Acta Mathematicae Applicatae Sinica, English Series
Vol. 38, No. 1 (2022) 44–67
https://doi.org/10.1007/s10255-022-1061-0
http://www.ApplMath.com.cn & www.SpringerLink.com

Acta Mathematicae Applicatae Sinica, English Series

Novel Kernel Function With a Hyperbolic Barrier Term to Primal-dual Interior Point Algorithm for SDP Problems

Imene TOUIL†, Wided CHIKOUCHE

LMPA, Mohammed Seddik Ben Yahia University, Jijel 18000, Algeria (E-mail: i_touil@yahoo.fr)

Abstract In this paper, we introduce for the first time a new eligible kernel function with a hyperbolic barrier term for semidefinite programming (SDP). This add a new type of functions to the class of eligible kernel functions. We prove that the interior-point algorithm based on the new kernel function meets $\mathcal{O}(n^{\frac{3}{4}}\log\frac{n}{\varepsilon})$ iterations as the worst case complexity bound for the large-update method. This coincides with the complexity bound obtained by the first kernel function with a trigonometric barrier term proposed by El Ghami et al. in 2012, and improves with a factor $n^{\frac{1}{4}}$ the obtained iteration bound based on the classic kernel function. We present some numerical simulations which show the effectiveness of the algorithm developed in this paper.

Keywords Linear Semidefinite Programming; Primal-Dual Interior Point Methods; Hyperbolic Kernel Function; Complexity Analysis; Large and small-update methods
2000 MR Subject Classification 90C22; 90C51

1 Introduction

The study of semidefinite programming (SDP) is currently one of the most active research areas in optimization since many practical problems in operations research, stastistics, structural design and combinatorial optimization can be modeled or approximated as semidefinite programming problems.

SDP is a subfield of convex optimization concerned with the optimization of a linear objective function over the intersection of the cone of positive semidefinite matrices with an affine space.

For solving SDP problems, a basic scheme of the primal-dual interior point methods (IPMs) (proposed by Karmarkar in his famous paper for linear programming (LP)[17]) is to follow the central path to reach an optimal solution. IPMs provide a powerful tool to solve optimization problems and are now among the most effective methods from computational point of view.

Most IPMs for LP and SDP are based on the logarithmic barrier function[11, 31] with complexity $\mathcal{O}(n\log\frac{n}{\varepsilon})$ for large-update methods, where n is the size of the problem and ε is the accuracy parameter. Peng et al. in [24] were the first to propose new variants of IPMs for LP and SDP problems based on a class of self-regular kernel functions with a nonlogarithmic barrier term (see also [25, 26]). They obtained $\mathcal{O}(\sqrt{n}\log\frac{n}{\varepsilon})$ for small-update methods, and improves significantly the iteration bound for large-update methods, obtaining the currently best known iteration bound for these types of methods, namely, $\mathcal{O}(\sqrt{n}\log n\log\frac{n}{\varepsilon})$.

At present, the best known theoretical iteration bound for small-update method is better than the one for large-update method by a factor of $\log n$. However, in practice, large-update methods are much more efficient than small-update methods for which the theoretical iteration

Manuscript received December 23, 2020. Accepted on November 4, 2021.
†Corresponding author.

bound is only $\mathcal{O}(\sqrt{n}\log\frac{n}{\varepsilon})$[31, 38, 39]. In 2005, Wang et al.[36] presented a primal-dual interior point algorithm for SDP problems based on a simple non self-regular kernel function which was first introduced in [1] for LP. They established that the complexity bound for large- and small-update methods are $\mathcal{O}\ (qn\log\frac{n}{\varepsilon})$ and $\mathcal{O}(q^2\sqrt{n}\log\frac{n}{\varepsilon})$ respectively, which are as good as those in linear case. Later on, Qian et al.[30] proposed a new kernel function with simple algebraic expression and derived the iteration complexity as $\mathcal{O}(n^{\frac{3}{4}}\log\frac{n}{\varepsilon})$ for large-update method for SDP. In 2004, Bai et al[3] introduced the first kernel function with a trigonometric barrier term. The evaluation of this function has been done furthermore by El Ghami et al[12] in 2012. They established the worst case iteration complexity as $\mathcal{O}\ (n^{\frac{3}{4}}\log\frac{n}{\varepsilon})$.

Since then, a number of various kernel functions with a trigonometric barrier term has been proposed and analyzed. For these, we refer the reader for example to Kheirfam[18], Li et al.[19], Bouafia et al.[5] and Peyghami et al[28]. The authors in [5, 28] are the first to reach the best known complexity bound for large-update methods for trigonometric kernel function. In 2014, Peyghami et al [29] and Cai et al[9] proposed new kernel functions with trigonometric-logarithmic barrier term for LP. In the same year, Peyghami et al[27] presented an other kernel function with an exponential-trigonometric barrier term which has $\mathcal{O}(\sqrt{n}(\log n)^2\log\frac{n}{\varepsilon})$ complexity bounds for large-update method. The complexity bound derived in [29], namely $\mathcal{O}(n^{\frac{2}{3}}\log\frac{n}{\varepsilon})$, improves the complexity bounds obtained in [9, 12, 27]. Note that the kernel function proposed in [9] yields the same complexity derived in [12].

We end this overview by mentioning the works of Bai et al.[2, 4], Zhang[40], Bouafia et al.[6] and Fathi-Hafshejani et al.[13], where the authors proposed kernel functions with exponential barrier term.

In view of the precedent, most of the kernel functions used in IPMs can be classified into four types : logarithmic, simple algebraic, exponential and trigonometric. The remaining kernel functions are just a binary combination of these types, see [7, 8, 14, 20, 22, 35] for recent proposed kernel functions.

In this paper, we propose a new kernel function (neither logarithmic, neither simple algebraic, neither trigonometric nor exponential). This function has the following simple expression

$$\psi(t)=\frac{t^2-1}{2\sinh^2(1)}+\coth(t)-\coth(1),\qquad \forall\, t>0. \tag{1.1}$$

To our knowledge it's the first kernel function with a hyperbolic barrier term. Furthermore, this kernel function is a member of the class of eligible kernel functions introduced by Bai et al. in [3].

We show that the iteration bounds for large- and small-update methods are $\mathcal{O}\big(n^{\frac{3}{4}}\log\frac{n}{\varepsilon}\big)$, and $\mathcal{O}\left(\sqrt{n}\log\frac{n}{\varepsilon}\right)$, respectively. This coincides with the complexity bound obtained by the first kernel function with a trigonometric barrier term[12].

The reminder of this paper is organized as follows. In Section 2, we first describe special matrix functions used in later sections. Then, we briefly recall the basic concepts of IPMs for SDP, such as central path, Nesterov and Todd (NT) search directions, etc. In Section 3, we present the properties of our hyperbolic kernel function $\psi(t)$ and study the matrix functions $\psi(V)$ and $\Psi(V)$. In section 4, we describe a primal-dual interior point algorithm based on $\Psi(V)$ for SDP. We analyze the algorithm to derive the complexity bound with large and small-update methods in Section 5. In Section 6, we present numerical tests on two different examples to illustrate the effectiveness of the proposed algorithm and compare the results with other available kernel functions. In the final section we conclude with some remarks.

Let us finish this introduction with some notations used in the whole paper: The set of all $(n\times n)$ matrices with real entries is denoted by $\mathbb{R}^{n\times n}$. Given $M\in\mathbb{R}^{n\times n}$, M^T denotes the transpose of M. $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$) denotes the cone of positive semidefinite (positive definite)

matrices in the real space of $(n \times n)$ symmetrical matrices $\mathbb{S}^n$. The scalar product of two matrices A and B in $\mathbb{S}^n$ is the trace of their product i.e., $\langle A, B\rangle = \mathrm{tr}(AB) = \sum_{i,j=1}^{n} a_{ij}b_{ij}$. For an $M \in \mathbb{S}^n$, we denote by $\lambda_i(M),\ i = 1, \cdots, n$, $\lambda_{\max}(M)$ and $\lambda_{\min}(M)$ the eigenvalues, the largest eigenvalue and the smallest eigenvalue of the matrix M. The diagonal matrix with diagonal entries $\lambda_i(M),\ i = 1, \cdots, n$, is denoted by $\mathrm{diag}(\lambda_1(\mathrm{M}), \cdots, \lambda_\mathrm{n}(\mathrm{M}))$. The Frobenius norm of $M \in \mathbb{S}^n$ is $\|M\| = \langle M, M\rangle^{\frac{1}{2}} = \sqrt{\sum_{i=1}^{n} \lambda_i^2(M)}$. For $M \in \mathbb{S}^n$, $M \succeq 0$ ($M \succ 0$) means that M belongs to $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$), respectively. For any $M \in \succ 0$, the expression $M^{\frac{1}{2}}$ (or $\sqrt{M}$) denotes its symmetric square root. Finally, if $f(x) \geq 0$ is a real valued function of a real nonnegative variable, the notation $f(x) = \mathcal{O}(x)$ means that $f(x) \leq Cx$ for some positive constant C and $f(x) = \Theta(x)$ means that $C_1 x \leq f(x) \leq C_2 x$ for two positive constants C_1 and C_2.

2 Preliminaries

2.1 Matrices and Matrix Functions

To introduce matrix functions which will be useful for designing primal-dual interior point algorithm, first of all, let us recall some known facts from linear algebra. For more details we refer to the books[16, 21, 37].

The next definition shows how a matrix function can be obtained from $\psi(t)$, where $\psi(t)$ is a real function on $[0, +\infty)$.

Definition 2.1. *Let $V \succ 0$ and*

$$V = Q^T \mathrm{diag}(\lambda_1(\mathrm{V}), \lambda_2(\mathrm{V}), \cdots, \lambda_\mathrm{n}(\mathrm{V}))\mathrm{Q},$$

where Q is an orthonormal matrix ($Q^T = Q^{-1}$) that diagonalizes V.
Then, for given real function $\psi(t)$ defined for $t > 0$, the matrix function $\psi(V) : \mathbb{S}^n_{++} \to \mathbb{S}^n$ is defined by

$$\psi(V) = Q^T \mathrm{diag}(\psi(\lambda_1(\mathrm{V})), \psi(\lambda_2(\mathrm{V})), \cdots, \psi(\lambda_\mathrm{n}(\mathrm{V})))\mathrm{Q}. \tag{2.1}$$

Furthermore, the proximity function (measure) for SDP is defined from $\mathbb{S}^n_{++}$ to $\mathbb{R}_+$ as follows

$$\Phi\left(X, S; \mu\right) := \Psi\left(V\right) := \mathrm{tr}\left(\psi\left(\mathrm{V}\right)\right) = \sum_{\mathrm{i}=1}^{\mathrm{n}} \psi\left(\lambda_\mathrm{i}\left(\mathrm{V}\right)\right). \tag{2.2}$$

Remark 2.2. If the function $\psi(t)$ is differentiable on the interval $(0, +\infty)$ such that $\psi'(t) > 0$, $\forall t > 0$, we can obtain the matrix function $\psi'(V)$ by replacing $\psi\left(\lambda_i(V)\right)$ in (2.1) by $\psi'(\lambda_i(V))$ for each i.

Definition 2.3. *A matrix $M(t)$ is said to be a matrix of functions if each entry of $M(t)$ is a function of t, i.e., $M(t) = [M_{ij}(t)]$.*

The usual concepts of continuity, differentiability and integrability can be naturally extended to matrices of functions, by interpreting them entry-wise. Then, it can easily be understood that

$$\frac{d}{dt}(M(t)) := \Big[\frac{d}{dt}M_{ij}(t)\Big] = M'(t).$$

Suppose that the matrix-valued functions $M(t), N(t)$ are differentiable with respect to t. Then we have

$$\frac{d}{dt}(\mathrm{tr}(M(t))) = \mathrm{tr}\Big(\frac{d}{dt}M(t)\Big) = \mathrm{tr}(M'(t)), \tag{2.3}$$

$$\frac{d}{dt}\mathrm{tr}(\psi(M(t))) = \mathrm{tr}[\psi'(M(t))(M'(t))], \tag{2.4}$$

$$\frac{d}{dt}(M(t)N(t)) = \Big(\frac{d}{dt}M(t)\Big)N(t) + M(t)\Big(\frac{d}{dt}N(t)\Big) = M'(t)N(t) + M(t)N'(t). \tag{2.5}$$

2.2 Primal-Dual Central Trajectory Methods

We consider the standard form for SDP problems

$$(P)\begin{cases} \min\langle C, X\rangle \\ \mathcal{A}X = b, \\ X \in \mathbb{S}^n_+, \end{cases}$$

where $b \in \mathbb{R}^m$, $C \in \mathbb{S}^n$ and $\mathcal{A}$ is a linear operator from $\mathbb{S}^n$ to $\mathbb{R}^m$ defined by

$$\mathcal{A}X = \left(\langle A_1, X\rangle, \langle A_2, X\rangle, \cdots, \langle A_m, X\rangle\right)^T, \tag{2.6}$$

with A_i, $i = 1, \cdots, m$, in $\mathbb{S}^n$.

Its dual is written as follows

$$(D)\begin{cases} \max b^T y \\ \mathcal{A}^* y + S \quad = C, \\ S \in \mathbb{S}^n_+, \end{cases}$$

where $\mathcal{A}^*$ is the adjoint of $\mathcal{A}$ defined from $\mathbb{R}^m$ to $\mathbb{S}^n$ by $\mathcal{A}^* y = \sum_{i=1}^{m} y_i A_i$.

The sets of strictly feasible solutions of (P) and (D) are

$$\overset{0}{\mathcal{F}}(P) = \left\{X \in \mathbb{S}^n_{++} : \mathcal{A}X = b\right\},$$

$$\overset{0}{\mathcal{F}}(D) = \left\{(y, S) \in \mathbb{R}^m \times \mathbb{S}^n_{++} : \mathcal{A}^* y + S = C\right\},$$

respectively. Throughout this paper, we assume that both problems (P) and (D) satisfy the interior point condition (IPC), i.e., $\overset{0}{\mathcal{F}}(P) \times \overset{0}{\mathcal{F}}(D) \neq \emptyset$. Under this assumption, it is well known that (P) and (D) have optimal solutions $\overline{X}$ and $(\overline{S}, \overline{y})$ such that $\langle C, \overline{X}\rangle = b^T\overline{y}$ (that is, the optimal values of (P) and (D) are equal). This last condition, called strong duality, can be alternatively expressed as $\langle \overline{X}, \overline{S}\rangle = 0$ or $\overline{X}\,\overline{S} = 0$. For simplicity, we also assume that the matrices A_i, $i = 1, \cdots, m$, are linearly independent.

If the IPC holds, then the optimality conditions for (P) and (D) can be written as follows

$$\begin{cases} \mathcal{A}X = b, X \succ 0, \\ \mathcal{A}^* y + S = C, S \succ 0, \\ XS = 0. \end{cases} \tag{2.7}$$

The core idea of primal-dual IPMs is to replace the last equation in (2.7), the so-called complementarity condition by the parameterized equation $XS = \mu I$, where $\mu > 0$. Thus the modified system is given by

$$\begin{cases} \mathcal{A}X = b, X \succ 0, \\ \mathcal{A}^* y + S = C, S \succ 0, \\ XS = \mu I, \quad \mu > 0. \end{cases} \tag{2.8}$$

Under the IPC, the system (2.8) has a unique solution denoted by $(X(\mu), y(\mu), S(\mu))$, for each $\mu > 0$. The set of all solutions $(X(\mu), y(\mu), S(\mu))$, with $\mu > 0$ is known as the *central path* or *central trajectory*.

Since for $X, S \in \mathbb{S}^n$, the product XS is generally not in $\mathbb{S}^n$, so, the left-hand side of (2.8) is a map from $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{S}^n$ to $\mathbb{R}^{n\times n} \times \mathbb{R}^m \times \mathbb{S}^n$. Thus, the system (2.8) is not a square system when X and S are restricted to $\mathbb{S}^n$, which is needed for applying Newton's method. A remedy for this is to make the system (2.8) square by modifying the left-hand side to a map from $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{S}^n$ to itself. To this end, we use the so-called *similar symmetrization* operator $H_P : \mathbb{R}^{n\times n} \to \mathbb{S}^n$ introduced by Zhang[41] defined as

$$H_P(M) = \frac{1}{2}[PMP^{-1} + (PMP^{-1})^T], \qquad \forall\, M \in \mathbb{R}^{n\times n},$$

$P \in \mathbb{R}^{n\times n}$ is some nonsingular matrix. Zhang also observed that if P is invertible and M is similar to a (symmetric) positive definite matrix, then

$$H_P(M) = \mu I \Leftrightarrow M = \mu I.$$

Thus, for any given nonsingular matrix, system (2.8) is equivalent to

$$\begin{cases} \mathcal{A}X = b, X \succ 0, \\ \mathcal{A}^* y + S = C, S \succ 0, \\ H_P(XS) = \mu I, \quad \mu > 0. \end{cases} \tag{2.9}$$

Applying Newton method to system (2.9) generates the following linear system

$$\begin{cases} \mathcal{A}\Delta X = 0, \\ \mathcal{A}^* \Delta y + \Delta S = 0, \\ H_P(X\Delta S + \Delta X S) = \mu I - H_P(XS), \end{cases} \tag{2.10}$$

where $(\Delta X, \Delta y, \Delta S) = \Delta W \in \mathbb{S}^n \times \mathbb{R}^m \times \mathbb{S}^n$ is the search direction and $\mu = \frac{\langle X,S\rangle}{n}$ is the normalized duality gap corresponding to (X, y, S).

The search direction obtained by the above system is called Monteiro-Zhang (MZ) family. Several search directions are presented in the literature [15, 23, 33, 34]. One among the most popular, the Nesterov and Todd (NT) direction. One important reason for this choice is that the NT scaling technique transfers the primal variable X and the dual variable S into the same space: the so-called V-space. Let us define the matrix

$$P = \left[X^{\frac{1}{2}}(X^{\frac{1}{2}} S X^{\frac{1}{2}})^{-\frac{1}{2}} X^{\frac{1}{2}}\right]^{-\frac{1}{2}} = \left[S^{-\frac{1}{2}}\left(S^{\frac{1}{2}} X S^{\frac{1}{2}}\right)^{\frac{1}{2}} S^{-\frac{1}{2}}\right]^{-\frac{1}{2}}.$$

The matrix P can be used to scale X and S to the same matrix V because

$$V = \frac{1}{\sqrt{\mu}} PXP = \frac{1}{\sqrt{\mu}} P^{-1} S P^{-1}. \tag{2.11}$$

Note that the matrices P and V are symmetric and positive definite. Furthermore, we have

$$V^2 = \left(\frac{PXP}{\sqrt{\mu}}\right)\left(\frac{P^{-1}SP^{-1}}{\sqrt{\mu}}\right) = \left(\frac{PXSP^{-1}}{\mu}\right).$$

Let us further define

$$\overline{A_i} = \frac{1}{\sqrt{\mu}} P^{-1} A_i P^{-1}, \qquad i = 1, \cdots, m,$$

and

$$D_X = \frac{1}{\sqrt{\mu}} P\Delta X P, \qquad D_S = \frac{1}{\sqrt{\mu}} P^{-1}\Delta S P^{-1}. \tag{2.12}$$

Using the above notations, the system (2.10) is equivalent to the following system

$$\begin{cases} \overline{\mathcal{A}} D_X = 0, \\ \overline{\mathcal{A}}^* \Delta y + D_S = 0, \\ D_X + D_S = V^{-1} - V, \end{cases} \tag{2.13}$$

where $\overline{\mathcal{A}}$ is the operator defined in (2.6), replacing the matrices A_i by $\overline{A_i}$ for $i = 1, \cdots, m$, and $\overline{\mathcal{A}}^*$ is its adjoint.

The solution of (2.13) defines the (scaled) NT search direction $(D_X, \Delta y, D_S)$. Now, following [10] we replace the right-hand side in the last equation of (2.13) by $-\psi'(V)$, where $\psi(V)$ and $\psi'(V)$ are defined in Definition 2.1 and Remark 2.2 for our given kernel function $\psi(t)$ in (1.1). Thus we consider the following system

$$\begin{cases} \overline{\mathcal{A}} D_X = 0, \\ \overline{\mathcal{A}}^* \Delta y + D_S = 0, \\ D_X + D_S = -\psi'(V). \end{cases} \tag{2.14}$$

It is easy to verify that this system has a unique solution $(D_X, \Delta y, D_S)$. From (2.12), having D_X and D_S, we can compute ΔX and ΔS.

Due to the first two equations of the system (2.14), D_X and D_S are orthogonal, i.e., $\mathrm{tr}(D_X D_S) = \mathrm{tr}(D_S D_X) = 0$. Then, we can easily verify that the matrix function $\psi(V)$ determines in a natural way an interior point algorithm.

$$D_X = D_S = 0_{n\times n} \Leftrightarrow \psi'(V) = 0_{n\times n} \Leftrightarrow V = I \Leftrightarrow \Psi(V) = 0 \Leftrightarrow XS = \mu I,$$

i.e., if and only if $X = X(\mu)$ and $S = S(\mu)$, as it should. Otherwise $\Psi(V) > 0$. Hence, if $(X, y, S) \neq (X(\mu), y(\mu), S(\mu))$, then $(\Delta X, \Delta y, \Delta S) \neq (0_{n\times n}, 0_m, 0_{n\times n})$. By taking a step-size α along the search direction, the new iterate (X, y, S) is constructed according to

$$X_+ = X + \alpha\Delta X, \qquad y_+ = y + \alpha\Delta y, \qquad S_+ = S + \alpha\Delta S. \tag{2.15}$$

In what follows, we also often use the norm-based proximity measure, namely,

$$\sigma(V) := \frac{1}{2}\|D_X + D_S\| = \frac{1}{2}\|\psi'(V)\| = \frac{1}{2}\sqrt{\mathrm{tr}(\psi'(V)^2)}, \tag{2.16}$$

where the matrices D_X and D_S are defined in (2.12).

3 The New Parameterized Kernel Function and Its Properties

3.1 Properties of $\psi(V)$

Definition 3.1. *We call $\psi : (0, +\infty) \longrightarrow [0, +\infty)$ a kernel function if ψ is twice differentiable and the following conditions are satisfied:*

(i) $\psi'(1) = \psi(1) = 0$.

(ii) $\psi''(t) > 0$, $\forall t > 0$.

(iii) $\lim_{t\to 0^+} \psi(t) = \lim_{t\to+\infty} \psi(t) = +\infty$.

The first two conditions implies that $\psi(t)$ is completely determined by its second derivative:

$$\psi(t) = \int_1^t \int_1^\xi \psi''(\zeta)\, d\zeta d\xi. \tag{3.1}$$

In this section, we recall our new kernel function

$$\psi(t) = \frac{t^2-1}{2\sinh^2(1)} + \coth(t) - \coth(1), \qquad t > 0,$$

and develop some useful properties, which are essential to study the complexity analysis. For ease of reference, we give the first three derivatives of $\psi(t)$ for all $t > 0$.

$$\psi'(t) = \frac{t}{\sinh^2(1)} - \frac{1}{\sinh^2(t)}, \tag{3.2}$$

$$\psi''(t) = \frac{1}{\sinh^2(1)} + 2\frac{\coth(t)}{\sinh^2(t)} > \frac{1}{\sinh^2(1)}, \tag{3.3}$$

and

$$\psi'''(t) = -\frac{2}{\sinh^2(t)}\Big(2\coth^2(t) + \frac{1}{\sinh^2(t)}\Big) < 0, \tag{3.4}$$

where

$$\coth(t) > 1, \qquad \forall\, t > 0.$$

We can easily verify that ψ is a kernel function according to Definition 3.1.

Lemma 3.2. *Let $\psi(t)$ be the function defined in (1.1). Then, we have*

$$2t\coth(t) - 1 > 0, \qquad \forall\, t > 0, \tag{3.5}$$

$$\psi'(t) = \frac{t}{\sinh^2(1)} - \frac{1}{\sinh^2(t)} > 0, \qquad \forall\, t > 1. \tag{3.6}$$

Proof. For (3.5), we have

$$2t\coth(t) - 1 = \frac{(2t-1)e^{2t} + (2t+1)}{e^{2t}-1}.$$

Since $e^{2t} - 1 > 0$, $\forall t > 0$, it suffices to study the sign of the function

$$l(t) = (2t-1)e^{2t} + (2t+1).$$

We have

$$l'(t) = 4te^{2t} + 2 > 0, \qquad \forall\, t > 0,$$

this implies that the function $l(t)$ is increasing in the interval $(0, +\infty)$ and since

$$\lim_{t\to 0^+} l(t) = 0, \qquad \lim_{t\to+\infty} l(t) = +\infty,$$

we conclude that $l(t) > 0$, $\forall t > 0$, which proves (3.5).

For the inequality (3.6), we know that sinh is positive and increasing in $(0, +\infty)$ then in $(1, +\infty)$, this implies that

$$\frac{t}{\sinh^2(t)} < \frac{t}{\sinh^2(1)}, \qquad \forall\, t > 1.$$

From this last, the inequality (3.6) is a direct consequence, which completes the proof. □

Lemma 3.3. *Let $\psi(t)$ be our new kernel function, then we have*

(i) *$\psi(t)$ is convex exponentially for all $t > 0$; that is*

$$\psi(\sqrt{t_1 t_2}) \leq \frac{1}{2}(\psi(t_1) + \psi(t_2)), \qquad \forall\, t_1, t_2 > 0.$$

(ii) *$\psi''(t)$ is monotonically decreasing, $\forall\, t > 0$.*

(iii) *$t\psi''(t) - \psi'(t) > 0,\ \forall\, t > 0$.*

(iv) *$\psi''(t)\,\psi'(\beta t) - \beta\psi'(t)\,\psi''(\beta t) > 0,\ \forall\, t > 1,\ \beta > 1$.*

(v) *$2\psi''(t)^2 - \psi'(t)\,\psi'''(t) > 0,\ \forall\, t < 1$.*

Proof. For (i), by Lemma 2.1 in [10], it suffices to show that $t\psi''(t) + \psi'(t) \geq 0,\ \forall\, t > 0$. Indeed, using (3.2), (3.3) and (3.5) of Lemma 3.2, we have for all $t > 0$

$$t\psi''(t) + \psi'(t) = \frac{2t}{\sinh^2(1)} + \frac{2t\coth(t) - 1}{\sinh^2(t)} > 0.$$

For (ii), using (3.4), we have $\psi'''(t) < 0,\ \forall\, t > 0$, then $\psi''(t)$ decreases monotonically.
For (iii), using (3.2) and (3.3), we have

$$t\psi''(t) - \psi'(t) = \frac{2t\coth(t) + 1}{\sinh^2(t)} \geq 0.$$

For the next-to-last item and from Lemma 2.4 in [3], if $\psi(t)$ satisfies (3.4) and (iii) of this Lemma, then $\psi(t)$ satisfies (iv).

For the last item, using (3.2), (3.3) and (3.4), simple calculation leads to

$$\begin{aligned}
2\psi''(t)^2 - \psi'(t)\psi'''(t) =& 2\left(\frac{1}{\sinh^4(1)} + 2\,\frac{\coth^2(t)}{\sinh^4(t)} + 4\,\frac{\coth(t)}{\sinh^2(1)\sinh^2(t)}\right) \\
&+ 2\left(\frac{2t\coth^2(t)}{\sinh^2(1)\sinh^2(t)} + \frac{t}{\sinh^2(1)\sinh^4(t)} - \frac{1}{\sinh^6(t)}\right) \\
>& \frac{2}{\sinh^6(t)}\big(2\coth^2(t)\sinh^2(t) - 1\big) \\
=& \frac{2}{\sinh^6(t)}\big(2\cosh^2(t) - 1\big) > 0,
\end{aligned}$$

since $\cosh(t) > 1$ for all $t > 0$. This completes the proof. □

Note that any kernel function that satisfies the four conditions of Lemma 3.3 is an eligible function according to [3]. Thus, our kernel function is eligible.

From the exponential convexity property of the kernel function ψ, we can deduce the following result for the proximity function Ψ.

Proposition 3.4[10]**.** *For any $X_1 \succ 0$ and $X_2 \succ 0$, we have*

$$\Psi\big((X_1^{\frac{1}{2}} X_2 X_1^{\frac{1}{2}})^{\frac{1}{2}}\big) \leq \frac{1}{2}\big(\Psi(X_1) + \Psi(X_2)\big).$$

Lemma 3.5. *For $\psi(t)$, we have*

(i) $\frac{(t-1)^2}{2\sinh^2(1)} \leq \psi(t) \leq \frac{\sinh^2(1)}{2}(\psi'(t))^2, \ \ \forall\, t > 0.$

(ii) $\psi(t) \leq \big(\frac{2\coth(1)+1}{2\sinh^2(1)}\big)(t-1)^2, \ \forall\, t \geq 1.$

Proof. For (i), using (3.1) and (3.3), we have

$$\psi(t) = \int_1^T \int_1^\xi \psi''(\zeta)d\zeta d\xi \geq \frac{1}{\sinh^2(1)} \int_1^T \int_1^\xi d\zeta d\xi = \frac{(t-1)^2}{2\sinh^2(1)},$$

and

$$\psi(t) = \int_1^T \int_1^\xi \psi''(\zeta)d\zeta d\xi \leq \sinh^2(1) \int_1^T \int_1^\xi \psi''(\xi)\psi''(\zeta)d\zeta d\xi = \frac{\sinh^2(1)}{2}(\psi'(t))^2, \qquad t > 0.$$

For (ii), since $\psi(1) = \psi'(1) = 0, \psi'''(t) < 0$, $\psi''(1) = \frac{2\coth(1)+1}{\sinh^2(1)}$, and by using Taylor's theorem, we have for $1 \leq \xi \leq t$,

$$\begin{aligned}
\psi(t) &= \psi(1) + \psi'(1)(t-1) + \frac{(t-1)^2\psi''(1)}{2!} + \frac{(t-1)^3\psi'''(\xi)}{3!} \\
&= \frac{1}{2}\psi''(1)(t-1)^2 + \frac{\psi'''(\xi)}{3!}(t-1)^3 \\
&\leq \tfrac{1}{2}\psi''(1)(t-1)^2 \\
&= \Big(\frac{2\coth(1)+1}{2\sinh^2(1)}\Big)(t-1)^2.
\end{aligned}$$

This completes the proof. □

Lemma 3.6. *Let $\varrho : [0,+\infty) \longrightarrow [1,+\infty)$ be the inverse function of $\psi(t)$ for $t \geq 1$ and $\rho : [0,+\infty) \longrightarrow (0,1]$ be the inverse function of $-\frac{1}{2}\psi'(t)$ for $0 < t \leq 1$, then*

(i) $1 + \sinh(1)\sqrt{\frac{2z}{2\coth(1)+1}} \leq \varrho(z) \leq 1 + \sinh(1)\sqrt{2z}, \ \ \forall\, z \in [0,+\infty).$

(ii) $\coth(t) \leq \big(2z + 1 + \frac{1}{\sinh^2(1)}\big)^{\frac{1}{2}}, \ \ z = -\frac{1}{2}\psi'(t) \geq 0, \ \ \forall\, t \in (0,1].$

Proof. For (i), let $z \geq 0$ and let $t \in [1,+\infty)$ such that $z = \psi(t)$, then $\varrho(z) = t$, by (ii) of Lemma 3.5, we have

$$z \leq \Big(\frac{2\coth(1)+1}{2\sinh^2(1)}\Big)(t-1)^2,$$

this implies that

$$t - 1 \geq \sqrt{\frac{2\sinh^2(1)z}{2\coth(1)+1}}.$$

Hence, we have

$$t \geq 1 + \sqrt{\frac{2\sinh^2(1)z}{2\coth(1)+1}} = 1 + \sinh(1)\sqrt{\frac{2z}{2\coth(1)+1}}.$$

For the second hand, we use (i) of Lemma 3.5, we have $z = \psi(t) \geq \frac{(t-1)^2}{2\sinh^2(1)}$, then we obtain

$$t \leq 1 + \sinh(1)\sqrt{2z}.$$

For (ii), let $z \geq 0$ and let $t \in (0,1]$ such that $z = -\frac{1}{2}\psi'(t)$, for $t \in (0,1]$, then $\rho(z) = t$, so, by (3.2), we have

$$2z = -\psi'(t) = \frac{1}{\sinh^2(t)} - \frac{t}{\sinh^2(1)},$$

which is equivalent to

$$2z = \coth^2(t) - 1 - \frac{t}{\sinh^2(1)},$$

this implies

$$\coth^2(t) \leq 2z + 1 + \frac{1}{\sinh^2(1)}, \qquad \forall\, t \in (0,1],$$

which gives (ii). □

3.2 Properties of $\Psi(V)$

We have $\Psi(V) \leq \tau$ just before the update of μ with the factor $(1-\theta)$, at the start of each outer iteration. After updating μ in an outer iteration, the matrix V is divided by the factor $\sqrt{1-\theta}$, with $0 < \theta < 1$, which leads the proximity to be increased, in general. Then during the inner iteration, the value of $\Psi(V)$ decreases until it passes the threshold value τ again. The following Lemma gives an upper bound of $\Psi(\beta V) = \Psi(\frac{V}{\sqrt{1-\theta}})$ in terms of $\Psi(V)$.

Lemma 3.7[36]**.** *Let $\varrho : [0, +\infty) \to [1, +\infty)$ be the inverse function of $\psi(t)$ for $t \geq 1$. Then, for any $V \succ 0$ and $\beta > 1$, we have*

$$\Psi(\beta V) \leq n\psi\Big(\beta\varrho\Big(\frac{\Psi(V)}{n}\Big)\Big).$$

Corollary 3.8. *Let θ be such that $0 < \theta < 1$. If $\Psi(V) \leq \tau$, then*

$$\Psi(\beta V) \leq \Big(\frac{2\coth(1)+1}{2(1-\theta)\sinh^2(1)}\Big)\Big(\theta\sqrt{n} + \sinh(1)\sqrt{2\tau}\Big)^2, \qquad \beta = \frac{1}{\sqrt{1-\theta}} > 1.$$

Proof. Using (ii) of Lemma 3.5 for $t \geq 1$, we have

$$\psi(t) \leq \Big(\frac{2\coth(1)+1}{2\sinh^2(1)}\Big)(t-1)^2.$$

Consequently, by the above Lemma, we get

$$\Psi(\beta V) \leq \frac{n}{2}\Big(\frac{2\coth(1)+1}{\sinh^2(1)}\Big)\Big(\beta\varrho\Big(\frac{\Psi(V)}{n}\Big) - 1\Big)^2.$$

Hence, from (i) of Lemma 3.6, we have

$$\Psi(\beta V) \leq \frac{n}{2}\Big(\frac{2\coth(1)+1}{\sinh^2(1)}\Big)\Big(\beta\Big(1 + \sinh(1)\sqrt{\frac{2\Psi(V)}{n}}\Big) - 1\Big)^2.$$

Since $\beta = \frac{1}{\sqrt{1-\theta}}$, then

$$\begin{aligned}\Psi(\beta V) &\leq \frac{n}{2(1-\theta)}\Big(\frac{2\coth(1)+1}{\sinh^2(1)}\Big)\Big(\sinh(1)\sqrt{\tfrac{2\Psi(V)}{n}} + 1 - \sqrt{1-\theta}\Big)^2 \\ &\leq \Big(\frac{2\coth(1)+1}{2(1-\theta)\sinh^2(1)}\Big)\big(\theta\sqrt{n} + \sinh(1)\sqrt{2\Psi(V)}\big)^2.\end{aligned}$$

The last inequality is obtained from the fact that $1-\sqrt{1-\theta}=\frac{\theta}{1+\sqrt{1-\theta}}\leq\theta$.
By the assumption $\Psi(V)\leq\tau$ just before the $\mu-$update, we have

$$\Psi(\beta V)\leq\Big(\frac{2\coth(1)+1}{2(1-\theta)\sinh^2(1)}\Big)\big(\theta\sqrt{n}+\sinh(1)\sqrt{2\tau}\big)^2:=\Psi_0, \tag{3.7}$$

then Ψ_0 is an upper bound for $\Psi(\beta V)$ during the process of the algorithm. □

The following Lemma gives a lower bound of $\sigma(V)$ in terms of the proximity function $\Psi(V)$.

Lemma 3.9. *Let $\sigma(V)$ be defined by (2.16). Then, for any $V\succ 0$, we have*

$$\sigma(V)\geq\frac{1}{\sinh(1)}\sqrt{\frac{\Psi(V)}{2}}.$$

Proof. From (i) of Lemma 3.5, we have for all $t>0$

$$\psi(t)\leq\frac{\sinh^2(1)}{2}\psi'(t)^2,$$

using (2.16) and (2.2), we obtain

$$\sigma(V)^2=\frac{1}{4}\mathrm{tr}\big(\psi'(V)^2\big)=\frac{1}{4}\sum_{i=1}^{n}\psi'(\lambda_i(V))^2\geq\frac{1}{2\sinh^2(1)}\sum_{i=1}^{n}\psi(\lambda_i(V))=\frac{\Psi(V)}{2\sinh^2(1)}.$$

This proves the Lemma. □

4 The Generic Primal-dual Interior Point Algorithm for SDP

Algorithm 1. Primal-dual algorithm for SDP

Input
a threshold parameter $\tau\geq 1$;
an accuracy parameter $\varepsilon>0$;
a fixed barrier update parameter $\theta\in]0,1[$;
(X^0,y^0,S^0) satisfy the IPC and $\mu^0=1$ such that $\Phi\left(X^0,S^0;\mu^0\right)\leq\tau$.
begin
$X:=X^0;y:=y^0;S:=S^0;\mu:=\mu^0$;
while $n\mu\geq\varepsilon$ **do**
  begin (outer iteration)
  $\mu:=(1-\theta)\mu$;
  while $\Phi(X,S;\mu)=\Psi(V)>\tau$ **do**
    begin (inner iteration)
    Solve system (2.14) and use (2.12) to obtain $(\Delta X,\Delta y,\Delta S)$;
    Choose a suitable step-size α;
    $X:=X+\alpha\Delta X;y:=y+\alpha\Delta y;S:=S+\alpha\Delta S;V:=\sqrt{\frac{PXSP^{-1}}{\mu}}$;
  end while (inner iteration)
end while (outer iteration)

In general each kernel function gives rise to a primal-dual interior point algorithm. For the description of our algorithm, it is clear that closeness of (X,y,S) to $(X(\mu),y(\mu),S(\mu))$ is measured by the value of $\Psi(V)$ with τ as a threshold value: if $\Psi(V)\leq\tau$, then we start a new *outer iteration* by decreasing μ to $\mu:=(1-\theta)\mu$, for fixed $\theta\in(0,1)$, otherwise enter an *inner*

iteration by computing the search direction $(\Delta X, \Delta y, \Delta S)$ at the current iterate with respect to the current value of μ and apply (2.15) to get the new iterate. This process is repeated until μ is small enough, say until $n\mu < \varepsilon$ for a certain accuracy parameter ε, at this stage we have found an $\varepsilon-$optimal solution of (P) and (D). The algorithm of primal-dual IPM based on our kernel function is given in Algorithm 1.

5 Complexity Result

5.1 Computation of Displacement Step

The choice of the step-size α is another crucial issue in the analysis of the algorithm. In this section, we compute a default step-size α, the decrease of the proximity function during an inner iteration and give the complexity results of the algorithm.
From (2.15), (2.12) and (2.11) for fixed μ, one has

$$X_+ = X + \alpha\Delta X = X + \alpha\sqrt{\mu}P^{-1}D_X P^{-1} = \sqrt{\mu}P^{-1}(V + \alpha D_X)P^{-1},$$
$$S_+ = S + \alpha\Delta S = S + \alpha\sqrt{\mu}PD_S P = \sqrt{\mu}P(V + \alpha D_S)P,$$

So, by (2.11), we have $V_+ = V + \alpha D_X = V + \alpha D_S$. This implies that the eigenvalues of V_+ are the same as those of the matrix $((V + \alpha D_X)^{\frac{1}{2}}(V + \alpha D_S)(V + \alpha D_X)^{\frac{1}{2}})^{\frac{1}{2}}$.

The difference of proximities between a new iterate and a current iterate for fixed μ, called the function of the step-size α, is denoted by

$$f(\alpha) = \Psi(V_+) - \Psi(V).$$

Throughout the paper, we assume that the step-size α satisfied

$$V + \alpha D_X \succeq 0 \text{ and } V + \alpha D_S \succeq 0.$$

Then, by (2.2) and Proposition 3.4, we have

$$\begin{aligned}\Psi(V_+) =& \Psi\Big(\big((V + \alpha D_X)^{\frac{1}{2}}(V + \alpha D_S)(V + \alpha D_X)^{\frac{1}{2}}\big)^{\frac{1}{2}}\Big)\\ \leq& \frac{1}{2}\big(\Psi(V + \alpha D_X) + \Psi(V + \alpha D_S)\big).\end{aligned}$$

Therefore, $f(\alpha) \leq f_1(\alpha)$, where

$$f_1(\alpha) = \frac{1}{2}\left(\Psi(V + \alpha D_X) + \Psi(V + \alpha D_S)\right) - \Psi(V).$$

One can easily see that

$$f(0) = f_1(0) = 0.$$

Taking the first two derivatives of $f_1(\alpha)$ with respect to α, and using (2.3)–(2.5), we get

$$f_1'(\alpha) = \frac{1}{2}\mathrm{tr}\left(\psi'(V + \alpha D_X)D_X + \psi'(V + \alpha D_S)D_S\right).$$

$$f_1''(\alpha) = \frac{1}{2}\mathrm{tr}\left(\psi''(V + \alpha D_X)D_X^2 + \psi''(V + \alpha D_S)D_S^2\right).$$

Due to the third equality of (2.14), we have

$$f_1'(0) = -\frac{1}{2}\mathrm{tr}\big(\big(\psi'(V)\big)^2\big) = -2\sigma(V)^2, \tag{5.1}$$

where $\sigma(V)$ is the norm-based proximity measure defined in (2.16). For notational convenience, we put $\sigma(V) := \sigma$.

In the following, we cite Lemmas 4.1–4.4 in [3] without proof (see also [1, 10, 36]).

Lemma 5.1. *We have*

$$f_1''(\alpha) \leq 2\sigma^2\psi''(\lambda_{\min}(V) - 2\alpha\sigma).$$

Lemma 5.2. *If the step-size α satisfies the inequality*

$$\psi'(\lambda_{\min}(V)) - \psi'(\lambda_{\min}(V) - 2\alpha\sigma) \leq 2\sigma, \tag{5.2}$$

then

$$f_1'(\alpha) \leq 0.$$

Lemma 5.3. *Let ρ be as defined in Lemma 3.6, then the largest possible value of the step-size α^* satisfying (5.2) is given by*

$$\alpha^* = \frac{\rho(\sigma) - \rho(2\sigma)}{\sigma}.$$

Lemma 5.4. *Let ρ and α^* be as defined in Lemma 5.3. Then, we have*

$$\alpha^* \geq \frac{1}{\psi''(\rho(2\sigma))}.$$

Now, we give a suitable step-size α for our algorithm.

Lemma 5.5. *Let ρ and α^* be as defined in Lemma 5.4. If $\Psi(V) \geq \tau \geq 1$, then we have*

$$\alpha^* \geq \frac{\sinh^3(1)}{\sinh(1) + 2(1 + (4\sigma+1)\sinh^2(1))^{\frac{3}{2}}}.$$

Proof. From (3.3) with $t = \rho(2\sigma) \in (0, 1]$, we have

$$\begin{aligned}
\psi''(t) &= \frac{1}{\sinh^2(1)} + \frac{2\coth(t)}{\sinh^2(t)} \\
&= \frac{1}{\sinh^2(1)} + 2\coth(t)(\coth^2(t) - 1) \\
&\leq \frac{1}{\sinh^2(1)} + 2\Big(4\sigma + \frac{1}{\sinh^2(1)}\Big)\Big(4\sigma + 1 + \frac{1}{\sinh^2(1)}\Big)^{\frac{1}{2}} \\
&\leq \frac{1}{\sinh^2(1)} + 2\Big(4\sigma + 1 + \frac{1}{\sinh^2(1)}\Big)^{\frac{3}{2}} \\
&= \frac{1}{\sinh^2(1)} + \frac{2}{\sinh^3(1)}\left(1 + (4\sigma+1)\sinh^2(1)\right)^{\frac{3}{2}} \\
&= \frac{\sinh(1) + 2(1 + (4\sigma+1)\sinh^2(1))^{\frac{3}{2}}}{\sinh^3(1)},
\end{aligned}$$

where the next-to-last inequality follows from (ii) of Lemma 3.6. From Lemma 5.4, we have

$$\alpha^* \geq \frac{\sinh^3(1)}{\sinh(1) + 2(1 + (4\sigma+1)\sinh^2(1))^{\frac{3}{2}}},$$

which completes the proof. □

Denoting

$$\bar{\alpha} = \frac{\sinh^3(1)}{\sinh(1) + 2(1 + (4\sigma+1)\sinh^2(1))^{\frac{3}{2}}}, \tag{5.3}$$

we have that $\bar{\alpha}$ is the default step-size in the algorithm and that $\bar{\alpha} \leq \alpha^*$.

5.2 Decreasing of $\Psi(V)$ During an Inner Iteration

Lemma 5.6[36]**.** *Let $g(t)$ be a twice differentiable convex function with $g(0)=0$, $g'(0)<0$ and let $g(t)$ attain its (global) minimum at $t^*>0$. If $g''(t)$ is increasing for $t\in[0,t^*]$, then*

$$g(t)\leq\frac{g'(0)}{2}t,\qquad 0\leq t\leq t^*.$$

As a direct consequence, we get

Lemma 5.7. *If the step-size α satisfies $\alpha\leq\alpha^*$, then*

$$f(\alpha)\leq-\alpha\sigma^2.$$

Proof. It's easy to verify that $f_1(\alpha)$ satisfies the condition of the above Lemma, then

$$f(\alpha)\leq f_1(\alpha)\leq\frac{f_1'(0)}{2}\alpha,\qquad\text{for all}\quad 0\leq\alpha\leq\alpha^*.$$

The result follows from (5.1). □

We can obtain the upper bound for the decreasing value of the proximity in the inner iteration by the following theorem:

Theorem 5.8. *If $\bar{\alpha}$ is the default step-size as given by (5.3) and $\sigma\geq1$, then we have*

$$f(\bar{\alpha})\leq-\frac{\Psi(V)^{\frac{1}{4}}\sinh^{\frac{5}{2}}(1)}{2+4\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}.\tag{5.4}$$

Proof. Using (5.3) in Lemma 5.7 for $\bar{\alpha}\in[0,\alpha^*]$, we get

$$\begin{aligned}f(\bar{\alpha})&\leq-\sigma^2\bar{\alpha}=-\frac{\sigma^2\sinh^3(1)}{\sinh(1)+2(1+(4\sigma+1)\sinh^2(1))^{\frac{3}{2}}}\\&=-\frac{\sigma^2\sinh^3(1)}{\left(\frac{\sinh(1)}{\sigma}+2\left(\frac{1}{\sigma}+\left(4+\frac{1}{\sigma}\right)\sinh^2(1)\right)^{\frac{3}{2}}\right)\sigma^{\frac{3}{2}}}\\&\leq-\frac{\sigma^{\frac{1}{2}}\sinh^3(1)}{\sqrt{2}+2\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}.\end{aligned}$$

Then, thanks to Lemma 3.9, we obtain

$$\begin{aligned}f(\bar{\alpha})&\leq-\frac{\left(\frac{\Psi(V)}{2\sinh^2(1)}\right)^{\frac{1}{4}}\sinh^3(1)}{\sqrt{2}+2\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}\\&\leq-\frac{\left(\frac{\Psi(V)}{2}\right)^{\frac{1}{4}}\sinh^{\frac{5}{2}}(1)}{\sqrt{2}+2\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}\\&=-\frac{\Psi(V)^{\frac{1}{4}}\sinh^{\frac{5}{2}}(1)}{2^{\frac{1}{4}}\left(\sqrt{2}+2\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}\right)}\\&\leq-\frac{\Psi(V)^{\frac{1}{4}}\sinh^{\frac{5}{2}}(1)}{2+4\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}.\end{aligned}$$

This result holds the Theorem. □

5.3 Iteration Complexity

We need to compute how many inner iterations the algorithm requires to return to the situation where $\Psi(V) \leq \tau$ after $\mu-$update. We denote the value of $\Psi(V)$ after $\mu-$update by Ψ_0 and by $\Psi_k, k = 1, \cdots, K-1$, the subsequent values in the same outer iteration, where K stands for the total number of inner iterations in an outer iteration. By the definition of $f(\alpha)$ and according to (5.4), the decrease of Ψ in an inner iteration is given by

$$\Psi_{k+1} \leq \Psi_k - \frac{\Psi_k^{\frac{1}{4}} \sinh^{\frac{5}{2}}(1)}{2 + 4\left(\sqrt{2}\sinh(1) + (4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}, \qquad k = 0, 1, \cdots, K-1.$$

Lemma 5.9[10]. *Suppose $t_0, t_1, \cdots, t_k$ be a sequence of positive numbers such that*

$$t_{k+1} \leq t_k - \beta t_k^{1-\gamma}, \qquad k = 0, 1, \cdots, K-1,$$

where $\beta > 0$ and $0 < \gamma \leq 1$. Then $K \leq \left\lceil \frac{t_0^{\gamma}}{\beta\gamma} \right\rceil$.

As a consequence, by taking $t_k = \Psi_k$, $\beta = \frac{\sinh^{\frac{5}{2}}(1)}{2+4\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}$ and $\gamma = \frac{3}{4}$, we can get the following lemma.

Lemma 5.10. *Let K be the total number of inner iterations in the outer iteration. Then we have*

$$K \leq \left\lceil K_{\max} \Psi_0^{\frac{3}{4}} \right\rceil,$$

where $K_{\max} = \frac{8+16\left(\sqrt{2}\sinh(1)+(4+\sqrt{2}\sinh(1))\sinh^2(1)\right)^{\frac{3}{2}}}{3\sinh^{\frac{5}{2}}(1)}$ and Ψ_0 is the value of $\Psi(V)$ after the $\mu-$update in outer iteration.

Now, we derive the complexity bounds for large and small-update methods.

Theorem 5.11. *Let Ψ_0 be the value defined in (3.7) and let $\tau \geq 1$. Then, the total number of iterations to obtain an approximation solution with $n\mu \leq \varepsilon$ is bounded by*

$$\left\lceil K_{\max} \Psi_0^{\frac{3}{4}} \right\rceil \left\lceil \frac{1}{\theta} \log \frac{n}{\varepsilon} \right\rceil.$$

Proof. Recall that Ψ_0 is the upper bound according to (3.7). An upper bound for the total number of iterations is derived by multiplying the upper bound K by the number of barrier parameter updates, which is bounded above by $\frac{1}{\theta}(\log \frac{n}{\varepsilon})$ (see [10]), that gives the result thanks to the above lemma. □

For large-update method with $\tau = \mathcal{O}(n)$ and $\theta = \Theta(1)$, the complexity of the primal-dual IPM for SDP problem based on our new kernel function is $\mathcal{O}\left(n^{\frac{3}{4}} \log \frac{n}{\varepsilon}\right)$ iterations complexity. These iterations bound coincide with the primal-dual IPM in [12] which is based on the first proposed trigonometric kernel function.

For small-update method with $\tau = \mathcal{O}(1)$ and $\theta = \Theta\left(\frac{1}{\sqrt{n}}\right)$, we get the currently best known iteration bound, namely $\mathcal{O}\left(\sqrt{n} \log \frac{n}{\varepsilon}\right)$ iterations.

6 Numerical Tests

To prove the effectiveness of our new kernel function (denoted ψ_{new}) and evaluate its effect on the behavior of the algorithm, we conducted comparative numerical experiments on two test problems (**Problem 1** and **Problem 2** below):

• In a first stage, with the following kernel functions:

1. The classic kernel function proposed by Roos et al. [31] defined as follows:

$$\psi_c(t) = \frac{t^2-1}{2} - \log t.$$

2. The exponential kernel function proposed by Bai et al. [3] defined as follows:

$$\psi_e(t) = \frac{t^2-1}{2} + e^{\frac{1}{t}-1} - 1.$$

3. The trigonometric kernel function proposed by Bouafia et al. [5] defined as follows:

$$\psi_B(t) = \frac{t^2-1}{2} + \frac{4}{\pi p}\Big(\tan^p\Big(\frac{\pi}{2t+2}\Big) - 1\Big), \qquad p = 4.$$

• In a second stage, with a contemporary state of the art implementations, namely SeDuMi[32].

Problem 1[34]. $C = -I$, A_k, $k = 1, \cdots, m$, are defined as

$$A_k(i,j) = \begin{cases} 1, & \text{if } i = j = k, \\ 1, & \text{if } i = j \text{ and } i = m+k, \\ 0, & \text{otherwise,} \end{cases}$$

and $b(i) = 2, \ i = 1, \cdots, m$.

We start by an initial point (X^0, y^0, S^0) such that X^0 is defined as follows

$$X^0(i,j) = \begin{cases} 1.5, & \text{if } i \le j, \\ 0.5, & \text{if } i > j, \end{cases}$$

$y^0(i) = -2, \ i = 1, \cdots, m$ and $S^0 = I$.

Problem 2[5].

$$C(i,j) = \begin{cases} -1, & \text{if } i = j \le m, \\ 0, & \text{otherwise,} \end{cases}$$

A_k, $k = 1, \cdots, m$, are defined as

$$A_k(i,j) = \begin{cases} 1, & \text{if } i = j = k, \\ 1, & \text{if } i = j \text{ and } i = m+k, \\ 0, & \text{otherwise,} \end{cases}$$

and $b(i) = 2$, $i = 1, \cdots, m$.

We start by an initial point (X^0, y^0, S^0) such that $X^0 = I$, $y^0(i) = -2, \ i = 1, \cdots, m$, and S^0 is defined as follows

$$S^0(i,j) = \begin{cases} 1, & \text{if } i = j \le m, \\ 2, & \text{if } i = j > m, \\ 0, & \text{otherwise.} \end{cases}$$

The starting primal-dual point (X^0, y^0, S^0) in each problem is feasible and so the IPC is satisfied. We used this point to initialize problems 1 and 2 for Algorithm 1 but we have let SeDuMi use its own initialization process.

To be solved, we used the Software MATLAB 7.7.0 (R2008b) on Intel Core i3 (1.80 GHz) with 4.00 Go RAM. We select the following parameters for all experiments: $\varepsilon = 1e-008,\ \mu = 1,\ \theta \in \{0.1,\ 0.95\},\ \tau = \sqrt{n}$, where $n = 2m$ is the number of variables, m being the number of constraints.

6.1 Comparison with other Kernel Functions

In this section, we show the results of the implementation of our algorithm for each of the kernel functions ψ_c, ψ_e, ψ_B and ψ_{new} tested on each problem. This numerical study is summarized in Tables 1 and 2 (resp. Tables 3 and 4) for **Problem 1** (resp. **Problem 2**), where we show the average of the CPU time and the number of iterations required to reduce the duality gap below 10^{-8} for $\theta = 0.1$ and $\theta = 0.95$ respectively.

Table 1. CPU Time and Number of Iterations for Problem 1. $\theta = 0.1$.

size (m,n)		ψ_c	ψ_e	ψ_B	ψ_{new}
	Iter	194	192	254	**190**
(5,10)	CPU	0.0169	0.0184	0.0193	0.0162
	gap	9.3132e-9	9.3132e-9	2.7615e-9	9.3132e-9
	Iter	206	408	325	**202**
(10,20)	CPU	0.1025	0.0881	0.0866	0.0817
	gap	4.6566e-9	9.3992e-9	7.1835e-9	4.4631e-9
	Iter	205	1538	363	**203**
(15,30)	CPU	0.7879	0.5616	0.7347	0.5596
	gap	6.9849e-9	9.4374e-9	6.4231e-9	7.5125e-9
	Iter	205	1904	674	**201**
(20,40)	CPU	4.3212	2.4098	2.4363	2.4290
	gap	9.3132e-9	4.5846e-9	5.9208e-9	9.5150e-9
	Iter	**214**	1789	867	**214**
(25,50)	CPU	8.7237	8.7600	8.8470	8.7044
	gap	4.9750e-9	9.4121e-9	6.6742e-9	2.9734e-9

Table 2. CPU Time and Number of Iterations for Problem 1. $\theta = 0.95$.

size (m,n)		ψ_c	ψ_e	ψ_B	ψ_{new}
	Iter	**16**	18	20	18
(5,10)	CPU	0.0221	0.0217	0.0293	0.0208
	gap	8.6324e-9	5.9311e-9	9.5165e-9	4.7714e-9
	Iter	23	44	25	**18**
(10,20)	CPU	0.1540	0.1150	0.0878	0.0851
	gap	1.1571e-9	6.6908e-10	6.6016e-10	9.5428e-9
	Iter	23	59	32	**20**
(15,30)	CPU	0.5686	0.5656	0.5937	0.5470
	gap	1.7357e-9	1.8326e-9	1.9117e-9	7.8748e-10
	Iter	23	130	101	**20**
(20,40)	CPU	2.8046	2.4605	2.4429	2.4338
	gap	2.3142e-9	2.4626e-9	1.3112e-9	1.0500e-9
	Iter	**19**	135	137	21
(25,50)	CPU	8.7256	8.7519	8.7197	8.6582
	gap	2.8556e-9	3.0526e-9	1.7414e-9	1.7099e-9

In all tables, (Iter), (CPU) and (gap) stand for the number of iterations, CPU time (in second) and the value of the duality gap, respectively. Each column on the table corresponds to a kernel function and each row to a size (m, n).

Table 3. CPU Time and Number of Iterations for Problem 2. $\theta = 0.1$.

size (m,n)		ψ_c	ψ_e	ψ_B	ψ_{new}
	Iter	206	200	204	**190**
(5,10)	CPU	0.0221	0.0221	0.0258	0.0109
	gap	2.5662e-9	4.3388e-9	2.7199e-9	9.1855e-9
	Iter	206	265	263	**201**
(10,20)	CPU	0.1062	0.0845	0.0850	0.0785
	gap	4.3864e-9	7.4438e-9	6.0232e-9	7.5547e-9
	Iter	207	267	342	**204**
(15,30)	CPU	0.5434	0.5554	0.9274	0.5364
	gap	5.6731e-9	4.3826e-9	8.2069e-9	9.9190e-9
	Iter	216	275	485	**210**
(20,40)	CPU	3.3591	2.4619	2.4465	2.4340
	gap	2.7778e-9	5.2792e-9	7.4000e-9	6.0448e-9
	Iter	**208**	293	525	215
(25,50)	CPU	13.8561	8.6938	8.8756	8.7750
	gap	7.5002e-9	3.6691e-9	8.9742e-9	5.3219e-9

Table 4. CPU Time and Number of Iterations for Problem 2. $\theta = 0.95$.

size (m,n)		ψ_c	ψ_e	ψ_B	ψ_{new}
	Iter	**26**	37	29	29
(5,10)	CPU	0.0187	0.0237	0.0220	0.0153
	gap	9.9899e-9	4.1175e-10	5.2381e-9	8.4520e-9
	Iter	38	**31**	32	34
(10,20)	CPU	0.0859	0.0973	0.1028	0.0816
	gap	1.5563e-9	8.9886e-10	1.1287e-9	9.0337e-10
	Iter	39	34	**32**	35
(15,30)	CPU	0.5368	0.5399	0.5572	0.5422
	gap	1.8297e-9	7.7120e-10	1.6931e-9	1.6381e-9
	Iter	41	**30**	39	36
(20,40)	CPU	2.5094	2.4236	2.6641	2.4185
	gap	2.7039e-9	1.9767e-9	1.9773e-9	6.6331e-10
	Iter	41	2733	58	**36**
(25,50)	CPU	8.7026	8.9311	8.7098	8.6650
	gap	3.3798e-9	3.6691e-009	2.6433e-9	1.3601e-9

We now plot separately the average amount of CPU time and the number of iterations taken by each kernel function to obtain the optimal solution below 10^{-8} in terms of the dimension n.

6.1.1 Comments

From tables and figures, it becomes clear that smaller values of the parameter θ influence the iteration count negatively. Hence, it seems on the average, that the best number results occur in the case of around $\theta = 1$.

For each example, we used **bold** font to highlight the best, i.e., the smallest, iteration number.

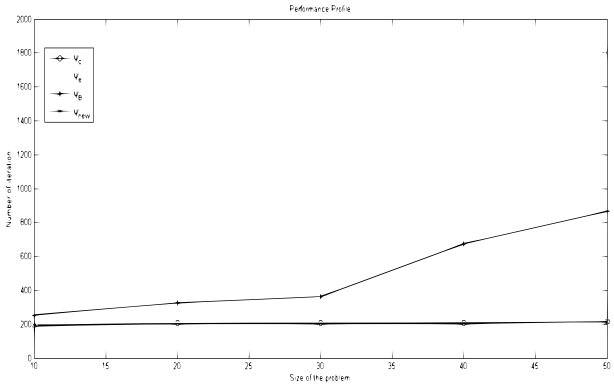

Figure 6.1. Problem 1: Number of Iterations Until Duality Gap Below to 10^{-8}. $\theta = 0.1$.

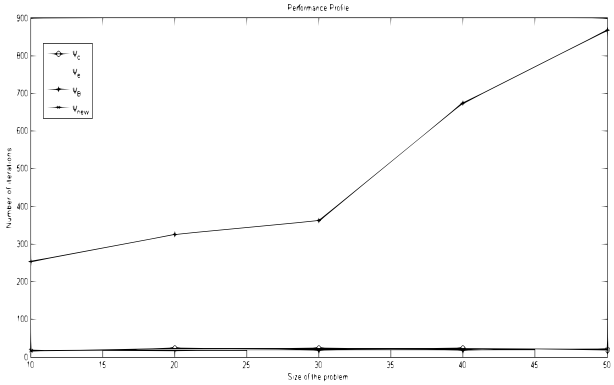

Figure 6.2. Problem 1: Number of Iterations Until Duality Gap Below to 10^{-8}. $\theta = 0.95$.

Recall that the numerical results were obtained by performing Algorithm 1 with the proposed kernel functions on two test problems for different sizes and in both cases $\theta = 0.1$ and $\theta = 0.95$. This left us with 20 experiments.

From this information, we conclude that the new kernel function gives the best results for 65% of the realized experiments.

Note that the kernel function ψ_B gives the smallest iteration number only one time. This confirm the effectiveness of the new kernel function ψ_{new}, particularly knowing that ψ_B has $\mathcal{O}\big(n^{\frac{3}{5}} \log \frac{n}{\varepsilon}\big)$ complexity bounds which is better than that of ψ_{new}.

In summary, numerical results show that by using our new kernel function, with hyperbolic barrier term, the best iteration complexity was achieved in most of the experiments. Their practical performance seems quite promising for SDP.

6.2 Comparison with SeDuMi Solver

In this section, we solve problems 1 and 2 by SeDuMi and compare with the results obtained by Algorithm 1 based on ψ_{new}.

In order to verify the correctness of the proposed implementation, we start by checking that the optimal solution is the same as the solution obtained by SeDuMi for each problem.

We compute the average of the CPU time of our algorithm and SeDuMi for both **Problem 1** and **Problem 2** with different sizes. The obtained results are shown in Table 5.

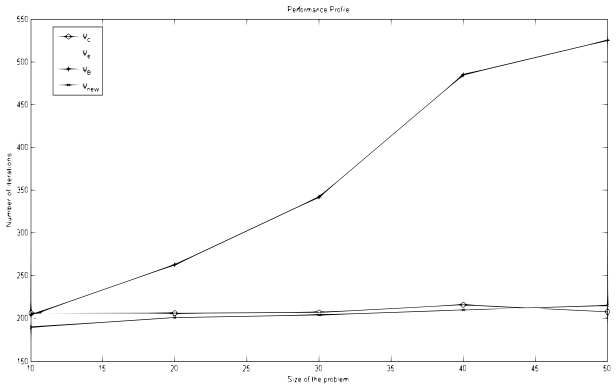

Figure 6.3. Problem 2: Number of Iterations Until Duality Gap Below to 10^{-8}. $\theta = 0.1$.

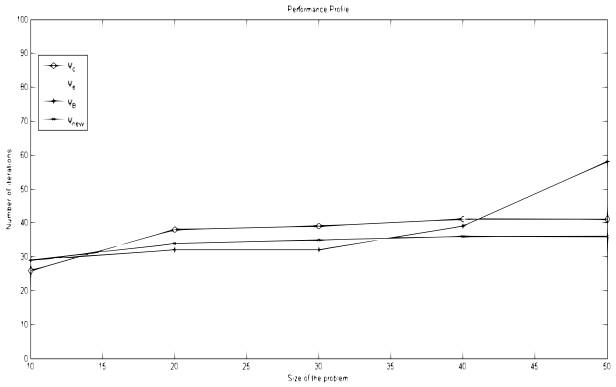

Figure 6.4. Problem 2: Number of Iterations Until Duality Gap Below to 10^{-8}. $\theta = 0.95$.

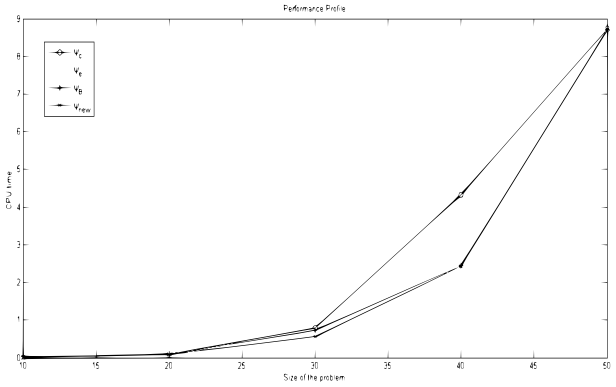

Figure 6.5. Problem 1: CPU time of Per Iteration for $\theta = 0.1$.

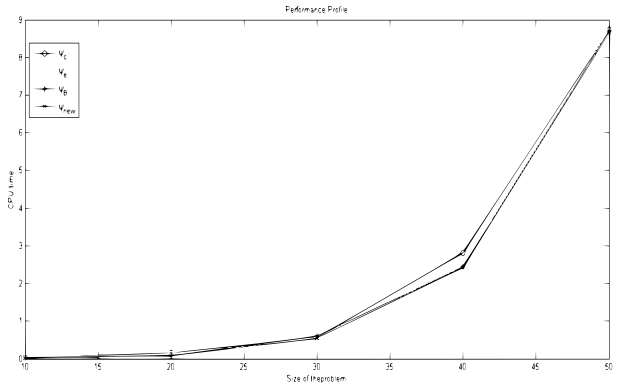

Figure 6.6. Problem 1: CPU Time of Per Iteration for $\theta = 0.95$.

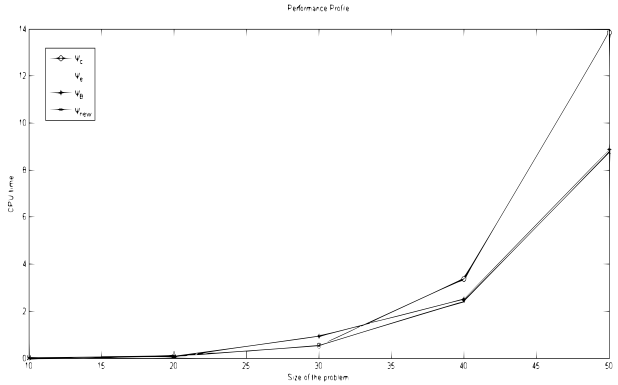

Figure 6.7. Problem 2: CPU time of Per Iteration for $\theta = 0.1$.

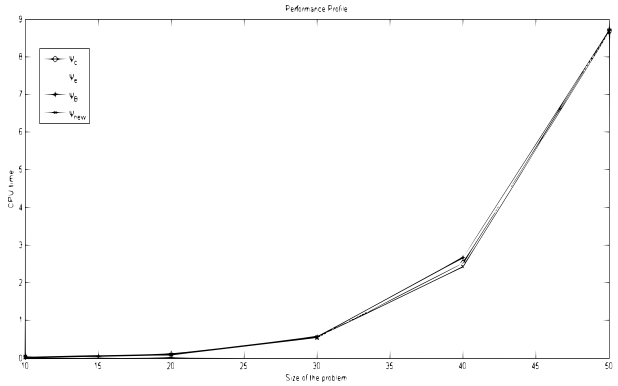

Figure 6.8. Problem 2: CPU Time of Per Iteration for $\theta = 0.95$.

Table 5. CPU to Obtain an Optimal Solution for Problems

Problem	size (m, n)	SeDuMi	$\psi_{\text{new},1}$ $\theta = 0.1$	$\psi_{\text{new},2}$ $\theta = 0.95$
Problem 1	(5,10)	0.8424	**0.0162**	0.0208
	(10,20)	0.9204	**0.0817**	0.0851
	(15,30)	0.9672	0.5596	**0.5470**
	(20,40)	**1.0452**	2.4290	2.4338
	(25,50)	**1.1232**	8.7044	8.6582
Problem 2	(5,10)	2.1528	**0.0109**	0.0153
	(10,20)	2.3088	**0.0785**	0.0816
	(15,30)	2.4024	**0.5364**	0.5422
	(20,40)	**2.4185**	2.4340	2.4960
	(25,50)	**2.7768**	8.7750	8.6650

6.2.1 Comments

By the look of the solving times shown in Table 5, we can see that the SeDuMi solver accomplishes good results for large sizes of problems. Furthermore, the CPU time grows slowly with the increasing size of the problem. However, our algorithm totally wins for problems sizes up to thirty.

7 Conclusions and Future Works

In this paper, we have combined a growth term and a hyperbolic barrier term to obtain a novel kernel function. We have analyzed a primal-dual algorithm for SDP based on this kernel function for both large-update and small-update versions. We have proved that the iteration bound for large-update methods is $\mathcal{O}\big(n^{\frac{3}{4}} \log \frac{n}{\varepsilon}\big)$. Compared with the classical method, which is based on the logarithmic barrier function, we see that complexity has been improved with a factor $n^{\frac{1}{4}}$. For small-update methods, we have obtained $\mathcal{O}\big(\sqrt{n} \log \frac{n}{\varepsilon}\big)$ iteration bound which matches the currently best known iteration bound for small-update methods. These results are the first contribution where the proposed kernel function is of hyperbolic type. To consolidate our theoretical results, we also provide some numerical performances of the proposed algorithm comparing the results with some other kernel functions. The realized numerical experiments show the effectiveness of our new kernel function.

Some interesting topics for further works remain. Firstly, we expect to find another hyperbolic kernel function to improve the complexity bound for large-update IPMs. Secondly, the extension to symmetric cone programming (SCP) deserves to be investigated.

Acknowledgments. The authors are very grateful and would like to thank the Editor-in-Chief and the anonymous referees for their suggestions and helpful comments which significantly improved the presentation of this paper.

References

[1] Bai, Y.Q., Roos, C. A primal-dual interior-point method based on a new kernel function with linear growth rate. Proceedings of Industrial Optimization Symposium and Optimization Day, Australia, 2002

[2] Bai, Y.Q., EL Ghami, M. Roos, C. A new efficient large-update primal-dual interior-point method based on a finite barrier. *SIAM J. Optim.*, 13(3): 766–782 (2003)

[3] Bai, Y.Q., EL Ghami, M. Roos, C. A comparative study of kernel functions for primal-dual interior-point algorithms in linear optimization. *SIAM J. Optim.*, 15(1): 101–128 (2004)

[4] Bai, Y.Q., Guo, J. Roos, C. A new kernel function yielding the best known iteration bounds for primal-dual interior-point algorithms. *Acta Math. Sin. (Engl. Ser.)*, 25(12): 2169–2178 (2009)

[5] Bouafia, M., Benterki, D. Yassine, A. An efficient primal-dual interior point method for linear programming problems based on a new kernel function with a trigonometric barrier term. *J. Optim. Theory Appl.*, 170(2): 528–545 (2016)
[6] Bouafia, M., Benterki, D. Yassine, A. Complexity analysis of interior point methods for linear programming based on a parameterized kernel function. *RAIRO-Oper. Res.*, 50: 935–949 (2016)
[7] Bouafia, M. Yassine, A. An efficient twice parameterized trigonometric kernel function for linear optimization. *Optim. Eng.*, 21: 651–672 (2020)
[8] Boudjellal, N., Roumili, H. Benterki, Dj. A primal-dual interior point algorithm for convex quadratic programming based on a new kernel function with an exponential barrier term. *Optim.*, https://doi.org/10.1080/02331934.2020.1751156 (2020)
[9] Cai, X.Z., Wang, G.Q., El Ghami, M. Yue, Y.J. Complexity analysis of primal-dual interior-point methods for linear optimization based on a new parametric kernel function with a trigonometric barrier term. *Abstr. Appl. Anal.*: 2014, 11 (2014). Art. ID 710158
[10] Choi, B.K. Lee, G. M. On complexity analysis of the primal-dual interior point method for semidefinite optimization problem based on a new proximity function. *Nonlinear Anal.*: 71(12): 2540–2550 (2009)
[11] El Ghami, M., Ivanov, I.D., Roos, C. Steihaug, T. A polynomial-time algorithm for LO based on generalized logarithmic barrier functions. *Int. J. Appl. Math.*, 21: 99–115 (2008)
[12] El Ghami, M., Guennoun, Z.A., Bouali, S. Steihaug, T. Interior point methods for linear optimization based on a kernel function with a trigonometric barrier term. *J. Comput. Appl. Math.*: 236 3613–3623 (2012)
[13] Fathi-Hafshejani, S. Fakharzadeh, J.A. An interior-point algorithm for semidefinite optimization based on a new parametric kernel function. *J. Nonlinear. Funct. Anal.*, 2018: 1–24 (2018) Art. ID 14
[14] Fathi-Hafshejani, S., Peyghami, M.R. Fakharzadeh, J.A. An interior-point method for linear optimization based on a trigonometric kernel function. *J. Nonlinear. Funct. Anal.*: 2019 1–17 (2019) Art. ID 46
[15] Helmberg, C, Rendl, F., Vanderbei, R.J., Wolkowicz, H. An interior-point method for semidefinite programming. *SIAM. J. Optim.*, 6: 342–361 (1996)
[16] Horn, R.A. Johnson, C.R. Topics in matrix analysis. Cambridge University Press, Cambridge, 1991
[17] Karmarkar, N.K. A new polynomial-time algorithm for linear programming. In: Proceedings of the 16th Annual ACM Symposium on Theory of Computing, 4: 373–395 (1984)
[18] Kheirfam, B. Primal-dual interior-point algorithm for semidefinite optimization based on a new kernel function with trigonometric barrier term. *Numer. Algorithms.*, 61: 659–680 (2012)
[19] Li, X., Zhang, M. Interior-point algorithm for linear optimization based on a new trigonometric kernel function. *Oper. Res. Lett.*, 43: 471–475 (2015)
[20] Li, M., Zhang, M., Huang, K. Huang, Z. A new primal-dual interior-point method for semidefinite optimization based on a parameterized kernel function. *Optim. Eng.*, 22: 293–319 (2021)
[21] Lütkepohl H. Handbook of matrices. Humboldt-Universität zu, Berlin, Germany, 1996
[22] Fathi-Hafshejani, S. Moaberfard, Z. A generic kernel function for interior point methods. *Optim. Eng.*, 22: 261–291 (2021)
[23] Monteiro, R.D.C. Zanjácomo, P.R. A note on the existence of Alizadeh-Heaberly-Overton direction for semidefinite programming. *Math. Program.*, 78(3): 393–396 (1997)
[24] Peng, J., Roos, C. Terlaky, T. A new and efficient large-update interior point method for linear optimization. *J. Comput. Technol.*, 6: 61–80 (2001)
[25] Peng, J., Roos, C. Terlaky, T. Self-regular functions and new search directions for linear and semidefinite optimization. *Math. Program.*, 93: 129–171 (2002)
[26] Peng, J., Roos, C. Terlaky, T. Self-regularity: A new paradigm for primal-dual interior-point algorithms. Princeton University Press, Princeton, 2002
[27] Peyghami, M.R. Hafshejani, S.F. Complexity analysis of an interior point algorithm for linear optimization based on a new proximity function. *Numer. Algorithms.*, 67: 33–48 (2014)
[28] Peyghami, M.R., Fathi-Hafshejani, S. Chen, S. A primal-dual interior-point method for semidefinite optimization based on a class of trigonometric barrier functions. *Oper. Res. Lett.*, 44(3): 319–323 (2016)
[29] Peyghami, M.R., Hafshejani, S.F. Shirvani, L. Complexity of interior point methods for linear optimization based on a new trigonometric kernel function. *J. Comput. Appl. Math.*, 255: 74–85 (2014)
[30] Qian, Z.G., Bai, Y.Q. Wang, G.Q. Complexity analysis of interior-point algorithm based on a new kernel function for semidefinite optimization. *J. Shanghai Univ.(Engl. Ed.)*, 12: 388–394 (2008)
[31] Roos, C., Terlaky, T. Vial, J.Ph. Theory and Algorithms for Linear Optimization, An interior point approach. Wiley, Chichester, 1997
[32] Sturm, J.F. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.*, 11(12): 625–653 (1999)
[33] Todd, M.J., Toh, K.C. Tütüncü, R.H. On the Nestrov-Todd direction in semidefinite programming. *SIAM J. Optim.*, 8(3): 769–796 (1998)
[34] Touil, I., Benterki, D. Yassine, A. A feasible primal-dual interior point method for linear semidefinite programming. *J. Comput. Appl. Math.*, 312: 216–230 (2017)
[35] Touil, I. Benterki, D. A primal-dual interior point method for the semidefinite programming problem based on a new kernel function. *J Nonlinear. Funct. Anal.*, 2019: 1–17 (2019) Art. ID 25

[36] Wang, G.Q., Bai, Y.Q. Roos, C. Primal-dual interior-point algorithms for semidefinite optimization based on a simple kernel function. *J. Math. Model. Algor.*, 4: 409–433 (2005)
[37] Wolkowicz, H., Saigal. R. Vandenberghe. L. Handbook of semidefinite programming, Theory, Algorithms, and Applications. Kluwer Academic Publishers, 2000
[38] Wright. S.J. Primal-dual interior point methods. SIAM Philadelphia USA, 1997
[39] Ye, Y. Interior Point Algorithms. Theory and Analysis. Wiley, Chichester, 1997
[40] Zhang, M.W. A large-update interior-point algorithm for convex quadratic semi-definite optimization based on a new kernel function. *Acta. Math. Sin.-English Ser.* 28: 2313–2328 (2012)
[41] Zhang, Y. On extending some primal-dual algorithms from linear programming to semidefinite programming. *SIAM J. Optim.*, 8(2): 365–386 (1998)