Numerical reduction method for doubly nonnegative optimization problems

Tanaka, Mirai

Nakata, Kazuhide

Waki, Hayato

出版情報：Journal of Math-for-Industry (JMI). 5 (A), pp.41-50, 2013. Faculty of Mathematics, Kyushu University
バージョン：
権利関係：

Journal of Math-for-Industry, Vol. 5 (2013A-6), pp. 41–50

Math-for-industry
Education & Research Hub

Numerical reduction method for doubly nonnegative optimization problems

Mirai Tanaka, Kazuhide Nakata and Hayato Waki

Received on March 15, 2013 / Revised on April 2, 2013

Abstract. Doubly nonnegative (DNN) optimization is one of the most important topics in convex optimization. Some approaches which use DNN optimization are effective for some NP-hard optimization problems, *e.g.*, the maximum stable set problem and the quadratic assignment problem (QAP). However, the obtained DNN optimization problems often become highly degenerate. This implies that it is numerically difficult to find accurate optimal values and solutions of such problems even by the state-of-the-art computational technology. We propose a numerical reduction method for such DNN optimization problems, which uses a simple idea based on facial reduction algorithms. We improve the numerical tractability of the DNN optimization problems by our proposed method. We present the improvement by presenting the preliminary numerical experiments for QAP.

Keywords. doubly nonnegative optimization, semidefinite optimization, facial reduction algorithm

1. Introduction

A *doubly nonnegative (DNN)* matrix is a positive semidefinite matrix with nonnegative entries. We denote the space of $n \times n$ symmetric matrices and the cone of $n \times n$ DNN matrices by $\mathcal{S}^n$ and $\mathcal{D}^n$, respectively. In this paper, we consider the following linear optimization problem over $\mathcal{D}^n$:

$$\left| \begin{array}{ll} \text{minimize} & \boldsymbol{C} \bullet \boldsymbol{Y} \\ \text{subject to} & \boldsymbol{A}_i \bullet \boldsymbol{Y} = b_i \ (i = 1, \ldots, m), \ \boldsymbol{Y} \in \mathcal{D}^n, \end{array} \right. \tag{1}$$

where $b_1, \ldots, b_m \in \mathbb{R}$, $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_m, \boldsymbol{C} \in \mathcal{S}^n$ are given, $\boldsymbol{Y} \in \mathcal{S}^n$ is a decision variable, and $\boldsymbol{C} \bullet \boldsymbol{Y} := \mathrm{tr}(\boldsymbol{C}^{\mathrm{T}}\boldsymbol{Y})$ is the standard trace inner product on $\mathcal{S}^n$. We call (1) a *DNN optimization problem*.

In recent studies [4, 5, 13, 14, 27], some approaches which use DNN optimization problem (1) for some NP-hard optimization problems are proposed to obtain lower bounds of the optimal values of the original problems. It is mentioned in [9, 26] that such DNN approaches provide tighter lower bounds than some existing approaches, such as ones that use linear optimization (LO) and semidefinite optimization (SDO). Since DNN optimization problem (1) can be reformulated into an SDO problem equivalently, we can solve it in polynomial time to any precision, theoretically. One can obtain a lower bound by applying SDO software, *e.g.*, SDPA [25], SeDuMi [17], SDPT3 [19], and CSDP [1], to the resulting SDO problem.

One of the difficulty in such DNN approaches is that the obtained DNN optimization problems are often highly degenerate, and virtually all existing SDO solvers would encounter numerical difficulties before they could obtain accurate solutions. As a result, the optimal value and solution obtained by many SDO solvers are often inaccurate and useless. For instance, Tanaka *et al.* [18] pointed out that DNN optimization problems proposed by Burer [4, 5] are highly degenerate.

Our contribution in this paper is to propose a numerical reduction method for DNN optimization problem (1) in order to remove the degeneracy. Our proposed method can be used as a preconditioner for improving the numerical tractability of the problem. In our proposed method, we use an approach proposed by Tanaka *et al.* [18] and a simple idea based on facial reduction algorithms [2, 3, 12, 15, 16, 20, 21, 22, 27]. Although our proposed method does not remove the degeneracy completely, we may hope that the numerical stability of software for solving the resulting problems is improved in practice. We see the improvement from the numerical experiments for the quadratic assignment problem (QAP) in Section 3. The result implies that using our proposed method for preconditioning to the original DNN optimization problem (1), one can solve the resulting problem more accurately than the original problem.

The remainder of this paper is organized as follows: In Section 2, we propose a numerical reduction method for DNN optimization problem (1) and give some remarks. We present the preliminary numerical experiments for the QAP in Section 3. Section 4 is devoted to conclusion.

2. Numerical reduction of DNN optimization problem

In this section, we propose a numerical reduction method for DNN optimization problem (1). We assume that the set of $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_m$ in (1) is linearly independent without loss of generality since one can remove all linearly dependent

matrices in $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_m$. Let $\mathcal{S}^n_+$ and $\mathcal{N}^n$ be the cones of $n \times n$ positive semidefinite matrices and nonnegative matrices, respectively. We remark that we have $\mathcal{D}^n = \mathcal{S}^n_+ \cap \mathcal{N}^n$.

A matrix $\boldsymbol{Y}$ is *an interior feasible solution* of DNN optimization problem (1) if $\boldsymbol{Y}$ is feasible in (1), $\boldsymbol{Y}$ is positive definite and all elements in $\boldsymbol{Y}$ are positive. (1) is *degenerate* if it has no interior feasible solutions.

Although our proposed method can be used to make (1) less degenerate, it does not work effectively for all degenerate problem (1). In fact, we present an example where the degeneracy is not be removed by our proposed method at all in Example 1. However, our proposed method is effective for degenerate DNN optimization problems obtained by the DNN approach by Burer [4, 5]. We may hope that the numerical stability of software for such DNN optimization problems is improved. To show the effectiveness, we present a numerical result for the QAP in Section 3.

In our proposed method, we reduce DNN optimization problem (1) to a more compact form by using a nonzero solution to the following system associated with problem (1):

$$\left| \begin{array}{ll} \text{find} & \boldsymbol{y} \in \mathbb{R}^m,\ \boldsymbol{S} \in \mathcal{S}^n_+,\ \boldsymbol{T} \in \mathcal{N}^n \\ \text{such that} & \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \geq 0,\ \displaystyle\sum_{i=1}^{m} \boldsymbol{A}_i y_i + \boldsymbol{S} + \boldsymbol{T} = \boldsymbol{O}. \end{array} \right. \tag{2}$$

It is clear that $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T}) = (\boldsymbol{0}, \boldsymbol{O}, \boldsymbol{O})$ is a trivial solution of (2). The following theorem plays an essential role in our proposed method.

Theorem 1. *We have the following relationships between DNN optimization problem* (1) *and its associated system* (2)*:*

(i) (2) *has a nonzero solution such that* $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$ *if and only if* (1) *is degenerate;*

(ii) If (2) *has a nonzero solution such that* $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} > 0$*, then* (1) *is infeasible.*

Proof. First, we give a proof of the only-if part of (i). Let $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ be a nonzero solution of (2) such that $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$. For any feasible solution $\boldsymbol{Y}$ to (1), it holds that

$$\begin{aligned} \boldsymbol{Y} \bullet \boldsymbol{S} + \boldsymbol{Y} \bullet \boldsymbol{T} &= -\boldsymbol{Y} \bullet \left(\sum_{i=1}^{m} \boldsymbol{A}_i y_i \right) \\ &= -\sum_{i=1}^{m} (\boldsymbol{A}_i \bullet \boldsymbol{Y})\, y_i = -\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}. \end{aligned}$$

It follows from $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$, $\boldsymbol{Y}, \boldsymbol{S} \in \mathcal{S}^n_+$, and $\boldsymbol{Y}, \boldsymbol{T} \in \mathcal{N}^n$, $\boldsymbol{Y} \bullet \boldsymbol{S} = \boldsymbol{Y} \bullet \boldsymbol{T} = 0$. Since the $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_m$ is linearly independent, we have $\boldsymbol{S} \neq \boldsymbol{O}$ or $\boldsymbol{T} \neq \boldsymbol{O}$. In fact, if $\boldsymbol{S} = \boldsymbol{T} = \boldsymbol{O}$ then $\boldsymbol{y} = \boldsymbol{0}$ since the set of $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_m$ is linearly independent. This contradicts that $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ is nonzero. If we have $\boldsymbol{S} \neq \boldsymbol{O}$, then $\boldsymbol{Y}$ cannot be positive definite, and if we have $\boldsymbol{T} \neq \boldsymbol{O}$, then all elements of $\boldsymbol{Y}$ cannot be positive simultaneously. Thus, (1) is degenerate.

We can prove the if part of (i) by using the result in [12, (2) of Theorem 3.2] or [22, Lemma 3.2]. To apply this result, we need to convert (1) to the so-called dual standard form. This is described in [22, Section 2]. See [12, 22] for completeness of the proof of the if part of (i).

To prove (ii), we suppose the contrary that (1) has a feasible solution $\boldsymbol{Y}$. Then for any solution $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ of (2) such that $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} > 0$, we have $\boldsymbol{Y} \bullet \boldsymbol{S} + \boldsymbol{Y} \bullet \boldsymbol{T} = -\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y}$. Since $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} > 0$, $\boldsymbol{Y}, \boldsymbol{S} \in \mathcal{S}^n_+$ and $\boldsymbol{Y}, \boldsymbol{T} \in \mathcal{N}^n$, this equation implies the contradiction. □

In the remainder of this section, we discuss how to reduce DNN optimization problem (1) by using a nonzero solution of system (2) such that $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$, how to find a nonzero solution of (2), and some remarks on our proposed method.

First, we give a way to reduce a degenerate DNN optimization problem (1) by using a nonzero solution of (2) such that $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$. For a nonzero solution $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ of (2) such that $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} = 0$, it follows from (i) in Theorem 1 that (1) is equivalent to the following DNN optimization problem:

$$\left| \begin{array}{ll} \text{minimize} & \boldsymbol{C} \bullet \boldsymbol{Y} \\ \text{subject to} & \boldsymbol{A}_i \bullet \boldsymbol{Y} = b_i\ (i = 1, \ldots, m),\ \boldsymbol{Y} \in \mathcal{D}^n, \\ & \boldsymbol{Y} \bullet \boldsymbol{S} = 0, \boldsymbol{Y} \bullet \boldsymbol{T} = 0. \end{array} \right. \tag{3}$$

By using $\boldsymbol{Y} \bullet \boldsymbol{S} = 0$ and $\boldsymbol{Y} \bullet \boldsymbol{T} = 0$, we reduce (3) into a more compact form. Let $I_+ := \{(i,j) : T_{ij} > 0\}$. It follows from $\boldsymbol{Y} \bullet \boldsymbol{T} = 0$ that $\boldsymbol{Y}_{ij} = 0$ for all $(i,j) \in I_+$. Therefore, (3) is equivalent to

$$\left| \begin{array}{ll} \text{minimize} & \boldsymbol{C} \bullet \boldsymbol{Y} \\ \text{subject to} & \boldsymbol{A}_i \bullet \boldsymbol{Y} = b_i\ (i = 1, \ldots, m),\ \boldsymbol{Y} \in \mathcal{D}^n, \\ & \boldsymbol{Y} \bullet \boldsymbol{S} = 0,\ \boldsymbol{Y}_{ij} = 0\ ((i,j) \in I_+). \end{array} \right. \tag{4}$$

Furthermore, we reduce (4) by using $\boldsymbol{Y} \bullet \boldsymbol{S} = 0$. This is based on [18, 22]. Since $\boldsymbol{S}$ is positive semidefinite, we can decompose it to

$$\boldsymbol{S} = \boldsymbol{R}\boldsymbol{R}^{\mathrm{T}},$$

where $\boldsymbol{R} \in \mathbb{R}^{n \times r}$ is full column rank. Then there exists $\boldsymbol{L} \in \mathbb{R}^{n \times (n-r)}$ such that the matrix $\boldsymbol{V} := (\boldsymbol{L}, \boldsymbol{R})$ is nonsingular. Since $\boldsymbol{Y} \bullet (\boldsymbol{R}\boldsymbol{R}^{\mathrm{T}}) = 0$, $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{R} = \boldsymbol{O}$, and thus for any feasible solution $\boldsymbol{Y}$ of (3), we have

$$\boldsymbol{V}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{V} = \begin{pmatrix} \boldsymbol{L}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{L} & \boldsymbol{L}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{R} \\ \boldsymbol{R}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{L} & \boldsymbol{R}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{R} \end{pmatrix} = \begin{pmatrix} \boldsymbol{L}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{L} & \boldsymbol{L}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{R} \\ \boldsymbol{R}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{L} & \boldsymbol{O} \end{pmatrix}.$$

It follows from the positive semidefiniteness of $\boldsymbol{Y}$ that $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{R} = \boldsymbol{O}$ and $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{L} = \boldsymbol{O}$. Let $\tilde{\boldsymbol{Z}} := \boldsymbol{V}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{V}$. Then we can express $\tilde{\boldsymbol{Z}}$ by

$$\tilde{\boldsymbol{Z}} = \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix},$$

where $\boldsymbol{Z} \in \mathcal{S}^{n-r}_+$. We define $\tilde{\boldsymbol{C}}$ and $\tilde{\boldsymbol{A}}_i$ as follows:

$$\tilde{\boldsymbol{C}} := \boldsymbol{V}^{-1}\boldsymbol{C}\boldsymbol{V}^{-\mathrm{T}}, \qquad \tilde{\boldsymbol{A}}_i := \boldsymbol{V}^{-1}\boldsymbol{A}_i\boldsymbol{V}^{-T}.$$

Since we have $(\boldsymbol{V}^{-1}\boldsymbol{E}\boldsymbol{V}^{-\mathrm{T}}) \bullet (\boldsymbol{V}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{V}) = \boldsymbol{E} \bullet \boldsymbol{F}$ for all $\boldsymbol{E}, \boldsymbol{F} \in \mathcal{S}^n$, (4) is equivalent to the following optimization problem:

$$\left| \begin{array}{ll} \text{minimize} & \tilde{\boldsymbol{C}} \bullet \tilde{\boldsymbol{Z}} \\ \text{subject to} & \tilde{\boldsymbol{A}}_i \bullet \tilde{\boldsymbol{Z}} = b_i\ (i = 1, \ldots, m), \\ & \tilde{\boldsymbol{Z}} = \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix},\ \boldsymbol{Z} \in \mathcal{S}^{n-r}_+, \\ & (\boldsymbol{V}^{-\mathrm{T}}\tilde{\boldsymbol{Z}}\boldsymbol{V}^{-1})_{ij} = 0\ ((i,j) \in I_+), \\ & (\boldsymbol{V}^{-\mathrm{T}}\tilde{\boldsymbol{Z}}\boldsymbol{V}^{-1})_{ij} \geq 0\ ((i,j) \in I_0), \end{array} \right. \tag{5}$$

where $I_0 := \{(i,j) : T_{ij} = 0\}$. We write $\tilde{\boldsymbol{A}}_i$, $\tilde{\boldsymbol{C}}$ and $\boldsymbol{V}^{-1}$ as follows:

$$\tilde{\boldsymbol{A}}_i = \begin{pmatrix} \tilde{\boldsymbol{A}}_{i1} & \tilde{\boldsymbol{A}}_{i2} \\ \tilde{\boldsymbol{A}}_{i2}^{\mathrm{T}} & \tilde{\boldsymbol{A}}_{i3} \end{pmatrix}, \tilde{\boldsymbol{C}} = \begin{pmatrix} \tilde{\boldsymbol{C}}_1 & \tilde{\boldsymbol{C}}_2 \\ \tilde{\boldsymbol{C}}_2^{\mathrm{T}} & \tilde{\boldsymbol{C}}_3 \end{pmatrix}, \boldsymbol{V}^{-1} = \begin{pmatrix} \tilde{\boldsymbol{V}}_1 \\ \tilde{\boldsymbol{V}}_2 \end{pmatrix}.$$

Then we reformulate (5) as follows:

$$\left| \begin{array}{ll} \text{minimize} & \tilde{\boldsymbol{C}}_1 \bullet \boldsymbol{Z} \\ \text{subject to} & \tilde{\boldsymbol{A}}_{i1} \bullet \boldsymbol{Z} = b_i \ (i = 1, \ldots, m), \\ & \boldsymbol{Z} \in \mathcal{S}_+^{n-r}, \\ & (\tilde{\boldsymbol{V}}_1^{\mathrm{T}} \boldsymbol{Z} \tilde{\boldsymbol{V}}_1)_{ij} = 0 \ ((i,j) \in I_+), \\ & (\tilde{\boldsymbol{V}}_1^{\mathrm{T}} \boldsymbol{Z} \tilde{\boldsymbol{V}}_1)_{ij} \geq 0 \ ((i,j) \in I_0). \end{array} \right. \tag{6}$$

One can convert (6) into an SDO problem. We remark that the resulting problem becomes less degenerate than the original (1) although the degeneracy may not be removed completely. We also note that the coefficient matrices in (6) could be denser than those in (1) since $\boldsymbol{V}^{-1}$ is often dense. Furthermore, the sizes of positive semidefinite constraint and nonnegative constraints in the resulting problem (6) are $n - r$ and $|I_0|$, while they are n and $n(n+1)/2$ in (1), respectively. In this sense, (5) is smaller than (1). If $\boldsymbol{S}$ has a higher rank and/or $\boldsymbol{T}$ contains many nonzero elements, then the resulting problem (6) may be small enough to be solved by SDO solvers.

Next, we discuss how to find a nonzero solution of (2). A nonzero solution $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ of (2) plays an essential role in our proposed method. To find such a solution $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T})$ of (2), we solve the following problem:

$$\left| \begin{array}{ll} \text{maximize} & \boldsymbol{b}^{\mathrm{T}} \boldsymbol{y} \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} \boldsymbol{A}_i y_i + \boldsymbol{S} + \boldsymbol{T} = \boldsymbol{O}, \ \boldsymbol{S} \in \mathcal{S}_+^n, \ \boldsymbol{T} \in \mathcal{N}^n. \end{array} \right. \tag{7}$$

Since $(\boldsymbol{y}, \boldsymbol{S}, \boldsymbol{T}) = (\boldsymbol{0}, \boldsymbol{O}, \boldsymbol{O})$ is one of feasible solutions of (7), the optimal value is nonnegative. If the optimal value is positive, then it follows from (ii) of Theorem 1 that (1) is infeasible. Since we can reformulate (7) to an SDO problem, we may obtain a nonzero solution by applying SDO solvers. However, the size is almost the same as an SDO problem which is converted from (1), so that finding a nonzero solution to (2) is as computationally expensive as solving (1).

In our method, we decompose (7) to the following two optimization problems to find a nonzero solution of (7):

$$\left| \begin{array}{ll} \text{maximize} & \boldsymbol{b}^{\mathrm{T}} \boldsymbol{y} \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} \boldsymbol{A}_i y_i + \boldsymbol{T} = \boldsymbol{O}, \ \boldsymbol{T} \in \mathcal{N}^n, \end{array} \right. \tag{8}$$

$$\left| \begin{array}{ll} \text{maximize} & \boldsymbol{b}^{\mathrm{T}} \boldsymbol{y} \\ \text{subject to} & \displaystyle\sum_{i=1}^{m} \boldsymbol{A}_i y_i + \boldsymbol{S} = \boldsymbol{O}, \ \boldsymbol{S} \in \mathcal{S}_+^n. \end{array} \right. \tag{9}$$

We remark that LO problem (8) and SDO problem (9) are much smaller than an SDO problem equivalent to (7). We observe that for any feasible solutions $(\boldsymbol{y}_{\mathrm{psd}}, \boldsymbol{S})$ to (8) and $(\boldsymbol{y}_{\mathrm{nn}}, \boldsymbol{T})$ to (9), $(\boldsymbol{y}_{\mathrm{psd}} + \boldsymbol{y}_{\mathrm{nn}}, \boldsymbol{S}, \boldsymbol{T})$ is one of the solutions of (2). Consequently, we obtain a nonzero solution of (7) if we can find nonzero solutions of $(\boldsymbol{y}_{\mathrm{psd}}, \boldsymbol{S})$ to (8) and/or $(\boldsymbol{y}_{\mathrm{nn}}, \boldsymbol{T})$ to (9).

For LO problem (8), if a nonzero solution exists, then we can obtain the densest solution of (8) with interior-point methods since the central path of an LO problem converges to the strong complementary solution, *i.e.*, for optimal solution $(\boldsymbol{y}^*, \boldsymbol{T}^*)$ and $\boldsymbol{Y}^*$ of (8) and its dual, we have $Y_{ij}^* T_{ij}^* = 0$ and $Y_{ij}^* + T_{ij}^* > 0$ for all i, j. See *e.g.*, [24] for more details on interior-point methods for LO problems.

For SDO problem (9), we can obtain a solution $\boldsymbol{S}$ which has the largest rank in the optimal solutions with an interior-point method since interior-point methods for SDO problems also converge the analytical center of the optimal set. See *e.g.*, [8] for more details about interior-point methods for SDO problems.

Tanaka *et al.* [18] pointed out that SDO problem (9) obtained from DNN optimization problem (1) for a DNN approach proposed by [4, 5] always has an analytical feasible solution. They reduced the semidefinite constraint in (1) by using the analytical feasible solution of (9), instead of using a solution obtained by SDO software. We use the same approach to find a nonzero solution of SDO problem (9) in the numerical experiments in Section 3.

Even if (1) is degenerate, then one may not be able to construct a nonzero solution of (7) from nonzero solutions of (8) and (9). We present such an example.

Example 1. In DNN optimization problem (1), let

$$\boldsymbol{A}_{\mathrm{psd}} := \begin{pmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \boldsymbol{A}_{\mathrm{nn}} := \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix},$$

$\boldsymbol{A} := \boldsymbol{A}_{\mathrm{psd}} + \boldsymbol{A}_{\mathrm{nn}}$, $b = 0$, and $\boldsymbol{C}$ be any constant matrix. From $\pm\boldsymbol{A} \notin \mathcal{S}_+^n$ and $\pm\boldsymbol{A} \notin \mathcal{N}^n$, we can see that problems (8) and (9) have no nonzero solutions. However, (7) has the nonzero solution $(-1, \boldsymbol{A}_{\mathrm{psd}}, \boldsymbol{A}_{\mathrm{nn}})$.

Finally, we give some remarks on our proposed method.

Remark 1. This reduction can be regarded as an incomplete FRA for DNN optimization problem (1). More precisely, it corresponds to the first iteration of FRA for (1). FRA is an algorithm which reduces conic optimization problems proposed by Borwein and Wolkowicz [2, 3] and later simplified by Pataki [12]. In general, if a given conic optimization problem is feasible, it generates an equivalent one which has interior feasible solutions in finitely many iterations; Otherwise, it returns a certificate of the infeasibility. For more details about FRAs, see also [15, 16, 20, 21, 22, 27].

Remark 2. Unlike SDO problem (9), we do not have any analytical solutions to LO problem (8). Since the solution obtained by LO software contains numerical errors occurred in floating-point computation, it may not be exact. In the numerical experiments in Section 3, we use the following numerical remedy: Let $(\boldsymbol{y}^\dagger, \boldsymbol{T}^\dagger)$ and $\boldsymbol{Y}^\dagger$ be solutions of (8) and its dual obtained by LO software in which an interior-point method is implemented. Then since they are approximation of optimal solutions of (8) and its dual, they

satisfy $Y_{ij}^\dagger T_{ij}^\dagger \simeq 0$ and $Y_{ij}^\dagger, T_{ij}^\dagger > 0$ for all i, j. For such solutions, we set I_0 and I_+ as follows:

$$I_+ = \{(i,j) : Y_{ij}^\dagger < T_{ij}^\dagger\}, \quad I_0 = \{(i,j) : T_{ij}^\dagger < Y_{ij}^\dagger\}.$$

Remark 3. We can generalize our method to the following linear optimization problem over the intersection of multiple closed convex cones $\mathcal{K}_k \subseteq \mathbb{R}^n$ $(k = 1, \ldots, K)$:

$$\left| \begin{array}{ll} \text{minimize} & \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \\ \text{subject to} & \boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{x} = b_i \ (i = 1, \ldots, m) \\ & \boldsymbol{x} \in \mathcal{K}_k, (k = 1, \ldots, K), \end{array} \right. \tag{10}$$

where $\boldsymbol{c}, \boldsymbol{a}_i \in \mathbb{R}^n$. Note that this problem includes problem (1) as a special case since we have $\mathcal{D}^n = \mathcal{S}_+^n \cap \mathcal{N}^n$. For problem (10), under a mild assumption, we can formulate the following problem corresponding to (2) as follows:

$$\left| \begin{array}{rl} \text{find} & \boldsymbol{y} \in \mathbb{R}^m, \ \boldsymbol{s}_k \in \mathcal{K}_k^* \ (k = 1, \ldots, K) \\ \text{such that} & \boldsymbol{b}^{\mathrm{T}}\boldsymbol{y} \geq 0, \ \displaystyle\sum_{i=1}^m \boldsymbol{a}_i y_i + \sum_{k=1}^K \boldsymbol{s}_k = \boldsymbol{0}, \end{array} \right.$$

where $\mathcal{K}_k^*$ is the dual cone of $\mathcal{K}_k$, *i.e.*, $\mathcal{K}_k^* = \{\boldsymbol{z} : \boldsymbol{x}^{\mathrm{T}}\boldsymbol{z} \geq 0 \ (\forall \boldsymbol{x} \in \mathcal{K}_k)\}$. We can decompose this system in a way similar to our method.

3. Preliminary numerical experiments

In this section, we verify the effectiveness of our method. To this end, we solved the following three types of DNN optimization problems (14) obtained from a DNN approach by Burer [4, 5] for the quadratic assignment problem (QAP) with SDPA, which is one of the fastest and the most widely used SDO software:

`original` DNN optimization problem (1) obtained by applying a DNN approach by Burer [4, 5] (for the formulation, see (14) in Appendix A),

`only psd` the reduced problem (6) by using a solution $(\boldsymbol{y}_{\mathrm{psd}}, \boldsymbol{S}, \boldsymbol{O})$ to (2) for (14), where $(\boldsymbol{y}_{\mathrm{psd}}, \boldsymbol{S})$ is an analytical solution of (9) derived by Tanaka *et al.* [18],

`psd + nn` the reduced problem (6) by using a solution $(\boldsymbol{y}_{\mathrm{psd}} + \boldsymbol{y}_{\mathrm{nn}}, \boldsymbol{S}, \boldsymbol{T})$ to (2) for (14), where $(\boldsymbol{y}_{\mathrm{psd}}, \boldsymbol{S})$ is an analytical solution of problem (9) derived by Tanaka *et al.* [18] and $(\boldsymbol{y}_{\mathrm{nn}}, \boldsymbol{T})$ is a numerical solution of problem (8).

We used the instances of the QAP in QAPLIB [6]. We reformulate the three types of DNN optimization problems to the following SDO problems: We convert (1) into the following primal SDO problem:

$$\left| \begin{array}{ll} \text{minimize} & \begin{pmatrix} \boldsymbol{C} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \bullet \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} \\ \text{subject to} & \begin{pmatrix} \boldsymbol{A}_i & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \bullet \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} = b_i \\ & \qquad\qquad (i = 1, \ldots, m), \\ & \operatorname{vech} \boldsymbol{Y} = \boldsymbol{y}, \\ & \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} \in \mathcal{S}_+^{n+n(n+1)/2}, \end{array} \right.$$

where, for $\boldsymbol{y} \in \mathbb{R}^d$, $\operatorname{Diag} \boldsymbol{y} \in \mathcal{S}^d$ denotes the diagonal matrix obtained by $\boldsymbol{y}$ and, for $\boldsymbol{Y} \in \mathcal{S}^d$, $\operatorname{vech} \boldsymbol{Y} \in \mathbb{R}^{d(d+1)/2}$ denotes the vector obtained by the lower triangular part of $\boldsymbol{Y}$. Similarly, we convert (6) into the following primal SDO problem:

$$\left| \begin{array}{ll} \text{minimize} & \begin{pmatrix} \tilde{\boldsymbol{C}}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \bullet \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} \\ \text{subject to} & \begin{pmatrix} \tilde{\boldsymbol{A}}_{i1} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \bullet \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} = b_i \\ & \qquad\qquad (i = 1, \ldots, m), \\ & \operatorname{vech}(\tilde{\boldsymbol{V}}_1^{\mathrm{T}} \boldsymbol{Z} \tilde{\boldsymbol{V}}_1)_{I_+} = \boldsymbol{0}, \\ & \operatorname{vech}(\tilde{\boldsymbol{V}}_1^{\mathrm{T}} \boldsymbol{Z} \tilde{\boldsymbol{V}}_1)_{I_0} = \boldsymbol{y}, \\ & \begin{pmatrix} \boldsymbol{Z} & \boldsymbol{O} \\ \boldsymbol{O} & \operatorname{Diag} \boldsymbol{y} \end{pmatrix} \in \mathcal{S}_+^{n-r+|\{(i,j) \in I_0 : i \leq j\}|}, \end{array} \right.$$

where, for $\boldsymbol{Y} \in \mathcal{S}^d$ and $I \subset \{1, \ldots, d\}^2$, $\operatorname{vech} \boldsymbol{Y}_I \in \mathbb{R}^{|\{(i,j) \in I : i \leq j\}|}$ is the vector obtained by the lower triangular part of $\boldsymbol{Y}_I$. We solved them with SDPA 7.3.6 [25] linked with OpenBLAS 0.2.4 with the default parameter. We executed all experiments on a computer with Intel® Xeon® CPU E5530 with 2.40 GHz (8 CPUs) and 24 GByte memory and we used 32 threads.

We compared the accuracy of the obtained solutions and the computational time in Table 1 and all tables in Appendix B. In all the tables, `instance` and `type` denote the names of instance and the types of DNN optimization problems. `dim psd`, `dim nn`, and `dim eq` denote the size of positive semidefinite variable matrices, the number of nonnegative constraints, and the number of linear equality constraints, respectively. Note that the size of linear equation systems which SDPA solves at each iteration (Schur complement equation) is equal to `dim eq`. `err1` to `err6` denotes DIMACS errors [11], which measure the accuracy of the solutions returned by SDPA. If all values are sufficiently close to zero, then the solution is accurate and one can regard it as an optimal solution of the DNN optimization problem. `err1` and `err3` correspond to the relative error to the equality constraints of primal and dual of SDO problems, respectively. `err5` and `err6` correspond to the relative duality gap. `err2` and `err4` are omitted since they are always equal to zero in the case of SDPA. For the formal definition of them, see *e.g.*, [11]. `time` denotes how much time were spent before SDPA stopped.

We observe in all tables that for `original` and `only psd`, SDPA could not reduce DIMACS errors sufficiently for all instances. Especially, `err5` and `err6` were still large for all instances. In contrast, for `psd + nn`, SDPA returned more accurate solutions than `original` and `only psd` in all instances. We observed from the numerical experiments that the sets of coefficient matrices in the resulting SDO problems obtained by `psd + nn` and `only psd` were linearly dependent. For this, we removed all redundant constraints corresponding to linearly dependent matrices from the SDO problems, so that `dim eq` of `psd + nn` and `only psd` are smaller than that of `original`.

Table 1: Numerical results for "taiXXa" in QAPLIB [6].

instance	type	dim psd	dim nn	dim eq	err1	err3	err5	err6	time [sec]
tai5a	original	26	351	431	1.4e-07	2.9e-11	4.1e-02	4.7e-02	0.12
	only psd	17	351	377	6.4e-07	1.4e-11	5.5e-02	5.9e-02	0.11
	psd + nn	17	251	327	1.5e-08	1.6e-13	1.0e-07	9.6e-08	0.11
tai6a	original	37	703	817	2.9e-07	3.8e-11	6.4e-02	7.1e-02	0.40
	only psd	26	703	740	1.7e-06	2.3e-11	6.5e-02	7.1e-02	0.38
	psd + nn	26	523	668	3.2e-10	3.4e-13	5.3e-08	5.3e-08	0.46
tai7a	original	50	1275	1429	4.6e-07	4.6e-11	7.5e-02	8.2e-02	1.42
	only psd	37	1275	1325	1.9e-06	7.0e-11	3.7e-02	4.1e-02	1.97
	psd + nn	37	981	1227	5.4e-09	5.2e-13	1.8e-07	1.8e-07	1.62
tai8a	original	65	2145	2345	3.2e-06	1.7e-10	1.4e-01	1.7e-01	3.80
	only psd	50	2145	2210	5.5e-06	2.3e-11	6.1e-02	7.0e-02	5.06
	psd + nn	50	1697	2082	1.4e-08	2.9e-13	1.1e-07	1.1e-07	6.09
tai9a	original	82	3403	3655	1.4e-06	5.9e-11	8.9e-02	1.1e-01	11.65
	only psd	65	3403	3485	9.2e-06	3.0e-11	6.7e-02	8.3e-02	13.82
	psd + nn	65	2755	3323	4.2e-09	6.2e-13	9.2e-07	9.2e-07	19.71
tai10a	original	101	5151	5461	1.6e-06	8.4e-11	6.3e-02	7.9e-02	27.39
	only psd	82	5151	5252	8.2e-06	2.0e-11	3.7e-02	4.8e-02	38.14
	psd + nn	82	4251	5052	2.7e-08	6.0e-13	4.3e-07	4.3e-07	47.44
tai11a	original	122	7503	7877	2.8e-06	8.2e-11	6.7e-02	8.9e-02	55.47
	only psd	101	7503	7625	1.7e-05	3.0e-11	3.8e-02	4.9e-02	78.53
	psd + nn	101	6293	7383	1.2e-08	2.6e-13	2.2e-07	2.2e-07	120.36
tai12a	original	145	10585	11029	2.8e-06	6.4e-11	6.5e-02	8.2e-02	124.30
	only psd	122	10585	10730	2.5e-05	3.1e-11	1.1e-01	1.2e-01	178.99
	psd + nn	122	9001	10442	2.0e-09	1.2e-12	1.1e-06	1.1e-06	220.68
tai13a	original	170	14535	15055	4.8e-06	1.2e-10	1.2e-01	1.5e-01	223.44
	only psd	145	14535	14705	3.7e-05	5.9e-11	1.3e-01	1.5e-01	281.18
	psd + nn	145	12507	14367	6.7e-09	6.6e-13	3.3e-08	3.3e-08	628.34
tai14a	original	197	19503	20105	2.9e-06	1.1e-10	1.1e-01	1.3e-01	442.23
	only psd	170	19503	19700	5.9e-05	1.8e-11	1.9e-01	2.1e-01	506.92
	psd + nn	170	16955	19308	4.8e-09	4.5e-13	1.3e-07	1.3e-07	1169.63
tai15a	original	226	25651	26341	1.5e-05	1.1e-10	2.2e-01	2.8e-01	447.18
	only psd	197	25651	25877	4.5e-05	2.8e-10	1.7e-01	2.0e-01	1012.57
	psd + nn	197	22501	25427	4.1e-09	4.8e-13	4.0e-08	3.9e-08	2231.11
tai16a	original	257	33153	33937	8.4e-06	2.6e-10	2.6e-01	3.0e-01	879.28
	only psd	226	33153	33410	5.4e-05	8.0e-11	2.1e-01	2.4e-01	1043.34
	psd + nn	226	29313	32898	2.4e-08	2.5e-13	6.6e-08	6.8e-08	2529.61
tai17a	original	290	42195	43079	1.1e-05	1.1e-10	2.2e-01	2.7e-01	1904.90
	only psd	257	42195	42485	7.0e-05	3.9e-11	2.9e-01	3.2e-01	1889.13
	psd + nn	257	37571	41907	5.4e-08	4.1e-13	2.7e-08	2.9e-08	4906.94

4. Conclusion

In this paper, we proposed the reduction method for DNN optimization problems (1). In our method, we decompose (2) to SDO problem (9) and LO problem (8) to obtain a nonzero solution of (2). We used an analytic solution of (9) by a way proposed in Tanaka *et al.* [18].

In Section 3, we applied our reduction method to the DNN optimization problem obtained by Burer [4, 5] for the QAP. We observed that the numerical stability of SDPA is improved for DNN optimization problems obtained by applying our proposed method.

To solve the reduced problem quickly is challenging future work. In fact, even for $N = 17$ in Table 1, SDPA took longer than one hour. This result suggests that it is still difficult to obtain lower bounds for large instances, say $N \geq 30$, using SDPA and other SDO solvers.

For (8), we use a numerical solution obtained by LO software. In this sense, our proposed method is not rigorous and depends on the numerical errors in computation, *e.g.*, round-off errors. In fact, our method may find a *wrong* nonzero solution of (2) for a non-degenerate DNN optimization problem due to numerical errors, despite the fact that the system (2) associated with a non-degenerate problem does not have any nonzero solutions. To establish a rigorous reduction method for DNN optimization problem (1) is one of the most important future work.

Acknowledgements

The second author was supported by Grant-in-Aid for Young Scientists (B) 22710136. The third author was supported by Grant-in-Aid for Young Scientists (B) 22740056.

References

[1] Borchers, B and Young, J. G: Implementation of a primal-dual method for SDP on a shared memory parallel architecture, *Comput. Optim. Appl.* **37** (2007) 355–369.

[2] Borwein, J. M. and Wolkowicz, H.: Facial reduction for a cone-convex programming problem, *J. Aust. Math. Soc.* **30** (1981) 369–380.

[3] Borwein, J. M. and Wolkowicz, H.: Regularizing the abstract convex program, *J. Math. Anal. Appl.* **83** (1981) 495–530.

[4] Burer, S.: On the copositive representation of binary and continuous nonconvex quadratic programs, *Math. Program.* **120** (2009) 479–495.

[5] Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs, *Math. Program. Comput.* **2** (2010) 1–19.

[6] Burkard, R. E., Çela, E., Karisch, S. E., and Rendl, F.: QAPLIB—A quadratic assignment problem library, *J. Grobal Optim.* **10** (1997) 391–403.

[7] Dickinson, P. J. C. and Gijben, L.: On the computational complexity of membership problems for the completely positive cone and its dual, *Technical Report*, 2011.

[8] de Klerk, E.: *Aspects of Semidefinite Programming,* Kluwer Academic Publishers, Dordrecht, 2002.

[9] Ge, D. and Ye, Y.: On doubly positive semidefinite programming relaxations, *Technical Report*, 2010.

[10] Kobayashi, K., Nakata, K., and Kojima, M.: A conversion of an SDP having free variables into the standard form SDP, *Comput. Optim. Appl.* **36** (2007) 289–307.

[11] Mittelmann, H. D.: An independent benchmarking of SDP and SOCP solvers, *Math. Program.* **95** (2003) 407–430.

[12] Pataki, G.: A simple derivation of a facial reduction algorithm and extended dual systems, *Technical Report*, 2000.

[13] Povh, J. and Rendl, F.: Copositive and semidefinite relaxation of the quadratic assignment problem, *Discrete Optim.* **6** (2009) 231–241.

[14] Peng, J. and Wei, J.: Approximating k-means-type clustering via semidefinite programming, *SIAM J. Optim.* **18** (2007) 186–205.

[15] Ramana, M. V.: An exact duality theory for semidefinite programming and its complexity implications, *Math. Program.* **77** (1997) 129–162.

[16] Ramana, M. V., Tunçel L., and Wolkowicz, H.: Strong duality for semidefinite programming, *SIAM J. Optim.* **7** (1997) 641–662.

[17] Sturm J. F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, *Optim. Methods and Softw.* **11 & 12** (1999) 625–653.

[18] Tanaka, M., Nakata, K., and Waki, H.: Application of a facial reduction algorithm and an inexact primal-dual path-following method for doubly nonnegative relaxation for mixed binary nonconvex quadratic optimization problems, *Pac. J. Optim.* **8** (2012) 699–724.

[19] Toh, K. C., Todd, M. J., and Tütüncü, R. H.: On the implementation and usage of SDPT3 — a MATLAB software package for semidefinite-quadratic-linear programming, version 4.0, in *Handbook of Semidefinite, Conic and Polynomial Optimization*, M. Anjos and J. B. Lasserre (eds.), Springer, New York, 2012, pp. 715–754.

[20] Tunçel, L.: On the Slater condition for the SDP relaxations of nonconvex sets, *Oper. Res. Lett.* **29** (2001) 181–186.

[21] Tunçel, L. and Wolkowicz, H.: Strong duality and minimal representations for cone optimization, *Comput. Optim. Appl.* **53** (2012) 619–648.

[22] Waki, H. and Muramatsu, M.: Facial reduction algorithm for conic optimization problems, *J. Optim. Theory and Appl.* DOI: 10.1007/s10957-012-0219-y (2012).

[23] Waki, H., Nakata, M., and Muramatsu, M.: Strange behaviors of interior-point methods for solving semidefinite programming problems in polynomial optimization, *Computational Optimization and Applications* **53** (2012) 823–844.

[24] Wright, S. J.: *Primal-Dual Interior-Point Methods,* Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[25] Yamashita, M., Fujisawa, K., Nakata, K., Nakata, M., Fukuda, M., Kobayashi, K., and Goto, K.: A high-performance software package for semidefinite programs: SDPA 7, *Technical Report*, 2010.

[26] Yoshise, A. and Matsukawa, Y.: On optimization over the doubly nonnegative cone, in *Proc. of 2010 IEEE MSC* (2010) 13–19.

[27] Zhao, Q., Karisch, S. E., Rendl, F., and Wolkowicz, H.: Semidefinite programming relaxations for the quadratic assignment problem, *J. Comb. Optim.* **2** (1998) 71–109.

A. DNN relaxation problem for QAP

Here, we introduce a DNN optimization problem appears in a DNN approach by [4, 5, 13] for QAP. QAP is a well-known NP-hard combinatorial optimization problem, which is formulated as follows:

$$\left|\begin{array}{lll} \text{minimize} & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}z_{ik}z_{jl} & \\ \text{subject to} & \displaystyle\sum_{k=1}^{n} z_{ik} = 1 & (i=1,\ldots,n), \\ & \displaystyle\sum_{i=1}^{n} z_{ik} = 1 & (k=1,\ldots,n), \\ & z_{ik} \in \{0,1\} & (i,k=1,\ldots,n), \end{array}\right. \tag{11}$$

where f_{ij} and d_{kl} are given real numbers. Defining $\boldsymbol{Q} \in \mathcal{S}^{n^2}$ and $\boldsymbol{a}_i \in \mathbb{R}^{n^2}$ $(i = 1,\ldots,2n)$ appropriately, we can represent QAP as follows:

$$\left|\begin{array}{lll} \text{minimize} & \boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} & \\ \text{subject to} & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{x} = 1 & (p=1,\ldots,2n), \\ & x_r \in \{0,1\} & (r=1,\ldots,n^2). \end{array}\right. \tag{12}$$

Let $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}$. If $\boldsymbol{x}$ is feasible in (12), then $\boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{X}\boldsymbol{a}_q = 1$ $(p,q=1,\ldots,2n)$, $\boldsymbol{x} \geq \boldsymbol{0}$, and $\boldsymbol{X} \in \mathcal{N}^n$. Also, $x_r \in \{0,1\}$ is equivalent to $X_{rr} = x_r^2 = x_r$ for $r = 1,\ldots,n^2$. Therefore, problem (12) is equivalent to the following problem:

$$\left|\begin{array}{lll} \text{minimize} & \boldsymbol{Q}\bullet\boldsymbol{X} & \\ \text{subject to} & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{x} = 1 & (p=1,\ldots,2n), \\ & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{X}\boldsymbol{a}_q = 1 & (p,q=1,\ldots,2n), \\ & X_{rr} = x_r & (r=1,\ldots,n^2), \\ & \boldsymbol{X} - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} = \boldsymbol{O},\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{X} \in \mathcal{N}^n. & \end{array}\right. \tag{13}$$

Replacing the nonlinear constraint $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}$ by a semidefinite constraint $\boldsymbol{X} - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \in \mathcal{S}_+^n$, we obtain the following problem:

$$\left|\begin{array}{lll} \text{minimize} & \boldsymbol{Q}\bullet\boldsymbol{X} & \\ \text{subject to} & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{x} = 1 & (p=1,\ldots,2n), \\ & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{X}\boldsymbol{a}_q = 1 & (p,q=1,\ldots,2n), \\ & X_{rr} = x_r & (r=1,\ldots,n^2), \\ & \boldsymbol{X} - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \in \mathcal{S}_+^n,\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{X} \in \mathcal{N}^n. & \end{array}\right.$$

We can represent it as the following DNN optimization problem:

$$\left|\begin{array}{lll} \text{minimize} & \boldsymbol{Q}\bullet\boldsymbol{X} & \\ \text{subject to} & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{x} = 1 & (p=1,\ldots,2n), \\ & \boldsymbol{a}_p^{\mathrm{T}}\boldsymbol{X}\boldsymbol{a}_q = 1 & (p,q=1,\ldots,2n), \\ & X_{rr} = x_r & (r=1,\ldots,n^2), \\ & \begin{pmatrix} 1 & \boldsymbol{x}^{\mathrm{T}} \\ \boldsymbol{x} & \boldsymbol{X} \end{pmatrix} \in \mathcal{D}^{1+n}. & \end{array}\right. \tag{14}$$

Let f^* and μ^*_{DNN} be the optimal values of (11) and (14), respectively. Since (14) is obtained by replacing $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}$ by the weaker constraint $\boldsymbol{X} - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \in \mathcal{S}_+^n$ and (13) is equivalent to (11), we have $f^* \geq \mu^*_{\mathrm{DNN}}$. Therefore, we can obtain a lower bound μ^*_{DNN} of f^* by solving DNN optimization problem (14).

B. Detail of numerical results

Here, we show all results of the numerical experiments in Section 3. We show the results for $5 \leq N \leq 8$, $9 \leq N \leq 12$, and $15 \leq N \leq 16$ in Tables 2, 3, and 4 respectively. We can see that we obtain the best results for `psd + nn` in most instances.

Mirai Tanaka and Kazuhide Nakata
Graduate School of Decision Science and Technology, Tokyo Institute of Technology, 2-12-1-W9-60, Ookayama, Meguro-ku, Tokyo, 152-8552, Japan
E-mail: tanaka.m.aa(at)m.titech.ac.jp
nakata.k.ac(at)m.titech.ac.jp

Hayato Waki
Institute of Mathematics for Industry, Kyushu University, 744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan
E-mail: waki(at)imi.kyushu-u.ac.jp

Table 2: Results of solving instances in QAPLIB [6] ($5 \leq N \leq 8$).

instance	type	dim psd	dim nn	dim eq	err1	err3	err5	err6	time [sec]
nug5	original	26	351	431	2.1e-07	2.3e-11	2.7e-02	3.1e-02	0.14
	only psd	17	351	377	3.5e-07	1.4e-11	1.1e-02	1.3e-02	0.12
	psd + nn	17	251	327	2.1e-10	2.2e-13	3.5e-08	3.5e-08	0.12
nug6	original	37	703	817	1.3e-06	6.7e-11	7.0e-02	9.1e-02	0.39
	only psd	26	703	740	2.2e-06	7.7e-11	1.4e-02	2.2e-02	0.47
	psd + nn	26	523	668	1.4e-09	5.4e-13	1.2e-07	1.2e-07	0.46
nug7	original	50	1275	1429	1.1e-06	4.3e-11	1.1e-01	1.4e-01	1.32
	only psd	37	1275	1325	1.8e-06	3.5e-11	3.4e-02	4.1e-02	1.44
	psd + nn	37	981	1227	2.0e-09	7.1e-13	2.3e-07	2.3e-07	1.80
esc8a	original	65	2145	2345	1.4e-06	3.8e-11	2.5e-01	3.6e-01	3.81
	only psd	50	2145	2210	5.2e-06	2.4e-11	2.6e-01	3.2e-01	4.07
	psd + nn	50	1697	2082	1.4e-12	8.7e-13	1.7e-07	1.7e-07	5.90
esc8b	original	65	2145	2345	2.0e-06	5.0e-11	1.4e-01	2.3e-01	4.73
	only psd	50	2145	2210	6.3e-06	8.1e-11	5.8e-02	9.3e-02	5.73
	psd + nn	50	1697	2082	7.2e-10	1.0e-12	2.3e-07	2.3e-07	6.98
esc8c	original	65	2145	2345	1.6e-06	3.6e-11	9.2e-02	1.3e-01	4.39
	only psd	50	2145	2210	7.0e-06	4.0e-11	1.1e-01	1.6e-01	4.39
	psd + nn	50	1697	2082	5.1e-08	1.3e-12	5.7e-08	7.2e-08	8.15
esc8d	original	65	2145	2345	2.9e-06	7.8e-11	1.8e-01	3.3e-01	5.07
	only psd	50	2145	2210	1.0e-05	1.3e-10	9.6e-02	1.7e-01	6.08
	psd + nn	50	1697	2082	2.1e-07	9.7e-13	1.0e-07	2.0e-07	8.13
esc8e	original	65	2145	2345	2.5e-06	4.7e-11	4.0e-01	7.0e-01	4.09
	only psd	50	2145	2210	2.4e-05	1.7e-10	5.9e-01	7.8e-01	4.09
	psd + nn	50	1697	2082	1.4e-08	2.5e-12	3.0e-07	2.9e-07	6.40
esc8f	original	65	2145	2345	2.9e-06	7.8e-11	1.8e-01	3.3e-01	5.08
	only psd	50	2145	2210	1.0e-05	1.3e-10	9.6e-02	1.7e-01	6.08
	psd + nn	50	1697	2082	2.1e-07	9.7e-13	1.0e-07	2.0e-07	8.11
nug8	original	65	2145	2345	1.0e-06	4.7e-11	1.1e-01	1.3e-01	3.81
	only psd	50	2145	2210	3.6e-06	5.4e-11	5.0e-02	6.2e-02	5.07
	psd + nn	50	1697	2082	1.7e-10	1.3e-12	6.6e-08	6.6e-08	7.82

Table 3: Results of solving instances in QAPLIB [6] ($10 \leq N \leq 12$).

instance	type	dim psd	dim nn	dim eq	err1	err3	err5	err6	time [sec]
lipa10a	original	101	5151	5461	1.6e-06	8.6e-11	6.2e-02	7.4e-02	23.84
	only psd	82	5151	5252	7.4e-06	1.1e-10	1.8e-01	1.9e-01	29.23
	psd + nn	82	4251	5052	1.2e-09	1.6e-13	4.0e-07	4.0e-07	39.60
lipa10b	original	101	5151	5461	1.4e-06	6.5e-11	8.3e-02	9.0e-02	23.56
	only psd	82	5151	5252	2.1e-05	1.4e-11	1.2e-01	1.3e-01	24.77
	psd + nn	82	4251	5052	2.9e-09	2.8e-13	3.1e-07	3.1e-07	37.72
rou10	original	101	5151	5461	2.3e-06	1.2e-10	1.0e-01	1.2e-01	27.44
	only psd	82	5151	5252	1.6e-05	4.0e-11	6.4e-02	7.8e-02	29.16
	psd + nn	82	4251	5052	1.5e-09	3.6e-13	6.5e-07	6.5e-07	49.66
scr10	original	101	5151	5461	1.2e-06	4.9e-11	7.2e-01	8.2e-01	23.84
	only psd	82	5151	5252	8.7e-06	2.4e-10	4.1e-01	4.8e-01	26.97
	psd + nn	82	4251	5052	2.4e-08	4.5e-12	2.0e-07	1.9e-07	48.00
tai10b	original	101	5151	5461	2.5e-06	4.9e-11	7.8e-01	9.2e-01	19.98
	only psd	82	5151	5252	8.1e-06	2.5e-09	7.4e-01	8.3e-01	35.94
	psd + nn	82	4251	5052	5.6e-09	5.2e-12	4.9e-06	4.9e-06	49.58
chr12a	original	145	10585	11029	2.3e-06	7.4e-11	8.3e-01	9.6e-01	106.71
	only psd	122	10585	10730	2.0e-05	2.6e-10	7.4e-01	9.5e-01	148.28
	psd + nn	122	9001	10442	7.7e-08	7.2e-13	6.0e-06	6.0e-06	240.19
chr12b	original	145	10585	11029	2.0e-06	1.7e-10	8.2e-01	9.6e-01	106.91
	only psd	122	10585	10730	2.1e-05	2.6e-11	8.3e-01	9.4e-01	137.79
	psd + nn	122	9001	10442	8.0e-09	1.4e-12	3.8e-06	3.8e-06	229.57
chr12c	original	145	10585	11029	2.3e-06	1.1e-10	8.6e-01	9.4e-01	100.45
	only psd	122	10585	10730	5.9e-05	1.6e-10	9.5e-01	9.9e-01	116.52
	psd + nn	122	9001	10442	1.9e-09	2.7e-12	5.0e-06	5.0e-06	229.65
nug12	original	145	10585	11029	2.4e-06	5.4e-11	1.2e-01	1.5e-01	123.49
	only psd	122	10585	10730	2.1e-05	2.0e-11	1.2e-01	1.5e-01	136.61
	psd + nn	122	9001	10442	1.1e-08	1.2e-12	8.5e-08	8.5e-08	309.28
rou12	original	145	10585	11029	5.4e-06	4.0e-11	4.1e-01	4.3e-01	92.23
	only psd	122	10585	10730	2.7e-05	2.6e-11	1.1e-01	1.3e-01	147.81
	psd + nn	122	9001	10442	1.9e-09	2.0e-13	2.6e-07	2.6e-07	304.30
scr12	original	145	10585	11029	3.5e-06	9.2e-11	7.9e-01	9.9e-01	117.47
	only psd	122	10585	10730	1.7e-05	3.6e-10	6.0e-01	7.0e-01	147.89
	psd + nn	122	9001	10442	5.6e-08	1.5e-11	1.3e-06	1.2e-06	286.15
tai12b	original	145	10585	11029	1.9e-06	7.4e-11	8.2e-01	9.4e-01	125.80
	only psd	122	10585	10730	3.6e-05	3.8e-10	3.6e-01	4.0e-01	137.64
	psd + nn	122	9001	10442	9.1e-10	9.1e-13	6.5e-07	6.5e-07	210.64

Table 4: Results of solving instances in QAPLIB [6] ($15 \leq N \leq 16$).

instance	type	dim psd	dim nn	dim eq	err1	err3	err5	err6	time [sec]
chr15a	original	226	25651	26341	3.7e-06	1.7e-10	9.2e-01	1.1e+00	442.65
	only psd	197	25651	25877	5.1e-05	1.6e-10	9.3e-01	1.1e+00	1020.00
	psd + nn	197	22501	25427	2.5e-07	3.6e-12	2.3e-05	2.3e-05	2304.14
chr15b	original	226	25651	26341	4.4e-06	9.2e-11	9.2e-01	1.1e+00	428.00
	only psd	197	25651	25877	4.3e-05	2.7e-10	9.2e-01	1.1e+00	1326.92
	psd + nn	197	22501	25427	5.3e-08	1.5e-12	1.3e-05	1.3e-05	2150.70
chr15c	original	226	25651	26341	4.0e-06	2.0e-10	9.5e-01	1.0e+00	556.49
	only psd	197	25651	25877	1.0e-04	1.2e-10	9.8e-01	1.0e+00	782.64
	psd + nn	197	22501	25427	7.3e-08	6.5e-12	9.4e-06	9.5e-06	1953.25
dre15	original	226	25651	26341	8.4e-06	1.4e-10	9.8e-01	1.0e+00	448.15
	only psd	197	25651	25877	4.5e-05	9.3e-11	9.2e-01	1.1e+00	1015.31
	psd + nn	197	22501	25427	1.0e-08	1.2e-12	5.3e-06	5.3e-06	1930.59
nug15	original	226	25651	26341	5.9e-06	1.2e-10	3.0e-01	3.7e-01	446.36
	only psd	197	25651	25877	5.5e-05	9.1e-11	3.5e-01	4.1e-01	1169.53
	psd + nn	197	22501	25427	2.1e-08	1.8e-12	1.2e-06	1.2e-06	2469.80
rou15	original	226	25651	26341	3.9e-06	8.4e-11	1.3e-01	1.6e-01	559.14
	only psd	197	25651	25877	5.6e-05	7.7e-12	1.8e-01	2.1e-01	938.71
	psd + nn	197	22501	25427	2.3e-09	8.2e-13	1.3e-07	1.3e-07	2290.06
scr15	original	226	25651	26341	5.1e-06	1.1e-10	8.7e-01	1.1e+00	426.34
	only psd	197	25651	25877	1.6e-05	7.5e-10	9.3e-01	9.7e-01	1016.17
	psd + nn	197	22501	25427	2.2e-08	4.6e-12	1.6e-06	1.6e-06	2155.56
tai15b	original	226	25651	26341	4.3e-06	5.8e-11	8.3e-01	1.0e+00	482.34
	only psd	197	25651	25877	6.1e-05	4.4e-10	6.5e-01	1.2e+00	937.62
	psd + nn	197	22501	25427	3.8e-10	1.1e-11	3.5e-06	3.5e-06	2082.33
esc16a	original	257	33153	33937	1.4e-05	8.3e-11	9.3e-01	1.2e+00	812.94
	only psd	226	33153	33410	4.3e-05	6.2e-11	9.3e-01	1.0e+00	920.67
	psd + nn	226	29313	32898	1.3e-07	1.2e-12	5.5e-08	3.8e-08	4478.04
esc16b	original	257	33153	33937	7.1e-06	1.2e-10	2.1e-01	2.4e-01	879.32
	only psd	226	33153	33410	3.6e-05	1.5e-10	2.1e-01	2.4e-01	919.10
	psd + nn	226	29313	32898	4.3e-08	1.3e-11	2.3e-07	2.4e-07	2608.03
esc16c	original	257	33153	33937	7.6e-06	2.6e-10	9.3e-01	1.1e+00	839.85
	only psd	226	33153	33410	9.9e-05	1.5e-09	9.5e-01	1.1e+00	918.95
	psd + nn	226	29313	32898	5.7e-07	2.6e-12	4.5e-07	5.1e-07	2435.09
esc16d	original	257	33153	33937	9.6e-06	3.4e-10	9.3e-01	1.0e+00	911.62
	only psd	226	33153	33410	5.9e-05	2.9e-10	9.4e-01	1.1e+00	921.18
	psd + nn	226	29313	32898	1.1e-08	9.4e-12	2.9e-06	2.9e-06	2510.38
esc16e	original	257	33153	33937	8.5e-06	1.9e-10	9.3e-01	1.1e+00	806.49
	only psd	226	33153	33410	4.6e-05	9.8e-11	9.3e-01	1.0e+00	1002.45
	psd + nn	226	29313	32898	1.6e-06	4.0e-12	7.8e-07	9.6e-07	3302.79
esc16f	original	257	33153	33937	6.3e-06	1.5e-10	8.6e-01	1.0e+00	890.67
	only psd	226	33153	33410	8.9e-05	2.8e-09	9.2e-01	1.1e+00	998.40
	psd + nn	226	29313	32898	1.3e-10	6.5e-11	7.5e-09	7.5e-09	1419.82
esc16g	original	257	33153	33937	1.4e-05	1.2e-10	9.1e-01	1.1e+00	947.54
	only psd	226	33153	33410	1.1e-04	1.6e-10	9.8e-01	1.0e+00	834.14
	psd + nn	226	29313	32898	4.5e-08	1.6e-11	5.1e-08	4.2e-08	3795.53
esc16h	original	257	33153	33937	1.1e-05	1.5e-10	8.9e-01	9.7e-01	1104.83
	only psd	226	33153	33410	6.7e-05	4.8e-10	6.0e-01	7.1e-01	1001.01
	psd + nn	226	29313	32898	3.7e-07	8.3e-12	9.3e-08	1.2e-07	2128.09
esc16i	original	257	33153	33937	6.1e-06	1.4e-10	9.2e-01	1.1e+00	801.59
	only psd	226	33153	33410	4.1e-05	7.3e-11	9.2e-01	1.0e+00	1002.68
	psd + nn	226	29313	32898	6.7e-08	2.2e-12	8.0e-09	2.1e-08	3699.14
esc16j	original	257	33153	33937	4.9e-06	1.2e-10	9.5e-01	1.0e+00	942.45
	only psd	226	33153	33410	4.7e-05	5.7e-11	9.3e-01	1.0e+00	919.85
	psd + nn	226	29313	32898	6.0e-07	7.3e-12	4.9e-06	4.9e-06	3091.60