Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 4545064, 18 pages
https://doi.org/10.1155/2019/4545064

Research Article

Neural Network for Solving SOCQP and SOCCVI Based on Two Discrete-Type Classes of SOC Complementarity Functions

Juhe Sun,[1] Xiao-Ren Wu,[2] B. Saheya,[3] Jein-Shan Chen ,[2] and Chun-Hsu Ko [4]

[1] *School of Science, Shenyang Aerospace University, Shenyang 110136, China*
[2] *Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan*
[3] *College of Mathematical Science, Inner Mongolia Normal University, Hohhot 010022, Inner Mongolia, China*
[4] *Department of Electrical Engineering, I-Shou University, Kaohsiung 840, Taiwan*

Correspondence should be addressed to Jein-Shan Chen; jschen@ntnu.edu.tw

Received 15 November 2018; Revised 19 January 2019; Accepted 3 February 2019; Published 14 February 2019

Academic Editor: Sabri Arik

This paper focuses on solving the quadratic programming problems with second-order cone constraints (SOCQP) and the second-order cone constrained variational inequality (SOCCVI) by using the neural network. More specifically, a neural network model based on two discrete-type families of SOC complementarity functions associated with second-order cone is proposed to deal with the Karush-Kuhn-Tucker (KKT) conditions of SOCQP and SOCCVI. The two discrete-type SOC complementarity functions are newly explored. The neural network uses the two discrete-type families of SOC complementarity functions to achieve two unconstrained minimizations which are the merit functions of the Karuch-Kuhn-Tucker equations for SOCQP and SOCCVI. We show that the merit functions for SOCQP and SOCCVI are Lyapunov functions and this neural network is asymptotically stable. The main contribution of this paper lies on its simulation part because we observe a different numerical performance from the existing one. In other words, for our two target problems, more effective SOC complementarity functions, which work well along with the proposed neural network, are discovered.

1. Introduction

In optimization community, it is well known that there are many computational approaches to solve the optimization problems such as linear programming, nonlinear programming, variational inequalities, and complementarity problems; see [1–6] and references therein. These approaches include the method using merit function, interior point method, Newton method, nonlinear equation method, projection method, and its variant versions. All the aforementioned methods rely on iterative schemes and usually only provide "approximate" solution(s) to the original optimization problems and do not offer real-time solutions. However, real-time solutions are eager in many applications, such as force analysis in robot grasping and control applications. Therefore, the traditional optimization methods may not be suitable for these applications due to stringent computational time requirements.

The neural network approach has an advantage in solving real-time optimization problems, which was proposed by Hopfield and Tank [7, 8] in the 1980s. Since then, neural networks have been applied to various optimization problems; see [9–30] and references therein. Unlike the traditional optimization algorithms, the essence of neural network approach for optimization 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 and its trajectory is likely convergent to an equilibrium point, which corresponds to the solution to the considered optimization problem.

Following the similar idea, researchers have also developed many continuous-time neural networks for second-order cone constrained optimization problems. For example, Ko, Chen and Yang [31] proposed two kinds of neural networks with different SOCCP functions for solving the second-order cone program; Sun, Chen, and Ko [32] gave two kinds of neural networks (the first one is based on the Fischer-Burmeister function and the second one relies on a projection function) to solve the second-order cone constrained variational inequality (SOCCVI) problem; Miao, Chen, and Ko [33] proposed a neural network model for efficiently solving general nonlinear convex programs with second-order cone constraints. In this paper, we are interested in employing neural network approach for solving two types of SOC constrained problems, the quadratic programming problems with second-order cone constraints (SOCQP for short) and the second-order cone constrained variational inequality (SOCCVI for short), whose mathematical formats are described as below.

The SOCQP is in the form of

$$\begin{aligned} \min \quad & \frac{1}{2}x^T Q x + c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \in \mathcal{K} \end{aligned} \tag{1}$$

where $Q \in \mathbb{R}^{n\times n}$, A is an $m \times n$ matrix with full row rank, $b \in \mathbb{R}^m$, and $\mathcal{K}$ is the Cartesian product of second-order cones (SOCs), also called Lorentz cones. In other words,

$$\mathcal{K} = \mathcal{K}^{n_1} \times \mathcal{K}^{n_2} \times \cdots \times \mathcal{K}^{n_q} \tag{2}$$

where $n_1, \dots, n_q, q$ are positive integers, $n_1 + \cdots + n_q = n$, and $\mathcal{K}^{n_i}$ denotes the SOC in $\mathbb{R}^{n_i}$ defined by

$$\mathcal{K}^{n_i} := \left\{ x_i = (x_{i1}, x_{i2}) \in \mathbb{R} \times \mathbb{R}^{n_i - 1} \mid \|x_{i2}\| \le x_{i1} \right\}. \tag{3}$$

with $\mathcal{K}^1$ denoting the nonnegative real number set $\mathbb{R}_+$. A special case of (3) corresponds to the nonnegative orthant cone $\mathbb{R}^n_+$, i.e., $q = n$ and $n_1 = \cdots = n_q = 1$. We assume that Q is a symmetric positive semidefinite matrix and problem (1) satisfies a suitable qualification [34], such as the generalized Slater condition that there exists $\hat{x}$ with strictly feasibility, then x is a solution to problem (1) if and only if there exists a Lagrange multiplier $(\mu, y) \in \mathbb{R}^m \times \mathbb{R}^n$ such that

$$\begin{gathered} Ax - b = 0 \\ c + Qx + A^T \mu - y = 0 \\ \mathcal{K} \ni y \perp x \in \mathcal{K} \end{gathered} \tag{4}$$

In Section 3, we will employ two new families of SOC complementarity functions and use (4) to build up the neural network model for solving SOCQP.

We say a few words about why we assume that Q is a symmetric positive semidefinite matrix. First, it is clear that the symmetric assumption is reasonable because Q can be replaced by $(1/2)(Q^T + Q)$ which is symmetric. Indeed, with Q being symmetric positive definite matrix, the SOCQP can be recast as a standard SOCP. To see this, we observe that

$$\frac{1}{2}x^T Q x + c^T x = \frac{1}{2}\left\|Q^{1/2}x + Q^{-1/2}c\right\|^2 - \frac{1}{2}c^T Q^{-1} c \tag{5}$$

which is done by completing the square. Then, the SOCQP (with $\mathcal{K} = \mathcal{K}^n$) is equivalent to

$$\begin{aligned} \min \quad & \left\|Q^{1/2}x + Q^{-1/2}c\right\| \\ \text{s.t.} \quad & Ax = b \\ & x \in \mathcal{K}^n \end{aligned} \tag{6}$$

which is also the same as

$$\begin{aligned} \min \quad & \|\overline{y}\| \\ \text{s.t.} \quad & Q^{1/2}x - \overline{y} = -Q^{-1/2}c \\ & Ax = b \\ & x \in \mathcal{K}^n \end{aligned} \tag{7}$$

This formulation is further equivalent to

$$\begin{aligned} \min \quad & y_1 \\ \text{s.t.} \quad & Q^{1/2}x - \overline{y} = -Q^{-1/2}c \\ & Ax = b \\ & x \in \mathcal{K}^n \\ & y_1 \ge \|\overline{y}\| \end{aligned} \tag{8}$$

Now, we let $y := (y_1, \overline{y})$ which says $y \in \mathcal{K}^{n+1}$ and denote

$$\begin{aligned} \hat{v} &:= (x, y) \in \mathcal{K}^n \times \mathcal{K}^{n+1}, \\ \hat{c} &:= (0, e) \in \mathbb{R}^{2n+1}, \\ \widehat{A} &:= \begin{bmatrix} A & 0 & 0 \\ Q^{1/2} & 0 & I_n \end{bmatrix}, \\ \hat{b} &:= \begin{bmatrix} b \\ Q^{-1/2} \end{bmatrix}. \end{aligned} \tag{9}$$

Thus, the above reformulation (8) is expressed as a standard SOCP as follows:

$$\begin{aligned} \min \quad & (\hat{c})^T \hat{v} \\ \text{s.t.} \quad & \widehat{A}\hat{v} = \hat{b} \\ & \hat{v} \in \mathcal{K}^n \times \mathcal{K}^{n+1} \end{aligned} \tag{10}$$

In view of this reformulation (10), we focus on SOCQP with Q being symmetric positive semidefinite in this paper.

The SOCCVI, our another target problem, is to find $x \in C$ satisfying

$$\langle F(x), y - x \rangle \ge 0 \quad \forall y \in C, \tag{11}$$

where the set C is finitely representable and is given by

$$C = \{x \in \mathbb{R}^n \mid h(x) = 0,\ -g(x) \in \mathcal{K}\}. \tag{12}$$

Here $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product, $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, $h : \mathbb{R}^n \longrightarrow \mathbb{R}^l$, and $g : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ are continuously differentiable functions, and $\mathscr{K}$ is a Cartesian product of second-order cones (or Lorentz cones), expressed as

$$\mathscr{K} = \mathscr{K}^{m_1} \times \mathscr{K}^{m_2} \times \cdots \times \mathscr{K}^{m_p}, \tag{13}$$

with $l \geq 0$, $m_1, m_2, \ldots, m_p \geq 1$, $m_1 + m_2 + \cdots + m_p = m$. When h is affine, an important special case of the SOCCVI corresponds to the KKT conditions of the convex second-order cone program (CSOCP):

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & Ax = b \\ & -g(x) \in \mathscr{K} \end{aligned} \tag{14}$$

where $A \in \mathbb{R}^{l \times n}$ has full row rank, $b \in \mathbb{R}^l$, $g : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, and $f : \mathbb{R}^n \longrightarrow \mathbb{R}$. Furthermore, when f is a convex twice continuously differentiable function, problem (14) is equivalent to the following SOCCVI which is to find $x \in C$ such that

$$\langle \nabla f(x), y - x \rangle \geq 0, \quad \forall y \in C, \tag{15}$$

where

$$C = \{x \in \mathbb{R}^n \mid Ax - b = 0, \ -g(x) \in \mathscr{K}\}. \tag{16}$$

In fact, the SOCCVI can be solved by analyzing its KKT conditions:

$$\begin{aligned} & L(x, \mu, \lambda) = 0, \\ & \langle g(x), \lambda \rangle = 0, \quad -g(x) \in \mathscr{K}, \ \lambda \in \mathscr{K}, \\ & h(x) = 0, \end{aligned} \tag{17}$$

where $L(x, \mu, \lambda) = F(x) + \nabla h(x)\mu + \nabla g(x)\lambda$ is the variational inequality Lagrangian function, $\mu \in \mathbb{R}^l$ and $\lambda \in \mathbb{R}^m$. We also point out that the neural network approach for SOCCVI was already studied in [32]. Here we revisit the SOCCVI with different neural models. More specifically, in our earlier work [32], we had employed neural network approach to the SOCCVI problem (11) and (13), in which the neural networks were aimed at solving system (17) whose solutions are candidates of SOCCVI problem (11) and (13). There were two neural networks considered in [32]. The first one is based on the smoothed Fischer-Burmeister function, while the other one is based on the projection function. Both neural networks possess asymptotical stability under suitable conditions. In Section 4, in light of (17) again, we adopt new and different SOC complementarity functions to construct our new neural networks.

As mentioned earlier, this paper studies neural networks by using two new classes of SOC complementarity functions to efficiently solve SOCQP and SOCCVI. Although the idea and the stability analysis for both problems are routine, we emphasize that the main contribution of this paper lies on its simulations. More specifically, from numerical performance and comparison, we observe a new phenomenon different from the existing one in the literature. This may suggest update choices of SOC complementarity functions to work with neural network approach.

2. Preliminaries

Consider the first-order differential equations (ODE):

$$\begin{aligned} \dot{w}(t) &= H(w(t)), \\ w(t_0) &= w_0 \in \mathbb{R}^n, \end{aligned} \tag{18}$$

where $H : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is a mapping. A point $w^* = w(t^*)$ is called an equilibrium point or a steady state of the dynamic system (18) if $H(w^*) = 0$. If there is a neighborhood $\Omega^* \subseteq \mathbb{R}^n$ of w^* such that $H(w^*) = 0$ and $H(w) \neq 0$ $\forall w \in \Omega^* \setminus \{w^*\}$, then w^* is called an isolated equilibrium point.

Lemma 1. *Suppose that $H : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is a continuous mapping. Then, for any $t_0 > 0$ and $w_0 \in \mathbb{R}^n$, there exists a local solution $w(t)$ to (18) with $t \in [t_0, \tau)$ for some $\tau > t_0$. If, in addition, H is locally Lipschitz continuous at x_0, then the solution is unique; if H is Lipschitz continuous in $\mathbb{R}^n$, then τ can be extended to ∞.*

Let $w(t)$ be a solution to dynamic system (18). An isolated equilibrium point w^* is Lyapunov stable if for any $w_0 = w(t_0)$ and any $\varepsilon > 0$, there exists a $\delta > 0$ such that $\|w(t) - w^*\| < \varepsilon$ for all $t \geq t_0$ and $\|w(t_0) - w^*\| < \delta$. An isolated equilibrium point w^* is said to be asymptotic stable if in addition to being Lyapunov stable, it has the property that $w(t) \longrightarrow w^*$ as $t \longrightarrow \infty$ for all $\|w(t_0) - w^*\| < \delta$. An isolated equilibrium point w^* is exponentially stable if there exists $\delta > 0$ such that arbitrary point $w(t)$ of (18) with the initial condition $w(t_0) = w_0$ and $\|w(t_0) - w^*\| < \delta$ is well defined on $[0, +\infty)$ and satisfies

$$\|w(t) - w^*\| \leq c e^{-\omega t} \|w(t_0) - w^*\| \quad \forall t \geq t_0, \tag{19}$$

where $c > 0$ and $\omega > 0$ are constants independent of the initial point.

Let $\Omega \subseteq \mathbb{R}^n$ be an open neighborhood of $\overline{w}$. A continuously differentiable function $V : \mathbb{R}^n \longrightarrow \mathbb{R}$ is said to be a Lyapunov function at the state $\overline{w}$ over the set Ω for (18) if

$$\begin{aligned} & V(\overline{w}) = 0, \\ & V(w) > 0, \\ & \qquad \forall w \in \Omega \setminus \{\overline{w}\}, \\ & \dot{V}(w) \leq 0, \quad \forall w \in \Omega \setminus \{\overline{w}\}. \end{aligned} \tag{20}$$

The Lyapunov stability and asymptotical stability can be verified by using Lyapunov function, which is a useful tool for analysis.

Lemma 2. *(a) An isolated equilibrium point w^* is Lyapunov stable if there exists a Lyapunov function over some neighborhood Ω^* of w^*.*

(b) An isolated equilibrium point w^* is asymptotically stable if there exists a Lyapunov function over some neighborhood Ω^* of w^* such that $\dot{V}(w) < 0$, $\forall w \in \Omega^* \setminus \{w^*\}$.

For more details, please refer to any usual ODE textbooks, e.g., [35].

Next, we briefly recall some concepts associated with SOC, which are helpful for understanding the target problems and our analysis techniques. We start with introducing the Jordan product and SOC complementarity function. For any $x = (x_1, x_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$ and $y = (y_1, y_2) \in \mathbb{R} \times \mathbb{R}^{n-1}$, we define their Jordan product associated with $\mathcal{K}^n$ as

$$x \circ y = \begin{bmatrix} x^T y \\ y_1 x_2 + x_1 y_2 \end{bmatrix}. \tag{21}$$

The Jordan product $\circ$, unlike scalar or matrix multiplication, is not associative, which is a main source of complication in the analysis of SOC constrained optimization. There exists an identity element under this product, which is denoted by $e := (1, 0, \ldots, 0)^T \in \mathbb{R}^n$. Note that x^2 means $x \circ x$ and $x + y$ means the usual componentwise addition of vectors. It is known that $x^2 \in \mathcal{K}^n$ for all $x \in \mathbb{R}^n$. Moreover, if $x \in \mathcal{K}^n$, then there exists a unique vector in $\mathcal{K}^n$, denoted by $x^{1/2}$, such that $(x^{1/2})^2 = x^{1/2} \circ x^{1/2} = x$. We also denote $|x| := (x^2)^{1/2}$.

A vector-valued function $\phi : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is called an SOC complementarity function if it satisfies

$$\begin{aligned} &\phi(x, y) = 0 \Longleftrightarrow \\ &\quad x \circ y = 0, \\ &\qquad\qquad x \in \mathcal{K}^n, \; y \in \mathcal{K}^n. \end{aligned} \tag{22}$$

There have been many SOC complementarity functions studied in the literature; see [36–40] and references therein. Among them, two popular ones are the Fischer-Burmeister function ϕ_{FB} and the natural residual function ϕ_{NR}, which are given by

$$\begin{aligned} \phi_{\mathrm{FB}}(x, y) &= \left(x^2 + y^2\right)^{1/2} - (x + y), \\ \phi_{\mathrm{NR}}(x, y) &= x - (x - y)_+. \end{aligned} \tag{23}$$

Some existing SOC complementarity functions are indeed variants of ϕ_{FB} and ϕ_{NR}. Recently, Ma, Chen, Huang, and Ko [41] explored the idea of "discrete generalization" to the Fischer-Burmeister function which yields the following class of functions (denoted by $\phi^p_{\mathrm{D-FB}}$):

$$\phi^p_{\mathrm{D-FB}}(x, y) = \left(\sqrt{x^2 + y^2}\right)^p - (x + y)^p, \tag{24}$$

where $p > 1$ is a positive odd integer. Applying similar idea, they also extended ϕ_{NR} to another family of SOC complementarity functions, $\phi^p_{\mathrm{NR}} : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$, whose formula is as follows:

$$\phi^p_{\mathrm{NR}}(x, y) = x^p - [(x - y)_+]^p, \tag{25}$$

where $p > 1$ is a positive odd integer and $(\cdot)_+$ means the projection onto $\mathcal{K}^n$. The functions $\phi^p_{\mathrm{D-FB}}$ and ϕ^p_{NR} are continuously differentiable SOC complementarity functions with computable Jacobian, which can be found in [41].

3. Neural Networks for SOCQP

In this section, we first show how we achieve the neural network model for SOCQP and prove various stabilities for it accordingly. Then, numerical experiments are reported to demonstrate the effectiveness of the proposed neural network.

3.1. *The model and Stability Analysis.*

As mentioned in Section 2, the KKT conditions are expressed in (4). With system (4) and using a given SOC complementarity function $\phi : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$, it is clear to see that system (4) is equivalent to

$$H(u) = \begin{bmatrix} Ax - b \\ c + Qx + A^T\mu - y \\ \phi(x, y) \end{bmatrix} = 0, \tag{26}$$

where $u = (x, \mu, y) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n$. Moreover, we can specifically describe $\nabla H(u)$ as the following:

$$\nabla H(u) = \begin{bmatrix} A^T & Q & \nabla_x \phi \\ 0 & A & 0 \\ 0 & -I & \nabla_y \phi \end{bmatrix}. \tag{27}$$

Here ϕ is a continuously differentiable SOC complementarity function such as $\phi^p_{\mathrm{D-FB}}$ and ϕ^p_{NR} introduced in Section 2. It is clear that if u^* solves $H(u) = 0$, then u^* solves $\nabla((1/2)\|H(u)\|^2) = 0$. Accordingly, we consider a specific first-order ordinary differential equation as follows:

$$\begin{aligned} \frac{du(t)}{dt} &= -\rho \nabla \left(\frac{1}{2}\|H(u)\|^2\right), \\ u(t_0) &= u_0, \end{aligned} \tag{28}$$

where $\rho > 0$ is a time scaling factor. In fact, letting $\tau = \rho t$, then $du(\tau)/d\tau = \rho(du(\tau)/d\tau)$. Hence, it follows from (28) that $du(\tau)/d\tau = -\nabla((1/2)\|H(u^*)\|^2)$. In view of this, we set $\rho = 1$ in the subsequent analysis. Next, we show that the equilibrium of the neural network (28) corresponds to the solution to system (4).

Lemma 3. *Let u^* be an equilibrium of the neural network (28) and suppose that $\nabla H(u^*)$ is nonsingular. Then u^* solves system (4).*

Proof. Since $\nabla((1/2)\|H(u^*)\|^2) = \nabla H(u^*) H(u^*)$ and $\nabla H(u^*)$ is nonsingular, it is clear to see that $\nabla((1/2)\|H(u^*)\|^2) = 0$ if and only if $H(u^*) = 0$. □

Besides, the following results address the existence and uniqueness of the solution trajectory of the neural network (28).

Theorem 4. *(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 (28).*

(b) If the level set $\mathcal{L}(u_0) := \{u \mid \|H(u)\|^2 \le \|H(u_0)\|^2\}$ is bounded, then τ can be extended to ∞.

Proof. This proof is exactly the same as the one in [32, Proposition 3.4], so we omit it here. □

Now, we are ready to analyze the stability of an isolated equilibrium u^* of the neural network (28), which means $\nabla((1/2)\|H(u^*)\|^2) = 0$ and $\nabla((1/2)\|H(u)\|^2) \neq 0$ for $u \in \Omega \setminus \{u^*\}$, with Ω being a neighborhood of u^*.

Theorem 5. *Let u^* be an isolated equilibrium point of the neural network (28).*

(a) *If $\nabla H(u^*)$ is nonsingular, then the isolated equilibrium point u^* is asymptotically stable and hence Lypunov stable.*

(b) *If $\nabla H(u)$ is nonsingular for all $u \in \Omega$, then the isolated equilibrium point u^* is exponentially stable.*

Proof. The desired results can be proved by using Lemma 3 and mimicking the arguments as in [32, Theorem 3.1]. □

3.2. Numerical Experiments. In order to demonstrate the effectiveness of the proposed neural network, we test three examples for our neural network (28). 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, in general the parameter ρ is set to be 1. For some special examples, the parameter ρ is set to be another value.

Example 6. Consider the following SOCQP problem:

$$\begin{aligned} \min \quad & (x_1-3)^2 + x_2^2 + (x_3-1)^2 + (x_4-2)^2 \\ & + (x_5+1)^2 \\ \text{s.t.} \quad & x \in \mathcal{K}^5 \end{aligned} \tag{29}$$

After suitable transformation, it can be recast as an SOCQP with $Q = 2I_5$, $c = [-6, 0, -2, -4, 2]^T$, $A = 0$, and $b = 0$. This problem has an optimal solution $x^* = [3, 0, 1, 2, -1]^T$. Now, we use the proposed neural network (28) with two cases $\phi = \phi^p_{\text{D-FB}}$ and $\phi = \phi^p_{\text{NR}}$, respectively, to solve the above SOCQP and their trajectories are depicted in Figures 1–4. For the sake of coding needs and check, the following expressions are presented.

For case of $\phi = \phi^p_{\text{D-FB}}$, we have

$$\begin{aligned} \frac{du(t)}{dt} &= -\rho \nabla H(u) H(u), \\ u(t_0) &= u_0 \\ H(u) &= \begin{bmatrix} c + 2x - y \\ \phi^p_{\text{D-FB}}(x, y) \end{bmatrix}, \quad u = (x, y) \\ \nabla H(u) &= \begin{bmatrix} 2I_5 & \nabla_x \phi^p_{\text{D-FB}}(u) \\ -I_5 & \nabla_y \phi^p_{\text{D-FB}}(u) \end{bmatrix} \end{aligned}$$

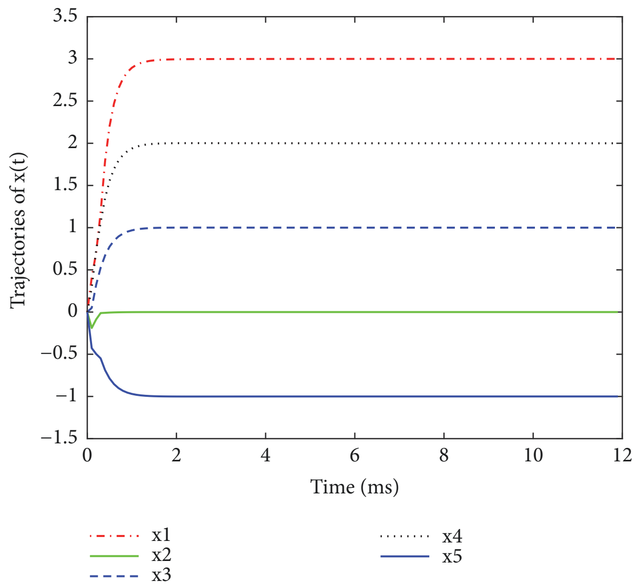


Figure 1: Transient behavior of the neural network with $\phi^p_{\text{D-FB}}$ function ($p = 3$) in Example 6.

$$\begin{aligned} \nabla_x \phi^p_{\text{D-FB}}(x, y) &= 2L_x \nabla g^{\text{soc}}(w) - 2L_{(x+y)} \nabla g^{\text{soc}}(v), \\ \nabla_y \phi^p_{\text{D-FB}}(x, y) &= 2L_y \nabla g^{\text{soc}}(w) - 2L_{(x+y)} \nabla g^{\text{soc}}(v). \\ w(x, y) &:= x^2 + y^2 = (w_1(x, y), w_2(x, y)) \\ &= \left(\|x\|^2 + \|y\|^2, 2(x_1 x_2 + y_1 y_2)\right) \\ &\in \mathbb{R} \times \mathbb{R}^4, \\ v(x, y) &:= (x + y)^2 \\ &= \left(\|x + y\|^2, 2(x_1 + y_1)(x_2 + y_2)\right) \\ &\in \mathbb{R} \times \mathbb{R}^4. \end{aligned} \tag{30}$$

Note that the element $w = (w_1, w_2) \in \mathbb{R} \times \mathbb{R}^4$ can also be expressed as

$$w := \lambda_1 e_1 + \lambda_2 e_2 \tag{31}$$

where $\lambda_i = w_1 + (-1)^i \|w_2\|$ and $e_i = (1/2)(1, (-1)^i (w_2/\|w_2\|))$ $(i = 1, 2)$ if $w_2 \neq 0$; otherwise $e_i = (1/2)(1, (-1)^i \nu$ with ν being any vector in $\mathbb{R}^4$ satisfying $\|\nu\| = 1$.

For case of $\phi = \phi^p_{\text{NR}}$, we replace $\phi^p_{\text{D-FB}}(x, y)$ as $\phi^p_{\text{NR}}(x, y)$, Hence, $H(u)$ and $\nabla H(u)$ have the forms as follows:

$$\begin{aligned} H(u) &= \begin{bmatrix} c + 2x - y \\ \phi^p_{\text{NR}}(x, y) \end{bmatrix}, \quad u = (x, y) \\ \nabla H(u) &= \begin{bmatrix} 2I_5 & \nabla_x \phi^p_{\text{NR}}(u) \\ -I_5 & \nabla_y \phi^p_{\text{NR}}(u) \end{bmatrix} \end{aligned}$$

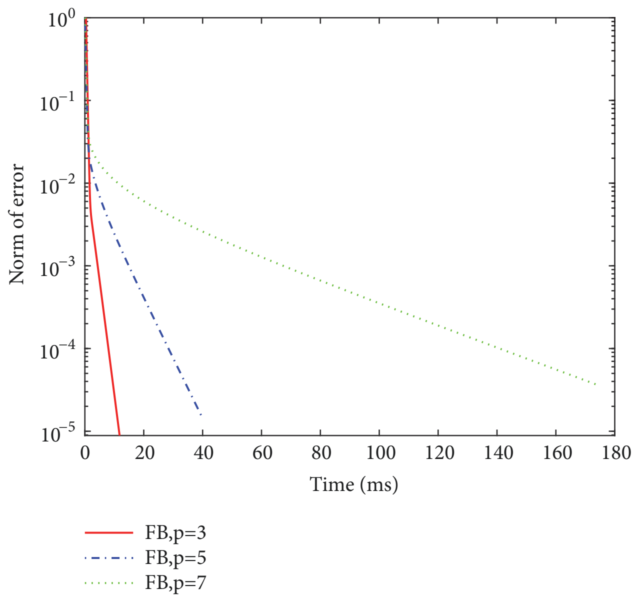


FIGURE 2: Convergence comparison of $\phi^p_{\text{D-FB}}$ function with different p value for Example 6.

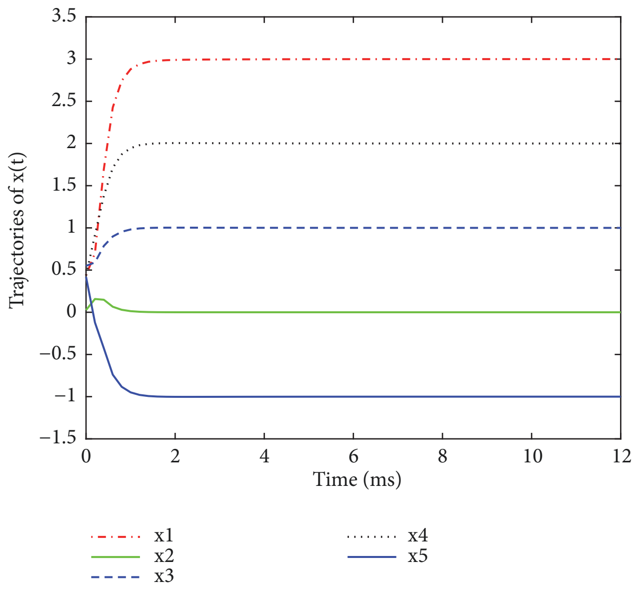


FIGURE 3: Transient behavior of the neural network with ϕ^p_{NR} function ($p = 3$) in Example 6.

$$\nabla_x \phi^p_{\text{NR}}(x, y) = \nabla h^{\text{soc}}(x) - \nabla l^{\text{soc}}(x - y),$$
$$\nabla_y \phi^p_{\text{NR}}(x, y) = \nabla l^{\text{soc}}(x - y). \tag{32}$$

Figures 1 and 3 show the transient behaviors of Example 6 for neural network model (28) based on smooth SOC complementarity functions $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} with initial states $x_0 = [0, 0, 0, 0, 0]^T$, respectively. In Figure 2, we see the convergence comparison of the neural network model using $\phi^p_{\text{D-FB}}$ function with different values of $p = 3, 5, 7$. Figure 4

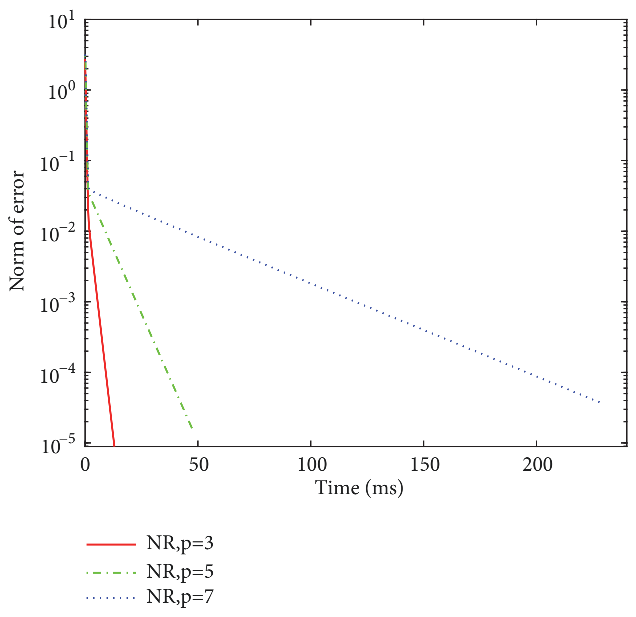


FIGURE 4: Convergence comparison of ϕ^p_{NR} function with different p value for Example 6.

depicts the influence of the parameter p on the value of norm of error for neural network model using ϕ^p_{NR} function.

Example 7. Consider the following SOCQP problem:

$$\begin{aligned} \min \quad & 4x_1^2 + 10x_2^2 + 4x_3^2 + 4x_1x_2 + 12x_2x_3 - x_1 + x_2 \\ & \quad + 5x_3 \\ \text{s.t.} \quad & 2x_1 + x_2 - 7 = 0 \\ & 3x_2 + 2x_3 - 1 = 0 \\ & x \in \mathcal{K}^3 \end{aligned} \tag{33}$$

For this SOCP, we have

$$Q = \begin{bmatrix} 8 & 4 & 0 \\ 4 & 20 & 12 \\ 0 & 12 & 8 \end{bmatrix}, \quad c = \begin{bmatrix} -1 \\ 1 \\ 5 \end{bmatrix}, \quad b = \begin{bmatrix} 7 \\ 1 \end{bmatrix}, \quad A = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 3 & 2 \end{bmatrix}. \tag{34}$$

This problem has an approximate solution $x^* = (2.6529, 1.6943, -2.0414)^T$. Note that the precise solution is $((22 - \sqrt{37})/6, (-2 + 2\sqrt{37})/6, (6 - 3\sqrt{37})/6)^T$. Indeed, we have

$$H(u) = \begin{bmatrix} Ax - b \\ c + Qx + A^T\mu - y \\ \phi(x, y) \end{bmatrix}, \quad u = (x, \mu, y),$$
$$\nabla H(u) = \begin{bmatrix} A^T & Q & \nabla_x\phi(x, y) \\ 0 & A & 0 \\ 0 & -I & \nabla_y\phi(x, y) \end{bmatrix}. \tag{35}$$

We also report numerical experiments for two cases when $\phi = \phi^p_{\text{D-FB}}$ and $\phi = \phi^p_{\text{NR}}$; see Figures 5–8.

Figures 5 and 7 show the transient behaviors of Example 7 for neural network model (28) based on $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} with initial states $x_0 = [0, 0, 0]^T$, respectively. Figure 6 provides the convergence comparison by using $\phi^p_{\text{D-FB}}$ function with different values of $p = 3, 5, 7$. Figure 8 shows the convergence of neural network model using ϕ^p_{NR} function, which indicates that this class of ϕ^p_{NR} functions performs not well for this problem.

Example 8. Consider the following SOCQP problem:

$$\begin{aligned} \min \quad & \frac{5}{2}x_1^2 + 2x_2^2 + \frac{5}{2}x_3^2 + 3x_1x_2 - 2x_2x_3 - x_1x_3 \\ & - 47x_1 - 35x_2 + 2x_3 \\ \text{s.t.} \quad & x \in \mathscr{K}^3 \end{aligned} \tag{36}$$

Here, we have

$$Q = \begin{bmatrix} 5 & 3 & -1 \\ 3 & 4 & -2 \\ -1 & -2 & 5 \end{bmatrix}, \qquad c = [-47, -35, 2]^T, \tag{37}$$

and $A = 0$, $b = 0$. This problem has an optimal solution $x^* = (7, 5, 3)^T$.

Figures 9 and 11 show the transient behaviors of Example 8 for neural network model (28) based on $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} with initial states $x_0 = [0, 0, 0]$, respectively. Figure 10 shows that there are no difference between the neural networks using $\phi^p_{\text{D-FB}}$ function with $p = 3, 5$. Figure 12 elaborates that when $p = 5$ the neural network based on ϕ^p_{NR} function produces fast decrease of norm of error. We point out that the neural network does not converge when $p = 7$ for both cases.

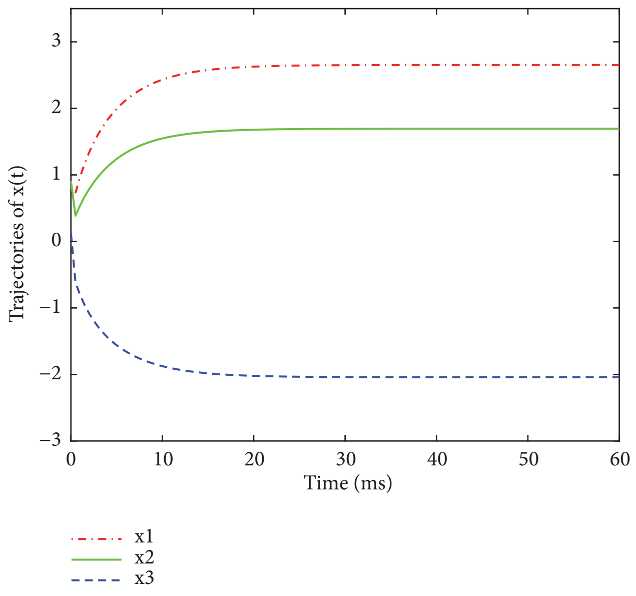


Figure 5: Transient behavior of the neural network with $\phi^p_{\text{D-FB}}$ function ($p = 3$) in Example 7.

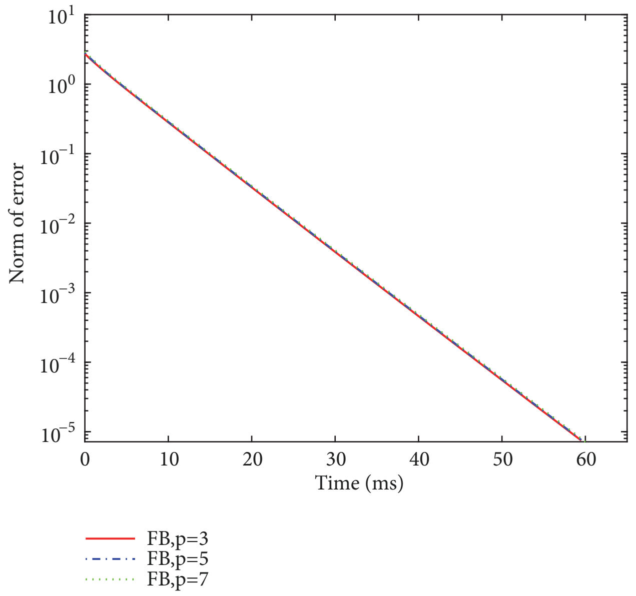


Figure 6: Convergence comparison of $\phi^p_{\text{D-FB}}$ function with different p value for Example 7.

4. Neural Networks for SOCCVI

This section is devoted to another type of SOC constrained problem, SOCCVI. Like what we have done for SOCQP, in this section, we first show how we build up the neural network model for SOCCVI and prove various stabilities for it accordingly. Then, numerical experiments are reported to demonstrate the effectiveness of the proposed neural network.

4.1. The Model and Stability Analysis. Let $\phi(x, y)$ be a SOC complementarity function like $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} defined as in (24) and (25), respectively. Mimicking the arguments described as in [42], we can verify that the KKT system

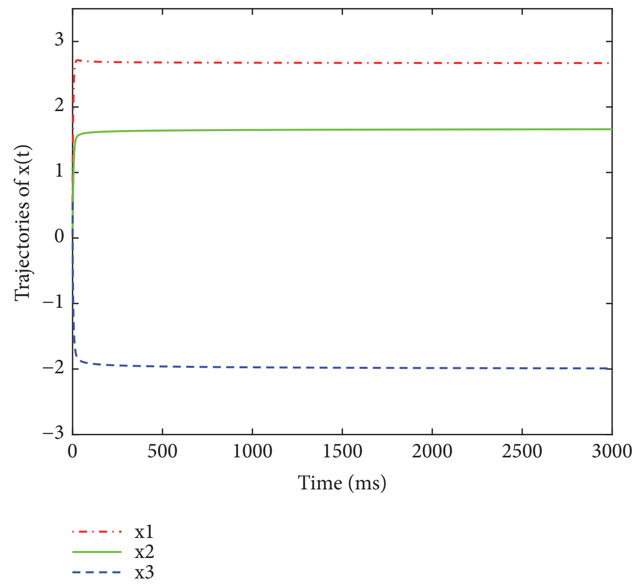


FIGURE 7: Transient behavior of the neural network with ϕ_{NR}^{p} function ($p = 3$) in Example 7.

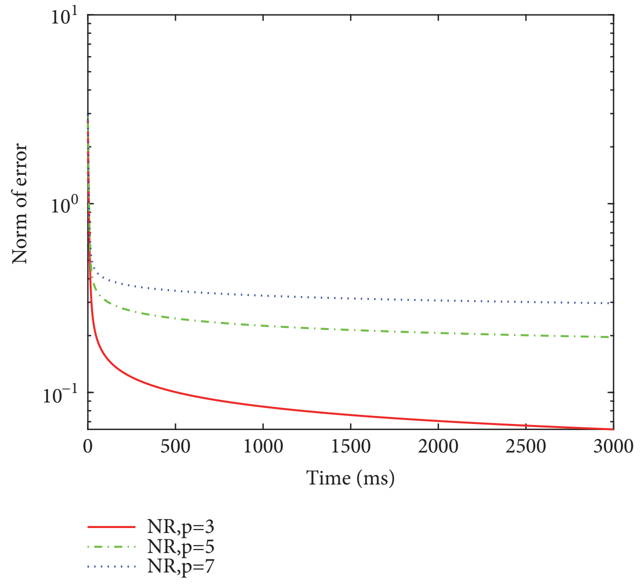


FIGURE 8: Convergence comparison of ϕ_{NR}^{p} function with different p value for Example 7.

(17) is equivalent to the following unconstrained smooth minimization problem:

$$\min \quad \Psi(z) := \frac{1}{2}\|S(z)\|^2, \tag{38}$$

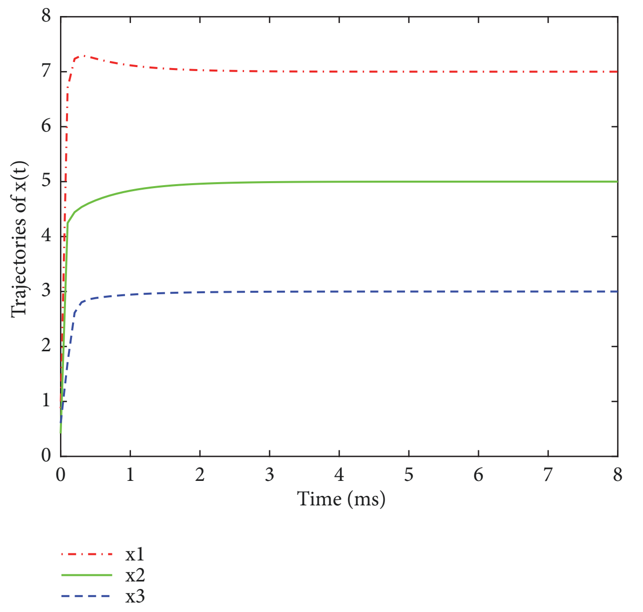


FIGURE 9: Transient behavior of the neural network with $\phi_{\mathrm{D-FB}}^{p}$ function ($p = 3$) in Example 8.

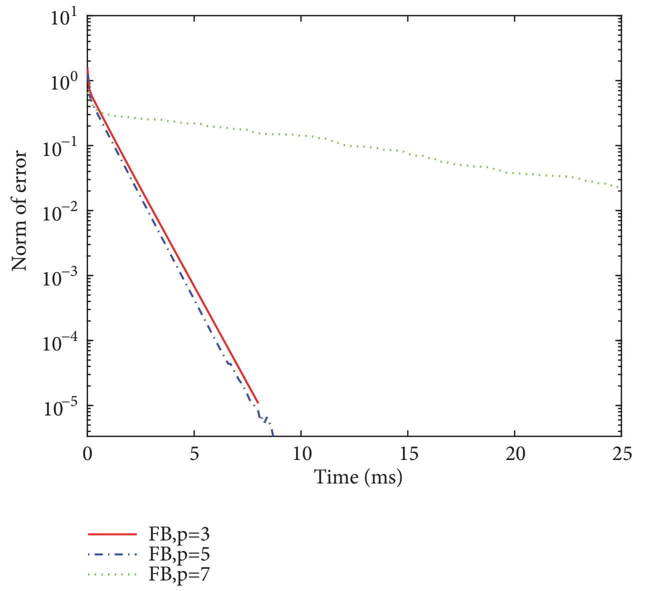


FIGURE 10: Convergence comparison of $\phi_{\mathrm{D-FB}}^{p}$ function with different p value for Example 8.

where $z = (x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ and $S(z)$ is given by

$$S(z) = \begin{bmatrix} L(x, \mu, \lambda) \\ -h(x) \\ \phi\left(-g_{m_1}(x), \lambda_{m_1}\right) \\ \vdots \\ \phi\left(-g_{m_q}(x), \lambda_{m_q}\right) \end{bmatrix}, \tag{39}$$

with $g_{m_i}(x), \lambda_{m_i} \in \mathbb{R}^{m_i}$. In other words, $\Psi(z)$ is a smooth merit function for the KKT system (17). Hence, based on the above smooth minimization problem (38), it is natural to propose a neural network for solving the SOCCVI as follows:

$$\begin{aligned} \frac{dz(t)}{dt} &= -\rho \nabla \Psi(z(t)), \\ z(t_0) &= z_0, \end{aligned} \tag{40}$$

where $\rho > 0$ is a scaling factor. To prove the stability of neural network (40), we need to present some properties of $\Psi(\cdot)$.

Proposition 9. *Let $\Psi : \mathbb{R}^{n+l+m} \longrightarrow \mathbb{R}_+$ be defined as in (38). Then, $\Psi(z) \geq 0$ for $z = (x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$. Moreover, $\Psi(z) = 0$ if and only if (x, μ, λ) solves the KKT system (17).*

Proof. The proof is straightforward. □

Proposition 10. *Let $\Psi : \mathbb{R}^{n+l+m} \longrightarrow \mathbb{R}_+$ be defined as in (38). Then, the following results hold.*

(a) *The function Ψ is continuously differentiable everywhere with*

$$\nabla \Psi(z) = \nabla S(z) S(z), \tag{41}$$

where

$$\nabla S(z) = \begin{bmatrix} \nabla_x L(x,\mu,\lambda)^T & -\nabla h(x) & -\nabla g(x) \operatorname{diag}\left\{\nabla_{g_{m_i}} \phi\left(-g_{m_i}(x), \lambda_{m_i}\right)\right\}_{i=1}^{q} \\ \nabla h(x)^T & 0 & 0 \\ \nabla g(x)^T & 0 & \operatorname{diag}\left\{\nabla_{\lambda_{m_i}} \phi\left(-g_{m_i}(x), \lambda_{m_i}\right)\right\}_{i=1}^{q} \end{bmatrix}. \tag{42}$$

(b) *If $\nabla S(z)$ is nonsingular, then for any stationary point $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ of Ψ, $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a KKT triple of the SOCCVI problem.*

(c) *$\Psi(z(t))$ is nonincreasing with respect to t.*

Proof. (a) It follows from the chain rule immediately.

(b) From $\nabla \Psi(z) = \nabla S(z) S(z)$ and the fact that matrix $\nabla S(z)$ is nonsingular, it is clear that $\nabla \Psi(z) = 0$ if and only if $S(z) = 0$. Hence, we see that $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a KKT triple of the SOCCVI problem provided $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a stationary point of Ψ.

(c) From the definition of $\Psi(z)$ and (40), it is easy to verify that

$$\begin{aligned} \frac{d\Psi(z(t))}{dt} &= \nabla \Psi(z(t))^T \frac{dz(t)}{dt} = -\rho \left\| \nabla \Psi(z(t)) \right\|^2 \\ &\leq 0, \end{aligned} \tag{43}$$

which says $\Psi(z(t))$ is a monotonically decreasing function with respect to t. □

Now, we are ready to analyze the behavior of the solution trajectory of neural network (40) and establish three kinds of stabilities for an isolated equilibrium point.

Proposition 11. *(a) If $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a KKT triple of the SOCCVI problem, then $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is an equilibrium point of neural network (40).*

(b) If $\nabla S(z)$ is nonsingular and $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is an equilibrium point of (40), then $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a KKT triple of the SOCCVI problem.

Proof. (a) From Proposition 9 and $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ being a KKT triple of SOCCVI problem, it is clear that $S(x, \mu, \lambda) = 0$, which implies $\nabla \Psi(x, \mu, \lambda) = 0$. Besides, by Proposition 10, we know that $\nabla \Psi(x, \mu, \lambda) \neq 0$. This shows that (x, μ, λ) is an equilibrium point of neural network (40).

(b) It follows from $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ being an equilibrium point of neural network (40) that $\nabla \Psi(x, \mu, \lambda) = 0$. In other words, (x, μ, λ) is the stationary point of Ψ. Then, the result is a direct consequence of Proposition 10(b). □

Proposition 12. *(a) For any initial state $z_0 = z(t_0)$, there exists exactly one maximal solution $z(t)$ with $t \in [t_0, \tau(x_0))$ for the neural network (40).*

(b) If the level set $\mathscr{L}(z_0) = \{z \in \mathbb{R}^{n+l+m} \mid \Psi(z) \leq \Psi(z_0)\}$ is bounded, then $\tau(x_0) = +\infty$.

Proof. (a) Since $S(\cdot)$ is continuous differentiable, it says that $\nabla S(\cdot)$ is continuous. This means $\nabla S(\cdot)$ is bounded on a local compact neighborhood of z, which implies that $\nabla \Psi(z)$ is locally Lipschitz continuous. Thus, applying Lemma 1 leads to the desired result.

(b) This proof is similar to the proof of Case(i) in [10, Proposition 4.2], so we omit it. □

Remark 13. A natural question arises here. When are the level sets

$$\mathscr{L}(\Psi, \gamma) := \left\{ z \in \mathbb{R}^{n+l+m} \mid \Psi(z) \leq \gamma \right\} \tag{44}$$

bounded for all $\gamma \in \mathbb{R}$? For the time being, we are not able to answer this question yet. We suspect that there needs more subtle properties of F, h, and g to finish it.

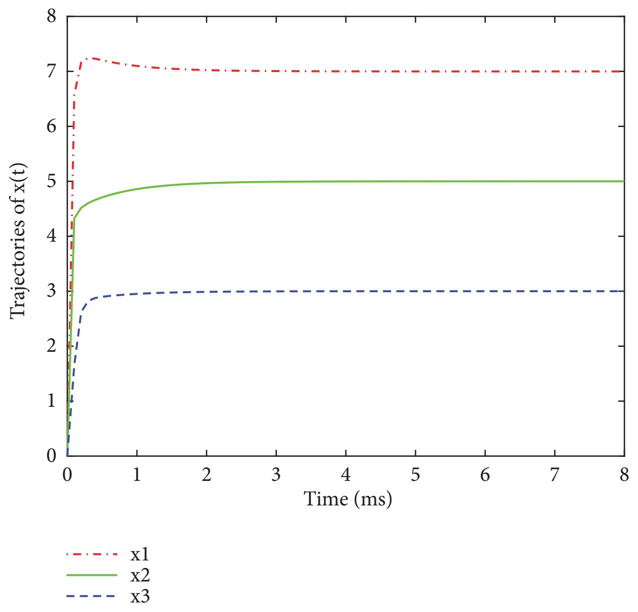


FIGURE 11: Transient behavior of the neural network with ϕ_{NR}^{p} function ($p = 3$) in Example 8.

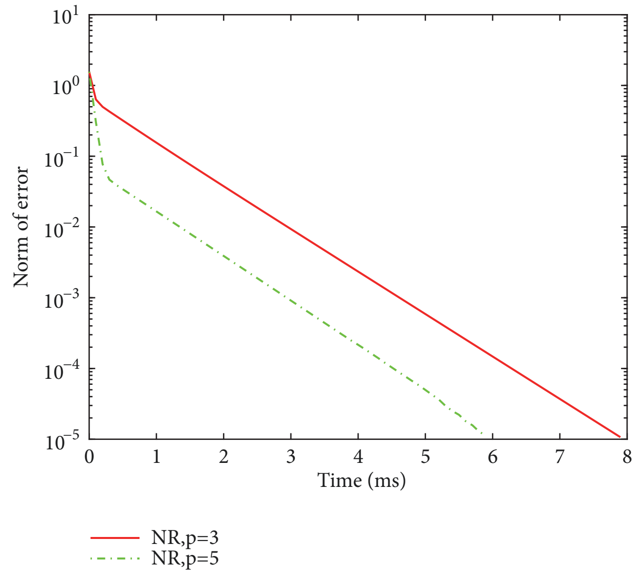


FIGURE 12: Convergence comparison of ϕ_{NR}^{p} function with different p value for Example 8.

Next, we investigate the convergence of the solution trajectory and stability of neural network (40), which are the main results of this section.

Theorem 14. *(a) Let $z(t)$ with $t \in [t_0, \tau(z_0))$ be the unique maximal solution to the neural network (40). If $\tau(z_0) = +\infty$ and $\{z(t)\}$ is bounded, then $\lim_{t\longrightarrow\infty} \nabla\Psi(z(t)) = 0$.*

(b) If $\nabla S(z)$ is nonsingular and $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is the accumulation point of the trajectory $z(t)$, then $(x, \mu, \lambda) \in \mathbb{R}^{n+l+m}$ is a KKT triple of the SOCCVI problem.

Proof. With Proposition 10(b) and 10(c) and Proposition 12, the arguments are exactly the same as those for [19, Corollary 4.3]. Thus, we omit them. □

Theorem 15. *Let z^* be an isolated equilibrium point of the neural network (40). Then, the following results hold.*

(a) *z^* is asymptotically stable and hence is also Lyapunov stable.*

(b) *If $\nabla S(z)$ is nonsingular, then it is exponentially stable.*

Proof. Again, the arguments are similar to those in [32, Theorem 3.1] and we omit them. □

To study the conditions for nonsingularity based on $\psi_{\mathrm{D-FB}}^{p}$ and ϕ_{NR}^{p}, we need the following assumptions.

Assumption 16. (a) The gradients $\{\nabla h_j(x) \mid j = 1, \dots, l\} \cup \{\nabla g_i(x) \mid i = 1, \dots, m\}$ are linear independent.

(b) $\nabla_x L(x, \mu, \lambda)$ is positive definite on the null space of the gradients $\{\nabla h_j(x) \mid j = 1, \dots, l\}$.

When SOCCVI problem corresponds to the KKT conditions of a convex second-order cone program (CSOCP) problem as (14) where both h and g are linear, the above Assumption 16(b) is indeed equivalent to the well-used condition of $\nabla^2 f(x)$ being positive definite, e.g., [22, Corollary 1].

Assumption 17. Let $\alpha := w_{m_i}^{p/2}$ and $\beta := v_{m_i}^{p/2}$, where $w_{m_i} = g_{m_i}^2 + \lambda_{m_i}^2$ and $v_{m_i} = (g_{m_i} + \lambda_{m_i})^2$. For $g_{m_i}(x), \lambda_{m_i} \in \mathcal{K}^{m_i}$, we have

(a) $L_{g_{m_i}}^2 - L_\beta L_\alpha^{-1} L_{g_{m_i}}^2 L_\alpha^{-1} L_\beta \succeq 0$ or $L_\alpha L_\beta^{-1} L_{g_{m_i}}^2 L_\beta^{-1} L_\alpha - L_{g_{m_i}}^2 \succeq 0$;

(b) $L^2_{\lambda_{m_i}} - L_\beta L_\alpha^{-1} L^2_{\lambda_{m_i}} L_\alpha^{-1} L_\beta \geq 0$ or $L_\alpha L_\beta^{-1} L^2_{\lambda_{m_i}} L_\beta^{-1} L_\alpha - L^2_{\lambda_{m_i}} \geq 0$.

Theorem 18. *Suppose $-g_{m_i} + \lambda_{m_i} \in \operatorname{int} \mathcal{K}^{m_i}$ for $i = 1, 2, \ldots, p$ and that Assumptions 16 and 17 hold. Then, the matrix*

$$\nabla S(z) = \begin{bmatrix} \nabla_x L(x,\mu,\lambda)^T & -\nabla h(x) & -\nabla g(x) \operatorname{diag}\left\{\nabla_{g_{m_i}} \psi^p_{\text{D-FB}}\left(-g_{m_i}(x), \lambda_{m_i}\right)\right\}_{i=1}^q \\ \nabla h(x)^T & 0 & 0 \\ \nabla g(x)^T & 0 & \operatorname{diag}\left\{\nabla_{\lambda_{m_i}} \psi^p_{\text{D-FB}}\left(-g_{m_i}(x), \lambda_{m_i}\right)\right\}_{i=1}^q \end{bmatrix} \quad (45)$$

is nonsingular.

Proof. We know that $\nabla S(z)$ is nonsingular if and only if the following equation only has zero solution:

$$\nabla S(z) \begin{bmatrix} u \\ v \\ t \end{bmatrix} = 0, \quad \text{where } (u,v,t) \in \mathbb{R}^n \times \mathbb{R}^l \times \mathbb{R}^m. \quad (46)$$

To reach the conclusion, we need to prove $u = 0, v = 0, t = 0$. First, plugging the components of $\nabla S(z)$ into (46), we have

$$(\nabla_x L)^T u - (\nabla h(x)) v - \nabla g(x) \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right) t = 0 \quad (47)$$

$$(\nabla h(x))^T u = 0 \quad (48)$$

$$(\nabla g(x))^T u + \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_\lambda \mathscr{L}_\alpha^{-1}\right) t = 0 \quad (49)$$

where

$$\begin{aligned}
\mathscr{L}_{g+\lambda} &= \operatorname{diag}\left\{L_{-g_{m_1}+\lambda_{m_1}}, L_{-g_{m_2}+\lambda_{m_2}}, \ldots, L_{-g_{m_q}+\lambda_{m_q}}\right\} \\
\mathscr{L}_\beta &= \operatorname{diag}\left\{\nabla g^{soc}(v_{m_1}), \nabla g^{soc}(v_{m_2}), \nabla g^{soc}(v_{m_q})\right\} \\
\mathscr{L}_{-g} &= \operatorname{diag}\left\{L_{-g_{m_1}}, L_{-g_{m_2}}, \ldots, L_{-g_{m_q}}\right\} \\
\mathscr{L}_\alpha &= \operatorname{diag}\left\{\nabla g^{soc}(w_{m_1}), \nabla g^{soc}(w_{m_2}), \nabla g^{soc}(w_{m_q})\right\} \\
\mathscr{L}_\lambda &= \operatorname{diag}\left\{L_{\lambda_{m_1}}, L_{\lambda_{m_2}}, \ldots, L_{\lambda_{m_q}}\right\} \\
\alpha &= \operatorname{diag}\left\{\alpha_{m_1}, \alpha_{m_2}, \ldots, \alpha_{m_q}\right\} \\
\alpha_{m_i} &= (v_{m_i})^{p/2} = \left(-g_{m_i} + \lambda_{m_i}\right)^p, \quad i = 1, 2, \ldots, q \\
\beta &= \operatorname{diag}\left\{\beta_{m_1}, \beta_{m_2}, \ldots, \beta_{m_q}\right\} \\
\beta_{m_i} &= (w_{m_i})^{p/2} = \left(-g^2_{m_i} + \lambda^2_{m_i}\right)^{p/2}, \quad i = 1, 2, \ldots, q
\end{aligned} \quad (50)$$

From (47) and (48), we see that

$$u^T (\nabla_x L)^T u - u^T \nabla g(x) \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right) t = 0, \quad (51)$$

while from (49), we have

$$t^T \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right)^T (\nabla g(x))^T u + t^T \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right)^T \cdot \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_\lambda \mathscr{L}_\alpha^{-1}\right) t = 0 \quad (52)$$

Next, we will claim that

$$t^T \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right)^T \cdot \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_\lambda \mathscr{L}_\alpha^{-1}\right) t \geq 0 \quad (53)$$

To see this, we note that

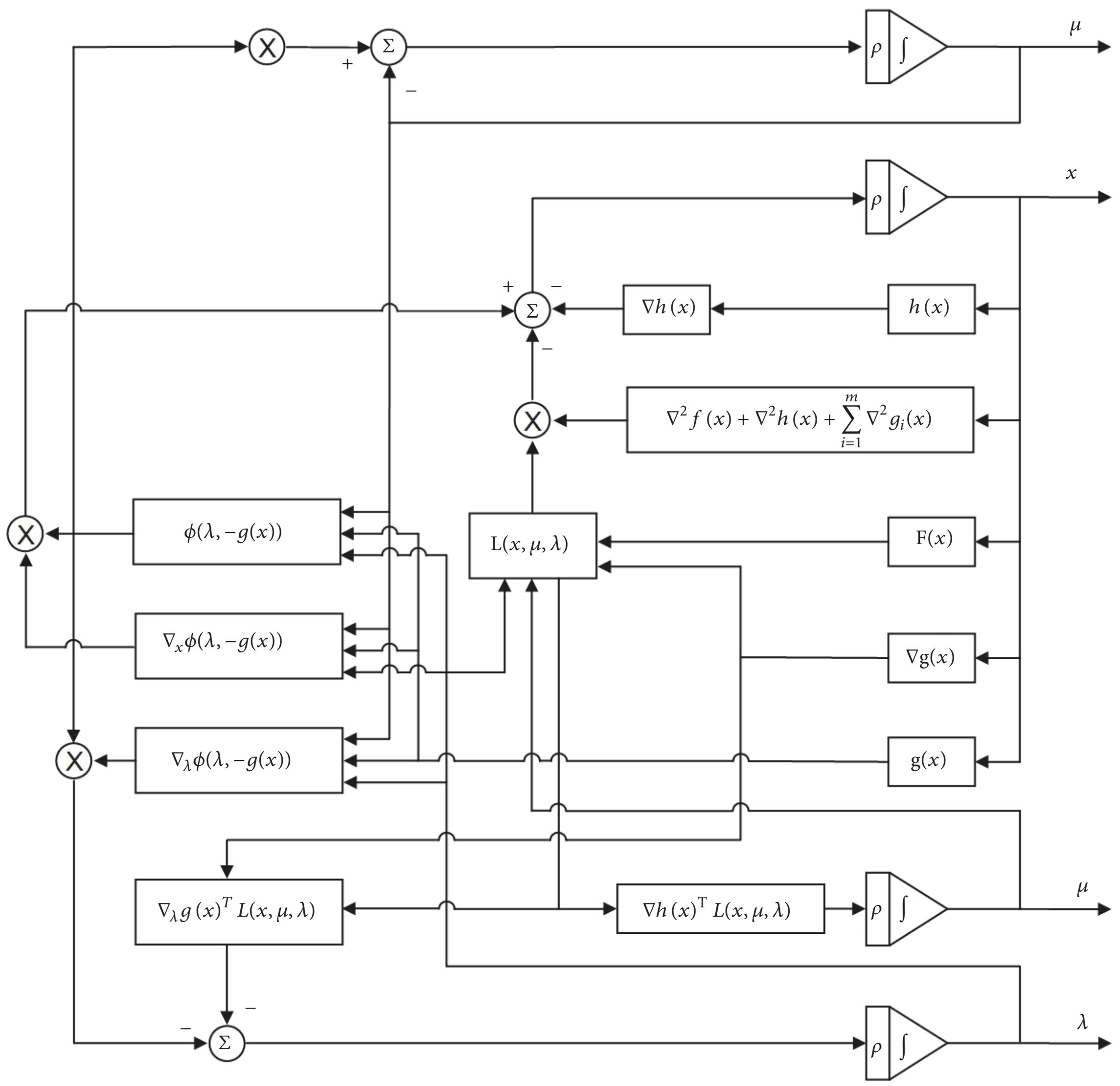
$$\begin{aligned}
&\left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_{-g} \mathscr{L}_\alpha^{-1}\right)^T \left(\mathscr{L}_{-g+\lambda} \mathscr{L}_\beta^{-1} - \mathscr{L}_\lambda \mathscr{L}_\alpha^{-1}\right) \\
&= \operatorname{diag}\left\{\left(L_{-g_{m_1}+\lambda_{m_1}} L^{-1}_{\beta_{m_1}} - L_{-g_{m_1}} L^{-1}_{\alpha_{m_1}}\right)^T \cdot \left(L_{-g_{m_1}+\lambda_{m_1}} L^{-1}_{\beta_{m_1}} - L_{\lambda_{m_1}} L^{-1}_{\alpha_{m_1}}\right), \ldots, \right. \\
&\left. \left(L_{-g_{m_q}+\lambda_{m_q}} L^{-1}_{\beta_{m_q}} - L_{-g_{m_q}} L^{-1}_{\alpha_{m_q}}\right)^T \cdot \left(L_{-g_{m_q}+\lambda_{m_q}} L^{-1}_{\beta_{m_q}} - L_{\lambda_{m_q}} L^{-1}_{\alpha_{m_q}}\right)\right\}.
\end{aligned} \quad (54)$$

In view of this, to prove inequality (53), it suffices to show that

$$t_i^T \left(L_{-g_{m_i}+\lambda_{m_i}} L^{-1}_{\beta_{m_i}} - L_{-g_{m_i}} L^{-1}_{\alpha_{m_i}}\right)^T \cdot \left(L_{-g_{m_i}+\lambda_{m_1}} L^{-1}_{\beta_{m_i}} - L_{\lambda_{m_i}} L^{-1}_{\alpha_{m_i}}\right) t_i \geq 0, \quad (55)$$

for $i = 1, 2, \ldots, q$. For convenience, we denote $X := -g_{m_i}$, $Y := \lambda_{m_i}$, $A := \alpha_{m_i}$, and $B := \beta_{m_i}$. With these notations, we have

$$\begin{aligned}
&\left(L_{-g_{m_i}+\lambda_{m_i}} L^{-1}_{\beta_{m_i}} - L_{-g_{m_i}} L^{-1}_{\alpha_{m_i}}\right)^T \cdot \left(L_{-g_{m_i}+\lambda_{m_1}} L^{-1}_{\beta_{m_i}} - L_{\lambda_{m_i}} L^{-1}_{\alpha_{m_i}}\right) \\
&= \left(L_{X+Y} L_B^{-1} - L_X L_A^{-1}\right)^T \left(L_{X+Y} L_B^{-1} - L_Y L_A^{-1}\right) \\
&= L_B^{-1} \left(L_{X+Y} - L_B L_A^{-1} L_X\right) \left(L_{X+Y} - L_Y L_A^{-1} L_B\right) L_B^{-1},
\end{aligned} \quad (56)$$



μ
ρ
$\int$
Σ
$+$
$-$
x
$\nabla h(x)$
$h(x)$
$\nabla^2 f(x) + \nabla^2 h(x) + \sum_{i=1}^{m} \nabla^2 g_i(x)$
$\phi(\lambda, -g(x))$
$\mathrm{L}(x, \mu, \lambda)$
$\mathrm{F}(x)$
$\nabla_x \phi(\lambda, -g(x))$
$\nabla \mathrm{g}(x)$
$\nabla_\lambda \phi(\lambda, -g(x))$
$\mathrm{g}(x)$
$\nabla_\lambda g(x)^T L(x, \mu, \lambda)$
$\nabla h(x)^{\mathrm{T}} L(x, \mu, \lambda)$
λ

FIGURE 13: Block diagram of the proposed neural network with ϕ (ϕ is $\phi^p_{\mathrm{D-FB}}$ or ϕ^p_{NR}).

which says that it is enough to show $M := (L_{X+Y} - L_B L_A^{-1} L_X)(L_{X+Y} - L_Y L_A^{-1} L_B)$ is semipositive definite in order to prove inequality (55). To this end, we compute that

$$\begin{aligned}
&\frac{1}{2}\left[\left(L_{X+Y} - L_B L_A^{-1} L_X\right)\left(L_{X+Y} - L_Y L_A^{-1} L_B\right)\right. \\
&\quad \left. + \left(\left(L_{X+Y} - L_B L_A^{-1} L_X\right)\left(L_{X+Y} - L_Y L_A^{-1} L_B\right)\right)^T\right] \\
&= \frac{1}{2}\left[2L^2_{X+Y} - L^2_{X+Y} L_A^{-1} L_B - L_B L_A^{-1} L^2_{X+Y}\right. \\
&\quad \left. + L_B L_A^{-1}\left(L_X L_Y + L_Y L_X\right) L_A^{-1} L_B\right] \\
&= \frac{1}{2}\left[\left(I - L_B L_A^{-1}\right) L^2_{X+Y}\left(I - L_A^{-1} L_B\right) + L^2_{X+Y}\right. \\
&\quad - L_B L_A^{-1} L^2_{X+Y} L_A^{-1} L_B \\
&\quad \left. + L_B L_A^{-1}\left(L_X L_Y + L_Y L_X\right) L_A^{-1} L_B\right] \\
&= \frac{1}{2}\left[\left(I - L_B L_A^{-1}\right) L^2_{X+Y}\left(I - L_A^{-1} L_B\right) + L^2_{X+Y}\right. \\
&\quad \left. - L_B L_A^{-1}\left(L^2_X + L^2_Y\right) L_A^{-1} L_B\right] \\
&= \frac{1}{2}\left[\left(I - L_B L_A^{-1}\right) L^2_{X+Y}\left(I - L_A^{-1} L_B\right)\right. \\
&\quad + \left(L^2_X - L_B L_A^{-1}\left(L^2_X\right) L_A^{-1} L_B\right) \\
&\quad \left. + \left(L^2_Y - L_B L_A^{-1}\left(L^2_Y\right) L_A^{-1} L_B\right) + \left(L_X L_Y + L_Y L_X\right)\right].
\end{aligned} \tag{57}$$

It can be verified that L^2_{X+Y} and $L_X L_Y + L_Y L_X$ are positive semidefinite. Then, from Assumption 17 and (57), we conclude that $M := (L_{X+Y} - L_B L_A^{-1} L_X)(L_{X+Y} - L_Y L_A^{-1} L_B)$ is semipositive definite; and hence inequality (55) holds. Thus, inequality (53) also holds accordingly. Now, tt follows from (51), (52), and (53) that $u^T(\nabla_x L)^T u = 0$ which implies that $u = 0$. Then, (47) and (48) become

$$\nabla h(x)v + \nabla g(x)\left(\mathscr{L}_{-g+\lambda}\mathscr{L}_\beta^{-1} - \mathscr{L}_{-g}\mathscr{L}_\alpha^{-1}\right)t = 0 \qquad (58)$$

$$\left(\mathscr{L}_{-g+\lambda}\mathscr{L}_\beta^{-1} - \mathscr{L}_{\lambda}\mathscr{L}_\alpha^{-1}\right)t = 0 \qquad (59)$$

In light of Assumption 16(a) and (58), we know

$$\begin{gathered} v = 0, \\ \left(\mathscr{L}_{-g+\lambda}\mathscr{L}_\beta^{-1} - \mathscr{L}_{-g}\mathscr{L}_\alpha^{-1}\right)t = 0. \end{gathered} \qquad (60)$$

Combining (59) and (60) together, it is clear to obtain

$$\mathscr{L}_{-g}\mathscr{L}_\alpha^{-1}t = \mathscr{L}_{\lambda}\mathscr{L}_\alpha^{-1}t \qquad (61)$$

Note that $-g$ and λ are strict complementary. Hence, it yields $t = 0$. In summary, from equation (46), we deduce $u = v = t = 0$, which says the matrix $\nabla S(z)$ is nonsingular. □

Theorem 19. *Suppose Assumption 16 holds and*

$$\begin{gathered} \nabla_{g_{m_i}}\phi_{\mathrm{NR}}^p\left(-g_{m_i}(x), \lambda_{m_i}\right) \cdot \nabla_{\lambda_{m_i}}\phi_{\mathrm{NR}}^p\left(-g_{m_i}(x), \lambda_{m_i}\right) \\ \succeq 0. \end{gathered} \qquad (62)$$

Then, the matrix

$$\nabla S(z) = \begin{bmatrix} \nabla_x L(x,\mu,\lambda)^T & -\nabla h(x) & -\nabla g(x)\,\mathrm{diag}\left\{\nabla_{g_{m_i}}\phi_{\mathrm{NR}}^p\left(-g_{m_i}(x),\lambda_{m_i}\right)\right\}_{i=1}^q \\ \nabla h(x)^T & 0 & 0 \\ \nabla g(x)^T & 0 & \mathrm{diag}\left\{\nabla_{\lambda_{m_i}}\phi_{\mathrm{NR}}^p\left(-g_{m_i}(x),\lambda_{m_i}\right)\right\}_{i=1}^q \end{bmatrix} \qquad (63)$$

is nonsingular.

Proof. The proof can be done by following the similar arguments as in Theorem 18. □

To close this subsection, we say a few words about the complexity of the proposed neural network. Since SOCQP can be transformed into an SOCCVI problem, we only take SOCCVI as an example to illustrate the complexity of the proposed neural network model. In light of the main ideas for constructing neural network (see [25] 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, we employ the neural network for solving the KKT system (17) of SOCCVI with the differential equation (40), where $\rho > 0$ is a time scaling factor. In fact, if $\tau = \rho t$, then $dz(t)/dt = \rho(dz(\tau)/d\tau)$. Hence, it follows from (40) that $dz(\tau)/d\tau = -\nabla\Psi(u)$. In view of this, for simplicity and convenience, we set $\rho = 1$ in this paper. Indeed, the dynamic system (40) can be realized by an architecture with the two discrete-type classes of SOC complementarity functions $\phi_{\mathrm{D-FB}}^p$ and ϕ_{NR}^p shown in Figure 13. 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 [25]. The circuit for (40) requires $n + l + m$ integrators, n processors for $F(x)$, m processors for $g(x)$, mn processors for $\nabla g(x)$, ln processors for $\nabla h(x)$, $(l^2 + m^2 + m + l)n$ processors for $\nabla_x L(x,\mu,\lambda)$, 1 processor for $\phi_{\mathrm{D-FB}}^p$ and ϕ_{NR}^p, n processors for $\nabla_x\phi_{\mathrm{D-FB}}^p$ and $\nabla_x\phi_{\mathrm{NR}}^p$, m processors for $\nabla_\lambda\phi_{\mathrm{D-FB}}^p$ and $\nabla_\lambda\phi_{\mathrm{NR}}^p$, $n^2 + 4ln + 3mn + m^2 + m$ connection weights, and some summers.

4.2. Numerical Experiments. In this subsection, to demonstrate effectiveness of the proposed neural networks, some illustrative SOCCVI problems are tested. The numerical implementation is coded by Matlab 2014b and the ordinary differential equation solver adopted is *ode23*, which uses Runge-Kutta (2; 3) formula. In the subsequent tests, the parameter ρ in neural networks of Example 22 is set to be 10 and others are set to be 1.

Example 20. Consider the SOCCVI problem where

$$\begin{gathered} F(x) = \left[2x_1 - 4, e^{x_1} - 1, 2x_3 - 4, -\sin(x_4)\right]^T, \\ C = \left\{x \in \mathbb{R}^5 \mid -g(x) = x \in \mathscr{K}^5\right\}. \end{gathered} \qquad (64)$$

This problem has an approximate solution $x^* = [2, 0, 1.3333, 0, 0]^T$. Figures 14 and 16 show the transient behaviors of Example 20 for neural network model (40) based on $\phi_{\mathrm{D-FB}}^p$ and ϕ_{NR}^p with initial states $x_0 = [0,0,0,0,0]^T$, respectively. Figure 15 depicts the convergence comparison of the neural network model using $\phi_{\mathrm{D-FB}}^p$ function with different values of $p = 3, 5, 7$. From this, we see that the performance of $p = 3$ is significantly better than in other cases. Figure 17 depicts the influence of the parameter p on the value of norm of error for neural network model using $\phi_{\mathrm{NR}}^{\mathrm{p}}$ function. Again, when $p = 3$, the neural network based on ϕ_{NR}^p function produces fast decrease of norm of error.

Example 21. Consider the problem where

$$\begin{gathered} \min_{x\in C} f(x) = 2(x_1 - 3)^2 + \sin(x_1 - 3)\sin x_2 + 2x_2^2 \\ + x_3^2, \\ C = \left\{x \in \mathbb{R}^5 \mid -g(x) = x \in \mathscr{K}^5\right\}. \end{gathered} \qquad (65)$$

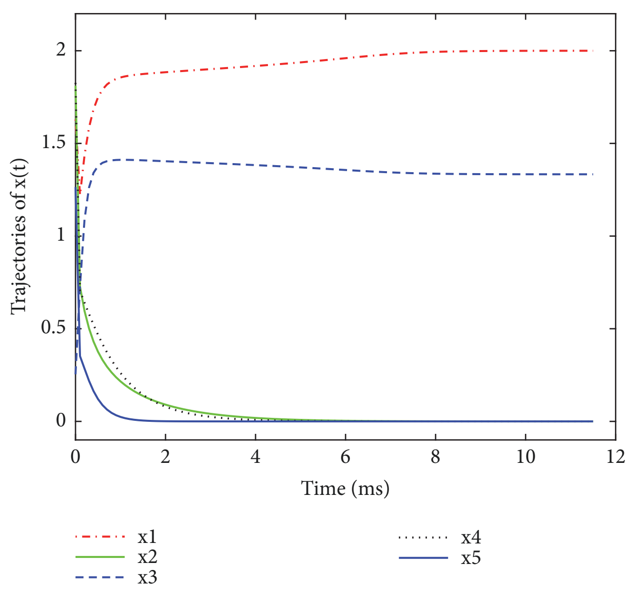


FIGURE 14: Transient behavior of neural network with $\phi^p_{\mathrm{D-FB}}$ function ($p = 3$) in Example 20.

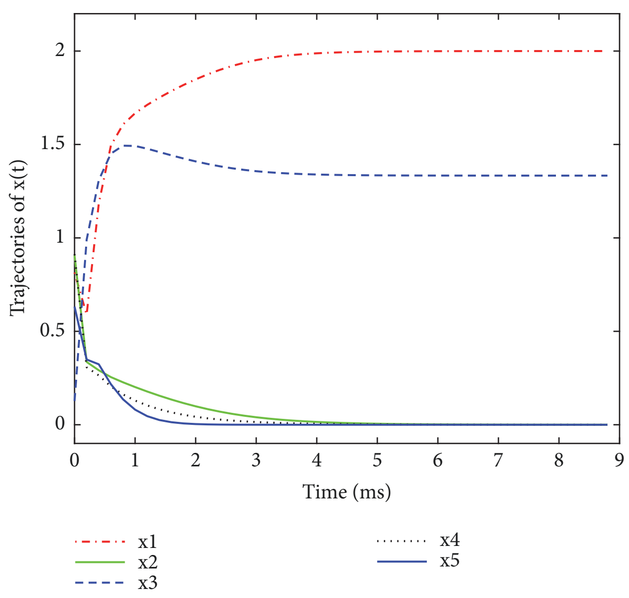


FIGURE 16: Transient behavior of the neural network with ϕ^p_{NR} function ($p = 3$) in Example 20.

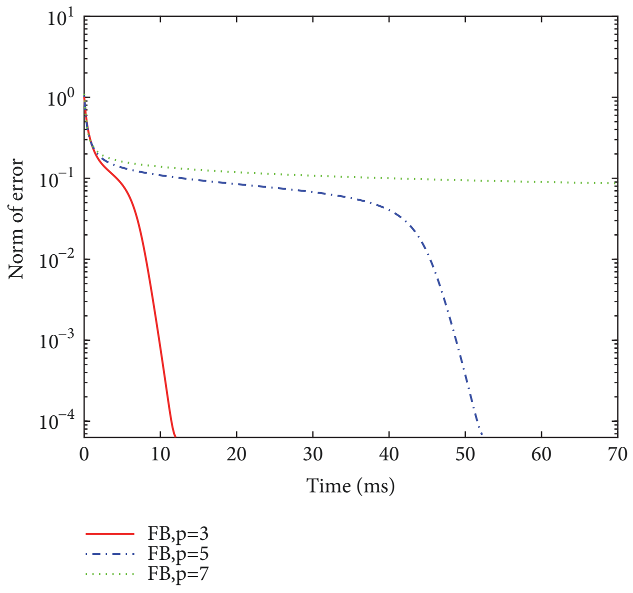


FIGURE 15: Convergence comparison of $\phi^p_{\mathrm{D-FB}}$ function with different p value for Example 20.

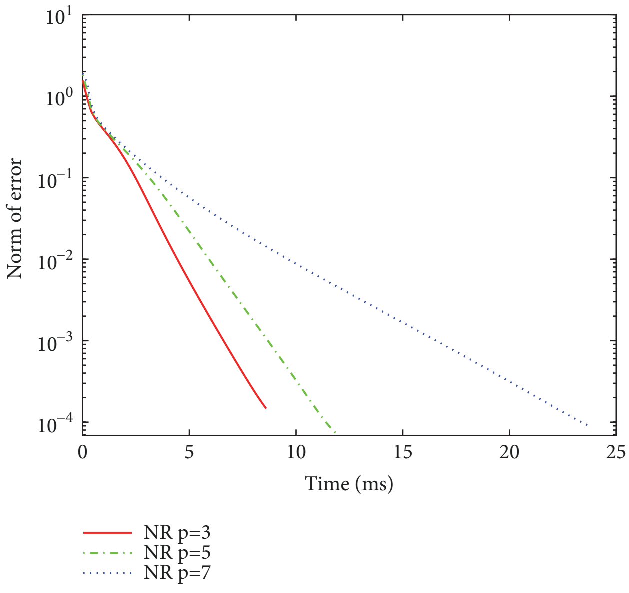


FIGURE 17: Convergence comparison of ϕ^p_{NR} function with different p value for Example 20.

This problem has an approximate solution $x^* = [3, 0, 0]^T$. In light of the analysis of previous section, this CSOCP in Example 21 can be transformed into an SOCCVI. We use the proposed neural networks to solve the problem with the trajectory obtained by it shown in Figures 18–21.

Figures 18 and 20 show the transient behaviors of Example 21 for neural network model (40) based on $\phi^p_{\mathrm{D-FB}}$ and ϕ^p_{NR} with initial states $x_0 = [0, 0, 0, 0, 0]^T$, respectively. From Figure 19, we see the convergence comparison of the neural network model using $\phi^p_{\mathrm{D-FB}}$ function with $p = 3, 5, 7$. There is no significant difference when perturbing the values of p. Figure 21 depicts the influence of the parameter p on the value of norm of error for neural network model using ϕ^p_{NR} function.

Example 22. We consider the following SOCCVI problem:

$$\langle Dx, y - x \rangle \geq 0, \quad \forall y \in C \tag{66}$$

where

$$C = \left\{ x \in \mathbb{R}^{10} \mid Ax - a = 0,\ Bx - b \preceq 0 \right\}, \tag{67}$$

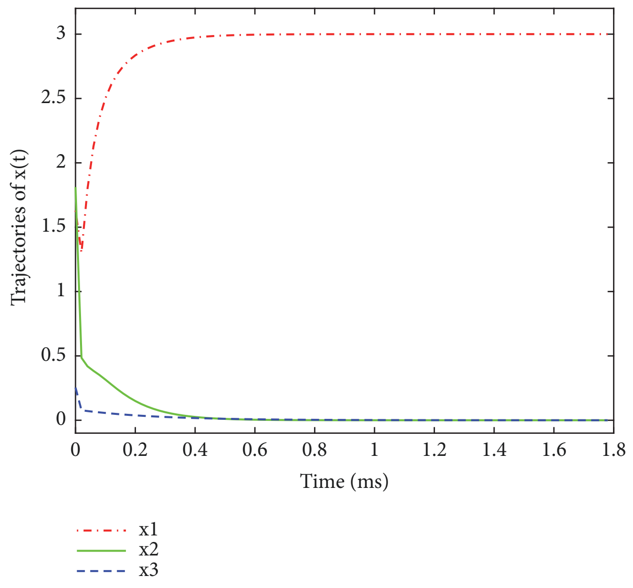


FIGURE 18: Transient behavior of neural network with $\phi^p_{\text{D-FB}}$ function ($p = 3$) in Example 21.

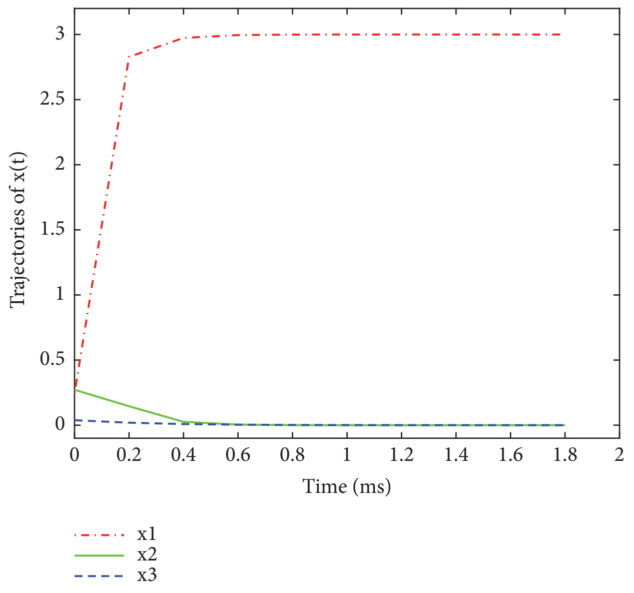


FIGURE 20: Transient behavior of the neural network with ϕ^p_{NR} function ($p = 3$) in Example 21.

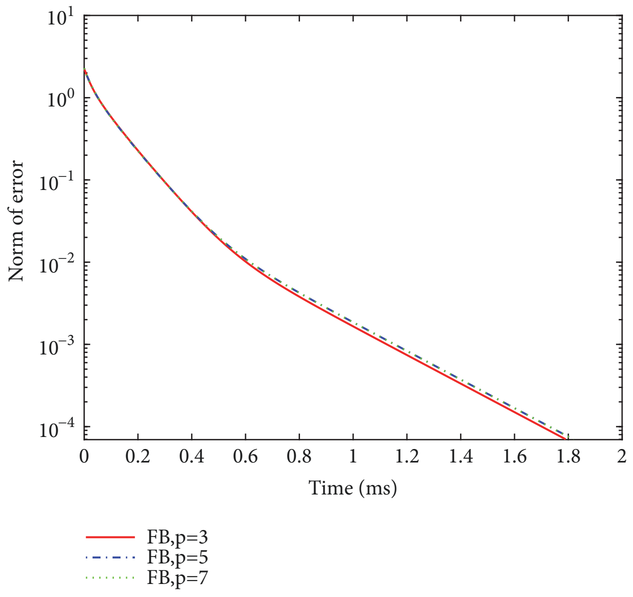


FIGURE 19: Convergence comparison of $\phi^p_{\text{D-FB}}$ function with different p value for Example 21.

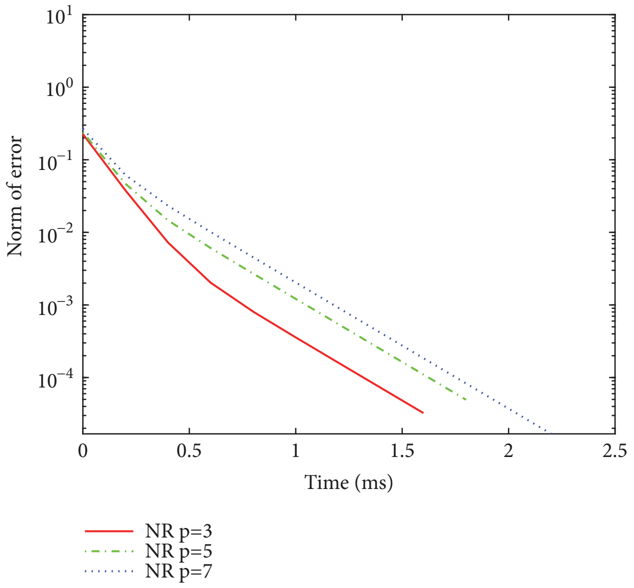


FIGURE 21: Convergence comparison of ϕ^p_{NR} functions with different p value for Example 21.

D is an 10×10 symmetric matrix and

$$D = (D_{ij})_{10\times10}, \quad \text{where } D_{ij} = \begin{cases} 2, & i = j \\ 1, & |i - j| = 1 \\ 0, & otherwise \end{cases}, \qquad (68)$$

$A = [I_{5\times5} \;\; 0_{5\times5}]_{5\times5}$, $B = [0_{5\times5} \;\; I_{5\times5}]_{5\times10}$, $a = 0_{5\times5}$, $b = [1, 0, 1, 0, 0]^T \in \mathcal{K}^2 \times \mathcal{K}^3$. Clearly, A and B are full row rank and $\operatorname{rank}([A^T \;\; B^T]) = 10$.

The problem has an solution $x^* = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$. It can be verified that the Lagrangian function for this example is

$$L(x, \mu, \lambda) = Dx + A^T\mu + B^T\lambda. \qquad (69)$$

Note that $\nabla_x L(x, \mu, \lambda)$ is positive definite.

Figures 22 and 24 show the transient behaviors of Example 22 for neural network model (40) based on $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} with initial states $x_0 = [0, 0, 0, 0, 0]^T$, respectively. Figures 23 and 25 depict the influence of the parameter p on the value of norm of error for neural network model based on $\phi^p_{\text{D-FB}}$

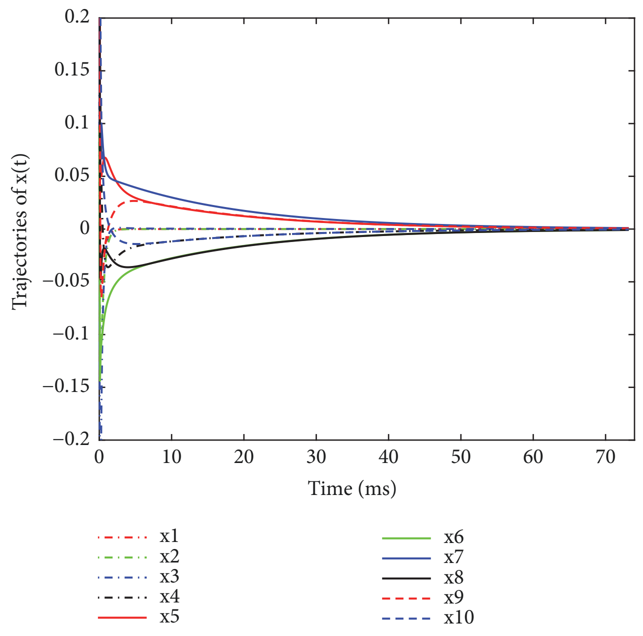


FIGURE 22: Transient behavior of neural network with $\phi^p_{\text{D-FB}}$ function ($p = 3$) in Example 22.

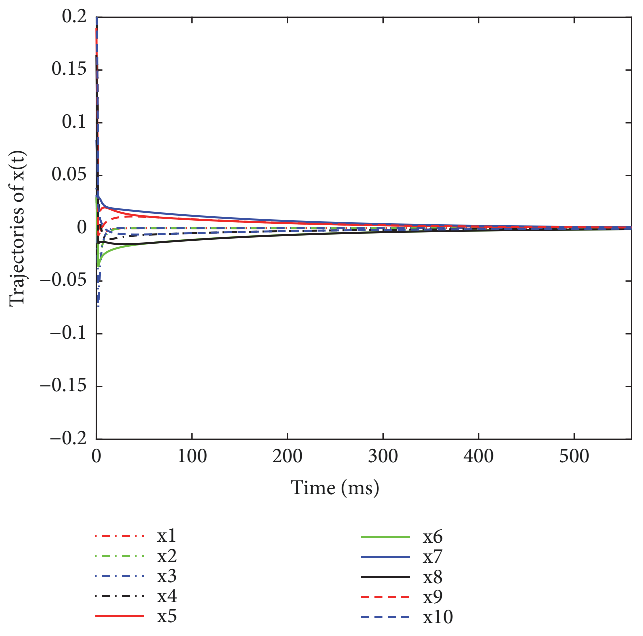


FIGURE 24: Transient behavior of the neural network with ϕ^p_{NR} function ($p = 3$) in Example 22.

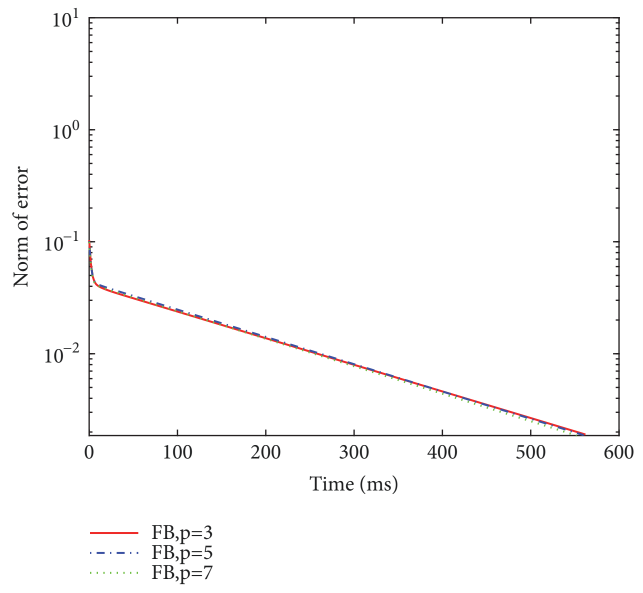


FIGURE 23: Convergence comparison of $\phi^p_{\text{D-FB}}$ functions with different p value for Example 22.

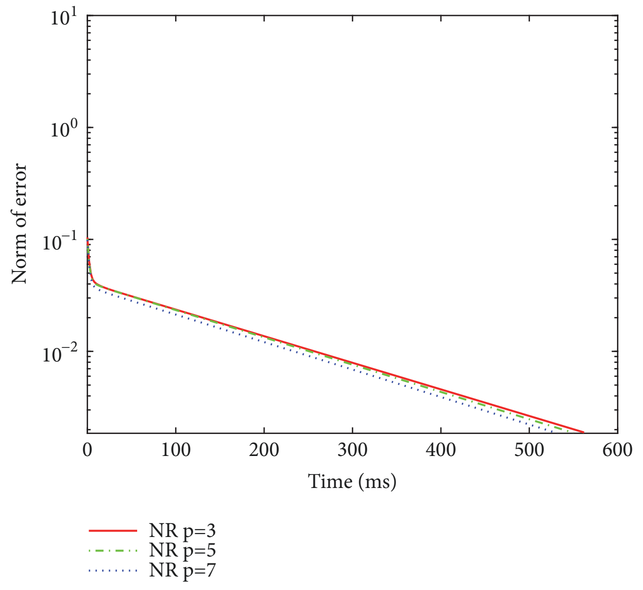


FIGURE 25: Convergence comparison of ϕ^p_{NR} functions with different p value for Example 22.

and ϕ^p_{NR} functions. We see that the influence of perturbing parameter p is tiny.

We summarize some observations based on the above experiments. First, we provide the simulation diagrams of convergence comparison of $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} functions with different p value for 6 examples. It can be seem from the Figures 2, 4, 6, 8, 10, 12, 15, 17, and 21 that the convergence speed is the fastest when $p = 3$. Under the same calculation time, the norm error in the case of $p = 3$ is generally smaller than that in the cases of $p = 5$ and $p = 7$. But from Figures 19, 23, and 25, we find that the convergence of case $p = 3$, $p = 5$, and $p = 7$ is similar. In other words, we observe that, for our examples, $p = 3$ is basically the best choice. However, the theoretical evidence is not clear yet, which will be left for future study.

5. Conclusions

In this paper, we propose a unified neural network model for solving two types of second-order cone optimization problems. We implement the neural network model with two classes of SOC complementarity functions $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR}, which are freshly discovered. Overall, when p is larger, the performance of the neural network based on $\phi^p_{\text{D-FB}}$ and ϕ^p_{NR} gets poorer. This suggests that the parameter $p = 3$ is the best choice to work with the neural network no matter $\phi^p_{\text{D-FB}}$ or ϕ^p_{NR} is employed. On the other hand, we observe that the neural network using the family of $\phi^p_{\text{D-FB}}$ functions performs better than the one using ϕ^p_{NR}, in general. This is a very interesting phenomenon and new discovery, which is different from the observation in [31]. Note that in [31], for standard SOCP (second-order cone program), 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). Our numerical experiments show that for SOCQP and SOCCVI, the family of $\phi^p_{\text{D-FB}}$ (a variant of FB function) may be considered to be employed in the neural network approach.

Another point that we want to clarify is the significance of this paper. By using two new SOC complementarity functions, this paper can be viewed as a follow-up of our previous works (see [31, 32]). From the natural feature of neural network, we pay more attention to the convergence path. This kind of convergence path varies with different SOC complementarity functions. In engineering, an important application of neural network is the design of robot arm. Therefore, compared with previous works, we achieve different convergence paths and the convergence paths are very good. In addition, we provide more detailed graphs regarding error analysis in this paper.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Juhe Sun's work is supported by National Natural Science Foundation of China (Grant no. 11301348). B. Saheya's work is supported by Natural Science Foundation of Inner Mongolia (Award no. 2017MS0125) and research fund of IMNU (Award no. 2017YJRC003). Jein-Shan Chen's work is supported by Ministry of Science and Technology, Taiwan.

References

[1] J.-S. Chen, "The semismooth-related properties of a merit function and a descent method for the nonlinear complementarity problem," *Journal of Global Optimization*, vol. 36, no. 4, pp. 565–580, 2006.

[2] J. Chen, H. Gao, and S. Pan, "A derivative-free *R*-linear convergent algorithm based on the generalized Fischer-Burmeister merit function," *Journal of Computational and Applied Mathematics*, vol. 232, no. 2, pp. 455–471, 2009.

[3] J.-S. Chen and S.-H. Pan, "A family of NCP functions and a descent method for the nonlinear complementarity problem," *Computational Optimization and Applications*, vol. 40, no. 3, pp. 389–404, 2008.

[4] X. Chen, L. Qi, and D. Sun, "Global and superlinear convergence of the smoothing Newton method and its application to general box constrained variational inequalities," *Mathematics of Computation*, vol. 67, no. 222, pp. 519–540, 1998.

[5] F. Facchinei and J.-S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer, New York, NY, USA, 2003.

[6] S. J. Wright, "An infeasible-interior-point algorithm for linear complementarity problems," *Mathematical Programming*, vol. 67, no. 1, pp. 29–51, 1994.

[7] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, no. 3, pp. 141–152, 1985.

[8] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 33, no. 5, pp. 533–541, 1986.

[9] Y.-L. Chang, J.-S. Chen, and C.-Y. Yang, "Symmetrization of generalized natural residual function for NCP," *Operations Research Letters*, vol. 43, no. 4, pp. 354–358, 2015.

[10] J.-S. Chen, C.-H. Ko, and S. Pan, "A neural network based on the generalized Fischer-Burmeister function for nonlinear complementarity problems," *Information Sciences*, vol. 180, no. 5, pp. 697–711, 2010.

[11] C. Dang, Y. Leung, X.-B. Gao, and K.-Z. Chen, "Neural networks for nonlinear and mixed complementarity problems and their applications," *Neural Networks*, vol. 17, no. 2, pp. 271–283, 2004.

[12] S. Effati, A. Ghomashi, and A. R. Nazemi, "Application of projection neural network in solving convex programming problems," *Applied Mathematics and Computation*, vol. 188, no. 2, pp. 1103–1114, 2007.

[13] S. Effati and A. R. Nazemi, "Neural network models and its application for solving linear and quadratic programming problems," *Applied Mathematics and Computation*, vol. 172, no. 1, pp. 305–331, 2006.

[14] Q. Han, L.-Z. Liao, H. Qi, and L. Qi, "Stability analysis of gradient-based neural networks for optimization problems," *Journal of Global Optimization*, vol. 19, no. 4, pp. 363–381, 2001.

[15] X. Hu and J. Wang, "A recurrent neural network for solving nonlinear convex programs subject to linear constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 16, no. 2, pp. 379–386, 2005.

[16] X. Hu and J. Wang, "Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 17, no. 6, pp. 1487–1499, 2006.

[17] X. Hu and J. Wang, "A recurrent neural network for solving a class of general variational inequalities," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 3, pp. 528–539, 2007.

[18] M. P. Kennedy and L. O. Chua, "Neural networks for nonlinear programming," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 35, no. 5, pp. 554–562, 1988.

[19] L.-Z. Liao, H. Qi, and L. Qi, "Solving nonlinear complementarity problems with neural networks: a reformulation method approach," *Journal of Computational and Applied Mathematics*, vol. 131, pp. 342–359, 2001.

[20] Y. Xia, H. Leung, and J. Wang, "A projection neural network and its application to constrained optimization problems," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 4, pp. 447–458, 2002.

[21] Y. Xia and J. Wang, "A general projection neural network for solving monotone variational inequalities and related optimization problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 15, no. 2, pp. 318–328, 2004.

[22] Y. Xia and J. Wang, "A recurrent neural network for solving nonlinear convex programs subject to linear constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 16, no. 2, pp. 379–386, 2005.

[23] M. Yashtini and A. Malek, "Solving complementarity and variational inequalities problems using neural networks," *Applied Mathematics and Computation*, vol. 190, no. 1, pp. 216–230, 2007.

[24] S. H. Zak and V. Upatising, "Solving linear programming problems with neural networks: a comparative study," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 6, no. 1, pp. 94–104, 1995.

[25] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*, John Wiley, New York, NY, USA, 1993.

[26] Y. Xia and J. Wang, "Robust regression estimation based on low-dimensional recurrent neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 5935–5946, 2018.

[27] S. Zhang and Y. Xia, "Solving nonlinear optimization problems of real functions in complex variables by complex-valued iterative methods," *IEEE Transactions on Cybernetics*, vol. 48, no. 1, pp. 277–287, 2018.

[28] Y. Xia and J. Wang, "A bi-projection neural network for solving constrained quadratic optimization problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 214–224, 2016.

[29] Y. Xia, H. Leung, and M. S. Kamel, "A discrete-time learning algorithm for image restoration using a novel L2-norm noise constrained estimation," *Neurocomputing*, vol. 198, pp. 155–170, 2016.

[30] S. Zhang, Y. Xia, and J. Wang, "A complex-valued projection neural network for constrained optimization of real functions in complex variables," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3227–3238, 2015.

[31] C.-H. Ko, J.-S. Chen, and C.-Y. Yang, "Recurrent neural networks for solving second-order cone programs," *Neurocomputing*, vol. 74, no. 17, pp. 3646–3653, 2011.

[32] J. Sun, J.-S. Chen, and C.-H. Ko, "Neural networks for solving second-order cone constrained variational inequality problem," *Computational optimization and applications*, vol. 51, no. 2, pp. 623–648, 2012.

[33] X. Miao, J.-S. Chen, and C.-H. Ko, "A smoothed NR neural network for solving nonlinear convex programs with second-order cone constraints," *Information Sciences*, vol. 268, pp. 255–270, 2014.

[34] V. Jeyakumar and H. Wolkowicz, "Generalizations of Slater's constraint qualification for infinite convex programs," *Mathematical Programming*, vol. 57, no. 1, pp. 85–101, 1992.

[35] R. K. Miller and A. N. Michel, *Ordinary Differential Equations*, Academic Press, 1982.

[36] J.-S. Chen and S. Pan, "A survey on SOC complementarity functions and solution methods for SOCPS and SOCCPS," *Pacific Journal of Optimization. An International Journal*, vol. 8, no. 1, pp. 33–74, 2012.

[37] J.-S. Chen and P. Tseng, "An unconstrained smooth minimization reformulation of the second-order cone complementarity problem," *Mathematical Programming*, vol. 104, no. 2-3, Ser. B, pp. 293–327, 2005.

[38] M. Fukushima, Z.-Q. Luo, and P. Tseng, "Smoothing functions for second-order-cone complementarity problems," *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 436–460, 2002.

[39] S. Hayashi, N. Yamashita, and M. Fukushima, "A combined smoothing and regularization method for monotone second-order cone complementarity problems," *SIAM Journal on Optimization*, vol. 15, no. 2, pp. 593–615, 2005.

[40] S. Pan and J.-S. Chen, "A semismooth Newton method for the SOCCP based on a one-parametric class of SOC complementarity functions," *Computational optimization and applications*, vol. 45, no. 1, pp. 59–88, 2010.

[41] P.-F. Ma, J.-S. Chen, C.-H. Huang, and C.-H. Ko, "Discovery of new complementarity functions for NCP and SOCCP," *Computational & Applied Mathematics*, vol. 37, no. 5, pp. 5727–5749, 2018.

[42] J. Sun and L. Zhang, "A globally convergent method based on Fischer-Burmeister operators for solving second-order cone constrained variational inequality problems," *Computers & Mathematics with Applications. An International Journal*, vol. 58, no. 10, pp. 1936–1946, 2009.