720 +2	209	9.8 -7	1.0 -8	41
RAND-0.7k50k	700|50000	3.1320370 +2	3.1320280 +2	236	9.7 -7	4.7 -9	65
RAND-0.6k50k	600|50000	-3.8641380 +2	-3.8641360 +2	196	9.6 -7	8.2 -9	42
RAND-0.5k50k	500|50000	3.6494490 +2	3.6494520 +2	177	9.6 -7	4.9 -9	66
RAND-0.6k60k	600|60000	6.4173730 +2	6.4173680 +2	183	9.2 -7	5.6 -9	57
RAND-0.8k70k	800|70000	2.3313969 +3	2.3313957 +3	220	9.8 -7	9.4 -9	80
RAND-0.7k70k	700|70000	-3.6955780 +2	-3.6955870 +2	190	9.9 -7	6.6 -9	63
RAND-0.7k90k	700|90000	-2.6758200 +1	-2.6755500 +1	183	9.5 -7	7.9 -9	145
RAND-0.9k100k	900|100000	9.5422350 +2	9.5422290 +2	206	1.0 -6	7.9 -9	102
RAND-0.8k100k	800|100000	2.2592888 +3	2.2592881 +3	196	9.6 -7	5.8 -9	113
RAND-1.0k100k	1000|100000	3.0963606 +3	3.0963602 +3	235	9.7 -7	8.9 -9	137
RAND-0.8k110k	800|110000	1.8579204 +3	1.8579207 +3	187	9.8 -7	9.3 -9	165
RAND-0.9k140k	900|140000	2.3198295 +3	2.3198295 +3	193	9.6 -7	9.4 -9	264
RAND-1.0k150k	1000|150000	1.0528840 +3	1.0528864 +3	205	9.7 -7	4.9 -9	194

Table 3: Comparison of the methods on FAPs

	Problem	$\max\{\epsilon_P, \epsilon_D\}$		Time	
Instance	$n_s; n_l \mid m$	DSA-BD	SDPNAL	DSA-BD	SDPNAL
fap05	84;3263|3570	9.5 -6	1.7 -5*	8	18*
		9.9 -7	1.7 -5*	14	18*
fap06	93;3997|4371	9.9 -6	9.4 -6	8	10
		1.0 -6	6.7 -7	11	13
fap07	98;4139|4851	1.0 -5	9.4 -6	5	8
		1.0 -6	9.9 -7	10	13
fap08	120;6668|7260	1.0 -5	7.6 -6	9	6
		9.9 -7	9.5 -7	16	10
fap09	174;14025|15225	9.7 -6	8.8 -6	15	15
		9.9 -7	9.4 -7	19	19
fap10	183;13754|14479	9.9 -6	6.8 -6	38	17
		1.0 -6	9.3 -7	74	32
fap11	252;23275|24292	9.9 -6	6.9 -6	65	39
		1.0 -6	9.7 -7	132	78
fap12	369;24410|26462	1.0 -5	8.2 -6	103	79
		1.0 -6	2.5 -6*	259	188*
fap25	2118;311044|322924	1.0 -5	7.4 -6	7444	14517
		9.9 -7	5.1 -6*	23572	29557*
fap36	4110;1112293|1154467	1.0 -5	8.5 -6	49617	29485
		9.8 -7	5.6 -6*	148855	72063*

Table 4: DSA-BD results on FAPs

Instance	$n_s; n_l \mid m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
fap05	84;3263|3570	3.0830000 -1	3.0830000 -1	2424	9.9 -7	7.8 -7	14
fap06	93;3997|4371	4.5933000 -1	4.5934000 -1	1694	1.0 -6	5.5 -7	11
fap07	98;4139;|4851	2.1176000 +0	2.1176000 +0	1766	1.0 -6	4.2 -7	10
fap08	120;6668|7260	2.4363000 +0	2.4363000 +0	1735	9.9 -7	8.2 -7	16
fap09	174;14025;|15225	1.0797800 +1	1.0797800 +1	1128	9.9 -7	8.6 -7	19
fap10	183;13754;|14479	9.6383000 -3	9.7289000 -3	3315	1.0 -6	8.0 -7	74
fap11	252;23275|24292	2.9713000 -2	2.9856000 -2	3380	1.0 -6	8.0 -7	132
fap12	369;24410|26462	2.7317000 -1	2.7341000 -1	4004	1.0 -6	8.2 -7	259
fap25	2118;311044|322924	1.2877800 +1	1.2880000 +1	5576	9.9 -7	8.1 -7	23572
fap36	4110;1112293|1154467	6.9857000 +1	6.9860700 +1	4013	9.8 -7	9.4 -7	148855

Table 5: Comparison of the methods on BIQ problems

Problem		$\max\{\epsilon_P, \epsilon_D\}$		Time	
Instance	$n_s; n_l \| m$	DSA-BD	SDPNAL	DSA-BD	SDPNAL
be100.1	101;5151|5252	1.0 -6	7.3 -7	21	55
be100.10	101;5151|5252	1.0 -6	8.3 -7	13	41
be100.2	101;5151|5252	1.0 -6	8.3 -7	17	51
be100.3	101;5151|5252	1.0 -6	9.0 -7	18	69
be100.4	101;5151|5252	1.0 -6	6.7 -6*	19	83*
be100.5	101;5151|5252	1.0 -6	6.8 -7	17	51
be100.6	101;5151|5252	9.9 -7	8.2 -6*	14	68*
be100.7	101;5151|5252	1.0 -6	8.0 -7	17	30
be100.8	101;5151|5252	9.2 -7	8.2 -7	13	21
be100.9	101;5151|5252	1.0 -6	5.0 -7	16	58
be120.3.1	121;7381|7502	9.9 -7	7.5 -7	25	80
be120.3.10	121;7381|7502	9.9 -7	6.6 -7	21	100
be120.3.2	121;7381|7502	1.0 -6	7.7 -6*	25	88*
be120.3.3	121;7381|7502	1.0 -6	6.0 -7	19	36
be120.3.4	121;7381|7502	9.8 -7	8.3 -7	20	51
be120.3.5	121;7381|7502	1.0 -6	1.3 -5*	30	98*
be120.3.6	121;7381|7502	1.0 -6	1.5 -5*	29	93*
be120.3.7	121;7381|7502	1.0 -6	4.1 -5*	46	102*
be120.3.8	121;7381|7502	1.0 -6	3.5 -5*	40	104*
be120.3.9	121;7381|7502	1.0 -6	5.5 -5*	38	74*
be120.8.1	121;7381|7502	9.9 -7	5.9 -7	21	52
be120.8.10	121;7381|7502	1.0 -6	9.9 -7	26	65
be120.8.2	121;7381|7502	1.0 -6	2.9 -5*	32	105*
be120.8.3	121;7381|7502	1.0 -6	7.6 -7	27	49
be120.8.4	121;7381|7502	1.0 -6	9.5 -6*	25	82*
be120.8.5	121;7381|7502	1.0 -6	8.9 -6*	27	111*
be120.8.6	121;7381|7502	9.9 -7	9.8 -7	24	91
be120.8.7	121;7381|7502	9.9 -7	6.6 -7	20	45
be120.8.8	121;7381|7502	9.7 -7	8.5 -7	19	19
be120.8.9	121;7381|7502	1.0 -6	9.9 -7	19	34
be150.3.1	151;11476|11627	1.0 -6	1.5 -5*	35	146*
be150.3.10	151;11476|11627	1.0 -6	1.3 -5*	53	177*
be150.3.2	151;11476|11627	1.0 -6	2.0 -5*	45	125*
be150.3.3	151;11476|11627	1.0 -6	6.1 -7	36	133
be150.3.4	151;11476|11627	1.0 -6	6.7 -7	36	60
be150.3.5	151;11476|11627	1.0 -6	9.0 -6*	44	135*
be150.3.7	151;11476|11627	1.0 -6	5.2 -7	36	94
be150.3.8	151;11476|11627	1.0 -6	2.2 -5*	53	84*
be150.3.9	151;11476|11627	9.5 -7	7.1 -7	35	49
be150.8.1	151;11476|11627	1.0 -6	8.2 -7	32	140
be150.8.10	151;11476|11627	1.0 -6	6.2 -7	38	80
be150.8.2	151;11476|11627	1.0 -6	6.9 -7	34	63
be150.8.3	151;11476|11627	1.0 -6	1.0 -6	36	119
be150.8.4	151;11476|11627	1.0 -6	5.7 -7	40	78
be150.8.5	151;11476|11627	9.9 -7	1.1 -5*	35	146*
be150.8.6	151;11476|11627	1.0 -6	9.4 -6*	40	119*
be150.8.7	151;11476|11627	1.0 -6	2.3 -5*	45	143*
be150.8.8	151;11476|11627	9.8 -7	1.3 -5*	52	136*
be150.8.9	151;11476|11627	1.0 -6	8.4 -7	46	108
be200.3.1	201;20301|20502	1.0 -6	4.8 -6*	67	204*
be200.3.10	201;20301|20502	1.0 -6	1.9 -5*	92	242*
be200.3.2	201;20301|20502	1.0 -6	6.7 -7	73	134
be200.3.3	201;20301|20502	1.0 -6	3.6 -5*	138	271*
be200.3.4	201;20301|20502	1.0 -6	1.6 -5*	87	258*
be200.3.5	201;20301|20502	1.0 -6	1.4 -5*	111	266*
be200.3.6	201;20301|20502	1.0 -6	7.2 -7	68	182
be200.3.7	201;20301|20502	1.0 -6	6.8 -7	84	261
be200.3.8	201;20301|20502	1.0 -6	9.1 -7	76	158
be200.3.9	201;20301|20502	1.0 -6	3.2 -5*	161	213*
be200.8.1	201;20301|20502	1.0 -6	3.0 -5*	117	234*
be200.8.10	201;20301|20502	1.0 -6	1.6 -5*	77	243*
be200.8.2	201;20301|20502	1.0 -6	5.3 -7	61	74
be200.8.3	201;20301|20502	1.0 -6	9.5 -6*	94	261*
be200.8.4	201;20301|20502	1.0 -6	5.8 -7	66	133
be200.8.5	201;20301|20502	1.0 -6	1.2 -5*	81	255*
be200.8.6	201;20301|20502	1.0 -6	1.7 -5*	94	269*
be200.8.7	201;20301|20502	1.0 -6	7.0 -7	69	119
be200.8.8	201;20301|20502	1.0 -6	6.6 -7	78	153
be200.8.9	201;20301|20502	1.0 -6	1.2 -5*	87	246*
be250.1	251;31626|31877	1.0 -6	1.0 -4*	209	360*
be250.10	251;31626|31877	1.0 -6	4.8 -5*	230	387*
be250.2	251;31626|31877	1.0 -6	3.0 -5*	183	450*
be250.3	251;31626|31877	9.9 -7	4.8 -5*	161	420*
be250.4	251;31626|31877	1.0 -6	6.5 -5*	362	420*
be250.5	251;31626|31877	1.0 -6	3.3 -5*	193	358*
be250.6	251;31626|31877	1.0 -6	4.2 -5*	188	400*
be250.7	251;31626|31877	1.0 -6	3.6 -5*	207	367*
be250.8	251;31626|31877	1.0 -6	1.6 -5*	176	444*
be250.9	251;31626|31877	1.0 -6	1.4 -4*	239	359*
bqp100-1	101;5151|5252	9.9 -7	7.6 -7	15	58
bqp100-10	101;5151|5252	1.0 -6	1.7 -5*	27	72*
bqp100-2	101;5151|5252	1.0 -6	1.4 -4*	28	60*
bqp100-3	101;5151|5252	1.0 -6	8.0 -7	43	65
bqp100-4	101;5151|5252	1.0 -6	7.7 -6*	25	84*
bqp100-5	101;5151|5252	1.0 -6	4.0 -5*	26	64*
bqp100-6	101;5151|5252	1.0 -6	8.6 -7	16	74
bqp100-7	101;5151|5252	1.0 -6	8.7 -7	14	61
bqp100-8	101;5151|5252	1.0 -6	2.1 -5*	25	75*

Table 6: Comparison of the methods on BIQ problems

Problem		$\max\{\epsilon_P, \epsilon_D\}$		Time	
Instance	$n_s; n_l \| m$	DSA-BD	SDPNAL	DSA-BD	SDPNAL
bqp100-9	101;5151|5252	1.0 -6	3.7 -5*	35	76*
bqp250-1	251;31626|31877	1.0 -6	5.2 -7	182	391
bqp250-10	251;31626|31877	1.0 -6	9.5 -7	131	469
bqp250-2	251;31626|31877	1.0 -6	6.1 -5*	191	404*
bqp250-3	251;31626|31877	1.0 -6	8.0 -7	209	266
bqp250-4	251;31626|31877	1.0 -6	2.2 -5*	151	398*
bqp250-5	251;31626|31877	1.0 -6	5.2 -5*	272	418*
bqp250-6	251;31626|31877	1.0 -6	4.0 -5*	221	421*
bqp250-7	251;31626|31877	1.0 -6	5.5 -7	193	400
bqp250-8	251;31626|31877	1.0 -6	9.4 -6*	144	392*
bqp250-9	251;31626|31877	1.0 -6	1.7 -5*	195	396*
bqp50-1	51;1326|1377	1.0 -6	1.2 -5*	9	32*
bqp50-10	51;1326|1377	9.5 -7	5.8 -7	6	8
bqp50-2	51;1326|1377	9.9 -7	2.5 -5*	20	19*
bqp50-3	51;1326|1377	1.0 -6	8.2 -7	7	23
bqp50-4	51;1326|1377	1.0 -6	4.9 -5*	42	37*
bqp50-5	51;1326|1377	1.0 -6	4.4 -5*	9	19*
bqp50-7	51;1326|1377	1.0 -6	4.9 -7	7	8
bqp50-8	51;1326|1377	9.6 -7	6.9 -7	7	14
bqp50-9	51;1326|1377	1.0 -6	6.2 -7	6	8
bqp500-1	501;125751|126252	1.0 -6	4.2 -7	1230	1385
bqp500-10	501;125751|126252	1.0 -6	9.7 -7	1241	1893
bqp500-2	501;125751|126252	1.0 -6	6.2 -5*	1364	2113*
bqp500-3	501;125751|126252	1.0 -6	9.2 -7	1232	1433
bqp500-4	501;125751|126252	1.0 -6	9.6 -7	1303	1852
bqp500-5	501;125751|126252	1.0 -6	6.6 -5*	1294	2210*
bqp500-6	501;125751|126252	1.0 -6	9.3 -7	1236	2198
bqp500-7	501;125751|126252	1.0 -6	5.2 -5*	1280	1916*
bqp500-8	501;125751|126252	1.0 -6	7.7 -7	1314	1871
bqp500-9	501;125751|126252	1.0 -6	5.7 -5*	1187	2195*
gka10b	126;8001|8127	1.0 -6	2.3 -4*	22	68*
gka10d	101;5151|5252	1.0 -6	5.8 -7	17	31
gka1b	21;231|252	9.2 -7	2.9 -7	3	1
gka1c	41;861|902	1.0 -6	1.4 -5*	11	21*
gka1d	101;5151|5252	1.0 -6	1.6 -5*	33	68*
gka1e	201;20301|20502	1.0 -6	7.8 -5*	181	364*
gka1f	501;125751|126252	1.0 -6	7.4 -5*	2195	3531*
gka2b	31;496|527	1.0 -6	7.9 -5*	6	13*
gka2c	51;1326|1377	9.9 -7	5.4 -7	7	17
gka2d	101;5151|5252	1.0 -6	7.7 -6*	17	76*
gka2e	201;20301|20502	1.0 -6	9.2 -7	121	239
gka2f	501;125751|126252	1.0 -6	7.9 -5*	2488	3729*
gka3a	71;2556|2627	9.9 -7	7.9 -7	11	28
gka3b	41;861|902	9.7 -7	1.6 -7	17	2
gka3c	61;1891|1952	9.9 -7	8.8 -7	8	14
gka3d	101;5151|5252	1.0 -6	1.2 -5*	30	85*
gka3e	201;20301|20502	1.0 -6	3.6 -5*	161	305*
gka3f	501;125751|126252	1.0 -6	3.7 -5*	2222	3514*
gka4a	81;3321|3402	1.0 -6	5.6 -6*	19	72*
gka4b	51;1326|1377	9.9 -7	5.0 -7	21	2
gka4c	71;2556|2627	1.0 -6	7.6 -6*	14	64*
gka4d	101;5151|5252	1.0 -6	7.2 -6*	19	64*
gka4e	201;20301|20502	1.0 -6	1.2 -5*	187	382*
gka4f	501;125751|126252	1.0 -6	3.4 -5*	2348	3908*
gka5a	51;1326|1377	9.9 -7	6.0 -7	7	10
gka5b	61;1891|1952	9.9 -7	1.4 -7	25	3
gka5c	81;3321|3402	1.0 -6	5.6 -6*	22	36*
gka5d	101;5151|5252	1.0 -6	8.6 -7	19	59
gka5e	201;20301|20502	1.0 -6	4.0 -5*	149	320*
gka5f	501;125751|126252	1.0 -6	1.8 -5*	2210	3798*
gka6a	31;496|527	9.9 -7	9.4 -7	5	6
gka6b	71;2556|2627	9.9 -7	5.3 -8	27	6
gka6c	91;4186|4277	1.0 -6	2.7 -5*	31	69*
gka6d	101;5151|5252	9.9 -7	7.0 -6*	18	46*
gka7a	31;496|527	1.0 -6	9.1 -7	5	4
gka7b	81;3321|3402	9.9 -7	5.2 -7	3	10
gka7c	101;5151|5252	1.0 -6	5.9 -5*	26	62*
gka7d	101;5151|5252	1.0 -6	5.3 -7	16	20
gka8a	101;5151|5252	1.0 -6	4.2 -5*	33	99*
gka8b	91;4186|4277	9.9 -7	8.5 -7	49	17
gka8d	101;5151|5252	1.0 -6	7.0 -7	31	52
gka9b	101;5151|5252	9.5 -7	3.3 -7	46	21
gka9d	101;5151|5252	1.0 -6	7.0 -7	15	38

Table 7: DSA-BD results on BIQ problems

Instance	n_{s};n_{l}|m	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
be100.1	101;5151|5252	-2.0021326 +4	-2.0021284 +4	2762	4.3 -7	1.0 -6	21
be100.10	101;5151|5252	-1.6408506 +4	-1.6408506 +4	1757	9.8 -7	1.0 -6	13
be100.2	101;5151|5252	-1.7988702 +4	-1.7988693 +4	2249	1.0 -6	6.8 -7	17
be100.3	101;5151|5252	-1.8231054 +4	-1.8231144 +4	2307	4.5 -7	1.0 -6	18
be100.4	101;5151|5252	-1.9841800 +4	-1.9841754 +4	2483	1.0 -6	9.0 -7	19
be100.5	101;5151|5252	-1.6888702 +4	-1.6888698 +4	2158	9.9 -7	1.0 -6	17
be100.6	101;5151|5252	-1.8148225 +4	-1.8148220 +4	1907	9.1 -7	9.9 -7	14
be100.7	101;5151|5252	-1.9700850 +4	-1.9700852 +4	2298	1.0 -6	8.9 -7	17
be100.8	101;5151|5252	-1.9946409 +4	-1.9946451 +4	1759	6.8 -7	9.2 -7	13
be100.9	101;5151|5252	-1.4263373 +4	-1.4263380 +4	2054	1.0 -6	7.4 -7	16
be120.3.1	121;7381|7502	-1.3803561 +4	-1.3803555 +4	2338	9.9 -7	9.7 -7	25
be120.3.10	121;7381|7502	-1.2930857 +4	-1.2930864 +4	1969	9.9 -7	9.0 -7	21
be120.3.2	121;7381|7502	-1.3626630 +4	-1.3626637 +4	2407	1.0 -6	7.6 -7	25
be120.3.3	121;7381|7502	-1.2987898 +4	-1.2987905 +4	1851	1.0 -6	6.8 -7	19
be120.3.4	121;7381|7502	-1.4511234 +4	-1.4511320 +4	1867	7.4 -7	9.8 -7	20
be120.3.5	121;7381|7502	-1.1991909 +4	-1.1991906 +4	2834	9.5 -7	1.0 -6	30
be120.3.6	121;7381|7502	-1.3432058 +4	-1.3432057 +4	2717	1.0 -6	9.3 -7	29
be120.3.7	121;7381|7502	-1.4564113 +4	-1.4564109 +4	4381	8.1 -7	1.0 -6	46
be120.3.8	121;7381|7502	-1.5303024 +4	-1.5303018 +4	3775	1.0 -6	6.7 -7	40
be120.3.9	121;7381|7502	-1.1241319 +4	-1.1241320 +4	3679	1.0 -6	6.6 -7	38
be120.8.1	121;7381|7502	-2.0193959 +4	-2.0193956 +4	1968	9.9 -7	7.5 -7	21
be120.8.10	121;7381|7502	-2.0024004 +4	-2.0024041 +4	2434	1.0 -6	9.8 -7	26
be120.8.2	121;7381|7502	-2.0074132 +4	-2.0074127 +4	3013	1.0 -6	9.4 -7	32
be120.8.3	121;7381|7502	-2.0505900 +4	-2.0505900 +4	2521	1.0 -6	8.5 -7	27
be120.8.4	121;7381|7502	-2.1779801 +4	-2.1779786 +4	2300	1.0 -6	9.2 -7	25
be120.8.5	121;7381|7502	-2.1316282 +4	-2.1316282 +4	2541	1.0 -6	6.1 -7	27
be120.8.6	121;7381|7502	-1.9676959 +4	-1.9676955 +4	2256	9.9 -7	7.4 -7	24
be120.8.7	121;7381|7502	-2.3732388 +4	-2.3732401 +4	1914	9.9 -7	7.4 -7	20
be120.8.8	121;7381|7502	-2.1204746 +4	-2.1204757 +4	1788	8.4 -7	9.7 -7	19
be120.8.9	121;7381|7502	-1.9284427 +4	-1.9284426 +4	1776	7.9 -7	1.0 -6	19
be150.3.1	151;11476|11627	-1.9849179 +4	-1.9849134 +4	2240	7.1 -7	1.0 -6	35
be150.3.10	151;11476|11627	-1.9230920 +4	-1.9230899 +4	3327	1.0 -6	6.8 -7	53
be150.3.2	151;11476|11627	-1.8864852 +4	-1.8864824 +4	2860	7.9 -7	1.0 -6	45
be150.3.3	151;11476|11627	-1.8043715 +4	-1.8043695 +4	2288	1.0 -6	7.9 -7	36
be150.3.4	151;11476|11627	-2.0652667 +4	-2.0652658 +4	2264	1.0 -6	7.1 -7	36
be150.3.5	151;11476|11627	-1.7768649 +4	-1.7768626 +4	2797	1.0 -6	8.3 -7	44
be150.3.7	151;11476|11627	-1.9101308 +4	-1.9101291 +4	2268	7.2 -7	1.0 -6	36
be150.3.8	151;11476|11627	-1.9698060 +4	-1.9698030 +4	3325	6.6 -7	1.0 -6	53
be150.3.9	151;11476|11627	-1.4103367 +4	-1.4103365 +4	2207	9.5 -7	7.5 -7	35
be150.8.1	151;11476|11627	-2.9143686 +4	-2.9143682 +4	2004	1.0 -6	7.2 -7	32
be150.8.10	151;11476|11627	-3.0047974 +4	-3.0047971 +4	2400	1.0 -6	6.8 -7	38
be150.8.2	151;11476|11627	-2.8821096 +4	-2.8821093 +4	2098	1.0 -6	7.2 -7	34
be150.8.3	151;11476|11627	-3.1060369 +4	-3.1060374 +4	2243	1.0 -6	8.5 -7	36
be150.8.4	151;11476|11627	-2.8729295 +4	-2.8729280 +4	2500	1.0 -6	7.4 -7	40
be150.8.5	151;11476|11627	-2.9482071 +4	-2.9482048 +4	2206	7.9 -7	9.9 -7	35
be150.8.6	151;11476|11627	-3.1437234 +4	-3.1437227 +4	2503	1.0 -6	7.4 -7	40
be150.8.7	151;11476|11627	-3.3252109 +4	-3.3252049 +4	2827	8.7 -7	1.0 -6	45
be150.8.8	151;11476|11627	-3.1599994 +4	-3.1599918 +4	3311	9.4 -7	9.8 -7	52
be150.8.9	151;11476|11627	-2.7110727 +4	-2.7110667 +4	2897	1.0 -6	6.9 -7	46
be200.3.1	201;20301|20502	-2.7716091 +4	-2.7716020 +4	2459	1.0 -6	9.2 -7	67
be200.3.10	201;20301|20502	-2.5760689 +4	-2.5760654 +4	3443	1.0 -6	8.4 -7	92
be200.3.2	201;20301|20502	-2.6760792 +4	-2.6760748 +4	2685	9.0 -7	1.0 -6	73
be200.3.3	201;20301|20502	-2.9478639 +4	-2.9478586 +4	5007	1.0 -6	8.8 -7	138
be200.3.4	201;20301|20502	-2.9106208 +4	-2.9106169 +4	3204	1.0 -6	7.6 -7	87
be200.3.5	201;20301|20502	-2.8072990 +4	-2.8072942 +4	4143	9.7 -7	1.0 -6	111
be200.3.6	201;20301|20502	-2.7928345 +4	-2.7928321 +4	2529	1.0 -6	8.3 -7	68
be200.3.7	201;20301|20502	-3.1620504 +4	-3.1620469 +4	3093	8.4 -7	1.0 -6	84
be200.3.8	201;20301|20502	-2.9244286 +4	-2.9244248 +4	2825	1.0 -6	6.7 -7	76
be200.3.9	201;20301|20502	-2.6437048 +4	-2.6436998 +4	5932	1.0 -6	7.0 -7	161
be200.8.1	201;20301|20502	-5.0869496 +4	-5.0869383 +4	4314	8.9 -7	1.0 -6	117
be200.8.10	201;20301|20502	-4.5743067 +4	-4.5742974 +4	2849	1.0 -6	8.3 -7	77
be200.8.2	201;20301|20502	-4.4336053 +4	-4.4335972 +4	2267	9.5 -7	1.0 -6	61
be200.8.3	201;20301|20502	-4.6253974 +4	-4.6253907 +4	3507	1.0 -6	8.1 -7	94
be200.8.4	201;20301|20502	-4.6621241 +4	-4.6621195 +4	2409	1.0 -6	9.2 -7	66
be200.8.5	201;20301|20502	-4.4271235 +4	-4.4271203 +4	2955	1.0 -6	9.5 -7	81
be200.8.6	201;20301|20502	-5.1218884 +4	-5.1218773 +4	3448	1.0 -6	6.7 -7	94
be200.8.7	201;20301|20502	-4.9352768 +4	-4.9352679 +4	2581	1.0 -6	8.2 -7	69
be200.8.8	201;20301|20502	-4.7689168 +4	-4.7689152 +4	2871	1.0 -6	7.1 -7	78
be200.8.9	201;20301|20502	-4.5495602 +4	-4.5495544 +4	3203	1.0 -6	6.8 -7	87
be250.1	251;31626|31877	-2.5119464 +4	-2.5119438 +4	5051	1.0 -6	6.9 -7	209
be250.10	251;31626|31877	-2.4355024 +4	-2.4354984 +4	5625	1.0 -6	8.0 -7	230
be250.2	251;31626|31877	-2.3681493 +4	-2.3681451 +4	4519	1.0 -6	7.3 -7	183
be250.3	251;31626|31877	-2.4000002 +4	-2.3999963 +4	3908	9.3 -7	9.9 -7	161
be250.4	251;31626|31877	-2.5720317 +4	-2.5720252 +4	8912	1.0 -6	7.4 -7	362
be250.5	251;31626|31877	-2.2374710 +4	-2.2374661 +4	4736	7.9 -7	1.0 -6	193
be250.6	251;31626|31877	-2.4018844 +4	-2.4018822 +4	4561	1.0 -6	6.7 -7	188
be250.7	251;31626|31877	-2.5118959 +4	-2.5118946 +4	5022	1.0 -6	7.5 -7	207
be250.8	251;31626|31877	-2.5020398 +4	-2.5020366 +4	4358	1.0 -6	6.8 -7	176
be250.9	251;31626|31877	-2.1397059 +4	-2.1396994 +4	5774	9.3 -7	1.0 -6	239
bqp100-1	101;5151|5252	-8.3803844 +3	-8.3803874 +3	1951	9.9 -7	9.4 -7	15
bqp100-10	101;5151|5252	-1.2980272 +4	-1.2980253 +4	3594	1.0 -6	7.0 -7	27
bqp100-2	101;5151|5252	-1.1489258 +4	-1.1489238 +4	3706	1.0 -6	8.2 -7	28
bqp100-3	101;5151|5252	-1.3153184 +4	-1.3153176 +4	5757	3.9 -7	1.0 -6	43
bqp100-4	101;5151|5252	-1.0731890 +4	-1.0731883 +4	3279	1.0 -6	9.7 -7	25
bqp100-5	101;5151|5252	-9.4870275 +3	-9.4870180 +3	3452	1.0 -6	8.3 -7	26
bqp100-6	101;5151|5252	-1.0824760 +4	-1.0824748 +4	2176	1.0 -6	7.8 -7	16
bqp100-7	101;5151|5252	-1.0689155 +4	-1.0689137 +4	1817	1.0 -6	9.1 -7	14
bqp100-8	101;5151|5252	-1.1769990 +4	-1.1769980 +4	3311	1.0 -6	7.0 -7	25

Table 8: DSA-BD results on BIQ problems

Instance	$n_s; n_l \mid m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
bqp100-9	101;5151|5252	-1.1733253 +4	-1.1733258 +4	4696	2.9 -7	1.0 -6	35
bqp250-1	251;31626|31877	-4.7663112 +4	-4.7662955 +4	4527	8.8 -7	1.0 -6	182
bqp250-10	251;31626|31877	-4.3014529 +4	-4.3014392 +4	3154	1.0 -6	9.9 -7	131
bqp250-2	251;31626|31877	-4.7222380 +4	-4.7222306 +4	4677	1.0 -6	7.2 -7	191
bqp250-3	251;31626|31877	-5.1076751 +4	-5.1076552 +4	5100	5.3 -7	1.0 -6	209
bqp250-4	251;31626|31877	-4.3312555 +4	-4.3312472 +4	3666	8.9 -7	1.0 -6	151
bqp250-5	251;31626|31877	-5.0004327 +4	-5.0004210 +4	6642	9.6 -7	1.0 -6	272
bqp250-6	251;31626|31877	-4.3668862 +4	-4.3668712 +4	5322	8.2 -7	1.0 -6	221
bqp250-7	251;31626|31877	-4.8921743 +4	-4.8921564 +4	4765	1.0 -6	8.6 -7	193
bqp250-8	251;31626|31877	-3.8779549 +4	-3.8779496 +4	3473	6.8 -7	1.0 -6	144
bqp250-9	251;31626|31877	-5.1497554 +4	-5.1497497 +4	4650	1.0 -6	9.0 -7	195
bqp50-1	51;1326|1377	-2.1439177 +3	-2.1439146 +3	2795	1.0 -6	6.9 -7	9
bqp50-10	51;1326|1377	-3.6263711 +3	-3.6263710 +3	1938	9.1 -7	9.5 -7	6
bqp50-2	51;1326|1377	-3.7425229 +3	-3.7425060 +3	6076	9.9 -7	9.2 -7	20
bqp50-3	51;1326|1377	-4.6372395 +3	-4.6372264 +3	2220	4.7 -7	1.0 -6	7
bqp50-4	51;1326|1377	-3.5839746 +3	-3.5839694 +3	12719	9.7 -7	1.0 -6	42
bqp50-5	51;1326|1377	-4.0776076 +3	-4.0776061 +3	2716	1.0 -6	8.7 -7	9
bqp50-6	51;1326|1377	-3.7125866 +3	-3.6959056 +3	20000	6.4 -4	1.4 -3	69
bqp50-7	51;1326|1377	-4.6496912 +3	-4.6496901 +3	2218	4.3 -7	1.0 -6	7
bqp50-8	51;1326|1377	-4.2692350 +3	-4.2692377 +3	2073	9.6 -7	9.6 -7	7
bqp50-9	51;1326|1377	-3.9216432 +3	-3.9216406 +3	1792	1.0 -6	9.9 -7	6
bqp500-1	501;125751|126252	-1.2596423 +5	-1.2596374 +5	7353	7.0 -7	1.0 -6	1230
bqp500-10	501;125751|126252	-1.3853448 +5	-1.3853384 +5	7411	7.9 -7	1.0 -6	1241
bqp500-2	501;125751|126252	-1.3601108 +5	-1.3601080 +5	8252	1.0 -6	7.5 -7	1364
bqp500-3	501;125751|126252	-1.3845346 +5	-1.3845287 +5	7329	7.7 -7	1.0 -6	1232
bqp500-4	501;125751|126252	-1.3932842 +5	-1.3932790 +5	7910	6.8 -7	1.0 -6	1303
bqp500-5	501;125751|126252	-1.3409217 +5	-1.3409177 +5	7790	1.0 -6	8.2 -7	1294
bqp500-6	501;125751|126252	-1.3076439 +5	-1.3076411 +5	7363	7.3 -7	1.0 -6	1236
bqp500-7	501;125751|126252	-1.3149149 +5	-1.3149108 +5	7621	6.9 -7	1.0 -6	1280
bqp500-8	501;125751|126252	-1.3348988 +5	-1.3348960 +5	7911	1.0 -6	9.8 -7	1314
bqp500-9	501;125751|126252	-1.3028828 +5	-1.3028790 +5	7166	1.0 -6	9.5 -7	1187
gka10b	126;8001|8127	-1.5557650 +2	-1.5555950 +2	1822	6.8 -7	1.0 -6	22
gka10d	101;5151|5252	-2.0108576 +4	-2.0108575 +4	2008	1.0 -6	8.9 -7	17
gka1a	51;1326|1377	-3.5374674 +3	-3.5366291 +3	20000	9.4 -6	3.4 -5	74
gka1b	21;231|252	-1.3300000 +2	-1.3300000 +2	1130	8.9 -7	9.2 -7	3
gka1c	41;861|902	-5.1138290 +3	-5.1138227 +3	3866	5.4 -7	1.0 -6	11
gka1d	101;5151|5252	-6.5284315 +3	-6.5283639 +3	3937	1.0 -6	9.9 -7	33
gka1e	201;20301|20502	-1.7069817 +4	-1.7069805 +4	4971	1.0 -6	7.2 -7	181
gka1f	501;125751|126252	-6.5559070 +4	-6.5558956 +4	7631	1.0 -6	7.7 -7	2195
gka2b	31;496|527	-1.2130600 +2	-1.2129990 +2	2220	1.0 -6	7.3 -7	6
gka2c	51;1326|1377	-6.3200104 +3	-6.3199986 +3	2136	5.2 -7	9.9 -7	7
gka2d	101;5151|5252	-6.9907097 +3	-6.9907100 +3	2042	7.1 -7	1.0 -6	17
gka2e	201;20301|20502	-2.4917636 +4	-2.4917596 +4	3231	1.0 -6	7.4 -7	121
gka2f	501;125751|126252	-1.0793177 +5	-1.0793138 +5	8508	1.0 -6	9.4 -7	2488
gka3a	71;2556|2627	-6.3859990 +3	-6.3859859 +3	2202	6.4 -7	9.9 -7	11
gka3b	41;861|902	-1.1799980 +2	-1.1801510 +2	6358	9.7 -7	9.5 -7	17
gka3c	61;1891|1952	-6.8138990 +3	-6.8138886 +3	1988	3.5 -7	9.9 -7	8
gka3d	101;5151|5252	-9.7343322 +3	-9.7343347 +3	3600	1.0 -6	8.7 -7	30
gka3e	201;20301|20502	-2.6898741 +4	-2.6898705 +4	4302	1.0 -6	6.8 -7	161
gka3f	501;125751|126252	-1.5015105 +5	-1.5015061 +5	7522	9.5 -7	1.0 -6	2222
gka4a	81;3321|3402	-8.8809678 +3	-8.8809583 +3	2936	2.8 -7	1.0 -6	19
gka4b	51;1326|1377	-1.2899980 +2	-1.2902000 +2	6382	9.9 -7	9.8 -7	21
gka4c	71;2556|2627	-7.5650115 +3	-7.5650110 +3	2735	8.6 -7	1.0 -6	14
gka4d	101;5151|5252	-1.1278414 +4	-1.1278415 +4	2279	1.0 -6	7.6 -7	19
gka4e	201;20301|20502	-3.7225147 +4	-3.7225072 +4	4524	9.4 -7	1.0 -6	187
gka4f	501;125751|126252	-1.8708790 +5	-1.8708748 +5	7948	1.0 -6	7.9 -7	2348
gka5a	51;1326|1377	-5.8970505 +3	-5.8970492 +3	2050	3.2 -7	9.9 -7	7
gka5b	61;1891|1952	-1.4999980 +2	-1.5002340 +2	6393	9.8 -7	9.9 -7	25
gka5c	81;3321|3402	-7.5762319 +3	-7.5762289 +3	3702	1.0 -6	9.7 -7	22
gka5d	101;5151|5252	-1.2398864 +4	-1.2398866 +4	2234	1.0 -6	6.8 -7	19
gka5e	201;20301|20502	-3.8002313 +4	-3.8002274 +4	3945	1.0 -6	8.5 -7	149
gka5f	501;125751|126252	-2.0691429 +5	-2.0691388 +5	7445	7.0 -7	1.0 -6	2210
gka6a	31;496|527	-4.1032065 +3	-4.1032066 +3	1951	1.8 -7	9.9 -7	5
gka6b	71;2556|2627	-1.4600020 +2	-1.4597220 +2	5673	9.9 -7	9.7 -7	27
gka6c	91;4186|4277	-5.9619429 +3	-5.9619530 +3	4360	1.0 -6	8.9 -7	31
gka6d	101;5151|5252	-1.4929358 +4	-1.4929358 +4	2131	7.3 -7	9.9 -7	18
gka7a	31;496|527	-4.6386078 +3	-4.6386032 +3	2134	1.0 -6	6.4 -7	5
gka7b	81;3321|3402	-1.6035690 +2	-1.6035880 +2	493	9.2 -7	9.9 -7	3
gka7c	101;5151|5252	-7.3164493 +3	-7.3164551 +3	3144	9.0 -7	1.0 -6	26
gka7d	101;5151|5252	-1.5375790 +4	-1.5375801 +4	1850	9.5 -7	1.0 -6	16
gka8a	101;5151|5252	-1.1197217 +4	-1.1197215 +4	3678	7.2 -7	1.0 -6	33
gka8b	91;4186|4277	-1.4499980 +2	-1.4503580 +2	6797	9.9 -7	9.7 -7	49
gka8d	101;5151|5252	-1.7005361 +4	-1.7005352 +4	3643	6.9 -7	1.0 -6	31
gka9b	101;5151|5252	-1.3700010 +2	-1.3696100 +2	6155	9.5 -7	9.4 -7	46
gka9d	101;5151|5252	-1.6533903 +4	-1.6533898 +4	1776	9.6 -7	1.0 -6	15

Table 9: Comparison of the methods on QAPs

Problem		$\max\{\epsilon_P, \epsilon_D\}$		Time	
Instance	$n_s; n_l \vert m$	DSA-BD	SDPNAL	DSA-BD	SDPNAL
bur26a	676;228826|229877	1.0 -4	1.4 -4*	453	4669*
		1.0 -5	1.4 -4*	13845	4669*
bur26b	676;228826|229877	1.0 -4	1.5 -4*	600	3577*
		1.0 -5	1.5 -4*	13330	3577*
bur26c	676;228826|229877	1.0 -4	2.4 -4*	641	5790*
		1.0 -5	2.4 -4*	14684	5790*
bur26d	676;228826|229877	1.0 -4	2.5 -4*	735	4436*
		1.0 -5	2.5 -4*	16765	4436*
bur26e	676;228826|229877	1.0 -4	2.3 -4*	750	6240*
		1.0 -5	2.3 -4*	5168	6240*
bur26f	676;228826|229877	1.0 -4	2.1 -4*	856	4509*
		1.0 -5	2.1 -4*	10088	4509*
bur26g	676;228826|229877	1.0 -4	1.9 -4*	311	4115*
		1.0 -5	1.9 -4*	6814	4115*
bur26h	676;228826|229877	9.9 -5	3.4 -5	195	1552
		1.0 -5	2.5 -5*	1284	4419*
chr12a	144;10440|10672	1.1 -4*	8.0 -5	2664*	106
		1.1 -4*	4.8 -6	2664*	110
chr12b	144;10440|10672	8.9 -5	1.6 -5	27	132
		4.3 -5*	9.2 -6	2762*	144
chr12c	144;10440|10672	9.9 -5	1.5 -4*	42	284*
		9.7 -6	1.5 -4*	769	284*
chr15a	225;25425|25783	1.0 -4	2.8 -4*	252	656*
		1.0 -5	2.8 -4*	1947	656*
chr15b	225;25425|25783	1.0 -4	4.4 -4*	129	325*
		2.4 -5*	4.4 -4*	5768*	325*
chr15c	225;25425|25783	1.0 -4	4.7 -7	168	331
		6.5 -5*	4.7 -7	5534*	331
chr18a	324;52650|53161	9.9 -5	7.0 -4*	477	1475*
		1.0 -5	7.0 -4*	3488	1475*
chr18b	324;52650|53161	1.0 -4	4.0 -5	54	314
		1.0 -5	4.0 -5*	101	797*
chr20a	400;80200|80828	1.0 -4	4.1 -4*	522	2133*
		1.0 -5	4.1 -4*	1109	2133*
chr20b	400;80200|80828	1.0 -4	1.4 -4*	941	1736*
		2.6 -5*	1.4 -4*	17235*	1736*
chr20c	400;80200|80828	1.0 -4	4.3 -4*	650	1622*
		9.8 -6	4.3 -4*	5365	1622*
chr22a	484;117370|118127	1.0 -4	3.1 -4*	302	2911*
		1.2 -5*	3.1 -4*	25802*	2911*
chr22b	484;117370|118127	1.0 -4	2.2 -4*	308	2273*
		1.3 -5*	2.2 -4*	24400*	2273*
nug12	144;10440|10672	1.0 -4	1.6 -4*	31	46*
		1.0 -5	1.6 -4*	239	46*
nug14	196;19306|19619	1.0 -4	3.1 -4*	91	119*
		1.0 -5	3.1 -4*	1144	119*
nug15	225;25425|25783	1.0 -4	2.2 -4*	108	159*
		1.0 -5	2.2 -4*	1019	159*
nug16a	256;32896|33302	1.0 -4	1.8 -4*	203	336*
		1.0 -5	1.8 -4*	2514	336*
nug16b	256;32896|33302	1.0 -4	2.7 -4*	103	194*
		1.0 -5	2.7 -4*	914	194*
nug17	289;41905|42362	1.0 -4	1.6 -4*	185	272*
		1.0 -5	1.6 -4*	1742	272*
nug18	324;52650|53161	1.0 -4	2.2 -4*	206	297*
		1.0 -5	2.2 -4*	1827	297*
nug20	400;80200|80828	1.0 -4	1.8 -4*	299	471*
		1.0 -5	1.8 -4*	2466	471*
nug21	441;97461|98152	1.0 -4	2.2 -4*	420	665*
		1.0 -5	2.2 -4*	3780	665*
nug21	441;97461|98152	1.0 -4	2.2 -4*	418	663*
		1.0 -5	2.2 -4*	3814	663*
nug22	484;117370|118127	1.0 -4	2.3 -4*	659	961*
		1.0 -5	2.3 -4*	5562	961*
nug22	484;117370|118127	1.0 -4	2.3 -4*	651	1030*
		1.0 -5	2.3 -4*	5473	1030*
tai25a	625;195625|196598	1.0 -4	1.4 -4*	471	1035*
		1.0 -5	1.4 -4*	3688	1035*
tai25b	625;195625|196598	1.0 -4	5.1 -4*	1891	2438*
		1.2 -5*	5.1 -4*	28402*	2438*
tai30a	900;405450|406843	1.0 -4	1.0 -4*	1059	2452*
		1.0 -5	1.0 -4*	8223	2452*

Table 10: DSA-BD results on QAPs

Instance	$n_s; n_l \mid m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
bur26a	676;228826|229877	5.4263208 +6	5.4268401 +6	50000	5.6 -6	7.0 -6	22578
bur26b	676;228826|229877	3.8172495 +6	3.8176224 +6	50000	6.2 -6	6.1 -6	23085
bur26c	676;228826|229877	5.4270012 +6	5.4277912 +6	50000	7.9 -6	9.4 -6	22290
bur26d	676;228826|229877	3.8206639 +6	3.8211360 +6	50000	6.4 -6	8.2 -6	22612
bur26e	676;228826|229877	5.3874085 +6	5.3877112 +6	50000	3.2 -6	4.3 -6	22577
bur26f	676;228826|229877	3.7820836 +6	3.7821498 +6	47127	6.6 -7	1.0 -6	21173
bur26g	676;228826|229877	1.0117250 +7	1.0117274 +7	32438	1.0 -6	3.9 -7	14530
bur26h	676;228826|229877	7.0986748 +6	7.0987800 +6	18086	9.9 -7	7.3 -7	7645
chr12a	144;10440|10672	9.5602473 +3	9.6751178 +3	100000	2.7 -5	1.1 -4	2664
chr12b	144;10440|10672	9.7420318 +3	9.8474864 +3	100000	1.3 -5	4.3 -5	2762
chr12c	144;10440|10672	1.1156025 +4	1.1163345 +4	100000	5.4 -7	3.2 -6	2578
chr15a	225;25425|25783	9.8957979 +3	9.9101536 +3	100000	5.5 -6	4.4 -6	5152
chr15b	225;25425|25783	7.9898869 +3	7.8800531 +3	100000	3.3 -6	2.4 -5	5768
chr15c	225;25425|25783	9.5039347 +3	9.2142006 +3	100000	1.3 -5	6.5 -5	5534
chr18a	324;52650|53161	1.1097999 +4	1.1085506 +4	100000	2.2 -7	1.8 -6	10592
chr18b	324;52650|53161	1.5339652 +3	1.5340022 +3	1493	9.1 -7	1.0 -6	149
chr20a	400;80200|80828	2.1919994 +3	2.1902299 +3	100000	4.0 -7	1.8 -6	16496
chr20b	400;80200|80828	2.2980107 +3	2.2705528 +3	100000	3.9 -6	2.6 -5	17235
chr20c	400;80200|80828	1.4144561 +4	1.4156713 +4	100000	6.2 -7	1.6 -6	16440
chr22a	484;117370|118127	6.1565369 +3	6.1744853 +3	100000	6.1 -6	1.2 -5	25802
chr22b	484;117370|118127	6.1946119 +3	6.2137604 +3	100000	6.1 -6	1.3 -5	24400
nug12	144;10440|10672	5.6778580 +2	5.6788770 +2	100000	1.5 -6	1.2 -6	1776
nug14	196;19306|19619	1.0095760 +3	1.0098620 +3	100000	4.2 -6	2.8 -6	2877
nug15	225;25425|25783	1.1399857 +3	1.1402761 +3	100000	2.8 -6	2.1 -6	3692
nug16a	256;32896|33302	1.5983130 +3	1.5988500 +3	100000	5.5 -6	3.9 -6	4671
nug16b	256;32896|33302	1.2176912 +3	1.2179803 +3	100000	2.1 -6	1.6 -6	4388
nug17	289;41905|42362	1.7062557 +3	1.7066925 +3	100000	3.0 -6	2.3 -6	5733
nug18	324;52650|53161	1.8927180 +3	1.8931364 +3	100000	2.6 -6	1.8 -6	7055
nug20	400;80200|80828	2.5054307 +3	2.5058998 +3	100000	2.1 -6	1.7 -6	11029
nug21	441;97461|98152	2.3808229 +3	2.3813933 +3	100000	2.7 -6	2.0 -6	13311
nug21	441;97461|98152	2.3808229 +3	2.3813933 +3	100000	2.7 -6	2.0 -6	13435
nug22	484;117370|118127	3.5266774 +3	3.5277129 +3	100000	3.4 -6	2.4 -6	16183
nug22	484;117370|118127	3.5266774 +3	3.5277129 +3	100000	3.4 -6	2.4 -6	15980
tai25a	625;195625|196598	1.0964898 +6	1.0965735 +6	100000	1.3 -6	8.4 -7	28954
tai25b	625;195625|196598	3.3674678 +8	3.3752145 +8	100000	1.0 -5	1.2 -5	28402
tai30a	900;405450|406843	1.7066136 +6	1.7067425 +6	100000	1.2 -6	8.9 -7	63839

Table 11: Comparison of the methods on $\theta(G)$

Problem		$\max\{\epsilon_P, \epsilon_D\}$			Time		
Instance	$n_s \mid m$	BP	DSA-BD	SDPNAL	BP	DSA-BD	SDPNAL
1dc.1024	1024|24064	1.00 -6	1.00 -6	1.62 -6*	10029	5192	585*
1dc.512	512|9728	1.00 -6	1.00 -6	2.94 -6*	2258	1205	152*
1et.1024	1024|9601	1.00 -6	1.00 -6	2.51 -6*	17026	4129	862*
1et.512	512|4033	1.00 -6	9.69 -7	7.03 -7	1880	529	60
1tc.1024	1024|7937	1.09 -6*	9.95 -7	2.70 -6*	22786*	6996	1322*
1tc.512	512|3265	9.99 -7	1.00 -6	8.94 -6*	4645	1577	337*
1zc.1024	1024|16641	9.27 -7	9.66 -7	8.71 -7	911	434	35
1zc.512	512|6913	9.14 -7	9.53 -7	8.56 -7	138	62	8
2dc.1024	1024|169163	9.97 -7	9.92 -7	6.47 -6*	15086	9297	1638*
2dc.512	512|54896	9.97 -7	9.68 -7	1.51 -5*	1856	1162	448*
G43	1000|9991	8.68 -7	9.67 -7	7.51 -7	1347	665	35
G44	1000|9991	9.32 -7	9.64 -7	6.61 -7	1340	703	35
G45	1000|9991	9.98 -7	9.76 -7	5.70 -7	1356	696	36
G46	1000|9991	9.60 -7	9.67 -7	6.78 -7	1384	706	34
G47	1000|9991	8.50 -7	9.33 -7	6.36 -7	1365	651	53
G51	1000|5910	9.99 -7	9.99 -7	9.74 -7	2976	3217	852
G52	1000|5917	2.76 -6*	3.07 -6*	1.09 -5*	11794*	9157*	1591*
G53	1000|5915	1.64 -5*	3.40 -6*	9.78 -6*	12053*	8995*	1199*
G54	1000|5917	9.99 -7	9.99 -7	5.94 -7	4875	2149	476
hamming-10-2	1024|23041	8.92 -7	9.62 -7	1.56 -7	837	1219	16
hamming-9-5-6	512|53761	9.55 -7	9.74 -7	2.15 -8	250	346	3
hamming-9-8	512|2305	8.73 -7	9.85 -7	2.02 -8	583	1785	3
theta12	600|17979	9.63 -7	8.84 -7	7.28 -7	163	98	13
theta123	600|90020	9.96 -7	9.81 -7	2.68 -7	160	65	19
theta162	800|127600	9.93 -7	9.84 -7	4.03 -7	306	128	31

Table 12: DSA-BD results on $\theta(G)$

Instance	$n_s \mid m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
1dc.1024	1024|24064	-9.598590 +1	-9.598590 +1	11431	1.00 -6	5.48 -7	5192
1dc.512	512|9728	-5.303110 +1	-5.303110 +1	11084	1.00 -6	2.91 -7	1205
1et.1024	1024|9601	-1.842274 +2	-1.842274 +2	9751	1.00 -6	2.99 -7	4129
1et.512	512|4033	-1.044244 +2	-1.044244 +2	4770	5.03 -7	9.69 -7	529
1tc.1024	1024|7937	-2.063055 +2	-2.063055 +2	15907	9.70 -7	9.95 -7	6996
1tc.512	512|3265	-1.134009 +2	-1.134009 +2	15003	1.00 -6	3.38 -7	1577
1zc.1024	1024|16641	-1.286670 +2	-1.286670 +2	1059	9.66 -7	8.75 -7	434
1zc.512	512|6913	-6.875000 +1	-6.875000 +1	615	9.53 -7	4.94 -7	62
2dc.1024	1024|169163	-1.863900 +1	-1.863900 +1	18519	9.92 -7	9.79 -7	9297
2dc.512	512|54896	-1.176820 +1	-1.176820 +1	9762	9.61 -7	9.68 -7	1162
G43	1000|9991	-2.806257 +2	-2.806257 +2	1477	9.21 -7	9.67 -7	665
G44	1000|9991	-2.805837 +2	-2.805837 +2	1561	9.20 -7	9.64 -7	703
G45	1000|9991	-2.801862 +2	-2.801862 +2	1556	9.76 -7	9.74 -7	696
G46	1000|9991	-2.798379 +2	-2.798379 +2	1576	9.67 -7	8.40 -7	706
G47	1000|9991	-2.818943 +2	-2.818943 +2	1424	9.33 -7	8.33 -7	651
G51	1000|5910	-3.490001 +2	-3.490001 +2	7163	9.99 -7	2.74 -7	3217
G52	1000|5917	-3.484016 +2	-3.483917 +2	20000	3.07 -6	1.46 -6	9157
G53	1000|5915	-3.483613 +2	-3.483514 +2	20000	3.40 -6	3.39 -6	8995
G54	1000|5917	-3.410002 +2	-3.410002 +2	4732	9.99 -7	6.72 -7	2149
hamming-10-2	1024|23041	-1.024005 +2	-1.024005 +2	2913	9.32 -7	9.62 -7	1219
hamming-9-5-6	512|53761	-8.533340 +1	-8.533340 +1	3219	4.60 -7	9.74 -7	346
hamming-9-8	512|2305	-2.239997 +2	-2.239997 +2	18818	6.00 -7	9.85 -7	1785
theta12	600|17979	-9.280180 +1	-9.280180 +1	613	8.57 -7	8.84 -7	98
theta123	600|90020	-2.466880 +1	-2.466880 +1	358	9.81 -7	8.87 -7	65
theta162	800|127600	-3.700990 +1	-3.700990 +1	398	9.84 -7	8.58 -7	128

Table 13: Comparison of the methods on $\theta_+(G)$

Problem		$\max\{\epsilon_P, \epsilon_D\}$		Time	
Instance	$n_s \vert m$	DSA-BD	SDPNAL	DSA-BD	SDPNAL
1dc.1024	1024|548864	1.00 -6	3.84 -6*	2882	3563*
1dc.512	512|141056	9.99 -7	2.75 -6*	532	619*
1et.1024	1024|534401	9.98 -7	5.31 -6*	2066	5346*
1et.512	512|135361	9.98 -7	2.08 -5*	347	1145*
1tc.1024	1024|532737	1.00 -6	1.09 -4*	6816	7962*
1tc.512	512|134593	9.99 -7	6.06 -5*	644	1791*
1zc.1024	1024|541441	9.78 -7	2.70 -7	1256	740
1zc.512	512|138241	9.33 -7	9.22 -7	241	82
2dc.1024	1024|693963	1.00 -6	1.40 -4*	1741	8027*
2dc.512	512|186224	1.00 -6	1.19 -4*	564	1954*
G43	1000|510491	8.35 -7	6.92 -7	1310	668
G44	1000|510491	9.92 -7	6.35 -7	1404	594
G45	1000|510491	9.45 -7	8.36 -7	1414	563
G46	1000|510491	9.85 -7	9.13 -7	1450	588
G47	1000|510491	9.38 -7	6.86 -7	1178	602
G51	1000|506410	1.00 -6	5.89 -4*	6796	10261*
G52	1000|506417	1.00 -6	2.84 -4*	7826	10501*
G53	1000|506415	1.19 -6*	2.19 -4*	14918*	9885*
G54	1000|506417	9.99 -7	3.69 -4*	3430	7488*
hamming-10-2	1024|547841	9.87 -7	3.78 -7	2575	58
hamming-9-5-6	512|185089	9.62 -7	4.94 -7	538	17
hamming-9-8	512|133633	1.94 -2*	2.36 -7	3224*	5
theta12	600|198279	9.94 -7	6.42 -7	145	110
theta123	600|270320	9.99 -7	9.02 -7	110	96
theta162	800|448000	9.76 -7	9.34 -7	222	257

Table 14: DSA-BD results on $\theta_+(G)$

Instance	$n_s \vert m$	$\langle C, X \rangle$	$b^T y$	Iter	ϵ_P	ϵ_D	Time
1dc.1024	1024|548864	-9.555200 +1	-9.555200 +1	3982	1.00 -6	3.74 -7	2882
1dc.512	512|141056	-5.269540 +1	-5.269540 +1	3058	9.99 -7	2.99 -7	532
1et.1024	1024|534401	-1.820735 +2	-1.820735 +2	2964	9.98 -7	8.04 -7	2066
1et.512	512|135361	-1.035492 +2	-1.035492 +2	2144	9.91 -7	9.98 -7	347
1tc.1024	1024|532737	-2.042061 +2	-2.042061 +2	9192	1.00 -6	4.35 -7	6816
1tc.512	512|134593	-1.125343 +2	-1.125343 +2	3807	9.99 -7	8.42 -7	644
1zc.1024	1024|541441	-1.280007 +2	-1.280007 +2	1832	9.31 -7	9.78 -7	1256
1zc.512	512|138241	-6.800010 +1	-6.800010 +1	1462	9.33 -7	5.96 -7	241
2dc.1024	1024|693963	-1.771020 +1	-1.771020 +1	2332	8.99 -7	1.00 -6	1741
2dc.512	512|186224	-1.138370 +1	-1.138370 +1	3107	1.00 -6	3.57 -7	564
G43	1000|510491	-2.797359 +2	-2.797359 +2	1824	7.91 -7	8.35 -7	1310
G44	1000|510491	-2.797468 +2	-2.797468 +2	1963	7.82 -7	9.92 -7	1404
G45	1000|510491	-2.793186 +2	-2.793186 +2	1977	9.45 -7	8.39 -7	1414
G46	1000|510491	-2.790333 +2	-2.790333 +2	1979	7.78 -7	9.85 -7	1450
G47	1000|510491	-2.808927 +2	-2.808927 +2	1657	9.38 -7	8.44 -7	1178
G51	1000|506410	-3.490002 +2	-3.490002 +2	9269	1.00 -6	3.96 -7	6796
G52	1000|506417	-3.483868 +2	-3.483868 +2	10655	1.00 -6	6.93 -7	7826
G53	1000|506415	-3.482135 +2	-3.482114 +2	20000	8.85 -7	1.19 -6	14918
G54	1000|506417	-3.410004 +2	-3.410004 +2	4776	9.44 -7	9.99 -7	3430
hamming-10-2	1024|547841	-8.533390 +1	-8.533390 +1	3626	8.87 -7	9.87 -7	2575
hamming-9-5-6	512|185089	-5.866660 +1	-5.866660 +1	2973	3.85 -7	9.62 -7	538
hamming-9-8	512|133633	-1.644936 +2	-2.693922 +2	20000	1.94 -2	1.12 -2	3224
theta12	600|198279	-9.209090 +1	-9.209090 +1	583	9.94 -7	9.61 -7	145
theta123	600|270320	-2.449530 +1	-2.449530 +1	399	9.99 -7	9.80 -7	110
theta162	800|448000	-3.671160 +1	-3.671160 +1	445	9.76 -7	9.44 -7	222