Information Sciences 268 (2014) 255–270

Contents lists available at ScienceDirect

Information Sciences

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

A smoothed NR neural network for solving 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 84001, Taiwan

ARTICLE INFO

Article history:
Received 2 January 2012
Received in revised form 22 June 2013
Accepted 16 October 2013
Available online 24 October 2013

Keywords:
Merit function
Neural network
NR function
Second-order cone
Stability

ABSTRACT

This paper proposes a neural network approach for efficiently solving general nonlinear convex programs with second-order cone constraints. The proposed neural network model was developed based on a smoothed natural residual merit function involving an unconstrained minimization reformulation of the complementarity problem. We study the existence and convergence of the trajectory of the neural network. Moreover, we show some stability properties for the considered neural network, such as the Lyapunov stability, asymptotic stability, and exponential stability. The examples in this paper provide a further demonstration of the effectiveness of the proposed neural network. This paper can be viewed as a follow-up version of [20,26] because more stability results are obtained.

1. Introduction

In this paper, we are interested in finding a solution to the following nonlinear convex programs with second-order cone constraints (henceforth SOCP):

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

where $A \in \mathbb{R}^{m\times n}$ has full row rank, $b \in \mathbb{R}^m$, $f : \mathbb{R}^n \to \mathbb{R}$, $g = [g_1, \ldots, g_l]^T : \mathbb{R}^n \to \mathbb{R}^l$ with f and g_i's being two order continuous differentiable and convex on $\mathbb{R}^n$, and $\mathcal{K}$ is a Cartesian product of second-order cones (also called Lorentz cones), expressed as

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

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

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

* Corresponding author. Address: National Center Theoretical Sciences, Taipei Office, Taiwan. Tel.: +886 2 77346641; fax: +886 2 29332342.
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 The Seed Foundation of Tianjin University (No. 60302041).
[2] The author's work is supported by National Science Council of Taiwan.

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.10.017

Here, $\|\cdot\|$ denotes the Euclidean norm and $\mathcal{K}^1$ means the set of nonnegative reals $\mathbb{R}_+$. In fact, the problem (1) is equivalent to the following variational inequality problem, which is to find $x \in D$ satisfying

$$\langle \nabla f(x), y - x \rangle \geqslant 0, \quad \forall y \in D,$$

where $D = \{x \in \mathbb{R}^n | Ax = b, \ -g(x) \in \mathcal{K}\}$. Many problems in the engineering, transportation science, and economics communities can be solved by transforming the original problems into the mentioned convex optimization problems or variational inequality problems, see [1,7,10,17,23].

Many studies have proposed computational approaches to solve convex optimization problems. Examples of these methods include the interior-point method [29], merit function method [5,16], Newton method [18,25], and projection method [10]. However, real-time solutions are imperative in many applications, such as force analysis in robot grasping and control applications. The traditional optimization methods may not be suitable for these applications because of stringent computational time requirements. Therefore, a feasible and efficient method is required to solve real-time optimization problems. The neural network method is an ideal method for solving real-time optimization problems. Compared with previous methods, the neural network method has an advantage in solving real-time optimization problems. Hence, researchers have developed many continuous-time neural networks for constrained optimization problems. The literature contains many studies on neural networks for solving real-time optimization problems, please see [4,9,12,14,15,19–22,27,30–34,36] and references therein.

Neural networks stemmed back from McCulloch and Pitts' pioneering work a half century ago, and neural networks were first introduced to the optimization domain in the 1980s [13,28]. The essence of neural network method for optimization [6] is to establish a nonnegative Lyapunov function (or energy function) and a dynamic system that represents an artificial neural network. This dynamic system usually adopts the form of a first-order ordinary differential equation. For an initial point, the neural network is likely to approach its equilibrium point, which corresponds to the solution to the considered optimization problem.

This paper presents a neural network method to solve general nonlinear convex programs with second-order cone constraints. In particular, we consider the Karush–Kuhn–Tucker (KKT) optimality conditions of the problem (1), which can be transformed into a second-order cone complementarity problems (SOCCP), as well as some equality constraints. Following a reformulation of the complementarity problem, an unconstrained optimization problem is formulated. A smoothed natural residual (NR) complementarity function is then used to construct a Lyapunov function and a neural network model. At the same time, we show the existence and convergence of the solution trajectory for the dynamic system. This study also investigates the stability results, such as the Lyapunov stability, the asymptotic stability, and the exponential stability. We want to point out that the optimization problem considered in this paper is more general than the one studied in [20] where $g(x) = -x$ is investigated therein. From [20], for solving the specific SOCP (i.e., $g(x) = -x$), we know that the neural network based on the cone projection function has better performance than the one based on the Fischer–Burmeister function in most cases (except for some oscillating cases). In light of considering this phenomenon, we employ a neural network model based on the cone projection function for a more general SOCP. Thus, this paper can be viewed as a follow-up of [20] in this sense. Nevertheless, the neural network model studied here is not exactly the same as the one considered in [20]. More specifically, we consider a neural network based on the smoothed NR function which was studied in [16]. Why do we make such a change? As in Section 4, we can establish various stability results, including exponential stability for the proposed neural network that were not achieved in [20]. In addition, the second neural network studied in [27] (for various types of problems) is also similar to the proposed network. Again, the stability is not guaranteed in that study, but three stabilities are proved here.

The remainder of this paper is organized as follows. Section 2 presents stability concepts and provides related results. Section 3 describes the neural network architecture, which is based on the smoothed NR function, to solve the problem (1). Section 4 presents the convergence and stability results of the proposed neural network. Section 5 shows the simulation results of the new method. Finally, Section 6 gives the conclusion of this paper.

2. Preliminaries

In this section, we briefly recall background materials of the ordinary differential equation (ODE) and some stability concepts regarding the solution of ODE. We also present some related results that play an essential role in the subsequent analysis.

Let $H : \mathbb{R}^n \to \mathbb{R}^n$ be a mapping. The first order differential equation (ODE) means

$$\frac{du}{dt} = H(u(t)), \quad u(t_0) = u_0 \in \mathbb{R}^n. \tag{2}$$

We start with the existence and uniqueness of the solution of Eq. (2). Then, we introduce the equilibrium point of (2) and define various stabilities. All of these materials can be found in a typical ODE textbook, such as [24].

Lemma 2.1 (*The existence and uniqueness 21, Theorem 2.5*). *Assume that $H : \mathbb{R}^n \to \mathbb{R}^n$ is a continuous mapping. Then for arbitrary $t_0 \geqslant 0$ and $u_0 \in \mathbb{R}^n$, there exists a local solution $u(t)$, $t \in [t_0, \tau)$ to (2) for some $\tau > t_0$. Furthermore, if H is locally Lipschitz continuous at u_0, then the solution is unique; if H is Lipschitz continuous in $\mathbb{R}^n$, then τ can be extended to ∞.*

Remark 2.1. For Eq. (2), 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 (21, Theorem 2.6). *Let $H:\mathbb{R}^n \to \mathbb{R}^n$ be 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 $\lim_{t\uparrow\tau}\|u(t)\| = +\infty$.*

For the first-order differential Eq. (2), a point $u^* \in \mathbb{R}^n$ is called an **equilibrium point** of (2) 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 $u \in \Omega\backslash\{u^*\}$, then u^* is called an **isolated equilibrium point**. The following are definitions of various stabilities, and related materials can be found in [21,24,27].

Definition 2.1 (*Lyapunov stability and Asymptotic stability*). Let $u(t)$ be a solution to Eq. (2).

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

$$\|u_0 - u^*\| < \delta \;\Rightarrow\; \|u(t) - u^*\| < \varepsilon \text{ for } t \geqslant t_0.$$

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

Definition 2.2 (*Lyapunov function*). Let $\Omega \subseteq \mathbb{R}^n$ be an open neighborhood of $\bar{u}$. A continuously differentiable function $g:\mathbb{R}^n \to \mathbb{R}$ is said to be a Lyapunov function (or energy function) at the state $\bar{u}$ (over the set Ω) for Eq. (2) if

$$\begin{cases} g(\bar{u}) = 0, \\ g(u) > 0 \ \forall u \in \Omega \setminus \{\bar{u}\}, \\ \frac{dg(u(t))}{dt} \leqslant 0, \ \forall u \in \Omega. \end{cases}$$

The following Lemma shows the relationship between stabilities and a Lyapunov function, see [3,8,35].

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^* that satisfies*

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

Definition 2.3 (*Exponential stability*). An isolated equilibrium point u^* is exponentially stable for Eq. (2) if there exist $\omega < 0$, $\kappa > 0$, $\delta > 0$ such that arbitrary solution $u(t)$ to Eq. (2), with the initial condition $u(t_0) = u_0$, $\|u_0 - u^*\| < \delta$, is defined on $[0,\infty)$ and satisfies

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

From the above definitions, it is obvious that exponential stability is asymptotic stable.

3. NR neural network model

This section shows how the dynamic system in this study was formed. As mentioned previously, the key steps in the 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:

$$\begin{cases} \nabla f(x) - A^T y + \nabla g(x) z = 0, \\ z \in \mathcal{K}, \quad -g(x) \in \mathcal{K}, \quad z^T g(x) = 0, \\ Ax - b = 0, \end{cases} \tag{3}$$

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 (1), i.e., there exists an $x \in \mathbb{R}^n$ such that $-g(x) \in int(\mathcal{K})$ and $Ax = b$. Then x^* is a solution of the problem (1) if and only if there exist y^*, z^* such that (x^*,y^*,z^*) satisfies the KKT conditions (3). Hence, we assume that the problem (1) satisfies the Slater's condition in this paper.

The following paragraphs provide a brief review of particular properties of the spectral factorization with respect to a 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 [5,11,25].

For any vector $z = (z_1, z_2) \in \mathbb{R} \times \mathbb{R}^{l-1} (l \geqslant 2)$, its spectral factorization with respect to the second-order cone $\mathcal{K}$ is defined as

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

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

$$e_i = \begin{cases} \frac{1}{2}(1, (-1)^i \frac{z_i}{\|z_i\|}), & z_2 \neq 0 \\ \frac{1}{2}(1, (-1)^i w), & z_2 = 0 \end{cases}$$

with $w \in \mathbb{R}^{l-1}$ such that $\|w\| = 1$. The terms e_1, e_2 are called the spectral vectors of z. The spectral values of z and the vector z have the following properties: for any $z \in \mathbb{R}^l$, there have $\lambda_1 \leqslant \lambda_2$ and

$$\lambda_1 \geqslant 0 \iff z \in \mathcal{K}.$$

Now we review the concept of metric projection onto $\mathcal{K}$. For arbitrary element $z \in \mathbb{R}^l$, the *metric projection* of z onto $\mathcal{K}$ is denoted by $P_{\mathcal{K}}(z)$ and defined as

$$P_{\mathcal{K}}(z) := \arg\min_{w \in \mathcal{K}} \|z - w\|.$$

Combining the spectral decomposition of z with the metric projection of z onto $\mathcal{K}$ yields the expression of metric projection $P_{\mathcal{K}}(z)$ in [11]:

$$P_{\mathcal{K}}(z) = \max\{0, \lambda_1\} e_1 + \max\{0, \lambda_2\} e_2.$$

The projection function $P_{\mathcal{K}}$ has the following property, which is called the Projection Theorem (see [2]).

Lemma 3.1. *Let Ω be a closed convex set of $\mathbb{R}^n$. Then, for all $x,\ y \in \mathbb{R}^n$ and any $z \in \Omega$,*

$$(x - P_\Omega(x))^T (P_\Omega(x) - z) \geqslant 0 \quad \text{and} \quad \|P_\Omega(x) - P_\Omega(y)\| \leqslant \|x - y\|.$$

Given the definition of the projection, suppose z_+ denotes the metric projection $P_{\mathcal{K}}(z)$ of $z \in \mathbb{R}^l$ onto $\mathcal{K}$. Then, the natural residual (NR) function is given as follows [11]:

$$\Phi_{\mathrm{NR}}(x, y) := x - (x - y)_+ \quad \forall x, \quad y \in \mathbb{R}^l.$$

The NR function is a popular SOC-complementarity function, i.e.,

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

Because of the non-differentiability of Φ_{NR}, we consider a class of smoothed NR complementarity function. To this end, we employ a continuously differentiable convex function $\hat{g} : \mathbb{R} \to \mathbb{R}$ such that

$$\lim_{a \to -\infty} \hat{g}(a) = 0, \quad \lim_{a \to \infty} (\hat{g}(a) - a) = 0 \quad \text{and} \quad 0 < \hat{g}'(a) < 1. \tag{4}$$

What kind of functions satisfies the condition (4)? Here we present two examples:

$$\hat{g}(a) = \frac{\sqrt{a^2 + 4} + a}{2} \quad \text{and} \quad \hat{g}(a) = \ln(e^a + 1).$$

Suppose $z = \lambda_1 e_1 + \lambda_2 e_2$, where λ_i and e_i $(i = 1,2)$ are the spectral values and spectral vectors of z, respectively. By applying the function $\hat{g}$, we define the following function:

$$P_\mu(z) := \mu \hat{g}\left(\frac{\lambda_1}{\mu}\right) e_1 + \mu \hat{g}\left(\frac{\lambda_2}{\mu}\right) e_2. \tag{5}$$

Fukushima et al. [11] show that P_μ is smooth for any $\mu > 0$; moreover P_μ is a smoothing function of the projection $P_{\mathcal{K}}$, i.e., $\lim_{\mu \downarrow 0} P_\mu = P_{\mathcal{K}}$. Hence, a smoothed NR complementarity function is given in the form of

$$\Phi_\mu(x, y) := x - P_\mu(x - y).$$

In particular, from [11, Proposition 5.1], there exists a positive constant $\gamma > 0$ such that

$$\|\Phi_\mu(x, y) - \Phi_{\mathrm{NR}}(x, y)\| \leqslant \gamma \mu$$

for any $\mu > 0$ and $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$.

Now we look into the KKT conditions (3) of the problem (1). Let

$$L(x,y,z)=\nabla f(x)-A^{T}y+\nabla g(x)z,\quad H(u):=\begin{bmatrix}\mu\\ Ax-b\\ L(x,y,z)\\ \Phi_{\mu}(z,-g(x))\end{bmatrix}$$

and

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

where $u=(\mu,x^{T},y^{T},z^{T})^{T}\in\mathbb{R}_{+}\times\mathbb{R}^{n}\times\mathbb{R}^{m}\times\mathbb{R}^{l}$. It is known that $\Psi_{\mu}(u)$ serves as a smoothing function of the merit function Ψ_{NR} which means the KKT conditions (3) are shown to be equivalent to the following unconstrained minimization problem via the merit function approach:

$$\min\Psi_{\mu}(u):=\frac{1}{2}\|H(u)\|^{2}. \tag{6}$$

Theorem 3.1.

(a) *Let P_{μ} be defined by (5). Then, $\nabla P_{\mu}(z)$ and $I-\nabla P_{\mu}(z)$ are positive definite for any $\mu>0$ and $z\in\mathbb{R}^{l}$.*
(b) *Let Ψ_{μ} be defined as in (6). Then, the smoothed merit function Ψ_{μ} is continuously differentiable everywhere with $\nabla\Psi_{\mu}(u)=\nabla H(u)H(u)$ where*

$$\nabla H(u)=\begin{bmatrix}1 & 0 & 0 & -\left(\frac{\partial P_{\mu}(z+g(x))}{\partial\mu}\right)^{T}\\ 0 & A^{T} & \nabla^{2}f(x)+\nabla^{2}g_{1}(x)+\cdots+\nabla^{2}g_{l}(x) & -\nabla_{x}P_{\mu}(z+g(x))\\ 0 & 0 & -A & 0\\ 0 & 0 & \nabla g(x)^{T} & I-\nabla_{z}P_{\mu}(z+g(x))\end{bmatrix}. \tag{7}$$

Proof. Form the proof of [16, Proposition 3.1], it is clear that $\nabla P_{\mu}(z)$ and $I-\nabla P_{\mu}(z)$ are positive definite for any $\mu>0$ and $z\in\mathbb{R}^{l}$. With the help of the definition of the smoothed merit function Ψ_{μ}, part (b) easily follows from the chain rule. □

In light of the main ideas for constructing artificial neural networks (see [6] for details), we establish a specific first order ordinary differential equation, i.e., an artificial neural network. More specifically, based on the gradient of the objective function Ψ_{μ} in minimization problem (6), we propose the neural network for solving the KKT system (3) of nonlinear SOCP (1) with the following differential equation:

$$\frac{du(t)}{dt}=-\rho\nabla\Psi_{\mu}(u),\quad u(t_{0})=u_{0}, \tag{8}$$

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 (8) that $\frac{du(\tau)}{d\tau}=-\nabla\Psi_{\mu}(u)$. In view of this, for simplicity and convenience, we set $\rho=1$ in this paper. Indeed, the dynamic system (8) can be realized by an architecture with the cone projection function shown in Fig. 1. Moreover, the architecture of this artificial neural network is categorized as a "recurrent" neural network according to the classifications of artificial neural networks as in [6, Chapter 2.3.1]. The circuit for (8) requires $n+m+l+1$ integrators, n processors for $\nabla f(x)$, l processors for $g(x)$, ln processors for $\nabla g(x)$, $(l+1)^{2}n$ processors for $\nabla^{2}f(x)+\sum_{i=1}^{l}\nabla^{2}g_{i}(x)$, 1 processor for Φ_{μ}, 1 processor for $\frac{\partial P_{\mu}}{\partial\mu}$, n processors for $\nabla_{x}P_{\mu}$, l processors for $\nabla_{z}P_{\mu}$, $n^{2}+4mn+3ln+l^{2}+l$ connection weights and some summers.

4. Stability analysis

In this section, in order to study the stability issues of the proposed neural network (8) for solving the problem (1), we first make an assumption that will be required in our subsequent analysis.

Assumption 4.1.

(a) The problem (1) satisfies the Slater's condition.
(b) The matrix $\nabla^{2}f(x)+\nabla^{2}g_{1}(x)+\cdots+\nabla^{2}g_{l}(x)$ is positive definite for each x.

Here we say a few words about Assumption 4.1(a and b). The Slater's condition is a standard condition that is widely used in optimization field. Assumption 4.1(b) seems stringent at first glance. Indeed, since f and g_{i}'s are two order continuously differentiable and convex functions on $\mathbb{R}^{n}$, if there exists at least one function which is strictly convex among these functions, then Assumption 4.1(b) is guaranteed.

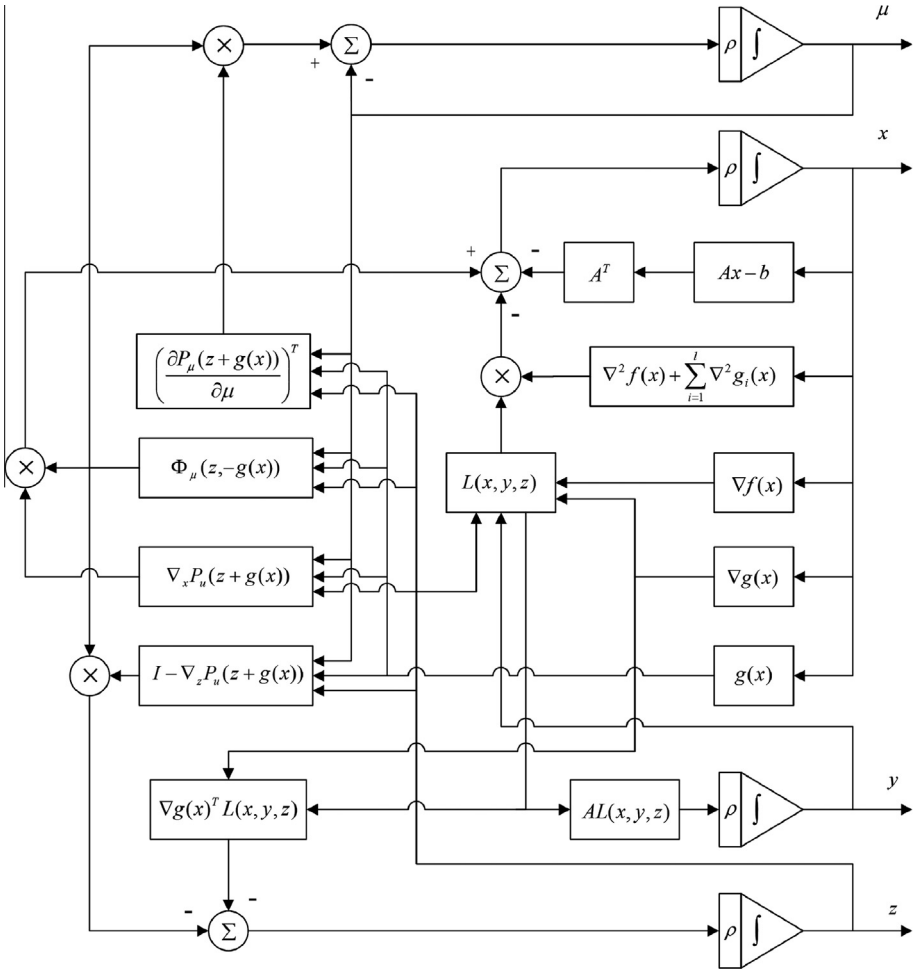

Fig. 1. Block diagram of the proposed neural network with smoothed NR function.

Lemma 4.1.

(a) *For any u, we have*

$$\|H(u)-H(u^*)-V(u-u^*)\|=o(\|u-u^*\|) \quad \text{for} \quad u\to u^* \text{ and } V\in\partial H(u)$$

where ∂H(u) denotes the Clarke generalized Jacobian at u.

(b) *Under Assumption 4.1, $\nabla H(u)^T$ is nonsingular for any $u=(\mu,x,y,z)\in\mathbb{R}_{++}\times\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^l$, where $\mathbb{R}_{++}$ denotes the set $\{\mu\,|\,\mu>0\}$.*

(c) *Under Assumption 4.1 and $V\in\partial P_0(w)$ being a positive definite matrix where $\partial P_0(w)$ denotes the Clarke generalized Jacobian of the project function P at w, there has*

$$T\in\partial H(u)=\left\{\begin{bmatrix}1 & 0 & 0 & -\left(\frac{\partial P_\mu(z+g(x))}{\partial\mu}\right)^T|_{\mu=0}\\ 0 & A^T & \nabla^2 f(x)+\nabla^2 g_1(x)+\cdots+\nabla^2 g_l(x) & -V^T\nabla g(x)\\ 0 & 0 & -A & 0\\ 0 & 0 & \nabla g(x)^T & I-V\end{bmatrix}\,|\,V\in\partial P_0(W)\right\}$$

is nonsingular for any $u=(0,x,y,z)\in\{0\}\times\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^l$.

(**d**) *$\Psi_\mu(u(t))$ is nonincreasing with respect to t.*

Proof. (a) This result follows directly from the definition of semismoothness of *H*, see [26] for more details.

(b) From the expression of $\nabla H(u)$ in Theorem 3.1, it follows that $\nabla H(u)^T$ is nonsingular if and only if the following matrix

$$M := \begin{bmatrix} A & 0 & 0 \\ \nabla^2 f(x) + \nabla^2 g_1(x) + \cdots + \nabla^2 g_l(x) & -A^T & \nabla g(x) \\ -\nabla_x P_\mu(z + g(x))^T & 0 & (I - \nabla_z P_\mu(z + g(x)))^T \end{bmatrix}$$

is nonsingular. Suppose $v = (x, y, z) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$. To show the nonsingularity of M, it is enough to prove that

$$Mv = 0 \Rightarrow x = 0, \quad y = 0 \quad \text{and} \quad z = 0.$$

Because $-\nabla_x P_\mu(z + g(x))^T = -\nabla P_\mu(w)^T \nabla\, g(x)^T$, where $w = z + g(x) \in \mathbb{R}^l$, from $Mv = 0$, we have

$$Ax = 0, \quad (\nabla^2 f(x) + \nabla^2 g_1(x) + \cdots + \nabla^2 g_l(x))x - A^T y + \nabla g(x) z = 0 \tag{9}$$

and

$$-\nabla P_\mu(w)^T \nabla g(x)^T x + (I - \nabla P_\mu(w))^T z = 0. \tag{10}$$

From (9), it follows that

$$x^T(\nabla^2 f(x) + \nabla^2 g_1(x) + \cdots + \nabla^2 g_l(x))x + (\nabla g(x)^T x)^T z = 0. \tag{11}$$

Moveover, Eq. (10) and Theorem 3.1 yield

$$\nabla g(x)^T x = (\nabla P_\mu(w)^T)^{-1}(I - \nabla P_\mu(w))^T z. \tag{12}$$

Combining (11) and (12) and Theorem 3.1, under the condition of Assumption 4.1, it is not hard to obtain that $x = 0$ and $z = 0$. By looking at Eq. (9) again, since A is full row rank, we have $y = 0$. Therefore, $\nabla H(u)^T$ is nonsingular.

(c) The proof of Part (c) is similar to that of Part (b), in which the only option is to replace $\nabla P_\mu(w)$ with $V \in \partial P_0(w)$.

(d) According to the definition of $\Psi_\mu(u(t))$ and Eq. (8), it is clear that

$$\frac{d\Psi_\mu(u(t))}{dt} = \nabla \Psi_\mu(u(t)) \frac{du(t)}{dt} = -\rho \|\nabla \Psi_\mu(u(t))\|^2 \leqslant 0.$$

Consequently, $\Psi_\mu(u(t))$ is nonincreasing with respect to t. □

Proposition 4.1. *Assume that $\nabla H(u)$ is nonsingular for any $u \in \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l$. Then,*

(a) *(x^*, y^*, z^*) satisfies the KKT conditions (3) if and only if $(0, x^*, y^*, z^*)$ is an equilibrium point of the neural network (8);*
(b) *Under the Slater's condition, x^* is a solution to the problem (1) if and only if $(0, x^*, y^*, z^*)$ is an equilibrium point of the neural network (8).*

Proof. (a) Because $\Phi_0 = \Phi_{\mathrm{NR}}$ when $\mu = 0$, it follows that (x^*, y^*, z^*) satisfies the KKT conditions (3) if and only if $H(u^*) = 0$, where $u^* = (0, x^*, y^*, z^*)^T$. Since $\nabla H(u)$ is nonsingular, we have that $H(u^*) = 0$ if and only if $\nabla\ \Psi_\mu(u^*) = \nabla H(u^*)^T H(u^*) = 0$. Thus, the desired result follows.

(b) Under the 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 (3). Hence, according to Part (a), it follows that $(0, x^*, y^*, z^*)$ is an equilibrium point of the neural network (8). □

The next result addresses the existence and uniqueness of the solution trajectory of the neural network (8).

Theorem 4.1.

(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 (8), where $[t_0, \tau)$ is the maximal interval of existence.*
(b) *If the level set $\mathcal{L}(u_0) := \{u | \Psi_\mu(u) \leqslant \Psi_\mu(u_0)\}$ is bounded, then τ can be extended to $+\infty$.*

Proof. This proof is exactly the same as the proof of [27, Proposition 3.4], and therefore omitted here. □

Theorem 4.2. *Assume that $\nabla H(u)$ is nonsingular and that u^* is an isolated equilibrium point of the neural network (8). Then the solution of the neural network (8) 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^*. Now, we consider the smoothed merit function

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

Since u^* is an isolated equilibrium point of (8), there is a neighborhood Ω of u^* such that

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

By the nonsingularity of $\nabla H(u)$ and the definition of Ψ_μ, it is easy to obtain that $\Psi_\mu(u^*)=0$. From the definition of Ψ_μ, we claim that $\Psi_\mu(u(t))>0$ for any $u(t)\in\Omega\setminus\{u^*\}$, where Ω is a neighborhood of u^*. Suppose not, namely, $\Psi_\mu(u(t))=0$. It follows that $H(u(t))=0$. Then, we have $\nabla\Psi_\mu(u(t))=0$ which contradicts with the assumption that u^* is an isolated equilibrium point of (8). Thus, $\Psi_\mu(u(t))>0$ for any $u(t)\in\Omega\setminus\{u^*\}$. Furthermore, by the proof of Lemma 4.1(d), we know that for any $u(t)\in\Omega$

$$\frac{d\Psi_\mu(u(t))}{dt}=\nabla\Psi_\mu(u(t))\frac{du(t)}{dt}=-\rho\|\nabla\Psi_\mu(u(t))\|^2\leqslant 0. \tag{13}$$

Consequently, the function Ψ_μ is a Lyapunov function over Ω. This implies that u^* is Lyapunov stable for the neural network (8). □

Theorem 4.3. *Assume that $\nabla H(u)$ is nonsingular and that u^* is an isolated equilibrium point of the neural network (8). Then u^* is asymptotically stable for neural network (8).*

Proof. From the proof of Theorem 4.2, we consider again the Lyapunov function Ψ_μ. By Lemma 2.3 again, we only need to verify that the Lyapunov function Ψ_μ over some neighborhood Ω of u^* satisfies

$$\frac{d\Psi_\mu(u(t))}{dt}<0, \quad \forall u(t)\in\Omega\setminus\{u^*\}. \tag{14}$$

In fact, by using (13) and the definition of the isolated equilibrium point, it is not hard to check that the Eq. (14) is true. Hence, u^* is asymptotically stable. □

Theorem 4.4. *Assume that u^* is an isolated equilibrium point of the neural network (8). If $\nabla H(u)^T$ is nonsingular for any $u=(\mu,x,y,z)\in\mathbb{R}_+\times\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^l$, then u^* is exponentially stable for the neural network (8).*

Proof. From the definition of $H(u)$, we know that H is semismooth. Hence, by Lemma 4.1, we have

$$H(u)=H(u^*)+\nabla H(u(t))^T(u-u^*)+o(\|u-u^*\|), \quad \forall u\in\Omega\setminus\{u^*\}, \tag{15}$$

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

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

Then, we have

$$\frac{dg(u(t))}{dt}=2(u(t)-u^*)^T\frac{du(t)}{dt}=-2\rho(u(t)-u^*)^T\nabla\Psi_\mu(u(t))=-2\rho(u(t)-u^*)^T\nabla H(u)H(u). \tag{16}$$

Substituting Eq. (15) into Eq. (16) yields

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

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

$$(u(t)-u^*)^T\nabla H(u)\nabla H(u)^T(u(t)-u^*)\geqslant\kappa\|u(t)-u^*\|^2. \tag{17}$$

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^* being an isolated equilibrium point. Consequently, there exists an $\kappa>0$ such that (17) holds. Moreover, for $o(\|u(t)-u^*\|^2)$, there is $\varepsilon>0$ such that $o(\|u(t)-u^*\|^2)\leqslant\varepsilon\|u(t)-u^*\|^2$. Hence,

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

This implies

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

Table 1
Stability comparisons of neural networks considered in current paper, [20,27].

	Current paper	[20]	[27]
Problem	min $f(x)$ s.t. $Ax = b$ $-g(x) \in \mathcal{K}$	min $f(x)$ s.t. $Ax = b$ $x \in \mathcal{K}$	$\langle F(x), y - x\rangle \geqslant 0, \quad \forall y \in C$ $C = \{x \mid h(x) = 0, -g(x) \in \mathcal{K}\}$
ODE	Based on smoothed NR-function	Based on NR-function and FB-function	Based on NR-function and smoothed FB-function
Stability	Lyapunov (smoothed NR) Asymptotical (smoothed NR) Exponential (smoothed NR)	Lyapunov (NR) Lyapunov (FB) asymptotical (FB)	Lyapunov (NR) Asymptotical (NR) Exponential (smoothed FB)

which means

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

Thus, u^* is exponentially stable for the neural network (8). □

To show the contribution of this paper, we present the stability comparisons of neural networks considered in the current paper, [20,27] in Table 1. More convergence comparisons will be presented in the next section. Generally speaking, we establish three stabilities for the proposed neural network, whereas not all three stabilities for the similar neural networks studied in [20,27] are guaranteed. Why do we choose to investigate the proposed neural network? Indeed, in [20], two neural networks based on NR function and FB function are considered which does not reach exponential stability. Our target optimization problem is a wider class than the one studied in [20]. In contrast, the smoothed FB has good performance as is shown in [27], but not all the three stabilities are established even though exponential stability is good enough. In light of these observations, we decide to look into the smoothed NR function for our problem which turns out to have better theoretical results. We summarize their differences in problems format, dynamical model, and stability issues in Table 1.

5. Numerical examples

In order to demonstrate the effectiveness of the proposed neural network, in this section we test several examples for our neural network (8). The numerical implementation is coded by Matlab 7.0 and the ordinary differential equation solver adopted here is *ode23*, which uses the 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 previous section, we know the solution converges with any initial point, we set initial $\mu = 1$ in the codes (and of course $\mu \to 0$, as seen in the trajectory behavior).

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 $g(x) = -x$. Then, we compute

$$L(x,z) = \nabla f(x) + \nabla g(x)z = 2f(x)\begin{bmatrix} x_1-3 \\ x_2 \\ x_3-1 \\ x_4-2 \\ x_5+1 \end{bmatrix} - \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix}. \tag{18}$$

Moreover, let $x := (x_1, \bar{x}) \in \mathbb{R} \times \mathbb{R}^4$ and $z := (z_1, \bar{z}) \in \mathbb{R} \times \mathbb{R}^4$. Then, the element $z - x$ can be expressed as

$$z - x := \lambda_1 e_1 + \lambda_2 e_2$$

where $\lambda_i = z_1 - x_1 + (-1)^i\|\bar{z} - \bar{x}\|$ and $e_i = \frac{1}{2}\left(1, (-1)^i\frac{\bar{z}-\bar{x}}{\|\bar{z}-\bar{x}\|}\right)$ $(i = 1, 2)$ if $\bar{z} - \bar{x} \neq 0$, otherwise $e_i = \frac{1}{2}(1, (-1)^i w)$ with w being any vector in $\mathbb{R}^4$ satisfying $\|w\| = 1$. This implies that

$$\Phi_\mu(z, -g(x)) = z - P_\mu(z + g(x)) = z - \mu\hat{g}\left(\frac{\lambda_1}{\mu}\right)e_1 + \mu\hat{g}\left(\frac{\lambda_2}{\mu}\right)e_2 \tag{19}$$

with $\hat{g}(a) = \frac{\sqrt{a^2+4}+a}{2}$ or $\hat{g}(a) = \ln(e^a + 1)$. Therefore, by Eqs. (18) and (19), we obtain the expression of $H(u)$ as follows:

$$H(u) = \begin{bmatrix} \mu \\ L(x,z) \\ \Phi_\mu(z, -g(x)) \end{bmatrix}.$$

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. 2. All simulation results show that the state trajectories with any initial point are always convergent to an optimal solution of the above problem x^*.

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

$$L(x,z) = \nabla f(x) + \nabla g(x)z = \begin{bmatrix} 2x_1 - 2x_2 - 10 \\ 4x_2 + 2x_1 - 12 \end{bmatrix} - \begin{bmatrix} -z_1 - 2(x_1+1)z_2 \\ 3z_3 + 2(1-x_2)z_2 \end{bmatrix}. \tag{20}$$

Since

$$z - g(x) = \begin{bmatrix} z_1 - 8 + x_1 - 3x_2 \\ z_2 - 3 + x_1^2 + 2x_1 - 2x_2 + x_2^2 \end{bmatrix},$$

the vector $z - g(x)$ can be expressed as

$$z - x \;:=\; \lambda_1 e_1 + \lambda_2 e_2$$

where $\lambda_i = z_1 - 8 + x_1 - 3x_2 + (-1)^i |z_2 - 3 + x_1^2 + 2x_1 - 2x_2 + x_2^2|$ and

$$e_i = \frac{1}{2}\left(1, (-1)^i \frac{z_2 - 3 + x_1^2 + 2x_1 - 2x_2 + x_2^2}{|z_2 - 3 + x_1^2 + 2x_1 - 2x_2 + x_2^2|}\right)(i = 1,2), \text{ if } z_2 - 3 + x_1^2 + 2x_1 - 2x_2 + x_2^2 \neq 0,$$

otherwise, $e_i = \frac{1}{2}(1, (-1)^i w)$ with w being any element in $\mathbb{R}$ satisfying $|w| = 1$. This implies that

$$\Phi_\mu(z, -g(x)) = z - P_\mu(z + g(x)) = z - \mu\hat{g}\left(\frac{\lambda_1}{\mu}\right)e_1 + \mu\hat{g}\left(\frac{\lambda_2}{\mu}\right)e_2 \tag{21}$$

with $\hat{g}(a) = \frac{\sqrt{a^2+4}+a}{2}$ or $\hat{g}(a) = \ln(e^a + 1)$. Therefore, by (20) and (21), we obtain the expression of $H(u)$ as follows:

$$H(u) = \begin{bmatrix} \mu \\ L(x,z) \\ \Phi_\mu(z, -g(x)) \end{bmatrix}.$$

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.

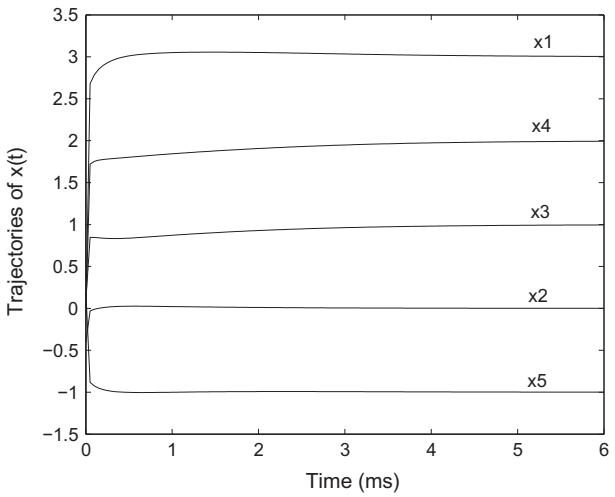

Fig. 2. Transient behavior of the neural network with the smoothed NR function in Example 5.1.

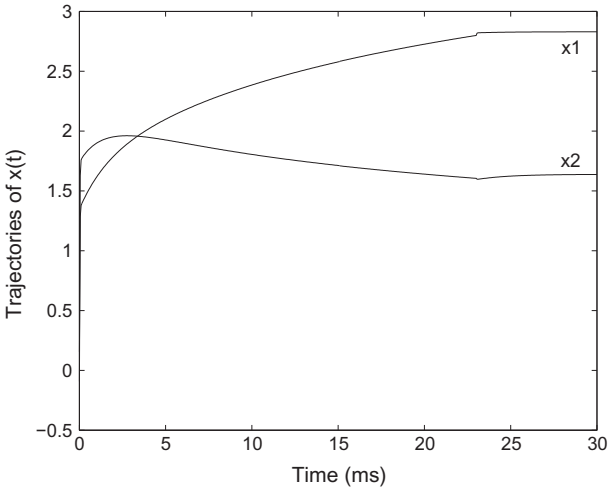

Fig. 3. Transient behavior of the neural network with the smoothed NR function in Example 5.2.

Example 5.3. Consider the following nonlinear convex program with second-order cone constraints [18]:

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

where

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

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

$$L(x,y,z) = \nabla f(x) + \nabla g(x)y - \nabla xz = \begin{bmatrix} e^{(x_1-x_3)} + 24(2x_1 - x_2)^3 \\ -12(2x_1 - x_2)^3 + \frac{3(3x_2+5x_3)}{\sqrt{1+(3x_2+5x_3)^2}} \\ -e^{(x_1-x_3)} + \frac{5(3x_2+5x_3)}{\sqrt{1+(3x_2+5x_3)^2}} \end{bmatrix} - \begin{bmatrix} 4y_1 - y_2 \\ 6y_1 + 7y_2 \\ 3y_1 - 5y_2 \end{bmatrix} - \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}. \tag{22}$$

Since

$$\begin{bmatrix} y + g(x) \\ z - x \end{bmatrix} = \begin{bmatrix} y_1 - 4x_1 - 6x_2 - 3x_3 + 1 \\ y_2 + x_1 - 7x_2 + 5x_3 - 2 \\ z_1 - x_1 \\ z_2 - x_2 \\ z_3 - x_3 \end{bmatrix},$$

$y + g(x)$ and $z - x$ can be expressed as follows, respectively,

$$y + g(x) := \lambda_1 e_1 + \lambda_2 e_2$$

and

$$z - x := \kappa_1 f_1 + \kappa_2 f_2$$

where $\lambda_i = y_1 - 4x_1 - 6x_2 - 3x_3 + 1 + (-1)^i |y_2 + x_1 - 7x_2 + 5x_3 - 2|$ and

$$e_i = \frac{1}{2}\left(1, (-1)^i \frac{y_2 + x_1 - 7x_2 + 5x_3 - 2}{|y_2 + x_1 - 7x_2 + 5x_3 - 2|}\right) \quad (i = 1,2) \quad \text{if} \quad y_2 + x_1 - 7x_2 + 5x_3 - 2 \neq 0;$$

otherwise, $e_i = \frac{1}{2}(1, (-1)^i w)$ with w being any element in $\mathbb{R}$ satisfying $|w| = 1$. Moveover, let $x := (x_1, \bar{x}) \in \mathbb{R} \times \mathbb{R}^2$ and $z := (z_1, \bar{z}) \in \mathbb{R} \times \mathbb{R}^2$. Then, we obtain that $\kappa_i = z_1 - x_1 + (-1)^i \|\bar{z} - \bar{x}\|$ and $f_i = \frac{1}{2}(1, (-1)^i \frac{\bar{z}-\bar{x}}{\|\bar{z}-\bar{x}\|})(i = 1,2)$ if $\bar{z} - \bar{x} \neq 0$; otherwise $f_i = \frac{1}{2}(1, (-1)^i \upsilon)$ with υ being any vector in $\mathbb{R}^2$ satisfying $\|\upsilon\| = 1$. This implies that

$$\Phi_\mu(\cdot) = \begin{bmatrix} y - P_\mu(y + g(x)) \\ z - P_\mu(z - x) \end{bmatrix} = \begin{bmatrix} y - \mu\hat{g}\left(\frac{\lambda_1}{\mu}\right)e_1 + \mu\hat{g}\left(\frac{\lambda_2}{\mu}\right)e_2 \\ z - \mu\hat{g}\left(\frac{\kappa_1}{\mu}\right)f_1 + \mu\hat{g}\left(\frac{\kappa_2}{\mu}\right)f_2 \end{bmatrix} \tag{23}$$

with $\hat{g}(a) = \frac{\sqrt{a^2+4}+a}{2}$ or $\hat{g}(a) = \ln(e^a + 1)$. Therefore, by (22) and (23), we obtain the expression of $H(u)$ as below:

$$H(u) = \begin{bmatrix} \mu \\ L(x,y,z) \\ \Phi_\mu(\cdot) \end{bmatrix}.$$

The approximate solution to this problem is $x^* = (0.2324, -0.07309, 0.2206)^T$. The trajectories are depicted in Fig. 4. We want to point out on thing: Assumption 4.1(a and b) are not both satisfied in this example. More specifically, for this example, the Assumption 4.1(a) is satisfied, which is obvious. However, $\nabla^2 f(x) + \nabla^2 g_1(x) + \cdots + \nabla^2 g_l(x)$ is not positive semidefinite for each x. To see this, we compute

$$\nabla^2 f(x) + \nabla^2 g_1(x) + \cdots + \nabla^2 g_l(x) = \nabla^2 f(x) = \begin{bmatrix} e^{(x_1 - x_3)} + 144(2x_1 - x_2)^2 & -72(2x_1 - x_2)^2 & -e^{(x_1 - x_3)} \\ -72(2x_1 - x_2)^2 & 36(2x_1 - x_2)^2 + \frac{9}{(1+(3x_2+5x_3)^2)^{\frac{3}{2}}} & \frac{15}{(1+(3x_2+5x_3)^2)^{\frac{3}{2}}} \\ e^{(x_1 - x_3)} & \frac{15}{(1+(3x_2+5x_3)^2)^{\frac{3}{2}}} & e^{(x_1 - x_3)} + \frac{25}{(1+(3x_2+5x_3)^2)^{\frac{3}{2}}} \end{bmatrix},$$

which is not positive semidefinite when $2x_1 - x_2 = 0$ (because the determinant equals zero). Hence, $H(u)$ is not guaranteed to be nonsingular and all the theorems in Section 4 do not apply for this example. Nonetheless, the solution trajectory does converge as depicted in Fig. 4. This phenomenon also occurs when it is solved by the second neural network studied in [27] (the stability is not guaranteed theoretically, but the solution trajectory does converges).

In addition, for Example 5.3, we also do comparisons among three neural networks based on FB function (considered in [20]), smoothed NR function (considered in this paper), and smoothed FB function (considered in [27]), respectively. Although Example 5.3 can be solved by all three neural networks, the neural network based on FB function does not behave as good as the other two neural networks, see Fig. 5.

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

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

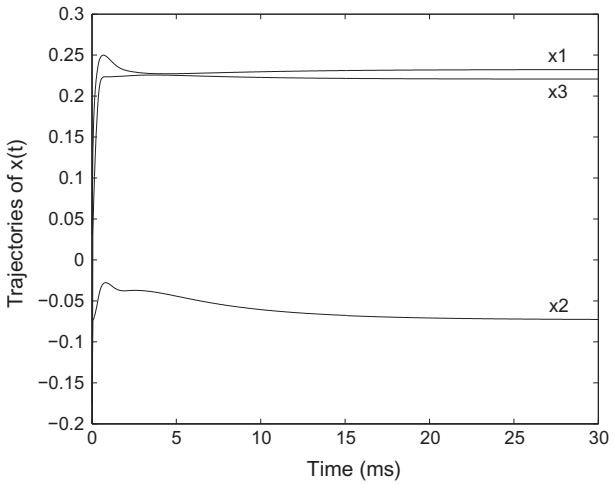

Fig. 4. Transient behavior of the neural network with the smoothed NR function in Example 5.3.

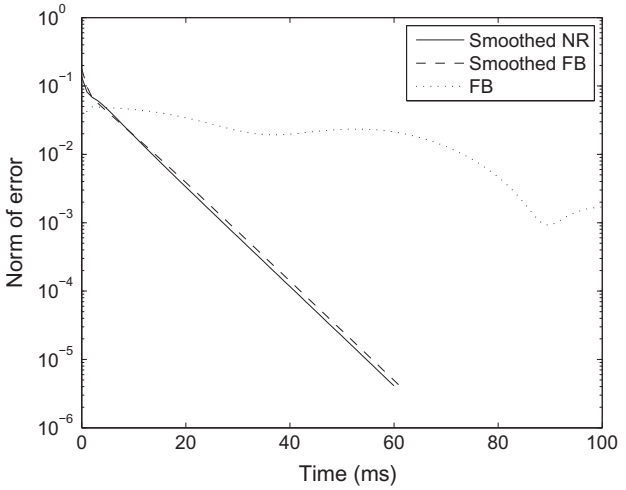

Fig. 5. Comparisons of three neural networks based on the FB function, smoothed NR function, and smoothed FB function in Example 5.3.

For this example, we compute

$$L(x,y,z)=\nabla f(x)+\begin{bmatrix}\nabla g_1(x)y\\ \nabla g_2(x)z\end{bmatrix}=\begin{bmatrix}x_3e^{(x_1x_3)}+6(x_1+x_2)\\ 6(x_1+x_2)+\frac{2(2x_2-x_3)}{\sqrt{1+(2x_2-x_3)^2}}\\ x_1e^{(x_1x_3)}+\frac{2x_2-x_3}{\sqrt{1+(2x_2-x_3)^2}}\\ x_4\\ x_5\end{bmatrix}-\begin{bmatrix}9x_1^2y_1-15x_1^2y_2\\ 2y_1+4y_2\\ (10x_3-1)y_1+(30x_3^2)y_2+y_3\\ z_1\\ 3z_2\end{bmatrix}. \tag{24}$$

Moreover, we know

$$\begin{bmatrix}y+g_1(x)\\ z+g_2(x)\end{bmatrix}=\begin{bmatrix}y_1-3x_1^3-2x_2+x_3-5x_3^2\\ y_2+5x_1^3-4x_2+2x_3-10x_3^3\\ y_3-x_3\\ z_1-x_4\\ z_2-x_5\end{bmatrix}.$$

Let $y+g_1(x):=(u,\bar{u})\in\mathbb{R}\times\mathbb{R}^2$ and $z+g_2(x):=(v,\bar{v})\in\mathbb{R}\times\mathbb{R}$, where $u=y_1-3x_1^3-2x_2+x_3-5x_3^2$,

$$\bar{u}=\begin{bmatrix}y_2+5x_1^3-4x_2+2x_3-10x_3^3\\ y_3-x_3\end{bmatrix}$$

and

$$v=z_1-x_4,\quad \bar{v}=z_2-x_5.$$

Then, $y+g_1(x)$ and $z+g_2(x)$ can be expressed as follows:

$$y+g_1(x):=\lambda_1e_1+\lambda_2e_2$$

and

$$z+g_2(x):=\kappa_1f_1+\kappa_2f_2$$

where $\lambda_i=u+(-1)^i\|\bar{u}\|$, $e_i=\frac{1}{2}\left(1,(-1)^i\frac{\bar{u}}{\|\bar{u}\|}\right)$ and $\kappa_i=v+(-1)^i|\bar{v}|$, $f_i=\frac{1}{2}\left(1,(-1)^i\frac{\bar{v}}{|\bar{v}|}\right)$ ($i=1,2$) if $\bar{u}\neq0$ and $\bar{v}\neq0$, otherwise $e_i=\frac{1}{2}(1,(-1)^iw)$ with w being any element in $\mathbb{R}^2$ satisfying $\|w\|=1$, and $f_i=\frac{1}{2}(1,(-1)^iv)$ with v being any vector in $\mathbb{R}$ satisfying $|v|=1$. This implies that

$$\Phi_\mu(\cdot)=\begin{bmatrix}y-P_\mu(y+g_1(x))\\ z-P_\mu(z+g_2(x))\end{bmatrix}=\begin{bmatrix}y-\mu\hat{g}\left(\frac{\lambda_1}{\mu}\right)e_1+\mu\hat{g}\left(\frac{\lambda_2}{\mu}\right)e_2\\ z-\mu\hat{g}\left(\frac{\kappa_1}{\mu}\right)f_1+\mu\hat{g}\left(\frac{\kappa_2}{\mu}\right)f_2\end{bmatrix} \tag{25}$$

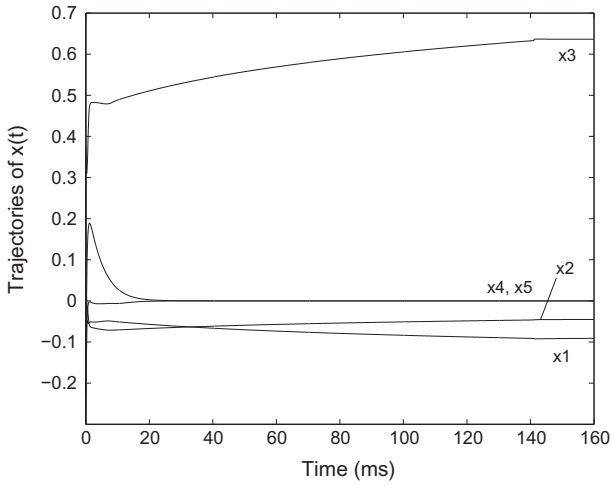

Fig. 6. Transient behavior of the neural network with the smoothed NR function in Example 5.4.

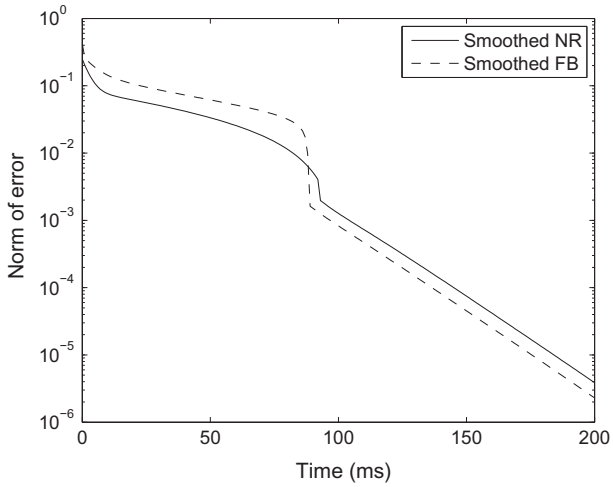

Fig. 7. Comparisons of two neural network based on smoothed NR function and smoothed FB function in Example 5.4.

with $\hat{g}(a) = \frac{\sqrt{a^2+4}+a}{2}$ or $\hat{g}(a) = \ln(e^a + 1)$. Consequently, by Eqs. (24) and (25), we obtain the expression of $H(u)$ as follows:

$$H(u) = \begin{bmatrix} \mu \\ L(x,y,z) \\ \Phi_\mu(\cdot) \end{bmatrix}.$$

This problem has an approximate solution $x^* = (-0.0903, -0.0449, 0.6366, 0.0001, 0)^T$ and Fig. 6 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, the neural network with the smoothed NR function has a fast convergence rate.

Furthermore, we also do comparisons between two neural networks based on smoothed NR function (considered in this paper) and smoothed FB function (considered in [27]) for Example 5.5. Note that Example 5.5 cannot be solved by the neural networks studied in [20]. Both neural networks possess exponential stability as shown in Table 1, which means the solution trajectories have the same order of convergence. This phenomenon is reflected in Fig. 7.

6. Conclusion

In this paper, we have studied a neural network approach for solving general nonlinear convex programs with second-order cone constraints. The proposed neural network is based on the gradient of the merit function derived from smoothed

NR complementarity function. In particular, from Definition 2.1 and Lemma 2.3, we know that there exists a stable equilibrium point u^* as long as there exists a Lyapunov function over some neighborhood of u^*, and the stable equilibrium point u^* is exactly the solution of our considering problem. In addition to studying the existence and convergence of the solution trajectory of the neural network, this paper shows that the merit function is a Lyapunov function. Furthermore, the equilibrium point of the neural network (8) is stable, including the stability in the sense of Lyapunov, asymptotic stability, and exponential stability under suitable conditions.

Indeed, this paper can be viewed as a follow-up of [20,27] because we establish three stabilities for the proposed neural network, but not all three stabilities for the similar neural networks studied in [20,27] are guaranteed. The numerical experiments presented in this study demonstrate the efficiency of the proposed neural network.

Acknowledgment

The authors are grateful to the reviewers for their valuable suggestions, which have considerably improved the paper a lot.

Appendix A

For the function $P_\mu(z)$ defined as in (5), the following Lemma provides the gradient matrix of $P_\mu(z)$, which will be used in numerical computation and coding.

Lemma 6.1. *For any $z = (z_1, z_2^T)^T \in \mathbb{R}^n$, the gradient matrix of $P_\mu(z)$ is written as*

$$\nabla P_\mu(z) = \begin{cases} \hat{g}'\left(\frac{z_1}{\mu}\right) I & \text{if } z_2 = 0; \\ \begin{bmatrix} b_\mu & \frac{c_\mu z_2^T}{\|z_2\|} \\ \frac{c_\mu z_2}{\|z_2\|} & a_\mu I + (b_\mu - a_\mu)\frac{z_2 z_2^T}{\|z_2\|^2} \end{bmatrix} & \text{if } z_2 \neq 0, \end{cases}$$

where

$$a_\mu = \frac{\hat{g}\left(\frac{\lambda_2}{\mu}\right) - \hat{g}\left(\frac{\lambda_1}{\mu}\right)}{\frac{\lambda_2}{\mu} - \frac{\lambda_1}{\mu}},$$

$$b_\mu = \frac{1}{2}\left(\hat{g}'\left(\frac{\lambda_2}{\mu}\right) + \hat{g}'\left(\frac{\lambda_1}{\mu}\right)\right),$$

$$c_\mu = \frac{1}{2}\left(\hat{g}'\left(\frac{\lambda_2}{\mu}\right) - \hat{g}'\left(\frac{\lambda_1}{\mu}\right)\right),$$

and I denotes the identity matrix.

Proof. See Proposition 5.2 in [11]. □

References

[1] F. Alizadeh, D. Goldfarb, Second-order cone programming, Mathematical Programming 95 (2003) 3–52.
[2] D.P. Bertsekas, Nonlinear Programming, Athena Scientific, 1995.
[3] Y.-H. Chen, S.-C. Fang, Solving convex programming problems with equality constraints by neural networks, Computers and Mathematics with Applications 36 (1998) 41–68.
[4] J.-S. Chen, C.-H. Ko, S.-H. Pan, A neural network based on the generalized Fischer–Burmeister function for nonlinear complementarity problems, Information Sciences 180 (2010) 697–711.
[5] J.-S. Chen, P. Tseng, An unconstrained smooth minimization reformulation of the second-order cone complementarity problem, Mathematical Programming 104 (2005) 293–327.
[6] A. Cichocki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, John wiley, New York, 1993.
[7] C. Dang, Y. Leung, X. Gao, K. Chen, Neural networks for nonlinear and mixed complementarity problems and their applications, Neural Networks 17 (2004) 271–283.
[8] S. Effati, A. Ghomashi, A.R. Nazemi, Application of projection neural network in solving convex programming problems, Applied Mathematics and Computation 188 (2007) 1103–1114.
[9] S. Effati, A.R. Nazemi, Neural network and its application for solving linear and quadratic programming problems, Applied Mathematics and Computation 172 (2006) 305–331.
[10] F. Facchinei, J. Pang, Finite-dimensional Variational Inequalities and Complementarity Problems, Springer, New York, 2003.
[11] M. Fukushima, Z.-Q. Luo, P. Tseng, Smoothing functions for second-order-cone complementarity problems, SIAM Journal on Optimization 12 (2002) 436–460.
[12] Q. Han, L.-Z. Liao, H. Qi, L. Qi, Stability analysis of gradient-based neural networks for optimization problems, Journal of Global Optimization 19 (2001) 363–381.
[13] J.J. Hopfield, D.W. Tank, Neural computation of decision in optimization problems, Biological Cybernetics 52 (1985) 141–152.

[14] X. Hu, J. Wang, A recurrent neural network for solving nonlinear convex programs subject to linear constraints, IEEE Transactions on Neural network 16 (3) (2005) 379–386.
[15] X. Hu, J. Wang, A recurrent neural network for solving a class of general variational inequalities, IEEE Transactions on Systems, Man, and Cybernetics-B 37 (2007) 528–539.
[16] S. Hayashi, N. Yamashita, M. Fukushima, A combined smoothing and regularization method for monotone second-order cone complementarity problems, SIAM Journal on Optimization 15 (2005) 593–615.
[17] N. Kalouptisidis, Sigunal Processing Systems, Theory and Design, Wiley, New York, 1997.
[18] 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 Journal on Optimization 20 (2009) 297–320.
[19] M.P. Kennedy, L.O. Chua, Neural network for nonlinear programming, IEEE Transactions on Circuits and Systems 35 (1988) 554–562.
[20] C.-H. Ko, J.-S. Chen, C.-Y. Yang, Recurrent neural networks for solving second-order cone programs, Neurocomputing 74 (2011) 3646–3653.
[21] L.-Z. Liao, H. Qi, L. Qi, Solving nonlinear complementarity problems with neural networks: a reformulation method approach, Journal of Computational and Applied Mathematics 131 (2001) 342–359.
[22] 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, Information Sciences 220 (2013) 331–342.
[23] O. Mancino, G. Stampacchia, Convex programming and variational inequalities, Journal of Optimization Theory and Applications 9 (1972) 3–23.
[24] R.K. Miller, A.N. Michel, Ordinary Differential Equations, Academic Press, 1982.
[25] S.-H. Pan, J.-S. Chen, A semismooth Newton method for the SOCCP based on a one-parametric class of SOC complementarity functions, Computational Optimization and Applications 45 (2010) 59–88.
[26] J.-S. Pang, L. Qi, Nonsmooth equation: motivation and algorithms, SIAM Journal on Optimization 3 (1993) 443–465.
[27] J.-H. Sun, J.-S. Chen, C.-H. Ko, Neural networks for solving second-order cone constrained variational inequality problem, Computational Optimization and Applications 51 (2012) 623–648.
[28] D.W. Tank, J.J. Hopfield, Simple neural optimization network: an A/D converter, signal decision circuit, and a linear programming circuit, IEEE Transactions on Circuits and Systems 33 (1986) 533–541.
[29] S.J. Wright, An infeasible-interior-point algorithm for linear complementarity problems, Mathematical Programming 67 (1994) 29–51.
[30] A.L. Wu, S.P. Wen, Z.G. Zeng, Synchronization control of a class of memristor-based recurrent neural networks, Information Sciences 183 (2012) 106–116.
[31] Y. Xia, H. Leung, J. Wang, A projection neural network and its application to constrained optimization problems, IEEE Transactions on Circuits and Systems-I 49 (2002) 447–458.
[32] Y. Xia, H. Leung, J. Wang, A general projection neural network for solving monotone variational inequalities and related optimization problems, IEEE Transactions on Neural Networks 15 (2004) 318–328.
[33] Y. Xia, J. Wang, A recurrent neural network for solving nonlinear convex programs subject to linear constraints, IEEE Transactions on Neural Networks 16 (2005) 379–386.
[34] M. Yashtini, A. Malek, Solving complementarity and variational inequalities problems using neural networks, Applied Mathematics and Computation 190 (2007) 216–230.
[35] J. Zabczyk, Mathematical Control Theorem: An Introduction, Birkhäuser, Boston, 1992.
[36] 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, Information Sciences 232 (2013) 386–396.