Neurocomputing 203 (2016) 62–72

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A neural network based on the generalized FB function for nonlinear convex programs with second-order cone constraints

Xinhe Miao [a,1], Jein-Shan Chen [b,*,2], Chun-Hsu Ko [c]

[a] Department of Mathematics, School of Science, Tianjin University, Tianjin 300072, PR China
[b] Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan
[c] Department of Electrical Engineering, I-Shou University, Kaohsiung 840, Taiwan

ARTICLE INFO

Article history:
Received 24 May 2015
Received in revised form
26 January 2016
Accepted 22 April 2016
Communicated by Ligang Wu
Available online 10 May 2016

Keywords:
Neural network
Generalized FB function
Stability
Second-order cone

ABSTRACT

This paper proposes a neural network approach to efficiently solve nonlinear convex programs with the second-order cone constraints. The neural network model is designed by the generalized Fischer–Burmeister function associated with second-order cone. We study the existence and convergence of the trajectory for the considered neural network. Moreover, we also show stability properties for the considered neural network, including the Lyapunov stability, the asymptotic stability and the exponential stability. Illustrative examples give a further demonstration for the effectiveness of the proposed neural network. Numerical performance based on the parameter being perturbed and numerical comparison with other neural network models are also provided. In overall, our model performs better than two comparative methods.

1. Introduction

The nonlinear convex programs with second-order cone constraints (we abbreviate it as SOCP in this paper) is given as below:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & Ax = b \\ & -g(x) \in \mathcal{K} \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{m \times n}$ has full row rank, $b \in \mathbb{R}^m$, $f : \mathbb{R}^n \to \mathbb{R}$ is two-order continuous differentiable and convex mapping, $g = [g_1, \ldots, g_l]^T : \mathbb{R}^n \to \mathbb{R}^l$ is two-order continuous differentiable $\mathcal{K}$-convex mapping which means for every $x, y \in \mathbb{R}^n$ and $t \in [0,1]$ such that

$$tg(x) + (1-t)g(y) - g(tx + (1-t)y) \in \mathcal{K},$$

and $\mathcal{K}$ is a Cartesian product of second-order cones (also called Lorentz cones), expressed as

$$\mathcal{K} = \mathcal{K}^{n_1} \times \mathcal{K}^{n_2} \times \ldots \times \mathcal{K}^{n_N}$$

with $N, n_1, \ldots, n_N \geq 1, n_1 + \cdots + n_N = l$ and

$$\mathcal{K}^{n_i} := \left\{ \left(x_{i1}, x_{i2}, \ldots, x_{in_i}\right)^T \in \mathbb{R}^{n_i} \mid \|(x_{i2}, \ldots, x_{in_i})\| \leq x_{i1} \right\}.$$

Here $\|\cdot\|$ denotes the Euclidean norm and $\mathcal{K}^1$ means the set of nonnegative reals $\mathbb{R}_+$.

It is well known that second-order cone programming problems (SOCP) have wide of applications in engineering, control and management science [1,23,26]. For example, the grasping force optimization problem for the multi-fingered robot hand can be recast as SOCP, see [23, Example 5.3] for real application data. For solving SOCP (1), there also exist many traditional optimization methods such as the interior point method [24], the merit function method [7,18], Newton method [21,31], and projection method [12] and so on. For a survey of solution methods, refer to [4]. In this paper, we are interested in the so-called neural network approach for solving SOCP (1), which is substantially different from the traditional ones. The main motivation to employ this approach arises from the following reason. In many applications, for example, force analysis in robot grasping and control applications, real-time solutions are usually imperative. For such applications, traditional optimization methods may not be competent due to the problem's stringent requirement on computational time. Compared with the traditional optimization methods, the neural network method has its advantage in dealing with real-time optimization problems. Hence, many continuous-time neural networks for constrained optimization problems have been widely developed. At present, there are many results on neural networks for solving real-

* Corresponding author.
E-mail addresses: xinhemiao@tju.edu.cn (X. Miao), jschen@math.ntnu.edu.tw (J.-S. Chen), chko@isu.edu.tw (C.-H. Ko).

[1] The author's work is also supported by National Young Natural Science Foundation (No. 11101302) and National Natural Science Foundation of China (No. 11471241).
[2] The author's work is supported by Ministry of Science and Technology, Taiwan.

http://dx.doi.org/10.1016/j.neucom.2016.04.008
0925-2312/

time optimization problems, see [6,9,11,14,16,17,19,22,23,25,27,33,35–39,41] and references therein.

Neural networks stemmed back from McCulloch and Pitts' pioneering work half century ago, and these were first introduced for optimization domain in the 1980s [15,20,34]. The essence of neural network method for solving optimization problems [8] is to establish a nonnegative Lyapunov function (or called energy function) and a dynamic system which represents an artificial neural network. Indeed, the dynamic system is usually in the form of the first order ordinary differential equations. When utilizing neural networks for solving optimization problems, we are usually much more interested in the stability of networks starting from an arbitrary point. It is expected that for an initial point, the neural network will approach its equilibrium point which corresponds to the solution for the considered optimization problem.

In fact, the neural network approach for solving SOCP has been studied in [23,29]. More specifically, the SOCP studied in [23] is

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & Ax=b \\ & x \in \mathcal{K} \end{aligned} \tag{2}$$

which is a special case of problem (1). Two kinds of neural networks were proposed in [23]. One is based on cone projection function (also called NR function) with which only Lyapunov stability is guaranteed. The other is based on the Fischer–Burmeister function (FB function) where Lyapunov stability and asymptotical stability are proved. Moreover, when solving problem (2), it was observed that the neural network based on the NR function has better performance than the one based on the FB function in most cases (except for some oscillating cases). However, compared to FB function, the NR function has a remarkable drawback, i.e., the non-differentiablity. In light of this phenomenon, the authors employed a neural network model based on “smoothed” NR function for solving more general SOCP (1), see [29]. In addition, all three kinds of stabilities including Lyapunov stability, asymptotical stability, and exponential stability are proved for such model in [29]. Moreover, the neural network based on generalized FB function can be regulated appropriately by perturbing its parameter p. Previous study [6] has demonstrated its efficiency for solving the nonlinear complementarity problems, which also motivates us to further explore its numerical performance for solving the SOCP. In view of the above discussions and the existing literature, we wish to keep tracking the performance of neural networks based on “smoothed” FB function, which is the main motivation of this paper. In particular, we consider a more general function, which is called the generalized FB function. In other words, we propose a neural network model based on the “smoothed” generalized FB function including FB function as a special case. With this function, we perturb the parameter p associated with the generalized FB function to see how it affects the numerical performance. In addition, all the aforementioned three types of stabilities are guaranteed in our proposed neural network. Numerical comparison between model based on smoothed NR function and model based on smoothed generalized FB function are provided.

The organization of this paper is as follows. In Section 2, we introduce concepts about the stability, and recall some background materials. In Section 3, based on the smoothed generalized FB function, the neural network architecture is proposed for solving the problem (1). In Section 4, we study the convergence and stability results of the proposed neural network. Simulation results of the new method are reported in Section 5. Section 6 gives the conclusion of this paper.

2. Preliminaries

For a given mapping $H:\mathbb{R}^n\to\mathbb{R}^n$, the first order differential equation (ODE) means

$$\frac{du}{dt}=H(u(t)), \quad u(t_0)=u_0\in\mathbb{R}^n. \tag{3}$$

In general, the most concerned issues regarding ODE (3) are the existence and uniqueness of the solution. Besides, the convergence of solution trajectory is also concerned. To this end, concepts regarding equilibrium point and stabilities are needed. As below, we recall background materials about ODE (3) as well as stability concepts about the solution to ODE (3). All these materials can be found in usual ODE's textbook, e.g., [30].

Lemma 2.1 (*The existence and uniqueness*). *Assume that $H:\mathbb{R}^n\to\mathbb{R}^n$ is a continuous mapping. Then, for arbitrary $t_0\ge 0$ and $u_0\in\mathbb{R}^n$, there exists a local solution $u(t)$, $t\in[t_0,\tau)$ to (3) for some $\tau>t_0$. Furthermore, if H is locally Lipschitz continuous at u_0, then the solution is unique; and if H is Lipschitz continuous in $\mathbb{R}^n$, then τ can be extended to ∞.*

Proof. See [25, Theorem 2.5]. □

Remark 2.1. For Eq. (3), if a local solution defined on $[t_0,\tau)$ cannot be extended to a local solution on a larger interval $[t_0,\tau_1)$, where $\tau_1>\tau$, then it is called a *maximal solution*, and this interval $[t_0,\tau)$ is the *maximal interval* of existence. It is obvious that an arbitrary local solution has an extension to a maximal one.

Lemma 2.2. *Let $H:\mathbb{R}^n\to\mathbb{R}^n$ is a continuous mapping. If $u(t)$ is a maximal solution, and $[t_0,\tau)$ is the maximal interval of existence associated with u_0 and $\tau<+\infty$, then $\lim_{t\uparrow\tau}\|u(t)\|=+\infty$.*

Proof. See [25, Theorem 2.6]. □

For ODE (3), a point $u^*\in\mathbb{R}^n$ is called an *equilibrium point* of (3) if $H(u^*)=0$. If there is a neighborhood $\Omega\subseteq\mathbb{R}^n$ of u^* such that $H(u^*)=0$ and $H(u)\neq 0$ for any $u\in\Omega\setminus\{u^*\}$, then u^* is called an *isolated equilibrium point*. The following are definitions of various stabilities. More related materials can be found in [25,30,33].

Definition 2.1. Let $u(t)$ be a solution of ODE (3).

(a) An isolated equilibrium point u^* is *Lyapunov stable* (or stability in the sense of Lyapunov) if for any $u_0=u(t_0)$ and $\varepsilon>0$, there exists a $\delta>0$ such that

$$\|u_0-u^*\|<\delta \implies \|u(t)-u^*\|<\varepsilon \quad \text{for } t\ge t_0.$$

(b) Under the condition that an isolated equilibrium point u^* is Lyapunov stable, u^* is said to be *asymptotic stable* if it has the property that if $\|u_0-u^*\|<\delta$, then $u(t)\to u^*$ as $t\to\infty$.

(c) An isolated equilibrium point u^* is **exponentially stable** for (3) if there exist $\omega<0$, $\kappa>0$, $\delta>0$ such that arbitrary solution $u(t)$ of ODE (3) with the initial condition $u(t_0)=u_0$, $\|u_0-u^*\|<\delta$ is defined on $[0,\infty)$ and satisfies

$$\|u(t)-u^*\|\le\kappa e^{\omega t}\|u(t_0)-u^*\|, \quad t\ge t_0.$$

Definition 2.2 (*Lyapunov function*). Let $\Omega\subseteq\mathbb{R}^n$ be an open neighborhood of $\overline{u}$. A continuously differentiable function $g:\mathbb{R}^n\to\mathbb{R}$ is said to be a Lyapunov function (or energy function) at the state $\overline{u}$ (over the set Ω) for Eq. (3) if

$$\begin{cases} g(\overline{u})=0, \\ g(u)>0 \quad \forall u\in\Omega\backslash\{\overline{u}\}, \\ \dfrac{dg(u(t))}{dt}\le 0 \quad \forall u\in\Omega. \end{cases}$$

From the above definition, it is obvious that exponentially stable is asymptotically stable. The next results show the relationship between stabilities and a Lyapunov function, see [5,10,40].

Lemma 2.3.

(a) *An isolated equilibrium point u^* is Lyapunov stable if there exists a Lyapunov function over some neighborhood Ω of u^*.*

(b) *An isolated equilibrium point u^* is asymptotically stable if there exists a Lyapunov function over some neighborhood Ω of u^* satisfying*

$$\frac{dg(u(t))}{dt}<0, \quad \forall u\in\Omega\backslash\{u^*\}.$$

To close this section, we briefly review some properties of the spectral factorization with respect to second-order cone, which will be used in the subsequent analysis. Spectral factorization is one of the basic concepts in Jordan algebra. For more details, see [7,13,31]. For any vector $z=(z_1,z_2)\in\mathbb{R}\times\mathbb{R}^{l-1}$ $(l\ge 2)$, its spectral factorization with respect to second-order cone $\mathcal{K}$ is defined as

$$z=\lambda_1(z)e_1(z)+\lambda_2(z)e_2(z),$$

where $\lambda_i(z)=z_1+(-1)^i\|z_2\|$ $(i=1,2)$ are called the spectral values of z, and

$$e_i(z)=\begin{cases} \dfrac{1}{2}\left(1,(-1)^i\dfrac{z_i}{\|z_i\|}\right), & z_2\ne 0 \\ \dfrac{1}{2}(1,(-1)^i w), & z_2=0 \end{cases}$$

with $w\in\mathbb{R}^{l-1}$ being an arbitrary element such that $\|w\|=1$. Here $e_1(z)$ and $e_2(z)$ are called the spectral vectors of z. It is well known that for any $z\in\mathbb{R}^l$, we have $\lambda_1(z)\le\lambda_2(z)$ and

$$\lambda_1(z)\ge 0 \Longleftrightarrow z\in\mathcal{K}.$$

Note that any closed convex cone can always yield a partial order. Suppose that the partial order "$\succcurlyeq_{\mathcal{K}}$" is induced by $\mathcal{K}$, i.e., $z\succcurlyeq_{\mathcal{K}}0 \Leftrightarrow z\in\mathcal{K}$. The following technical lemma is helpful towards the subsequent analysis.

Lemma 2.4 (*Pan et al. [32, Lemma 2.2]*). *For any $0\le r\le 1$ and $z\succcurlyeq_{\mathcal{K}}w\succcurlyeq_{\mathcal{K}}0$, we have $z^r\succcurlyeq_{\mathcal{K}}w^r$.*

For any $x=(x_1,x_2)\in\mathbb{R}\times\mathbb{R}^{n-1}$ and $y=(y_1,y_2)\in\mathbb{R}\times\mathbb{R}^{n-1}$, Jordan product of $x\circ y$ is defined as

$$x\circ y=\begin{bmatrix}\langle x,y\rangle \\ x_1y_2+y_1x_2\end{bmatrix}.$$

According to Jordan product and spectral factorization with respect to second-order cone $\mathcal{K}$, we often employ the following vector-valued functions (also called SOC-functions) associated with $|t|^p$ $(t\in\mathbb{R})$ and $\sqrt[p]{t}$ $(t\ge 0)$, respectively, which are expressed as

$$|x|^p=|\lambda_1(x)|^pe_1(x)+|\lambda_2(x)|^pe_2(x) \quad \forall x\in\mathbb{R}^n,$$
$$\sqrt[p]{x}=\sqrt[p]{\lambda_1(x)}e_1(x)+\sqrt[p]{\lambda_1(x)}e_2(x) \quad \forall x\in\mathcal{K}.$$

In light of the expressions of $|x|^p$ and $\sqrt[p]{x}$ as above, for any $p>1$, the generalized FB merit function $\phi_p:\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}^n$ associated with second-order cone is defined in [32]:

$$\phi_p(x,y):=\sqrt[p]{|x|^p+|y|^p}-(x+y).$$

In particular, in [32] the authors have shown that $\phi_p(x,y)$ is an SOC-complementarity function, i.e.,

$$\phi_p(x,y)=0 \Longleftrightarrow x\in\mathcal{K},\ y\in\mathcal{K} \quad\text{and}\quad \langle x,y\rangle=0.$$

This also yields that the function $\Phi_p:\mathbb{R}^n\to\mathbb{R}^n$ given by

$$\Phi_p(x):=\frac{1}{2}\|\phi_p(x,F(x))\|^2$$

is a merit function for second-order cone complementarity problems. Moreover, the following conclusions are obtained in [32].

Lemma 2.5. *For any $p>1$, let $w:=w(x,y):=|x|^p+|y|^p$, $t=t(x,y):=\sqrt[p]{w}$ and denote $g^{soc}(x):=|x|^p$. Then, $t(x,y)$ is continuously differentiable at (x,y) with $w\in\text{int}(\mathcal{K})$, and*

$$\nabla_x t(x,y)=\nabla g^{soc}(x)\nabla g^{soc}(t)^{-1}$$

and

$$\nabla_y t(x,y)=\nabla g^{soc}(y)\nabla g^{soc}(t)^{-1}$$

where

$$\nabla g^{soc}(x)=\begin{cases} p\,\text{sign}(x_1)|x_1|^{p-1}I, & x_2=0 \\ \begin{bmatrix} b(x) & c(x)\overline{x}_2^T \\ c(x)\overline{x}_2 & a(x)I+(b(x)-a(x))\overline{x}_2\ \overline{x}_2^T\end{bmatrix}, & x_2\ne 0\end{cases}$$

with $\overline{x}_2=\frac{x_2}{\|x_2\|}$ and

$$a(x)=\frac{|\lambda_2(x)|^p-|\lambda_1(x)|^p}{\lambda_2(x)-\lambda_1(x)},$$
$$b(x)=\frac{p}{2}\left[\text{sign}(\lambda_2(x))|\lambda_2(x)|^{p-1}+\text{sign}(\lambda_1(x))|\lambda_1(x)|^{p-1}\right],$$
$$c(x)=\frac{p}{2}\left[\text{sign}(\lambda_2(x))|\lambda_2(x)|^{p-1}-\text{sign}(\lambda_1(x))|\lambda_1(x)|^{p-1}\right].$$

Proof. See [32, Lemma 3.2].□

Lemma 2.6. *Let Φ_p be defined as $\Phi_p(x,y):=\frac{1}{2}\|\phi_p(x,y)\|^2$ and denote $w(x,y):=|x|^p+|y|^p$, $g^{soc}(x):=|x|^p$. Then, the function Φ_p for $p\in(1,4)$ is differentiable everywhere. Moreover, for any $x,y\in\mathbb{R}^n$,*

(a) *if $w(x,y)=0$, then $\nabla_x\Phi_p(x,y)=\nabla_y\Phi_p(x,y)=0$;*

(b) *if $w(x,y)\in\text{int}(\mathcal{K})$, then*

$$\begin{aligned}\nabla_x\Phi_p(x,y)&=\left(\nabla g^{soc}(x)\nabla g^{soc}(t)^{-1}-I\right)\phi_p(x,y)\\ &=\left(\nabla g^{soc}(x)-\nabla g^{soc}(t)\right)\nabla g^{soc}(t)^{-1}\phi_p(x,y),\ \nabla_y\Phi_p(x,y)\\ &=\left(\nabla g^{soc}(y)\nabla g^{soc}(t)^{-1}-I\right)\phi_p(x,y)\\ &=\left(\nabla g^{soc}(y)-\nabla g^{soc}(t)\right)\nabla g^{soc}(t)^{-1}\phi_p(x,y).\end{aligned}$$

(c) *if $w(x,y)\in\partial\mathcal{K}\backslash\{0\}$, where $\partial\mathcal{K}$ means the boundary of $\mathcal{K}$, then*

$$\nabla_x\Phi_p(x,y)=\left(\frac{\text{sign}(x_1)|x_1|^{p-1}}{\sqrt[q]{|x_1|^p+|y_1|^p}}-1\right)\phi_p(x,y),$$
$$\nabla_y\Phi_p(x,y)=\left(\frac{\text{sign}(y_1)|y_1|^{p-1}}{\sqrt[q]{|x_1|^p+|y_1|^p}}-1\right)\phi_p(x,y).$$

Proof. See [32, Proposition 3.1].□

3. Generalized FB neural network model

In this section, we will explain how we form the dynamic system. As is mentioned earlier, the key points for neural network method lie in constructing the dynamic system and Lyapunov function. To this end, we first look into the KKT conditions of the problem (1) which are presented as below:

$$\begin{cases} \nabla f(x) - A^T y + \nabla g(x) z = 0, \\ z \in \mathcal{K}, \quad -g(x) \in \mathcal{K}, \quad z^T g(x) = 0, \\ Ax - b = 0, \end{cases} \tag{4}$$

where $y \in \mathbb{R}^m$, $\nabla g(x)$ denotes the gradient matrix of g. According to the KKT condition, it is well known that if the problem (1) satisfies Slater's condition, which means there exists a strictly feasible point for the problem (1), i.e., there exists an $x \in \mathbb{R}^n$ such that $-g(x) \in \text{int}(\mathcal{K})$ and $Ax = b$. Then, for the nonlinear convex programs (1), x^* is a solution of the problem (1) if and only if there exist y^* and z^* such that (x^*, y^*, z^*) satisfying the KKT conditions (4), see [2]. Hence, we assume that the problem (1) satisfies Slater's condition in this paper.

Lemma 3.1. *For $z = (z_1, z_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$ and $x = (x_1, x_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$ with $z \succcurlyeq_{\mathcal{K}} x$, we have $\lambda_i(z) \geq \lambda_i(x)$ for $i = 1, 2$.*

Proof. Since $z \succcurlyeq_{\mathcal{K}} x$, we may express $z = x + y$ where $x = (x_1, x_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$, $y = (y_1, y_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$ and $y = z - x \succcurlyeq_{\mathcal{K}} 0$. This implies $y_1 \geq \|y_2\|$ and

$$\begin{aligned} \lambda_1(z) &= (x_1 + y_1) - \|x_2 + y_2\| \\ &\geq (x_1 + y_1) - \|x_2\| - \|y_2\| \\ &\geq x_1 - \|x_2\| = \lambda_1(x). \end{aligned}$$

Thus, we have

$$\begin{aligned} \lambda_2(z) &= (x_1 + y_1) + \|x_2 + y_2\| \geq (x_1 + y_1) + |\,\|x_2\| - \|y_2\|\,| \\ &= \begin{cases} x_1 + y_1 + \|x_2\| - \|y_2\|, & \text{if} \quad \|x_2\| \geq \|y_2\| \\ x_1 + y_1 - \|x_2\| + \|y_2\|, & \text{if} \quad \|x_2\| < \|y_2\| \end{cases} \\ &\geq \begin{cases} x_1 + \|x_2\|, & \text{if} \quad \|x_2\| \geq \|y_2\| \\ x_1 + y_1, & \text{if} \quad \|x_2\| < \|y_2\| \end{cases} \\ &\geq x_1 + \|x_2\| = \lambda_2(x) \end{aligned}$$

which is the desired result. □

Lemma 3.2. *Let $w := w(x, y) = |x|^p + |y|^p$, $t = t(x, y) := \sqrt[q]{w}$ and $g^{soc}(x) := |x|^p$. Then, the following three matrices*

$$\nabla g^{soc}(t) - \nabla g^{soc}(x), \nabla g^{soc}(t) - \nabla g^{soc}(y),$$
$$\left(\nabla g^{soc}(t) - \nabla g^{soc}(x)\right)\left(\nabla g^{soc}(t) - \nabla g^{soc}(y)\right)$$

are all positive semi-definite for $p = \frac{n}{2}$ with $n \in \mathbb{N}$.

Proof. From the expression of $\nabla g^{soc}(x)$ in Lemma 2.5 and the proof of [32, Lemma 3.2], we know that the eigenvalues of $\nabla g^{soc}(x)$ for $x_2 \neq 0$ are

$$b(x) - c(x), a(x), \ldots, a(x), \quad \text{and} \quad b(x) + c(x).$$

Let $w := (w_1, w_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$. Then applying [32, Lemma 3.1] gives

$$\begin{aligned} w_1 &= \frac{|\lambda_2(x)|^p + |\lambda_1(x)|^p}{2} + \frac{|\lambda_2(y)|^p + |\lambda_1(y)|^p}{2} \\ w_2 &= \frac{|\lambda_2(x)|^p - |\lambda_1(x)|^p}{2}\bar{x}_2 + \frac{|\lambda_2(y)|^p - |\lambda_1(y)|^p}{2}\bar{y}_2, \end{aligned}$$

where $\bar{x}_2 = \frac{x_2}{\|x_2\|}$ if $x_2 \neq 0$, and otherwise $\bar{x}_2$ is an arbitrary vector in $\mathbb{R}^{n-1}$ satisfying $\|\bar{x}_2\| = 1$. Similar situation applies for $\bar{y}_2$. Thus, we will proceed the proof by discussing two cases: $w_2 = 0$ or $w_2 \neq 0$.

Case 1: For $w_2 = 0$, we have $\nabla g^{soc}(t) = p\sqrt[q]{w_1} I$ where

$$w_1 = \frac{|\lambda_2(x)|^p + |\lambda_1(x)|^p}{2} + \frac{|\lambda_2(y)|^p + |\lambda_1(y)|^p}{2}. \tag{5}$$

Under the condition of $w_2 = 0$, there are the following two subcases.

(i) If $x_2 = 0$, then $w_1 = |x_1|^p + \frac{|\lambda_2(y)|^p + |\lambda_1(y)|^p}{2}$, which implies that $p\sqrt[q]{w_1} \geq p\text{sign}(x_1)|x_1|^{p-1}$. Hence, we see that the matrix $\nabla g^{soc}(t) - \nabla g^{soc}(x)$ is positive semi-definite. Indeed, if $x \neq 0$, $\nabla g^{soc}(t) - \nabla g^{soc}(x)$ is positive definite.

(ii) If $x_2 \neq 0$, it follows from $w_2 = 0$ that

$$\left|\frac{|\lambda_2(x)|^p - |\lambda_1(x)|^p}{2}\right| = \left|\frac{|\lambda_2(y)|^p - |\lambda_1(y)|^p}{2}\right|. \tag{6}$$

We want to prove that the matrix $\nabla g^{soc}(t) - \nabla g^{soc}(x)$ is positive semi-definite. It is sufficient to show that

$$p\sqrt[q]{w_1} \geq \max\{b(x) - c(x), a(x), b(x) + c(x)\}.$$

It is obvious that $p\sqrt[q]{w_1} - (b(x) - c(x)) > 0$ when $\lambda_1(x) < 0$. When $\lambda_1(x) \geq 0$, using (5) and $\lambda_2(x) \geq \lambda_1(x)$, we have

$$\begin{aligned} p\sqrt[q]{w_1} - (b(x) - c(x)) &\geq p\sqrt[q]{|\lambda_1(x)|^p} - p\text{sign}(\lambda_1(x))|\lambda_1(x)|^{p-1} \\ &\geq 0. \end{aligned}$$

Next, we verify that $p\sqrt[q]{w_1} - a(x) \geq 0$. For $|\lambda_1(x)| \geq |\lambda_2(x)|$, it is clear that $p\sqrt[q]{w_1} - a(x) \geq 0$. For $|\lambda_1(x)| < |\lambda_2(x)|$, it follows from $\lambda_2(x) \geq \lambda_1(x)$ that $x_1 > 0$, which yields

$$\frac{|\lambda_2(x)|^p - |\lambda_1(x)|^p}{\lambda_2(x) - \lambda_1(x)} \leq \frac{\lambda_2(x)^p - |\lambda_1(x)|^p}{\lambda_2(x) - |\lambda_1(x)|}.$$

Let $p = \frac{n}{m}$ $(n, m \in \mathbb{N})$, $a = \lambda_2(x)^{\frac{1}{m}}$ and $b = |\lambda_1(x)|^{\frac{1}{m}}$. From $p > 1$, it follows that $n > m$. Then, we have $0 \leq b < a$ and

$$a(x) = \frac{a^n - b^n}{a^m - b^m} = \frac{a^{n-1} + a^{n-2}b + \ldots + ab^{n-2} + b^{n-1}}{a^{m-1} + a^{m-2}b + \ldots + ab^{m-2} + b^{m-1}}.$$

Now, letting $f(v) = \frac{a^n - v^n}{a^m - v^m}$ with $v \in [0, a]$, we obtain

$$f'(v) = \frac{-nv^{n-1}(a^m - v^m) + mv^{m-1}(a^n - v^n)}{(a^m - v^m)^2}.$$

In addition, it follows from $f'(v) = 0$ that

$$\frac{a^n - v^n}{a^m - v^m} = \frac{n}{m} v^{n-m}.$$

Since $f(0) = \frac{a^n}{a^m} = a^{n-m}$ with $v = 0$ and $f(a) = \frac{n}{m} a^{n-m}$ with $v = a$, it is easy to verify that $f(b) \leq \frac{n}{m} a^{n-m}$ for $0 \leq b < a$, i.e.,

$$\frac{|\lambda_2(x)|^p - |\lambda_1(x)|^p}{\lambda_2(x) - \lambda_1(x)} \leq p \quad |\lambda_2(x)|^{p-1}.$$

Hence, we have

$$\begin{aligned} p\sqrt[q]{w_1} - a(x) &\geq p\sqrt[q]{\max\{|\lambda_2(x)|^p, |\lambda_1(x)|^p\} + \min\{|\lambda_2(y)|^p, |\lambda_1(y)|^p\}} \\ &\quad - \frac{|\lambda_2(x)|^p - |\lambda_1(x)|^p}{\lambda_2(x) - \lambda_1(x)} \\ &\geq p\sqrt[q]{\lambda_2(x)^p} - p|\lambda_2(x)|^{p-1} \\ &\geq 0, \end{aligned}$$

where the first inequality holds due to (6). Lastly, we also see that

$$\begin{aligned} p\sqrt[q]{w_1} - (b(x) + c(x)) &\geq p\sqrt[q]{\max\{|\lambda_2(x)|^p, |\lambda_1(x)|^p\} + \min\{|\lambda_2(y)|^p, |\lambda_1(y)|^p\}} \\ &\quad - p\text{sign}(\lambda_2(x))|\lambda_2(x)|^{p-1} \\ &\geq p\sqrt[q]{\max\{|\lambda_2(x)|^p, |\lambda_1(x)|^p\}} \\ &\quad - p\text{sign}(\lambda_2(x))|\lambda_2(x)|^{p-1} \\ &\geq 0. \end{aligned}$$

To sum up, under this case $x_2 \neq 0$, we prove that the matrix $\nabla g^{soc}(t) - \nabla g^{soc}(x)$ is positive semi-definite.

Case 2: For $w_2 \neq 0$, from the expression of $t(x, y)$ and the properties of the spectral values of the vector-valued function $|x|^p$ with $p = \frac{n}{2}$ for $n \in \mathbb{N}$, all the eigenvalues of the matrix $\nabla g^{soc}(t)$ are

$$b(t) - c(t) \leq a(t) \leq b(t) + c(t). \tag{7}$$

When $x_2=0$, we note that

$$\begin{aligned}
b(t)-c(t)-p\text{sign}(x_1)|x_1|^{p-1} &= p\left[\sqrt[p]{\lambda_1(w)}\right]^{p-1}-p\text{sign}(x_1)|x_1|^{p-1}\\
&= p\Bigg[\frac{|\lambda_2(x)|^p+|\lambda_1(x)|^p}{2}+\frac{|\lambda_2(y)|^p+|\lambda_1(y)|^p}{2}\\
&\quad -\left\|\frac{|\lambda_2(x)|^p-|\lambda_1(x)|^p}{2}\bar{x}_2+\frac{|\lambda_2(y)|^p-|\lambda_1(y)|^p}{2}\bar{y}_2\right\|\Bigg]^{\frac{p-1}{p}}\\
&\quad -p\text{sign}(x_1)|x_1|^{p-1}\\
&\geq p|x_1|^{p-1}-p\text{sign}(x_1)|x_1|^{p-1}\\
&\geq 0,
\end{aligned}$$

where $\bar{y}_2$ denotes $\bar{y}_2=\frac{y_2}{\|y_2\|}$ when $y_2\neq 0$, and otherwise $\bar{y}_2$ is an arbitrary vector in $\mathbb{R}^{n-1}$ satisfying $\|\bar{y}_2\|=1$. Now, applying the relation of the eigenvalues in (7), we have

$$b(t)+c(t)\geq a(t)\geq p\text{sign}(x_1)|x_1|^{p-1},$$

which implies that the matrix $\nabla g^{soc}(t)-\nabla g^{soc}(x)$ is positive semi-definite.

When $x_2\neq 0$, we also note that

$$b(t)-c(t)-(b(x)-c(x))=p\left[\sqrt[p]{\lambda_1(w)}\right]^{p-1}-p\text{sign}(\lambda_1(x_1))|\lambda_1(x_1)|^{p-1}.$$

For $\lambda_1(x)<0$, it is clear that $b(t)-c(t)-(b(x)-c(x))\geq 0$. For $\lambda_1(x)\geq 0$, we have $\lambda_2(x)\geq\lambda_1(x)\geq 0$, which leads to

$$\begin{aligned}
\lambda_1(w) &= \frac{|\lambda_2(x)|^p+|\lambda_1(x)|^p}{2}+\frac{|\lambda_2(y)|^p+|\lambda_1(y)|^p}{2}\\
&\quad -\left\|\frac{|\lambda_2(x)|^p-|\lambda_1(x)|^p}{2}\bar{x}_2+\frac{|\lambda_2(y)|^p-|\lambda_1(y)|^p}{2}\bar{y}_2\right\|\\
&\geq \frac{|\lambda_2(x)|^p+|\lambda_1(x)|^p}{2}-\frac{|\lambda_2(x)|^p-|\lambda_1(x)|^p}{2}\\
&\quad +\frac{|\lambda_2(y)|^p+|\lambda_1(y)|^p}{2}-\left|\frac{|\lambda_2(y)|^p-|\lambda_1(y)|^p}{2}\right|\\
&\geq |\lambda_1(x)|^p.
\end{aligned}$$

Thus, it follows that $b(t)-c(t)-(b(x)-c(x))\geq 0$. Moreover, since $t\succcurlyeq_{\mathcal{K}}|x|$, by Lemma 3.1 and the eigenvalue of $|x|$ being $|\lambda_1(x)|$ and $|\lambda_2(x)|$, we have

$$\begin{aligned}
&\lambda_2(t)\geq\max\{|\lambda_1(x)|,|\lambda_2(x)|\}\\
&\text{and}\lambda_1(t)\geq\min\{|\lambda_1(x)|,|\lambda_2(x)|\}.
\end{aligned}\tag{8}$$

When $p=\frac{n}{2}$ with $n\in\mathbb{N}$, then, we have

$$a(t)-a(x)=\frac{\lambda_2(t)^{\frac{n}{2}}-\lambda_1(t)^{\frac{n}{2}}}{\lambda_2(t)-\lambda_1(t)}-\frac{|\lambda_2(x)|^{\frac{n}{2}}-|\lambda_1(x)|^{\frac{n}{2}}}{\lambda_2(x)-\lambda_1(x)}.$$

If $|\lambda_2(x)|<|\lambda_1(x)|$, it is obvious that $a(t)-a(x)\geq 0$. If $|\lambda_2(x)|\geq|\lambda_1(x)|$, in light of $\lambda_2(x)\geq\lambda_1(x)$, we obtain that $x_1\geq 0$ and $\lambda_2(x)\geq 0$. Now, let

$$a:=\lambda_2(t)^{\frac{1}{2}},\quad b:=\lambda_1(t)^{\frac{1}{2}},\quad c:=\lambda_2(x)^{\frac{1}{2}}\quad\text{and}\quad d:=|\lambda_1(t)|^{\frac{1}{2}}.$$

Then, we get that

$$\begin{aligned}
a(t)-a(x) &= \frac{a^n-b^n}{a^2-b^2}-\frac{c^n-d^n}{c^2-d^2}\\
&= \frac{(a^{n-1}+a^{n-2}b+\ldots+ab^{n-2}+b^{n-1})(c+d)}{(a+b)(c+d)}\\
&\quad -\frac{(a+b)(c^{n-1}+c^{n-2}d+\ldots+cd^{n-2}+d^{n-1})}{(a+b)(c+d)}\\
&= \frac{a^{n-1}c+bc(a^{n-2}+a^{n-3}b+\ldots+ab^{n-3}+b^{n-2})}{(a+b)(c+d)}\\
&\quad +\frac{ad(a^{n-2}+a^{n-3}b+\ldots+ab^{n-3}+b^{n-2})+b^{n-1}d}{(a+b)(c+d)}\\
&\quad -\frac{ac^{n-1}+ad(c^{n-2}+c^{n-3}d+\ldots+cd^{n-3}+d^{n-2})}{(a+b)(c+d)}\\
&\quad -\frac{bc(c^{n-2}+c^{n-3}d+\ldots+cd^{n-3}+d^{n-2})+bd^{n-1}}{(a+b)(c+d)},
\end{aligned}$$

which together with (8) implies that

$$a\geq c,\quad b\geq d\geq 0\quad\text{and}\quad a(t)-a(x)\geq 0.$$

In addition, we also verity that

$$\begin{aligned}
b(t)+c(t)-(b(x)+c(x)) &= p(\lambda_2(t))^{p-1}-p\text{sign}(\lambda_2(x))|\lambda_2(x)|^{p-1}\\
&\geq 0.
\end{aligned}$$

Therefore, for any $x\in\mathbb{R}^n$, we have

$$\begin{aligned}
x^T(\nabla g^{soc}(t)-\nabla g^{soc}(x))x &= x^T\nabla g^{soc}(t)x-x^T\nabla g^{soc}(x)x\\
&= [b(t)-c(t)+(n-2)a(t)+b(t)+c(t)]x^Tx\\
&\quad -[b(x)-c(x)+(n-2)a(x)+b(x)+c(x)]x^Tx\\
&\geq 0,
\end{aligned}$$

which shows that the matrix $\nabla g^{soc}(t)-\nabla g^{soc}(x)$ is positive semi-definite.

With the same arguments, we can verify that the matrix $\nabla g^{soc}(t)-\nabla g^{soc}(y)$ is also positive semi-definite.

Finally, using the properties of eigenvalues of symmetric matrix product, i.e.,

$$\lambda_i(AB)\geq\lambda_i(A)\lambda_{\min}(B),\quad i=1,\ldots,n,\ \forall A,B\in S^{n\times n},$$

where $S^{n\times n}$ denotes n order symmetric matrix, we easily obtain that the matrix $(\nabla g^{soc}(t)-\nabla g^{soc}(x))(\nabla g^{soc}(t)-\nabla g^{soc}(y))$ is also positive semi-definite.□

Remark 3.1. From the above proof of Lemma 3.2, when $x\neq 0$ and $y\neq 0$, we have that the matrixes $\nabla g^{soc}(t)-\nabla g^{soc}(x)$, $\nabla g^{soc}(t)-\nabla g^{soc}(y)$ and $(\nabla g^{soc}(t)-\nabla g^{soc}(x))(\nabla g^{soc}(t)-\nabla g^{soc}(y))$ are all positive definite.

Now, we look into the KKT conditions (4) of the problem (1). Let

$$L(x,y,z)=\nabla f(x)-A^Ty+\nabla g(x)z,$$

$$H(u):=\begin{bmatrix} Ax-b\\ L(x,y,z)\\ \phi_p(z,-g(x))\end{bmatrix}\tag{9}$$

and

$$\Psi_p(u):=\frac{1}{2}\|H(u)\|^2=\frac{1}{2}\|\phi_p(z,-g(x))\|^2+\frac{1}{2}\|L(x,y,z)\|^2+\frac{1}{2}\|Ax-b\|^2,$$

where $u=(x^T,y^T,z^T)^T\in\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^l$. From Lemma 2.5 in [32], we know that

$$\phi_p(z,-g(x))=0\Longleftrightarrow z\in\mathcal{K},\quad -g(x)\in\mathcal{K},\quad -z^Tg(x)=0.$$

Hence, the KKT conditions (4) are equivalent to $H(u)=0$, i.e., $\Psi_p(u)=0$. Then, it follows that the KKT conditions (4) are equivalent to the following unconstrained minimization problem with zero optimal value via the merit function approach:

$$\min\ \Psi_p(u):=\frac{1}{2}\|H(u)\|^2.\tag{10}$$

However, the function ϕ_p is not $\mathcal{K}$-convex and the merit function Ψ_p is neither convex function for $p=2$, which is showed in Example 3.5 of [3].

Theorem 3.1. *Let Ψ_p be defined as in (10).*

(a) *The matrix $\nabla g^{soc}(x)$ is positive definite for all $0\neq x\in\mathcal{K}$.*

(b) *The function Ψ_p for $p \in (1,4)$ is continuously differentiable everywhere. Moreover, $\nabla \Psi_p(u) = \nabla H(u) H(u)$ where*

$$\nabla H(u) = \begin{bmatrix} A^T & \nabla_x L(x,y,z) & -\nabla g(x) V_1 \\ 0 & -A & 0 \\ 0 & \nabla g(x)^T & V_2 \end{bmatrix} \tag{11}$$

with

$$V_1 = \begin{cases} 0, & w(z,-g(x)) = |z|^p + |-g(x)|^p = 0, \\ \nabla g^{soc}(x) \nabla g^{soc}(t)^{-1} - I, & w(z,-g(x)) \in \mathrm{int}(\mathcal{K}), \\ \dfrac{\mathrm{sign}(-g_1(x))|-g_1(x)|^{p-1}}{\sqrt[p]{|-g_1(x)|^p + |z_1|^p}} - 1, & w(z,-g(x)) \in \partial\mathcal{K} \backslash \{0\} \end{cases}$$

and

$$V_2 = \begin{cases} 0, & w(z,-g(x)) = |z|^p + |-g(x)|^p = 0, \\ \nabla g^{soc}(z) \nabla g^{soc}(t)^{-1} - I, & w(z,-g(x)) \in \mathrm{int}(\mathcal{K}), \\ \dfrac{\mathrm{sign}(z_1)|z_1|^{p-1}}{\sqrt[p]{|-g_1(x)|^p + |z_1|^p}} - 1, & w(z,-g(x)) \in \partial\mathcal{K} \backslash \{0\} \end{cases}$$

with $t := \sqrt[p]{w(z,-g(x))}$.

Proof. (a) For all $0 \neq x \in \mathcal{K}$, if $x_2 = 0$, it is obvious that the matrix $\nabla g^{soc}(x) = p\mathrm{sign}(x_1)|x_1|^{p-1} I$ is positive definite. If $x \neq 0$, from the expression of $\nabla g^{soc}(x)$ in Lemma 2.5 and $x \in \mathcal{K}$, we have $b(x) > 0$. In order to prove that the matrix $\nabla g^{soc}(x)$ is positive definite, it suffices to show that the Schur complement of $b(x)$ in the matrix $\nabla g^{soc}(x)$ is positive definite. In fact, from the expression of $\nabla g^{soc}(x)$, the Schur complement has the form

$$a(x)I + (b(x) - a(x))\overline{x}_2\,\overline{x}_2^T - \frac{c^2(x)}{b(x)}\overline{x}_2\,\overline{x}_2^T = a(x)(I - \overline{x}_2\,\overline{x}_2^T) + b(x)\left(1 - \frac{c^2(x)}{b(x)}\right)\overline{x}_2\,\overline{x}_2^T.$$

Since $x \in \mathcal{K}$, we have $\lambda_2(x) \geq \lambda_1(x) \geq 0$, which implies that $a(x) > 0$ and $b(x) > c(x) \geq 0$. Note that the matrices $I - \overline{x}_2\,\overline{x}_2^T$ and $\overline{x}_2\,\overline{x}_2^T$ are positive semi-definite. Thus, the Schur complement is positive definite. Further, we get that $\nabla g^{soc}(x)$ is positive definite for all $0 \neq x \in \mathcal{K}$.

(b) From the proof of Proposition 3.1 and Lemma 3.2 of [32], we know that the function Ψ_p for $p \in (1,4)$ is continuously differentiable everywhere. Hence, in view of the definition of the function Ψ_p and the chain rule, the expression of $\nabla \Psi_p(u)$ is obtained.□

In light of the main ideas for constructing artificial neural networks (see [8] for details), we will establish a specific first order ordinary differential equation, i.e., an artificial neural network. Moreover, specifically, based on the gradient of the merit function Ψ_p in minimization problem (10), we propose the neural network for solving the KKT system (4) of nonlinear SOCP (1) with the following differential equation:

$$\frac{du(t)}{dt} = -\rho \nabla \Psi_p(u), \quad u(t_0) = u_0, \tag{12}$$

where $\rho > 0$ is a time scaling factor. In fact, if $\tau = \rho t$, then $\frac{du(t)}{dt} = \rho \frac{du(\tau)}{d\tau}$. Hence, it follows from (12) that $\frac{du(\tau)}{d\tau} = -\nabla \Psi_p(u)$. For simplicity and convenience, we set $\rho = 1$ in this paper.

4. Stability analysis

In this section, we are interested in the stability analysis about the proposed neural network (12). By these theoretical analyses, the desired optimal solution of SOCP (1) can always be obtained by setting the initial state of the network of an arbitrary value. In order to study the stability issues on the proposed neural network (12) for solving SOCP (1), we first make an assumption which will be needed in our subsequent analysis, in order to avoid the singularity of $\nabla H(u)$.

Assumption 4.1.

(a) The SOCP problem (1) satisfies Slater's condition.
(b) The matrix $[A^T \ \nabla g(x)]$ is full column rank, and the matrix $\nabla_x L(x,y,z)$ is positive definite on the null space $\{t \mid At = 0\}$ of A.

Here we say a few words about Assumption 4.1(a) and (b). Slater's condition is a standard condition which is widely used in optimization field. When g is linear, Assumption 4.1(b) is indeed equivalent to the well-used condition $\nabla^2 f(x)$ is positive definite.

Lemma 4.1. *Let $p = \frac{n}{2} \in (1,4)$ with $n \in \mathbb{N}$. Then, the following hold.*

(a) *Under the condition of Assumption 4.1, $\nabla H(u)$ is nonsingular for $u = (x,y,z) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$ with $(z,-g(x)) \neq 0$.*
(b) *Every stationary point of Ψ_p is a global minimizer of problem (10) for $(z,-g(x)) \neq 0$.*
(c) *$\Psi_p(u(t))$ is nonincreasing with respect to t.*

Proof. (a) Suppose $\xi = (s,t,v) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$. From the expression (11) of $\nabla H(u)$ in Theorem 3.1, to show the nonsingularity of $\nabla H(u)$, it is enough to prove that

$$\nabla H(u)\xi = 0 \implies s = 0, \ t = 0 \text{ and } v = 0.$$

Indeed, by $\nabla H(u)\xi = 0$, we have

$$-At = 0, \quad A^T s + \nabla_x L(x,y,z)\, t - \nabla g(x) V_1 v = 0 \tag{13}$$

and

$$\nabla g(x)^T t + V_2 v = 0. \tag{14}$$

From (13), it follows that

$$t^T \nabla_x L(x,y,z) t - t^T \nabla g(x) V_1 v = 0. \tag{15}$$

Moveover, by Eq. (14), we obtain

$$t^T \nabla g(x) = -v^T V_2^T. \tag{16}$$

Then, combining (15) and (16), this yields that

$$t^T \nabla_x L(x,y,z) t + v^T V_2^T V_1 v = 0.$$

By Lemma 3.2 and Assumption 4.1(b), it is not hard to see that t=0. In addition, from (13) and (14), we have

$$A^T s - \nabla g(x) V_1 v = 0 \quad \text{and} \quad V_2 v = 0.$$

By Assumption 4.1(b) again, we also get that

$$s = 0 \quad \text{and} \quad V_1 v = 0.$$

Thus, combining Lemma 3.2 with the expression V_1 and V_2 in Theorem 3.1, we have v=0. Therefore, $\nabla H(u)^T$ is nonsingular.

(b) Suppose that u^* is a stationary point of Ψ_p. This says $\nabla \Psi_p(u^*) = 0$, and from Theorem 3.1, we have $\nabla H(u^*) H(u^*) = 0$. According to part(a), $\nabla H(u)$ is nonsingular. Hence, it follows that $H(u^*) = 0$, i.e., $\Psi_p(u^*) = 0$, which says u^* is a global minimizer of (10).

(c) By the definition of $\Psi_p(u(t))$ and (12), it is clear that

$$\frac{d\Psi_p(u(t))}{dt} = \nabla \Psi_p(u(t)) \frac{du(t)}{dt} = -\rho \left\| \nabla \Psi_p(u(t)) \right\|^2 \leq 0.$$

Therefore, $\Psi_p(u(t))$ is nonincreasing with respect to t.□

Proposition 4.1. *Assume that $\nabla H(u)$ is nonsingular for any $u \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$ and $p = \frac{n}{2} \in (1,4)$ with $n \in \mathbb{N}$. Then,*

(a) *(x^*, y^*, z^*) satisfies the KKT conditions (4) if and only if (x^*, y^*, z^*) is an equilibrium point of the neural network (12);*

(b) *under Slater's condition, x^* is a solution of the problem (1) if and only if (x^*, y^*, z^*) is an equilibrium point of the neural network (12).*

Proof. (a) It is easy to prove that (x^*, y^*, z^*) satisfies the KKT conditions (4) if and only if $H(u^*) = 0$ where $u^* = (x^*, y^*, z^*)^T$. According to the condition that $\nabla H(u)$ is nonsingular, we have that $H(u^*) = 0$ if and only if $\nabla\Psi_p(u^*) = \nabla H(u^*)^T H(u^*) = 0$. Then the desired result follows.

(b) Under Slater's condition, it is well known that x^* is a solution of the problem (1) if and only if there exist y^* and z^* such that (x^*, y^*, z^*) satisfying the KKT conditions (4). Hence, by part (a), it follows that (x^*, y^*, z^*) is an equilibrium point of the neural network (12).□

The next result addresses the existence and uniqueness of the solution trajectory of the neural network (12).

Theorem 4.1. *For any fixed $p = \frac{n}{2} \in (1, 4)$ with $n \in \mathbb{N}$, the following hold.*

(a) *For any initial point $u_0 = u(t_0)$, there exists a unique continuously maximal solution $u(t)$ with $t \in [t_0, \tau)$ for the neural network (12), where $[t_0, \tau)$ is the maximal interval of existence.*
(b) *If the level set $\mathcal{L}(u_0) := \{u \mid \Psi_p(u) \le \Psi_p(u_0)\}$ is bounded, then τ can be extended to $+\infty$.*

Proof. This proof is exactly the same as the proof of [33, Proposition 3.4]. Hence, we omit it here.□

Theorem 4.2. *Assume that $\nabla H(u)$ is nonsingular and u^* is an isolated equilibrium point of the neural network (12). Then, the solution of the neural network (12) with any initial point u_0 is Lyapunov stable.*

Proof. From Lemma 2.3, we only need to argue that there exists a Lyapunov function over some neighborhood Ω of u^*. To this end, we consider the smoothed merit function for $p = \frac{n}{2} \in (1, 4)$ with $n \in \mathbb{N}$

$$\Psi_p(u) = \frac{1}{2}\|H(u)\|^2.$$

Since u^* is an isolated equilibrium point of (12), there is a neighborhood Ω of u^* such that

$$\nabla\Psi_p(u^*) = 0 \quad \text{and} \quad \nabla\Psi_p(u(t)) \neq 0, \quad \forall u(t) \in \Omega\backslash\{u^*\}.$$

By the nonsingularity of $\nabla H(u)$ and the definition of Ψ_p, it is easy to obtain that $\Psi_p(u^*) = 0$. From the definition of Ψ_p, we claim that $\Psi_p(u(t)) > 0$ for any $u(t) \in \Omega\backslash\{u^*\}$, where Ω is a neighborhood of u^*. If not, that is, $\Psi_p(u(t)) = 0$, it follows that $H(u(t)) = 0$. Then, we have $\nabla\Psi_p(u(t)) = 0$, which contradicts with the assumption that u^* is an isolated equilibrium point of (12). Thus, $\Psi_p(u(t)) > 0$ for any $u(t) \in \Omega\backslash\{u^*\}$. Moreover, by the proof of Lemma 4.1(c), we know that for any $u(t) \in \Omega$

$$\frac{d\Psi_p(u(t))}{dt} = \nabla\Psi_p(u(t))\frac{du(t)}{dt} = -\rho\|\nabla\Psi_p(u(t))\|^2 \le 0. \tag{17}$$

Therefore, the function Ψ_p is a Lyapunov function over Ω. This implies that u^* is Lyapunov stable for the neural network (12).□

Theorem 4.3. *Assume that $\nabla H(u)$ is nonsingular and u^* is an isolated equilibrium point of the neural network (12). Then, u^* is asymptotically stable for neural network (12).*

Proof. From the proof of Theorem 4.2, we consider again the Lyapunov function Ψ_p for $p = \frac{n}{2} \in (1, 4)$ with $n \in \mathbb{N}$. By Lemma 2.3 again, we only need to verify that the Lyapunov function Ψ_p over some neighborhood Ω of u^* satisfies

$$\frac{d\Psi_p(u(t))}{dt} < 0, \quad \forall u(t) \in \Omega\backslash\{u^*\}. \tag{18}$$

In fact, by using (17) and the definition of the isolated equilibrium point, it is not hard to check that Eq. (18) is true. Hence, u^* is asymptotically stable.□

Theorem 4.4. *Assume that u^* is an isolated equilibrium point of the neural network (12). If $\nabla H(u)^T$ is nonsingular for any $u = (x, y, z) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$, then u^* is exponentially stable for the neural network (12).*

Proof. From the definition of $H(u)$ and Lemma 2.6, we have

$$H(u) = H(u^*) + \nabla H(u(t))^T(u - u^*) + o(\|u - u^*\|), \quad \forall u \in \Omega\backslash\{u^*\}, \tag{19}$$

where $\nabla H(u(t))^T \in \partial H(u(t))$ and Ω is the neighborhood of u^*. Now, letting

$$g(u(t)) = \|u(t) - u^*\|^2, \quad t \in [t_0, \infty),$$

we have

$$\begin{aligned}\frac{dg(u(t))}{dt} &= 2\big(u(t) - u^*\big)^T\frac{du(t)}{dt} \\ &= -2\rho\big(u(t) - u^*\big)^T\nabla\Psi_p(u(t)) \\ &= -2\rho\big(u(t) - u^*\big)^T\nabla H(u)H(u).\end{aligned} \tag{20}$$

Substituting (19) into (20) yields

$$\begin{aligned}\frac{dg(u(t))}{dt} &= -2\rho\big(u(t) - u^*\big)^T\nabla H(u(t)) < \big(H(u^*) \\ &\quad + \nabla H(u(t))^T(u(t) - u^*) + o(\|u(t) - u^*\|)\big) \\ &= -2\rho\big(u(t) - u^*\big)^T\nabla H(u(t))\nabla H(u(t))^T\big(u(t) - u^*\big) \\ &\quad + o(\|u(t) - u^*\|^2).\end{aligned}$$

Since $\nabla H(u)$ and $\nabla H(u)^T$ are nonsingular, we claim that there exists an $\kappa > 0$ such that

$$\big(u(t) - u^*\big)^T\nabla H(u)\nabla H(u)^T\big(u(t) - u^*\big) \ge \kappa\|u(t) - u^*\|^2. \tag{21}$$

Otherwise, if $(u(t) - u^*)^T\nabla H(u(t))\nabla H(u(t))^T(u(t) - u^*) = 0$, it implies that

$$\nabla H(u(t))^T(u(t) - u^*) = 0.$$

Indeed, from the nonsingularity of $H(u)$, we have $u(t) - u^* = 0$, i.e., $u(t) = u^*$, which contradicts with the assumption of u^* that is an isolated equilibrium point. Therefore, there exists an $\kappa > 0$ such that (21) holds. Moreover, for $o(\|u(t) - u^*\|^2)$, there is $\varepsilon > 0$ such that $o(\|u(t) - u^*\|^2) \le \varepsilon\|u(t) - u^*\|^2$. Hence,

$$\frac{dg(u(t))}{dt} \le (-2\rho\kappa + \varepsilon)\|u(t) - u^*\|^2 = (-2\rho\kappa + \varepsilon)g(u(t)).$$

This implies

$$g(u(t)) \le e^{(-2\rho\kappa + \varepsilon)t}g(u(t_0)),$$

which means

$$\|u(t) - u^*\| \le e^{-\rho\kappa + \frac{\varepsilon}{2}}\|u(t_0) - u^*\|.$$

Thus, u^* is exponentially stable for the neural network (12).□

5. Numerical examples

In order to demonstrate the effectiveness of the proposed neural network, we test several examples for our neural network (12) in this section. The numerical implementation is coded by Matlab 7.0 and the ordinary differential equation solver adopted here is *ode23*, which uses Ruge–Kutta (2; 3) formula. As mentioned earlier, the parameter ρ is set to be 1. How is μ chosen initially? From Theorem 4.2 in last section, we know the solution will

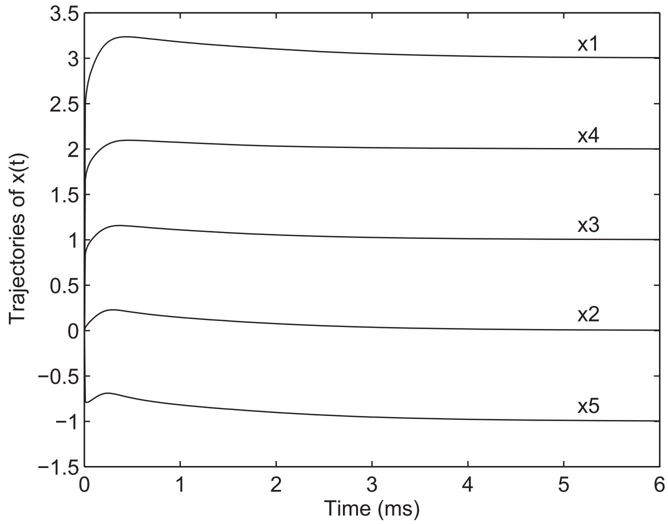

Fig. 1. Transient behavior of the neural network with the generalized FB function ($p=7$) in Example 5.1.

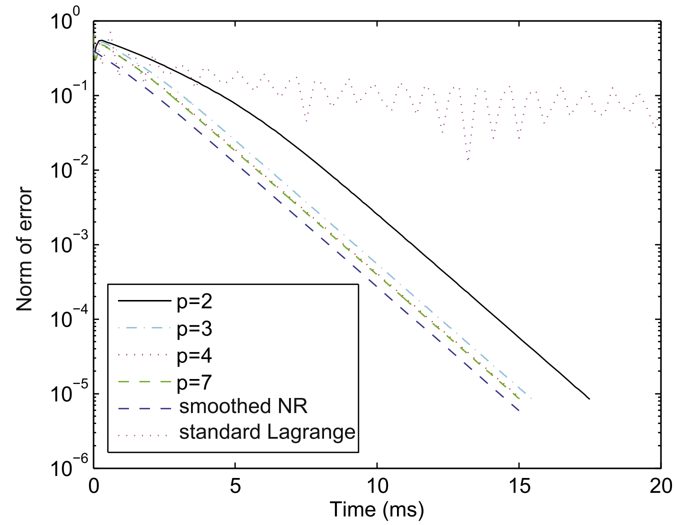

Fig. 2. Convergence comparison for Example 5.1.

converge with any initial point, we set initial $\mu=1$ in the codes (and of course $\mu \to 0$, as seen in the trajectory behavior).

To implement the proposed neural network (12), the calculation of $\nabla \Psi_p(u)$ is required. As below, we describe the step-by-step scheme for computing $\nabla \Psi_p(u)$.

Step 1. With $u=(x,y,z)^T$, we first calculate $g(x)$, $\nabla g(x)$, $\nabla f(x)$, $L(x,y,z)$, and $\nabla_x L(x,y,z)$.
Step 2. Compute $\phi_p(z,-g(x))$ and its gradient.
Step 3. Compute $H(u)$ and $\nabla H(u)$ given as in (9) and (11), respectively.
Step 4. Next, $\nabla \Psi_p(u)$ can be obtained by $\nabla H(u)H(u)$. Then, the ordinary differential equation solver Matlab *ode23*, which uses Runge–Kutta formula, is adopted for the numerical simulations.

Example 5.1. Consider the following nonlinear convex programming problem:

$$\begin{aligned} \min \quad & e^{(x_1-3)^2+x_2^2+(x_3-1)^2+(x_4-2)^2+(x_5+1)^2} \\ \text{s.t.} \quad & x \in \mathcal{K}^5 \end{aligned}$$

Here we denote

$$f(x) := e^{(x_1-3)^2+x_2^2+(x_3-1)^2+(x_4-2)^2+(x_5+1)^2}$$

and $g(x)=-x$. Hence, we compute that

$$L(x,z)=\nabla f(x)+\nabla g(x)z$$

$$=2f(x)\begin{bmatrix} x_1-3 \\ x_2 \\ x_3-1 \\ x_4-2 \\ x_5+1 \end{bmatrix} - \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix}.$$

This problem has an optimal solution $x^*=(3,0,1,2,-1)^T$. We use the proposed neural network to solve the above problem whose trajectories are depicted in Fig. 1. All simulation results show that the state trajectories with any initial point are always convergent to an optimal solution of the above problem x^*. From Fig. 2, we see that the performance in "good order" is the model based on smoothed NR function used in [29], the current model based on smoothed generalized FB function with $p=7$, the current model based on smoothed generalized FB function with $p=4$, the current model based on smoothed generalized FB function with $p=3$, the current model based on smoothed generalized FB function with $p=2$. The LPNN approach solves this problem, but its performance is not good.

Example 5.2. Consider the following nonlinear second-order cone programming problem:

$$\begin{aligned} \min \quad & f(x)=x_1^2+2x_2^2+2x_1x_2-10x_1-12x_2 \\ \text{s.t.} \quad & g(x)=\begin{bmatrix} 8-x_1+3x_2 \\ 3-x_1^2-2x_1+2x_2-x_2^2 \end{bmatrix} \in \mathcal{K}^2. \end{aligned}$$

For this example, we compute that

$$L(x,z)=\nabla f(x)+\nabla g(x)z$$

$$=\begin{bmatrix} 2x_1+2x_2-10 \\ 4x_2+2x_1-12 \end{bmatrix} - \begin{bmatrix} -z_1-2(x_1+1)z_2 \\ 3z_1+2(1-x_2)z_2 \end{bmatrix}.$$

This problem has an approximate solution $x^*=(2.8308, 1.6375)^T$. Note that the objective function is convex and the Hessian matrix $\nabla^2 f(x)$ is positive definite. Using the proposed neural network in this paper, we can easily obtain the approximate solution x^* of the above problem, see Fig. 3. From Fig. 4, we see that the performance in "good order" is the current model based on smoothed generalized FB function with $p=2$, the current model based on smoothed generalized FB function with $p=3$, the current model based on smoothed generalized FB function with $p=4$, the model based on smoothed NR function used in [29], the current model based on smoothed generalized FB function with $p=7$. Again, the LPNN approach solves this problem, but its performance is not good.

Example 5.3. Consider the following nonlinear convex program with second-order cone constraints [21]:

$$\begin{aligned} \min \quad & e^{(x_1-x_3)}+3(2x_1-x_2)^4+\sqrt{1+(3x_2+5x_3)^2} \\ \text{s.t.} \quad & Ax+b \in \mathcal{K}^2 \\ & 6x \in \mathcal{K}^3 \end{aligned}$$

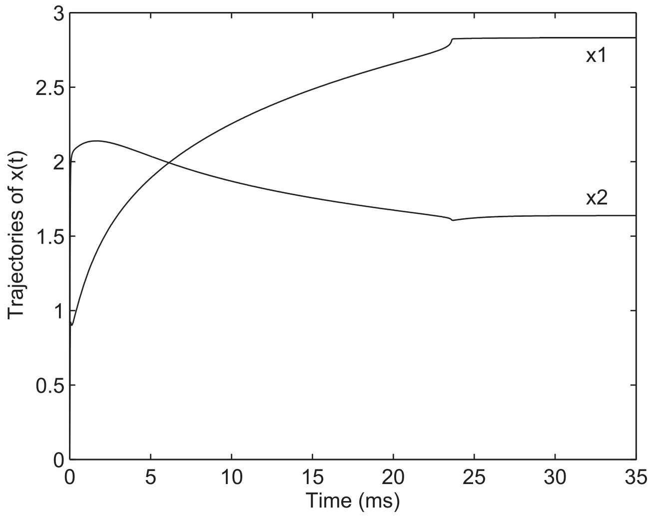

Fig. 3. Transient behavior of the neural network with the generalized FB function ($p=3$) in Example 5.2.

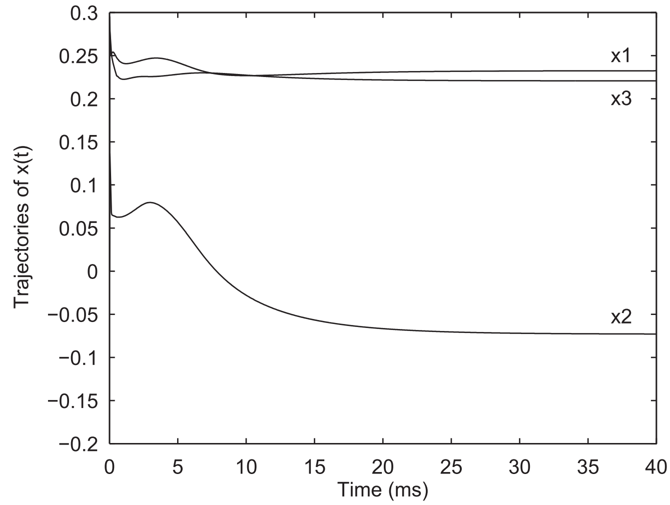

Fig. 5. Transient behavior of the neural network with the generalized FB function ($p=4$) in Example 5.3.

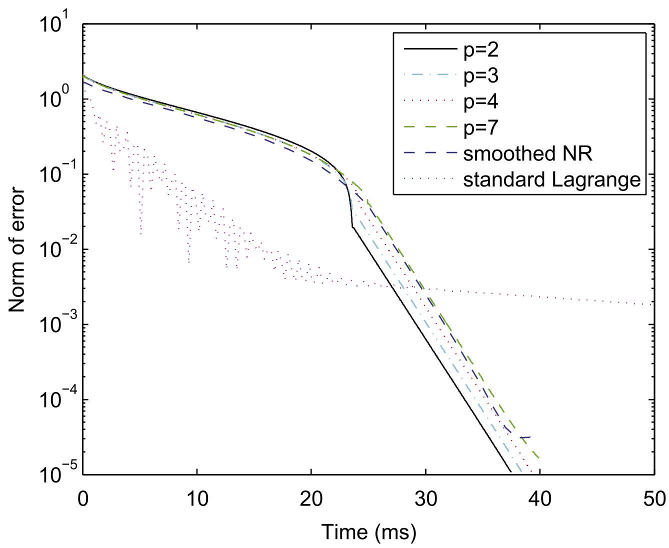

Fig. 4. Convergence comparison for Example 5.2.

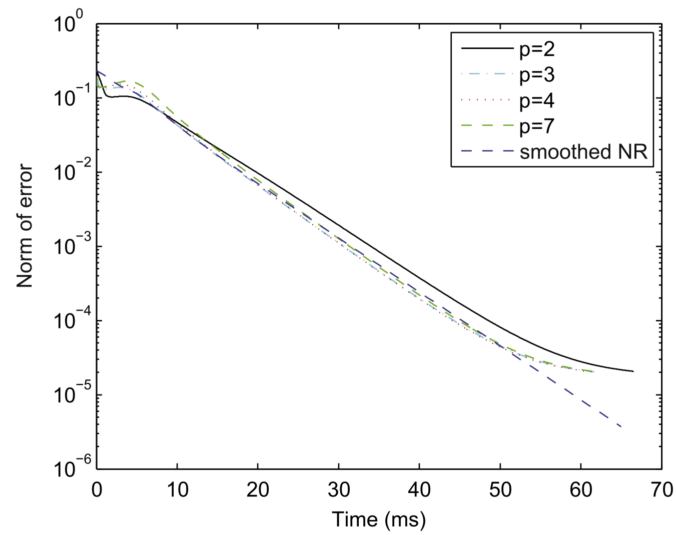

Fig. 6. Convergence comparison for Example 5.3.

where

$$A := \begin{bmatrix} 4 & 6 & 3 \\ -1 & 7 & -5 \end{bmatrix}, \quad b := \begin{bmatrix} -1 \\ 2 \end{bmatrix}.$$

For this example, $f(x) := e^{(x_1-x_3)} + 3(2x_1-x_2)^4 + \sqrt{1+(3x_2+5x_3)^2}$, from which we have

$$\begin{aligned} L(x,y,z) &= \nabla f(x) + \nabla g(x) y - 6 \nabla x z \\ &= \begin{bmatrix} e^{(x_1-x_3)} + 24(2x_1-x_2)^3 \\ -12(2x_1-x_2)^3 + \dfrac{3(3x_2+5x_3)}{\sqrt{1+(3x_2+5x_3)^2}} \\ -e^{(x_1-x_3)} + \dfrac{5(3x_2+5x_3)}{\sqrt{1+(3x_2+5x_3)^2}} \end{bmatrix} \\ &\quad - \begin{bmatrix} 4y_1 - y_2 \\ 6y_1 + 7y_2 \\ 3y_1 - 5y_2 \end{bmatrix} - 6 \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}. \end{aligned}$$

The approximate solution of this problem is $x^* = (0.2324, -0.07309, 0.2206)^T$, see Fig. 5. From Fig. 6, there is no marginal difference for all models. Note that the LPNN approach cannot solve this problem.

Example 5.4. Consider the following nonlinear second-order cone programming problem:

$$\begin{array}{ll} \min & f(x) = e^{x_1 x_3} + 3(x_1+x_2)^2 - \sqrt{1+(2x_2-x_3)^2} \\ & \quad + \frac{1}{2}x_4^2 + \frac{1}{2}x_5^2 \\ \text{s.t.} & h(x) = -24.51x_1 + 58x_2 - 16.67x_3 - x_4 - 3x_5 + 11 = 0 \\ & -g_1(x) = \begin{bmatrix} 3x_1^3 + 2x_2 - x_3 + 5x_3^2 \\ -5x_1^3 + 4x_2 - 2x_3 + 10x_3^3 \\ x_3 \end{bmatrix} \in \mathcal{K}^3 \\ & -g_2(x) = \begin{bmatrix} x_4 \\ 3x_5 \end{bmatrix} \in \mathcal{K}^2 \end{array}$$

For this example, we compute

$$L(x,y,z) = \nabla f(x) + \begin{bmatrix} \nabla g_1(x) y \\ \nabla g_2(x) z \end{bmatrix}$$

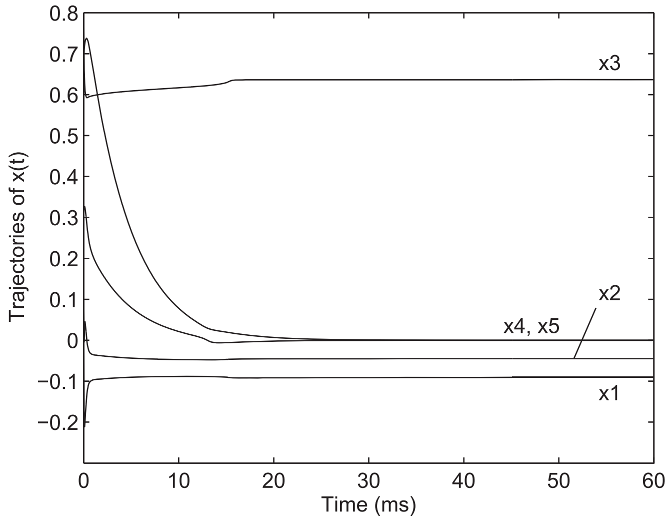

Fig. 7. Transient behavior of the neural network with the generalized FB function ($p=3$) in Example 5.4.

$$=\begin{bmatrix} x_3 e^{(x_1x_3)}+6(x_1+x_2) \\ 6(x_1+x_2)-\dfrac{2(2x_2-x_3)}{\sqrt{1+(2x_2-x_3)^2}} \\ x_1 e^{(x_1x_3)}+\dfrac{2x_2-x_3}{\sqrt{1+(2x_2-x_3)^2}} \\ x_4 \\ x_5 \end{bmatrix} - \begin{bmatrix} 9x_1^2y_1-15x_1^2y_2 \\ 2y_1+4y_2 \\ (10x_3-1)y_1+(30x_3^2-2)y_2+y_3 \\ z_1 \\ 3z_2 \end{bmatrix}.$$

This problem has an approximate solution $x^* = (-0.0903, -0.0449, 0.6366, 0.0001, 0)^T$ and Fig. 7 displays the trajectories obtained by using the proposed new neural network. All simulation results show that the state trajectory with any initial point are always convergent to the solution x^*. As observed in Fig. 8, the neural network with the smoothed NR function has a better convergence rate, and it is hard to see the effect when p is perturbed. Note that the LPNN approach cannot solve this problem.

In our numerical implementations, we test $p=2,3,4,7$ to see how it affects the numerical performance when it is perturbed. We also compare with the neural network model used in [29], which is based on "smoothed" NR function. In general, there is no big difference between our model based on "smoothed" generalized FB function and the one in [29]. Only slight better performance for the one used in [29] in Example 5.1, 5.3, 5.4 are observed. Another observation is that there is no regular change for numerical performance when p is perturbed. For Example 5.1, when p is increased, its performance becomes better. However, for Example 5.2, when p is increased, its performance becomes less better. These two phenomena do not occur in other two examples. Moreover, as suggested by one referee, we also have a comparison with the standard Lagrange programming neural networks (LPNN), which is studied in [42]. The LPNN for Example 5.1 and 5.2 has bad convergence shown as in Figs. 2 and 4 compared to other methods. For other examples, the LPNN does not even solve them successfully so that it is not depicted in other figures. The numerical comparisons verify the effectiveness of our proposed neural networks. To sum up, based on the numerical results, we

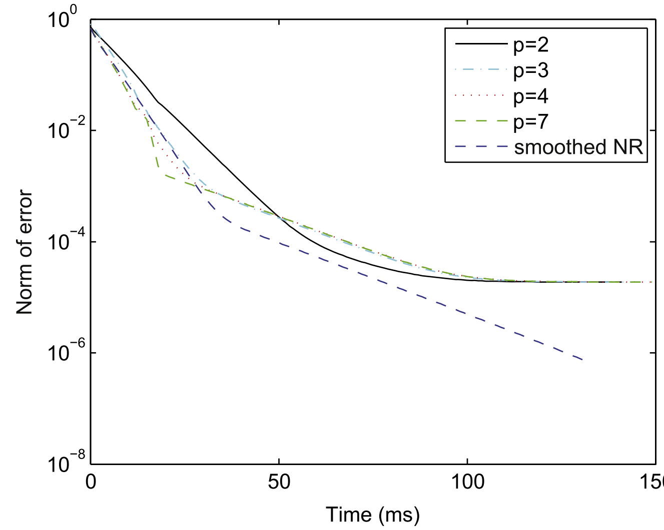

Fig. 8. Convergence comparison for Example 5.4.

can conclude that the proposed neural network model is definitely better than the standard LPNN model. In addition, although the difference between the proposed neural network model and the one based on "smoothed" NR function in [29] is very slight, it is generally true that our model is better than the aforementioned one when an appropriate p is chosen. How to determine an suitable p is a good topic for future study.

6. Concluding remarks

In this paper, we have studied a neural network approach for solving general nonlinear convex programs with second-order cone constraints. The neural network is based on the gradient of the merit function derived from the generalized FB merit function, which involves parameter $p \in (1,4)$. For such neural network, the Lyapunov stability, the asymptotic stability and the exponential stability are proved, which indicates its effectiveness. Moreover, numerical performance based on the parameter p being perturbed and numerical comparison with other neural network model are also provided. There is limited value of p ($p=\frac{n}{2} \in (1,4)$) that could be perturbed because Ψ_p is theoretically shown to be smooth only in $p \in (1,4)$ under SOC case, so far. Can we extend the above results to the case of general p? In other words, whether $p=\frac{n}{2} \in (1,4)$ can be relaxed to more general real value? This is one of our future directions. Moreover, we will try to show the smoothness of Ψ_p associated with SOC in a wider interval in the future. Recently, some other discrete types of complementarity functions associated with SOC have been proposed in [28]. Another direction is to design neural network based on "discrete" types of complementarity functions. Of course, it will be very interesting to see the comparisons of neural networks based on continuous type of complementarity functions (like the NR function and FB function) and discrete types of complementarity functions.

References

[1] F. Alizadeh, D. Goldfarb, Second-order cone programming, Math. Program. 95 (2003) 3–52.
[2] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, New York, 2004.
[3] J.-S. Chen, The convex and monotone functions associated with second-order cone, Optimization 55 (2006) 363–385.
[4] J.-S. Chen, S.-H. Pan, A survey on SOC complementarity functions and solution methods for SOCPs and SOCCPs, Pac. J. Optim. 8 (2012) 33–74.

[5] Y.-H. Chen, S.-C. Fang, Solving convex programming problems with equality constraints by neural networks, Comput. Math. Appl. 36 (1998) 41–68.
[6] J.-S. Chen, C.-H. Ko, S.-H. Pan, A neural network based on the generalized Fischer–Burmeister function for nonlinear complementarity problems, Inf. Sci. 180 (2010) 697–711.
[7] J.-S. Chen, P. Tseng, An unconstrained smooth minimization reformulation of the second-order cone complementarity problem, Math. Program. 104 (2005) 293–327.
[8] A. Cichocki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, John Wiley, New York, 1993.
[9] C. Dang, Y. Leung, X. Gao, K. Chen, Neural networks for nonlinear and mixed complementarity problems and their applications, Neural Netw. 17 (2004) 271–283.
[10] S. Effati, A. Ghomashi, A.R. Nazemi, Application of projection neural network in solving convex programming problems, Appl. Math. Comput. 188 (2007) 1103–1114.
[11] S. Effati, A.R. Nazemi, Neural network and its application for solving linear and quadratic programming problems, Appl. Math. Comput. 172 (2006) 305–331.
[12] F. Facchinei, J. Pang, Finite-Dimensional Variational Inequalities and Complementarity Problems, Springer, New York, 2003.
[13] M. Fukushima, Z.-Q. Luo, P. Tseng, Smoothing functions for second- order-cone complimentarity problems, SIAM J. Optim. 12 (2002) 436–460.
[14] Q. Han, L.-Z. Liao, H. Qi, L. Qi, Stability analysis of gradient-based neural networks for optimiation problems, J. Glob. Optim. 19 (2001) 363–381.
[15] J.J. Hopfield, D.W. Tank, Neural computation of decision in optimization problems, Biol. Cybern. 52 (1985) 141–152.
[16] X. Hu, J. Wang, A recurrent neural network for solving nonlinear convex programs subject to linear constraints, IEEE Trans. Neural Netw. 16 (2005) 379–386.
[17] X. Hu, J. Wang, A recurrent neural network for solving a class of general variational i nequalities, IEEE Trans. Syst. Man Cybern.-B 37 (2007) 528–539.
[18] S. Hayashi, N. Yamashita, M. Fukushima, A combined smoothing and regularization method for monotone second-order cone complementarity problems, SIAM J. Optim. 15 (2005) 593–615.
[19] Z. Hou, L. Cheng, and M. Tan, Coordination of two redundant robots using a dual neural network, in: Proceedings of IJCNN, Vancouver, BC, Canada, 2006, pp. 4187–4192.
[20] N. Kalouptisidis, Signal Processing Systems, Theory and Design, Wiley, New York, 1997.
[21] C. Kanzow, I. Ferenczi, M. Fukushima, On the local convergence of semismooth Newton methods for linear and nonlinear second-order cone programs without strict complementarity, SIAM J. Optim. 20 (2009) 297–320.
[22] M.P. Kennedy, L.O. Chua, Neural network for nonlinear programming, IEEE Trans. Circuits Syst. 35 (1988) 554–562.
[23] C.-H. Ko, J.-S. Chen, C.-Y. Yang, Recurrent neural networks for solving second-order cone programs, Neurocomputing 74 (2011) 3646–3653.
[24] Y.-J. Kuo, H.D. Mittelmann, Interior point methods for second-order cone programming and OR applications, Comput. Optim. Appl. 28 (2004) 255–285.
[25] L.-Z. Liao, H. Qi, L. Qi, Solving nonlinear complementarity problems with neural networks: a reformulation method approach, J. Comput. Appl. Math. 131 (2001) 342–359.
[26] M.S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, Applications of second-order cone programming, Linear Algebra Appl. 284 (1998) 193–228.
[27] D.R. Liu, D. Wang, X. Yang, An iterative adaptive dynamic programming algorithm for optimal control of unknown discrete-time nonlinear systems with constrained inputs, Inf. Sci. 220 (2013) 331–342.
[28] P.-F. Ma, J.-S. Chen, C.-H. Huang, C.-H. Ko, Discovery of new complementarity functions for NCP and SOCCP, Comput. Optim. Appl. (2016).
[29] X.-H. Miao, J.-S. Chen, C.-H. Ko, A smoothed NR neural network for solving nonlinear convex programs with second-order cone constraints, Inf. Sci. 268 (2014) 255–270.
[30] R.K. Miller, A.N. Michel, Ordinary Differential Equations, Academic Press, New York, 1982.
[31] S.-H. Pan, J.-S. Chen, A semismooth Newton method for the SOCCP based on a one-parametric class of SOC complementarity functions, Comput. Optim. Appl. 45 (2010) 59–88.
[32] S.-H. Pan, S.H. Kum, Y.D. Lim, J.-S. Chen, On the generalized Fischer–Burmeister merit function for the second-order cone complementarity problem, Math. Comput. 83 (2014) 1143–1171.
[33] J.-H. Sun, J.-S. Chen, C.-H. Ko, Neural networks for solving second-order cone constrained variational inequality problem, Comput. Optim. Appl. 51 (2012) 623–648.
[34] D.W. Tank, J.J. Hopfield, Simple neural optimization network: an A/D converter, signal decision circuit, and a linear programming circuit, IEEE Trans. Circuits Syst. 33 (1986) 533–541.
[35] A.L. Wu, S.P. Wen, Z.G. Zeng, Synchronization control of a class of memristor-based recurrent neural networks, Inf. Sci. 183 (2012) 106–116.
[36] Y. Xia, H. Leung, J. Wang, A projection neural network and its application to constrained optimization problems, IEEE Trans. Circuits Syst.-I 49 (2002) 447–458.
[37] Y. Xia, H. Leung, J. Wang, A general projection neural network for solving monotone variational inequalities and related optimization problems, IEEE Trans. Neural Netw. 15 (2004) 318–328.
[38] Y. Xia, J. Wang, A recurrent neural network for solving nonlinear convex programs subject to linear constraints, IEEE Trans. Neural Netw. 16 (2005) 379–386.
[39] M. Yashtini, A. Malek, Solving complementarity and variational inequalities problems using neural networks, Appl. Math. Comput. 190 (2007) 216–230.
[40] J. Zabczyk, Mathematical Control Theorem: An Introduction, Birkhäuser, Boston, 1992.
[41] G.D. Zhang, Y. Shen, Q. Yin, J.W. Sun, Global exponential periodicity and stability of a class of memristor-based recurrent neural networks with multiple delays, Inf. Sci. 232 (2013) 386–396.
[42] S. Zhang, A.G. Constantinides, Lagrange programming neural networks, IEEE Trans. Circuits Syst.-11: Analog Digit. Signal Process. 39 (1992) 441–452.

Xinhe Miao is an Associate Professor at Department of Mathematics of Tianjin University. He received his M.S. degree in Department of Mathematics of Tianjin University in 2003, and obtained Ph.D. degree in Institute of Systems Engineering of Tianjin University, Tianjin, China, in 2010. From 2010 to 2011, he was a Research Fellow at Department of Mathematics of National Taiwan University. His main scientific interests are in the field of conic complementarity problems, conic optimization problems and neural network, etc.

Jein-Shan Chen is currently a Distinguished Professor at the Mathematics Department of National Taiwan Normal University from where he received his B.S. degree and M.S. degree, in 1990 and 1993, respectively. In the fall of 1997, he attended the Mathematics Department of UCLA to pursue his Ph.D. degree. He only stayed one year over there and moved to the Mathematics Department, University of Washington in the fall of 1998, where he spent six years studying optimization with Professor Paul Tseng. He has published over 85 papers including a few in top journals like Mathematical Programming, SIAM Journal on Optimization. His research is mainly in continuous optimization with side interests in nonsmooth analysis and operations research.

Chun-Hsu Ko was born in Tainan, Taiwan, in 1967. He received the M.S. degree in power mechanical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1991, and the Ph.D. degree in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2003. From 1994 to 1998, he was with the Industrial Technology Research Institute, Hsinchu, Taiwan, as an Associate Researcher. He is currently a Professor in the Department of Electrical Engineering, I-Shou University, Kaohsiung, Taiwan. His research interests include robot control, robot walking helpers, and optimization.