Journal of Computational and Applied Mathematics 407 (2022) 114092

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

A new class of neural networks for NCPs using smooth perturbations of the natural residual function

Jan Harold Alcantara, Jein-Shan Chen [*,1]

Department of Mathematics, National Taiwan Normal University, Taiwan

ARTICLE INFO

Article history:
Received 10 February 2020
Received in revised form 4 January 2022

MSC:
37-N40
65-K10
65-K15

Keywords:
Complementarity problem
Smoothing method
Neural network
Stability

ABSTRACT

We present a new class of neural networks for solving nonlinear complementarity problems (NCPs) based on some family of real-valued functions (denoted by $\mathscr{F}$) that can be used to construct smooth perturbations of the level curve defined by $\Phi_{\text{NR}}(x, y) = 0$, where Φ_{NR} is the natural residual function (also called the "min" function). We introduce two important subclasses of $\mathscr{F}$, which deserve particular attention because of their significantly different theoretical and numerical properties. One of these subfamilies yields a smoothing function for Φ_{NR}, while the other subfamily only yields a smoothing curve for $\Phi_{\text{NR}}(x, y) = 0$. We also propose a simple framework for generating functions from these subclasses. Using the smoothing approach, we build two types of neural networks and provide sufficient conditions to guarantee asymptotic and exponential stability of equilibrium solutions. Finally, we present extensive numerical experiments to validate the theoretical results and to illustrate the difference in numerical performance of functions from the two subclasses. Numerical comparisons with existing neural networks for NCPs are also demonstrated.

1. Introduction

A nonlinear complementarity problem (NCP) consists of nonlinear inequalities with nonnegativity and orthogonality conditions on multiple variables and given multivariate functions. Such problems arise in constrained optimization and equilibrium problems [1]. Moreover, applications of NCPs in operations research, engineering, and economics motivated significant research efforts in the past decades [1–3], which resulted in various numerical techniques including the merit function approach [4–6], nonsmooth Newton method [7–9], and regularization approach [10,11]. We refer the interested reader to the monograph [1] and the paper [12] for a survey and thorough discussion of solution methods for complementarity problems.

Smoothing methods belong to another class of solution methods that have been extensively used in solving complementarity problems [10,13–17]. A natural smoothing technique for the NCP is to construct a differentiable approximation of

$$\phi_{\text{NR}}(s, t) := \min\{s, t\} = s - (s - t)_+ \tag{1.1}$$

using a smoothing function for the plus function. Meanwhile, Haddou and Maheux [18] recently introduced a novel smoothing approach to handle NCPs. In their approach, they utilized smooth perturbations of ϕ_{NR} that do not necessarily

* Corresponding author.
E-mail addresses: 80640005s@ntnu.edu.tw (J.H. Alcantara), jschen@math.ntnu.edu.tw (J.-S. Chen).

[1] The author's work is supported by Ministry of Science and Technology, Taiwan.

https://doi.org/10.1016/j.cam.2022.114092
0377-0427/

correspond to smoothing functions for ϕ_{NR}. These perturbations can be viewed as smooth approximations of the level curve $\phi_{\text{NR}}(s,t)=0$, which are then used to obtain approximate solutions of the NCP. Unfortunately, there have been no further studies on algorithmic procedure for NCPs following this smoothing framework. Moreover, the choice of smooth perturbations as well as a procedure for controlling the perturbation parameter are some numerical issues that were left for future studies in [18].

Meanwhile, neural network (NN) approaches for complementarity problems have also been explored in [19–21] mainly because it is desirable to have a real-time solution of the NCP, which may not be attainable with the usual approaches mentioned above. Neural networks are hardware-implementable, i.e. via integrated circuits, and therefore exhibit real-time processing. Hopfield and Tank originally introduced neural networks for optimization [22,23], and since then have been used in solving linear and nonlinear programming problems and variational inequalities [24–30]. Notice, however, that the NCP is not an optimization problem. Nevertheless, the NCP can be reformulated as a smooth minimization problem by constructing a merit function usually through the use of NCP-functions [4–6,19–21]. A nonnegative merit function usually serves as an energy function, which is then used to formulate a steepest-descent dynamical system whose equilibrium solutions correspond to NCP solutions under some suitable conditions.

A neural network based on the Fischer–Burmeister (FB) function was designed to handle P_0-NCPs in [21]. In [20], these results were extended to the generalized Fischer–Burmeister (GFB) function, an NCP function that involves a parameter $p\in(1,\infty)$. It was shown that for the latter NN, better numerical performance of the network can be achieved by choosing a larger value of p. These neural networks have good stability and convergence properties and are not very sensitive to initial conditions. Aside from FB and GFB neural networks, three classes of NNs based on the discrete generalization of ϕ_{NR} given in (1.1) were recently formulated in [19]. This neural network has relatively better convergence speed than FB and GFB neural networks, but its stability requires stricter assumptions and the NN is usually more sensitive to initial conditions as compared to FB and GFB networks.

In this paper, we build a new class of neural networks based on the smoothing method for NCP introduced by Haddou and Maheux [18] using some family $\mathscr{F}$ of smoothing functions. We introduce two subclasses of $\mathscr{F}$ and propose a simple method to generate functions from these subclasses. Interestingly, the aforementioned issues on how to choose the smooth perturbations of ϕ_{NR} and how to control the decrease of the smoothing parameter can be best addressed by considering these two subclasses. Sufficient conditions to guarantee stability of the neural network are provided and are illustrated through several numerical experiments. Finally, we compare the present neural networks with the well-known FB and GFB neural networks in solving P_0- and non-P_0-NCPs.

This paper is organized as follows: In Section 2, we recall the smoothing method proposed in [18] and introduce the family $\mathscr{F}$. In Section 3, we discuss a simple idea on how to generate smoothing functions. We also prove a characterization result for the two subfamilies of $\mathscr{F}$ introduced. In Section 4, two neural networks for NCPs are presented. We provide several sufficient conditions for Lyapunov and exponential stability of the neural networks. Several numerical simulations are presented in Section 5 to understand how to choose a smoothing function for the NCP. A comparative analysis with FB and GFB neural networks is also shown. Conclusions and some recommendations for future studies are presented in Section 6.

The notations used for this study are as follows. $\mathbb{R}^n$ denotes the n-dimensional Euclidean space endowed with the usual inner product, and $\mathbb{R}^{m\times n}$ denotes the space of $m\times n$ real matrices. We let $\mathbb{R}^n_+$ and $\mathbb{R}^n_{++}$ denote the nonnegative and positive orthant of $\mathbb{R}^n$, respectively. M^T denotes the transpose of a matrix M. M_{ij} and $M_{\cdot j}$ will be used to denote the (i,j)-entry of M and jth column of M, respectively. For $M\in\mathbb{R}^{n\times n}$ and $\Lambda\subseteq\{1,\dots,n\}$, we denote by M_Λ the principal submatrix of M indexed by Λ (i.e. the submatrix of M corresponding to rows and columns indexed by Λ). Λ^c denotes the complement of Λ. For any differentiable function $f:\mathbb{R}^2\to\mathbb{R}$, $\nabla_a f(s,t)$ and $\nabla_b f(s,t)$ means the partial derivative of f w.r.t. s and t, respectively. Given a differentiable mapping $F=(F_1,\dots,F_m)^T:\mathbb{R}^n\to\mathbb{R}^m$, $\nabla F(x)=[\nabla F_1(x)\ \cdots\ \nabla F_m(x)]\in\mathbb{R}^{n\times m}$ denotes the transposed Jacobian of F at x, where $\nabla F_i(x)$ denotes the gradient of F_i at x. Given a family of real-valued functions on $\mathbb{R}^n$: $\{\phi_r : r>0\}$, we denote by ϕ_0 the pointwise limit $\lim_{r\searrow 0}\phi_r$, whenever it exists.

2. Smoothing approach for NCP

In this section, we present the smoothing approach for NCP proposed by Haddou and Maheux [18] and introduce two important classes of smoothing functions. We also recall some concepts related to nonlinear mappings.

Let $F:\mathbb{R}^n\to\mathbb{R}^n$ be given. The nonlinear complementarity problem, which we denote by NCP(F), is to find a point $x\in\mathbb{R}^n$ such that

$$x\geq 0,\qquad F(x)\geq 0\qquad\text{and}\qquad\langle x,F(x)\rangle=0. \tag{2.1}$$

The set of all solutions of NCP(F) is denoted by SOL(F), which we assume to be nonempty. We can also show the equivalence of NCP(F) and the equation $x=P_K(x-F(x))$, where $K=\mathbb{R}^n_+$ and P_K denotes the projection onto K. In light of this equivalence, we see that x solves (2.1) if and only if $\Phi_{\text{NR}}(x,F(x))=0$ where $\Phi:\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}^n$ is a nonsmooth mapping given by

$$\Phi_{\text{NR}}(x,y)=\begin{pmatrix}\phi_{\text{NR}}(x_1,y_1)\\ \vdots\\ \phi_{\text{NR}}(x_n,y_n)\end{pmatrix}.$$

In this paper, we will consider nonlinear monotone and P_0-functions. We recall some basic types of nonlinear mappings.

Definition 2.1. Let $F = (F_1, \ldots, F_n)^T : \mathbb{R}^n \to \mathbb{R}^n$. Then, the mapping F is said to be

(i) monotone if $\langle x - y, F(x) - F(y)\rangle \geq 0$ for all $x, y \in \mathbb{R}^n$.

(ii) strictly monotone if $\langle x - y, F(x) - F(y)\rangle > 0$ for all $x, y \in \mathbb{R}^n$ and $x \neq y$.

(iii) strongly monotone with modulus $\mu > 0$ if $\langle x - y, F(x) - F(y)\rangle \geq \mu\|x - y\|^2$ for all $x, y \in \mathbb{R}^n$.

(iv) a P_0-function if $\max\limits_{\substack{1\leq i\leq n\\ x_i\neq y_i}}(x_i - y_i)(F_i(x) - F_i(y)) \geq 0$ for all $x, y \in \mathbb{R}^n$ and $x \neq y$.

(v) a P-function if $\max\limits_{1\leq i\leq n}(x_i - y_i)(F_i(x) - F_i(y)) > 0$ for all $x, y \in \mathbb{R}^n$ and $x \neq y$.

(vi) a uniform P-function with modulus $\kappa > 0$ if $\max\limits_{1\leq i\leq n}(x_i - y_i)(F_i(x) - F_i(y)) \geq \kappa\|x - y\|^2$, for all $x, y \in \mathbb{R}^n$.

A matrix is called a P_0-matrix (resp. P-matrix) if all its principals minors are nonnegative (resp. positive). Note that F is a P_0-function if and only if $\nabla F(x)$ is a P_0-matrix for all $x \in \mathbb{R}^n$. If $\nabla F(x)$ is a P-matrix for all $x \in \mathbb{R}^n$, then F is a P-function. However, the converse is not necessarily true.

We now recall the smoothing approach described in [18]. First, we consider a continuously differentiable strictly increasing function θ such that

$$\lim_{t\to-\infty}\theta(t) = -\infty, \qquad \theta(0) = 0 \qquad \text{and} \qquad \lim_{t\to+\infty}\theta(t) = 1 \tag{2.2}$$

We also impose a condition that $\theta(t)$ should approach 1 fast enough but with some uniformity for large values of t. This condition is precisely defined as follows.

Definition 2.2. Let $a \in (0, 1)$ and suppose $\theta(t)$ is a strictly increasing C^1 function such that $\theta(0) = 0$ and $\lim_{t\to\infty}\theta(t) = 1$. We say that θ satisfies condition (H_a) if there exists $t_a > 0$ such that

$$\frac{1}{2} + \frac{1}{2}\theta(at) \leq \theta(t) \quad \forall t \geq t_a.$$

We say that θ belongs to class $\mathscr{F}$ if it satisfies condition (H_a) for some $a \in (0, 1)$.

For instance, the following functions were considered in [18]:

$$\theta^{(1)}(t) = \begin{cases} \frac{t}{t+1} & \text{if } t \geq 0 \\ t & \text{if } t < 0 \end{cases} \qquad \text{and} \qquad \theta^{(2)}(t) = 1 - e^{-t}. \tag{2.3}$$

$\theta^{(1)}$ satisfies (H_a) for all $a \in (0, 1/2)$ while $\theta^{(2)}$ satisfies (H_a) for all $a \in (0, 1)$.

For each $r > 0$, we define the function $\phi_r : \mathbb{R}^2 \to \mathbb{R}$ by

$$\phi_r(s, t) := r\theta^{-1}\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right). \tag{2.4}$$

Note that strict monotonicity of θ was imposed to guarantee its invertibility as required in (2.4). We summarize here some important facts proved in [18].

Lemma 2.1. *Let θ be a strictly increasing C^1 function satisfying conditions (2.2). Then, the following holds.*

(i) $\phi_r(s, t) \leq \min\{s, t\}$ *for all* $s, t \in \mathbb{R}$ *and* $r > 0$.

(ii) *If $\theta \in \mathscr{F}$, then*

$$\lim_{r\searrow 0}\phi_r(s, t) = 0 \Longleftrightarrow \min\{s, t\} = 0 \qquad \forall s, t \in \mathbb{R}.$$

(iii) *If θ satisfies condition (H_a) for all $a \in (0, 1)$, then*

$$\lim_{r\searrow 0}\phi_r(s, t) = \min\{s, t\} \qquad \forall s, t \in \mathbb{R}_{++}.$$

Let $G_r : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ be given by

$$G_r(x, y) = \begin{pmatrix} \phi_r(x_1, y_1) \\ \vdots \\ \phi_r(x_n, y_n) \end{pmatrix}. \tag{2.5}$$

By Lemma 2.1(iii), G_r is a smoothing function for Φ_{NR} over $\mathbb{R}^n_{++} \times \mathbb{R}^n_{++}$ if θ satisfies (H_a) for all $a \in (0, 1)$. In contrast, we may not obtain a smoothing function for Φ_{NR} if there exists $a \in (0, 1)$ for which (H_a) does not hold. For instance, when $\theta = \theta^{(1)}$, one can verify that $\lim_{r \searrow 0} G_r(x, y) < \Phi_{\text{NR}}(x, y)$ (see also [18]). In view of these contrasting properties, we introduce two subclasses of $\mathscr{F}$.

Definition 2.3. Let $\theta \in \mathscr{F}$. We say that θ belongs to class $\mathscr{F}_1$ if it satisfies condition (H_a) for all $a \in (0, 1)$. Otherwise, we say that θ belongs to $\mathscr{F}_2$.

In Section 3, we will prove that the result of Lemma 2.1(iii) holds only for functions in $\mathscr{F}_1$. Despite not obtaining a smoothing function for Φ_{NR} when $\theta \in \mathscr{F}_2$, Lemma 2.1(ii) is a very useful result to apply Haddou and Maheux's smoothing strategy. To see this, define the functions

$$\Phi_r(x) := G_r(x, F(x)) \qquad \text{and} \qquad \Phi_0(x) := \lim_{r \searrow 0} \Phi_r(x),$$

provided that the limit exists. Then by Lemma 2.1(ii), (2.1) is equivalent to solving the system of equations $\Phi_0(x) = 0$. Moreover, it is shown in [18] that whenever F is a P_0-function, Φ_r is a P-function for any $r > 0$ while Φ_0 is a P_0-function whenever it exists. Consequently, one may use Gowda and Tawhid's theory for P_0-equations given in [31] by viewing Φ_0 as a continuous perturbation of Φ_r. Indeed, Lemma 2.3 which was given in [18] is an easy consequence of Theorem 4 of [31]. Before stating the convergence result, we make some assumptions.

Assumption 1. SOL(F) is nonempty and bounded, $\theta \in \mathscr{F}$ and Φ_0 exists.

Some conditions which will guarantee the existence of Φ_0 are given in the following result, whose proofs can be found in [18].

Lemma 2.2. *Let $s, t > 0$. Then,*

$$\phi_0(s, t) := \lim_{r \searrow 0} \phi_r(s, t) \quad \textit{exists} \quad \textit{and} \quad \phi_0(s, t) \le \min\{s, t\}$$

if any of the following condition holds:

(i) *There exists $\varepsilon > 0$ such that $\frac{\partial}{\partial r}\phi_r(s, t) \le 0$ for all $r \in (0, \varepsilon)$;*

(ii) *$V := (-\psi' \circ \psi^{-1}) \times \psi^{-1}$ is locally subadditive at 0^+; i.e. there exists $\eta > 0$ such that for all $0 < \alpha, \beta, \alpha + \beta < \eta$, we have*

$$V(\alpha + \beta) \le V(\alpha) + V(\beta),$$

where $\psi := 1 - \theta$.

Lemma 2.3. *Suppose Assumption 1 holds and that F is a P_0-function.*

(i) *There exists an $\hat{r} > 0$ such that for any $r \in (0, \hat{r})$, $\Phi_r(x) = 0$ has a unique solution $x^{(r)}$ and the mapping $r \mapsto x^{(r)}$ is continuous on $(0, \hat{r})$.*

(ii) $\lim_{r \searrow 0} \inf_{x^* \in \text{SOL}(F)} \|x^{(r)} - x^*\| = 0.$

Lemma 2.3(i) implies the well-posedness of the equation $\Phi_r(x) = 0$, while Lemma 2.3(ii) suggests the following typical algorithm employed in smoothing techniques, which is the one presented in [18].

Algorithm 1.

Let $\varepsilon > 0$, and $x^0 \in \mathbb{R}^n_{++}$. Set $r^0 = \max\{1, \sqrt{\max_{1 \le i \le n} |x_i^0 F_i(x^0)|}\}$ and $k = 0$.

For $k = 0, 1, \ldots$ until $\max_{1 \le i \le n} |x_i^k F_i(x^k)| \le \varepsilon$, **do**

1. Solve $\Phi_{r^k}(x) = G_{r_k}(x, F(x)) = 0$, where $G_r(\cdot, \cdot)$ is defined in (2.5), to get a solution x^k.
2. Set $r^{k+1} = \min\{0.1 r^k, (r^k)^2, \sqrt{\max_{1 \le i \le n} |x_i^k F_i(x^k)|}\}$.

We remark that if $x^{(r)}$ solves $\Phi_r(x) = 0$, then $x^{(r)}$ and $F(x^{(r)})$ are both in $\mathbb{R}^n_{++}$.

3. Designing smoothing functions

In Section 2, we introduced two subclasses of $\mathscr{F}$ based on whether or not condition (H_a) is satisfied for all $a \in (0, 1)$. In this section, we discuss some important remarks and results related to condition (H_a). Moreover, we present a simple and intuitive way to generate functions θ from $\mathscr{F}$. Under some suitable assumptions, the construction way also provides information on the classification of θ, i.e. whether θ belongs to $\mathscr{F}_1$ or $\mathscr{F}_2$.

We consider first a function θ satisfying (2.2) restricted to $\mathbb{R}_+$. Observe that there is a one-to-one correspondence between these functions and probability density functions. Indeed, if $p(t) := \theta'(t)$, then $p(t) \geq 0$ and

$$\int_0^\infty p(t)\,dt = \lim_{t\to\infty}\int_0^t \theta'(\tau)\,d\tau = \lim_{t\to\infty}(\theta(t)-\theta(0)) = 1.$$

Hence, the function θ corresponds to a probability density function on $\mathbb{R}_+$. Conversely, one natural way to generate θ is to consider a probability density function $p : \mathbb{R}_+ \to \mathbb{R}_+$ and taking

$$\theta(t) = \int_0^t p(\tau)\,d\tau$$

for all $t \in [0, \infty)$. This construction way was also mentioned in [32].

Another natural and intuitive way to generate θ with the desired asymptotic behavior at infinity is to consider a differential equation with an equilibrium solution at 1. Consider for instance the separable autonomous differential equation given by

$$\frac{d\theta}{dt} = f(\theta), \qquad \theta(0) = 0. \tag{3.1}$$

Throughout the paper, the following are our assumptions on f:

Assumption 2.

(i) f is C^1 on an open interval containing $[0, 1]$.
(ii) $f > 0$ on $[0, 1)$ and $f(1) = 0$.

Assumption 2(i) is important in ensuring that the initial-value problem (3.1) has a unique solution. On the other hand, Assumption 2(ii) is required to obtain an increasing function $\theta(t)$ which approaches 1 as $t \to \infty$.

Proposition 3.1. *Let f be any function satisfying Assumption 2. Then the initial-value problem (3.1) has a unique solution which is defined for all $t \geq 0$ and $\lim_{t\to\infty}\theta(t) = 1$.*

Proof. The existence and uniqueness of solution follows from Assumption 2(i) (see [33]). Let $[0, T)$ be the maximal interval of existence of the solution $\theta(t)$ of (3.1).

Since $f(\theta) > 0$ for all $\theta \in (0, 1)$, $\theta(t)$ is an increasing function. Thus, we may let $L = \lim_{t\to\infty}\theta(t)$. Note also that the constant function 1 is a solution of the differential equation $\frac{d\theta}{dt} = f(\theta)$. Since solution through a point is unique, $\theta(t) < 1$ for all $t \in [0, T)$. Consequently, $L \leq 1$ and $\{\theta(t) : t \in [0, T)\} \subseteq [0, 1]$. Hence, $\theta(t)$ is defined for all $t \geq 0$ (see [33]). Finally, note that from (3.1), we have

$$\theta(t) = \int_0^t f(\theta(\tau))\,d\tau.$$

Since $\theta(t) \to L$, we conclude that $\int_0^\infty f(\theta(\tau))\,d\tau$ is convergent. It follows that

$$f(L) = \lim_{t\to\infty} f(\theta(t)) = 0.$$

By Assumption 2(ii), we conclude that $L = 1$, as desired. □

It was mentioned in the preceding section that satisfying condition (H_a) is very important to guarantee the applicability of the smoothing approach. The benefit of constructing θ using (3.1) is that condition (H_a) can be easily deduced if f has a specific form. We describe this in the following theorem.

Theorem 3.1. *Let $\bar{f}$ be a continuously differentiable function on an open interval containing $[0, 1]$ such that $\bar{f} > 0$ on $[0, 1]$. Suppose that f takes the form $f(\theta) = \bar{f}(\theta)(1-\theta)^k$, where $k \geq 1$. If $\theta(t)$ denotes the unique solution of (3.1), then $\theta \in \mathscr{F}$. In particular,*

(i) *If $k = 1$, then $\theta \in \mathscr{F}_1$.*

(ii) *If $k > 1$, then $\theta \in \mathscr{F}_2$. Moreover, condition (H_a) is satisfied when $a \in \left(0, \frac{1}{2^{k-1}}\right)$ and is not satisfied when $a \in \left(\frac{1}{2^{k-1}}, 1\right)$.*

Before we prove Theorem 3.1, we need the following lemma. This lemma will also be useful later in Theorems 4.3 and 4.4.

Lemma 3.1. *Under the hypothesis of Theorem 3.1, we have*

$$\lim_{t\to\infty}\frac{1-\theta(t)}{1-\theta(at)} = \begin{cases} 0 & \text{if } k = 1 \\ a^{\frac{1}{k-1}} & \text{if } k > 1 \end{cases} \qquad \forall a \in (0, 1).$$

Proof. Let $a \in (0, 1)$ and define $\psi(t) := 1 - \theta(t)$. Note that

$$\frac{d}{dt}\psi(t) = -\frac{d\theta}{dt} = -\bar{f}(\theta(t))(1-\theta(t))^k = -\bar{f}(\theta(t))(\psi(t))^k. \tag{3.2}$$

Suppose first that $k = 1$. Since $\theta(0) = 0$, then $\psi(0) = 1$. Moreover, we have from (3.2) that

$$1 - \theta(t) = \psi(t) = \exp\left(-\int_0^t \bar{f}(\theta(\tau))\, d\tau\right).$$

Consequently, we have for any $a \in (0, 1)$ that

$$\frac{1-\theta(t)}{1-\theta(at)} = \frac{\exp\left(-\int_0^t \bar{f}(\theta(\tau))\, d\tau\right)}{\exp\left(-\int_0^{at} \bar{f}(\theta(\tau))\, d\tau\right)} = \exp\left(-\int_{at}^t \bar{f}(\theta(\tau))\, d\tau\right) \tag{3.3}$$

Since $\bar{f}(\theta) > 0$ in $[0, 1]$, there exists $m > 0$ such that $\bar{f}(\theta) > m$ for all $\theta \in [0, 1]$. Continuing from (3.3),

$$\frac{1-\theta(t)}{1-\theta(at)} \le \exp(-m(1-a)t) \qquad \forall t \ge 0.$$

Letting $t \to \infty$ gives the desired limit for the first case.

Now, suppose that $k > 1$. From (3.2), we have

$$(1-\theta(t))^{1-k} = (\psi(t))^{1-k} = 1 + (k-1)\int_0^t \bar{f}(\theta(\tau))\, d\tau.$$

Note that due to $\bar{f}(1) \neq 0$, we have $\int_0^\infty \bar{f}(\theta(\tau))\, d\tau = \infty$. Then, this leads to

$$\lim_{t\to\infty}\left(\frac{1-\theta(t)}{1-\theta(at)}\right)^{k-1} = \lim_{t\to\infty}\frac{1+(k-1)\int_0^{at}\bar{f}(\theta(\tau))\, d\tau}{1+(k-1)\int_0^t \bar{f}(\theta(\tau))\, d\tau} = \lim_{t\to\infty}\frac{a\bar{f}(\theta(at))}{\bar{f}(\theta(t))} = a,$$

where the last equality follows from the fact that $\bar{f}(1) \neq 0$ and $\theta(t) \to 1$ by Proposition 3.1. □

Proof of Theorem 3.1. Fix $a \in (0, 1)$ and define $h_a(t) = \frac{1}{2} + \frac{1}{2}\theta(at) - \theta(t)$ for all $t \ge 0$. Differentiating h_a, we obtain

$$h_a'(t) = \frac{a}{2}\theta'(at) - \theta'(t) = \frac{a}{2}f(\theta(at)) - f(\theta(t)). \tag{3.4}$$

Since $\bar{f}(1) \neq 0$ and $\theta(t) \to 1$, we have from Lemma 3.1 that

$$\lim_{t\to\infty}\frac{f(\theta(t))}{f(\theta(at))} = \lim_{t\to\infty}\frac{\bar{f}(\theta(t))(1-\theta(t))^k}{\bar{f}(\theta(at))(1-\theta(at))^k} = \begin{cases} 0 & \text{if } k = 1 \\ a^{\frac{k}{k-1}} & \text{if } k > 1. \end{cases} \tag{3.5}$$

Suppose now that $k = 1$. By (3.5), we can find $t_a > 0$ such that $f(\theta(t)) < \frac{a}{2}f(\theta(at))$ for all $t > t_a$. From (3.4), we see that $h_a'(t) > 0$ for all $t > t_a$ and therefore $h_a(t)$ is strictly increasing on (t_a, ∞). But clearly, $h_a(t)$ approaches 0 as $t \to \infty$. Thus, $h_a(t) < 0$ for all $t > t_a$. That is, $\theta(t)$ satisfies condition (H_a) for all $a \in (0, 1)$.

On the other hand, suppose $k > 1$. If $0 < a < \frac{1}{2^{k-1}}$, then

$$\frac{a}{2} - a^{\frac{k}{k-1}} = a\left(\frac{1}{2} - a^{\frac{1}{k-1}}\right) > 0.$$

Then, by (3.5), we can find $t_a > 0$ such that $\frac{f(\theta(t))}{f(\theta(at))} < \frac{a}{2}$ for all $t > t_a$. As in the preceding case, $\theta(t)$ satisfies condition (H_a) for all $a \in \left(0, \frac{1}{2^{k-1}}\right)$. If $\frac{1}{2^{k-1}} < a < 1$, then $a^{\frac{k}{k-1}} - \frac{a}{2} > 0$ and so there exists $t_a > 0$ for which $\frac{f(\theta(t))}{f(\theta(at))} > \frac{a}{2}$ for all $t > t_a$. It follows that $h_a(t) > 0$ over (t_a, ∞). This completes the proof. □

Remark 3.1. Given a function f satisfying Assumption 2, we may apply Theorem 3.1 provided that there exists $k \ge 1$ such that $L := \lim_{\theta\to 1} f(\theta)(1-\theta)^{-k} \neq 0$. In this case, we define $\bar{f}(\theta) := f(\theta)(1-\theta)^{-k}$ for $\theta \in [0, 1)$ and define $\bar{f}(1) := L$. We then obtain the desired factorization $f(\theta) = \bar{f}(\theta)(1-\theta)^k$.

In general, this factorization is not always achievable. To see this, consider the function

$$g(t) = t^3\left(1 + \sin\frac{1}{t}\right) + t(1 - \cos t).$$

It is easy to verify that g is continuously differentiable on $\mathbb{R}$. Moreover, since the first term of g is nonnegative on $(0, 1]$ while the second term is greater than 0 on $(0, 1]$, then g is strictly positive on $(0, 1]$. In addition,

$$\lim_{t\to 0^+}\frac{g(t)}{t^k} = 0 \qquad \forall k \in [1, 3),$$

and the above limit does not exist for $k \geq 3$. Defining $f(\theta) := g(1-\theta)$, we see that f satisfies Assumption 2. But, there is no $k \geq 1$ such that $\lim_{\theta \to 1^-} f(\theta)(1-\theta)^{-k}$ exists and is nonzero.

The following is an easy consequence of Theorem 3.1 and the above remark.

Corollary 3.1. *If $f'(1) < 0$, then $\theta \in \mathscr{F}_1$.*

Example 3.1. We generate five functions from (3.1), two of which are the ones used in [18] given by Eq. (2.3). It can be easily verified that V given in Lemma 2.2 is locally subadditive at 0^+ for each of the functions below.

The parameter k has a significant role in numerical simulations. Notice that $(1-\theta)^k$ decreases in value as k increases when $\theta \in (0,1)$. In view of (3.1) and the form of f given in Theorem 3.1, k controls the rate of growth of the function $\theta(t)$. In Section 5, it is shown that functions with faster growth rate yield faster convergence time for the neural networks. However, the rate of increase of θ should not be very fast so as to avoid ill-conditioning. Hence, the parameter k is useful when we take into account the conditioning of the problem. That is, a higher value of k may be adapted when the problem is ill-conditioned.

We now give an example of a function θ that does not belong to $\mathscr{F}$.

Example 3.2. Define $\theta(t) = \dfrac{2}{\pi}\displaystyle\int_0^t \frac{\sin^2 \tau}{\tau^2} d\tau$. Indeed, θ is a strictly increasing continuously differentiable function such that $\theta(0) = 0$ and $\theta \to 1$ as $t \to \infty$. The function h_a defined in the proof of Theorem 3.1 oscillates between positive and negative values of t, and therefore does not satisfy condition (H_a) for any $a \in (0,1)$.

Remark 3.2. One can observe that class $\mathscr{F}_1$ functions can be viewed, in some sense, as the limit case of class $\mathscr{F}_2$ functions. First, notice that in Lemma 3.1, the zero limit obtained for class $\mathscr{F}_1$ function can be viewed as the limit of $a^{\frac{1}{k-1}}$ as $k \searrow 1$, where $a^{\frac{1}{k-1}}$ is the calculated limit for class $\mathscr{F}_2$ function. In Theorem 3.1, we see that for $\theta \in \mathscr{F}_2$, condition (H_a) is satisfied only for $a \in I_k := (0, 2^{1-k})$. Note that $I_k \nearrow (0,1)$ as $k \searrow 1$ and that condition (H_a) holds for all $a \in (0,1)$ for class $\mathscr{F}_1$ functions.

Remark 3.3. We mention some remarks regarding the definition of θ over $(-\infty, 0)$. In order for G_r given by (2.5) to be smooth over $\mathbb{R}^n \times \mathbb{R}^n$ for any $r > 0$, we must use a differentiable extension of θ for $t < 0$. In this paper, we let $\theta(t) = tf(0)$ for $t < 0$ in this paper, similar to the construction of $\theta^{(1)}$ in (2.3).

We conclude this section with the following proposition which implies that the limit in Lemma 2.1(iii) is in fact a characterization of functions in $\mathscr{F}_1$. That is, a smoothing function of ϕ_{NR} over the positive orthant is obtained if and only if $\theta \in \mathscr{F}_1$.

Proposition 3.2. *Suppose $\theta \in \mathscr{F}_2$ and let $\bar{a} \in \inf S$, where*

$$S = \{\, a \in (0,1) \mid \textit{condition } (H_a) \textit{ does not hold} \}.$$

Let $P_{\bar{a}} = \{(s,t) \in \mathbb{R}^2_{++} \ : \ \bar{a}s < t < \frac{s}{\bar{a}}\}$. Then, we have

$$\lim_{r \searrow 0} \phi_r(s,t) < \min\{s,t\} \qquad \forall (s,t) \in P_{\bar{a}},$$

whenever the limit exists.

Proof. We note first that $(\bar{a}, 1) \subseteq S$. Indeed, the set S is nonempty and $\bar{a} \in (0,1)$ since $\theta \in \mathscr{F}_2$. If $\tilde{a} \in S$, for any $u > 0$, $\theta(v) < \frac{1}{2} + \frac{1}{2}\theta(\tilde{a}v)$ for some $v \geq u$. Since θ is increasing, we see that condition (H_a) also does not hold for any $a \in (\tilde{a}, 1)$.

Now, fix $(s,t) \in P_{\bar{a}}$ and suppose $s = \min\{s,t\}$. Given any $a \in (\bar{a}, 1)$, then by the above note, there exists a sequence $\{r_n\}$ such that $r_n \searrow 0$ and

$$\theta\left(\frac{t}{r_n}\right) < \frac{1}{2} + \frac{1}{2}\theta\left(\frac{at}{r_n}\right).$$

Since θ and θ^{-1} are increasing and $s \leq t$, the above inequality yields

$$\phi_{r_n}(s,t) = r_n \theta^{-1}\left(\theta\left(\frac{s}{r_n}\right) + \theta\left(\frac{t}{r_n}\right) - 1\right) \leq r_n \theta^{-1}\left(2\theta\left(\frac{t}{r_n}\right) - 1\right) < at$$

Letting $n \to \infty$, we obtain $\lim_{n\to\infty} \phi_{r_n}(s,t) \leq at$ for any $a \in (\bar{a}, 1)$. Since $(s,t) \in P_{\bar{a}}$, then $s > \bar{a}t$ and by letting $a \searrow \bar{a}$, we obtain

$$\lim_{n\to\infty} \phi_{r_n}(s,t) \leq \bar{a}t < s = \min\{s,t\}$$

which is the desired result. □

It is interesting to note that as in Remark 3.2, class $\mathscr{F}_1$ can again be viewed as the limit case of class $\mathscr{F}_2$ when $\bar{a} \searrow 0$. More precisely, the limit in Lemma 2.1(iii) holds on $\bigcup_{\bar{a} \in (0,1)} P_{\bar{a}} = \mathbb{R}^2_{++}$ when $\theta \in \mathscr{F}_1$.

4. Smoothed neural networks

In this section, we present two gradient dynamical systems to solve the NCP (2.1) using the smoothing approach presented in Section 2. Similar to the approach used in [19,21], the stability analysis of our neural networks relies on the use of Lyapunov functions. For more details, we refer the reader to [33,34].

4.1. The first neural network

We consider first a neural network which can be used to obtain approximate solutions of NCP(F). From Lemma 2.3, solutions of (2.1) can be obtained by successively solving for decreasing values of r the equation $\Phi_r(x) = 0$, which is exactly the motivation of Algorithm 1. Note that solving the aforementioned equation, if a solution exists, is equivalent to solving the smooth minimization problem

$$\min_{x\in\mathbb{R}^n} \Psi_r(x) := \frac{1}{2}\|\Phi_r(x)\|^2. \tag{4.1}$$

This motivates the steepest descent-based neural network

$$\frac{dx}{dt} = -\rho\nabla\Psi_r(x), \qquad x(0) = x_0, \tag{NN1}$$

where $\rho > 0$ is a time-scaling parameter and $r > 0$ is a sufficiently small fixed number. Consequently, neural network (NN1) can be used to deal with Step 1 of Algorithm 1.

Observe that a solution of (4.1) is an equilibrium point of (NN1). However, the converse is not necessarily true. We are interested on conditions that can be imposed on F so that an equilibrium point of the neural network (NN1) is also an optimal solution of (4.1), and consequently an approximated solution of (2.1). To this end, we establish first an important property of $\nabla\Phi_r$.

Theorem 4.1. *Let F be a P_0-function, $r > 0$ and suppose that $\theta'(u) \neq 0$ for all $u \in \mathbb{R}$. Then $\nabla\Phi_r(x)$ is a nonsingular for any $x \in \mathbb{R}^n$ and $r > 0$.*

Proof. First, we note that

$$\begin{aligned}\nabla_a\phi_r(s,t) = \frac{\partial}{\partial s}\phi_r(s,t) &= (\theta^{-1})'\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right)\theta'\left(\frac{s}{r}\right)\\ &= \frac{\theta'\left(\frac{s}{r}\right)}{\theta'\left(\theta^{-1}\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right)\right)}\\ &> 0,\end{aligned} \tag{4.2}$$

where we used the fact that $\theta'(\theta^{-1}(u))\cdot(\theta^{-1})'(u) = 1$. By symmetry, we also have $\nabla_b\phi_r(s,t) := \frac{\partial}{\partial t}\phi_r(s,t) > 0$.

Let $x \in \mathbb{R}^n$ and $r > 0$, and denote by $A_r(x)$ and $B_r(x)$ the $n \times n$ diagonal matrices such that

$$(A_r(x))_{ii} = \nabla_a\phi_r(x_i, F_i(x)) \qquad \text{and} \qquad (B_r(x))_{ii} = \nabla_b\phi_r(x_i, F_i(x)).$$

Note that $A_r(x)$ and $B_r(x)$ are invertible since $\nabla_a\phi_r(x_i, F_i(x))$ and $\nabla_b\phi_r(x_i, F_i(x))$ are both strictly positive for all i, as noted above. Furthermore, we have by Chain Rule that

$$\nabla\Phi_r(x) = A_r(x) + \nabla F(x)\cdot B_r(x) = (D_r(x) + \nabla F(x))\cdot B_r(x), \tag{4.3}$$

where $D_r(x) = A_r(x)B_r(x)^{-1}$. Since $D_r(x)$ is a diagonal matrix,

$$\det(D_r(x) + \nabla F(x)) = \sum_{\Lambda\subseteq\{1,\ldots,n\}} \det(D_\Lambda)\det(\nabla F(x)_{\Lambda^c}) \tag{4.4}$$

where D_Λ denotes the principal submatrix of $D_r(x)$ (see, for instance, Chapter 2 of [35]). Each term in the above summation is nonnegative since F is a P_0-function and the diagonal entries of $D_r(x)$ are positive. In particular, the term corresponding to $\alpha = \{1,\ldots,n\}$ is precisely $\det(D_r(x)) > 0$. Thus, $D_r(x) + \nabla F(x)$ is nonsingular. Since $B_r(x)$ is also nonsingular, we conclude from (4.3) that $\nabla\Phi_r(x)$ is nonsingular. □

Observe that the hypothesis on θ of the above theorem holds if it is generated from (3.1) using Assumption 2 and Remark 3.3. We now state some important consequences of the above result.

Corollary 4.1. *Under the hypotheses of Theorem 4.1, the following holds:*

(i) *Every equilibrium point of (NN1) is an optimal solution of (4.1).*

(ii) *An isolated equilibrium point of (NN1) is exponentially stable.*

(iii) *Suppose in addition that Assumption 1 holds. Then there exists $\hat{r} > 0$ such that for any $r \in (0, \hat{r})$, the neural network (NN1) has a unique equilibrium point $x^{(r)}$ which is exponentially stable.*

Proof. Suppose that x^* is an equilibrium point of (NN1), i.e. $\nabla\Psi_r(x^*) = 0$. Since $\nabla\Psi_r(x^*) = \nabla\Phi_r(x^*)\Phi_r(x^*)$ and $\nabla\Phi_r(x^*)$ is nonsingular by the preceding theorem, then $\Phi_r(x^*) = 0$. Hence, x^* is an optimal solution of (4.1). This proves part (i). If x^* is an isolated equilibrium point, then it is a strict local minimizer of Ψ_r by (i) and Theorem 4.1. Consequently, x^* is asymptotically stable (see also [19,34]). Exponential stability follows from noting the nonsingularity of $\nabla\Phi_r$ and from the same arguments used in [19]. This establishes (ii). Finally, part (iii) directly follows from Lemma 2.3, Theorem 4.1, and part (ii). □

Remark 4.1. Corollary 4.1(i) implies the equivalence of the systems $\Phi_r(x) = 0$ and $\nabla\Psi_r(x) = 0$. This justifies the use of the neural network (NN1) as a means to proceed with Step 1 of Algorithm 1 whenever F is a P_0-function. Meanwhile, the exponential stability proved in Corollary 4.1(ii) and (iii) asserts the efficiency of using the neural network (NN1) in solving the subproblems. Motivated by these, we may consider a sequence of neural network implementations as follows: Starting with an arbitrary initial point $x(0) = x^0 \in \mathbb{R}^n_{++}$ and smoothing parameter r^0 as in the initialization of Algorithm 1, we use the neural network (NN1) with $r = r^0$ to obtain an equilibrium point x^1. We then reduce the smoothing parameter r^0 as in Step 2 of Algorithm 1. Then, we use the neural network (NN1) with $r = r^1 < r^0$ and $x(0) = x^1$ to obtain a better approximated solution of (2.1) . Proceeding in this manner, we obtain a sequence $\{x^k\}$ of approximated solutions for NCP(F) which converges to an NCP solution by Corollary 4.1(i) and Lemma 2.3(ii). Due to the exponential stability of the neural network and the tendency of the solutions x^k to approach an NCP solution, the latter simulations of the neural network are expected to converge faster.

We now tackle the existence of equilibrium points of (NN1). When F is a P_0-function, this is equivalent to existence of solutions of $\Phi_r(x) = 0$ for $r > 0$. Recall that if F is a P_0-function, then Φ_r is a P-function by Lemma 2.3. If in addition, Φ_r is coercive, then it is known that $\Phi_r(x) = 0$ has a unique solution [31], say x^*. Meanwhile, the Lyapunov function Ψ_r for (NN1) is also coercive whenever Φ_r is. Invoking the Barbashin–Krasovskii Theorem [34], we conclude that x^* is a globally asymptotically stable equilibrium point of (NN1). This establishes the following lemma.

Lemma 4.1. *Suppose that F is a P_0-function and Φ_r is coercive for some $r > 0$. Then, the neural network (NN1) has a unique equilibrium point which is globally asymptotically stable.*

It is therefore worthwhile to determine conditions under which the function Φ_r is coercive.

Remark 4.2. For any $r > 0$, coerciveness of Φ_r in fact holds if F is a uniform P-function. To see this, the same arguments as those in [1] show that if $\{x^k\}_{k=1}^{\infty}$ with $\|x^k\| \to \infty$, then there exists an index j such that $|x_j^k| \to \infty$ and $|F_j(x^k)| \to \infty$. Meanwhile, observe from (2.4) that for any sequence $\{(s^k, t^k)\}_{k=1}^{\infty} \subseteq \mathbb{R}^2$ such that $|s^k| \to \infty$ and $|t^k| \to \infty$, we have $|\phi_r(s^k, t^k)| \to \infty$. It follows that $|\phi_r(x_j^k, F_j(x^k))| \to \infty$. Hence, $\Phi_r(x)$ is coercive. Apparently, we can also consider a class of functions larger than the class of uniform P-functions, which is described in the following definition.

Definition 4.1. The mapping $F : \mathbb{R}^n \to \mathbb{R}^n$ is called an R_0-function if for any sequence $\{x^k\}_{k=1}^{\infty}$ such that $\|x^k\| \to \infty$ and

$$\liminf_{k\to\infty} \frac{\min_{1\le i\le n} x_i^k}{\|x^k\|} \ge 0, \quad \liminf_{k\to\infty} \frac{\min_{1\le i\le n} F_i(x^k)}{\|x^k\|} \ge 0,$$

there exists an index j such that $x^k \to \infty$ and $F_j(x^k) \to \infty$.

The concept of R_0-function was introduced in [13] as a generalization of an affine function $F(x) = Ax + b$ where A is an R_0-matrix [35]. It was proved in [13] that a uniform P-function is an R_0-function. Hence, the following result is more general than the one given in Remark 4.2.

Theorem 4.2. *Suppose that F is both a P_0-function and an R_0-function. Then for any $r > 0$, the neural network (NN1) has a unique equilibrium point which is globally asymptotically stable.*

Proof. By Lemma 4.1, it is enough to show that Φ_r is coercive for any $r > 0$. To this end, let $\{x^k\}_{k=1}^{\infty}$ be such that $\|x^k\| \to \infty$. If $\{x_i^k\}_{k=1}^{\infty}$ and $\{F_i(x^k)\}_{k=1}^{\infty}$ are both bounded below for all $i = 1, \ldots, n$, then there exists $c \in \mathbb{R}$ such that $c \le x_i^k$ and $c \le F_i(x^k)$ for all $i = 1, \ldots, n$ and $k \ge 1$. Consequently,

$$0 = \lim_{k\to\infty} \frac{c}{\|x^k\|} \le \liminf_{k\to\infty} \frac{\min_{1\le i\le n} x_i^k}{\|x^k\|}$$

and similarly, $\liminf_{k\to\infty} \frac{\min_{1\le i\le n} F_i(x^k)}{\|x^k\|} \ge 0$. Since F is an R_0-function, there exists an index j such that $|x_j^k| \to \infty$ and $|F_j(x^k)| \to \infty$. As in Remark 4.2, we conclude that Φ_r is coercive as desired.

Table 1
θ functions generated from (3.1).

$\bar{f}$	k	θ
(1) $\bar{f}(\theta) = 1$	1	$\theta_1(t) = 1 - e^{-t}$
(2) $\bar{f}(\theta) = 1 + \theta$	1	$\theta_2(t) = \tanh t$
(3) $\bar{f}(\theta) = 1$	2	$\theta_3(t) = \dfrac{t}{t+1}$
(4) $\bar{f}(\theta) = 1$	3/2	$\theta_4(t) = \dfrac{t^2+4t}{(t+2)^2}$
(5) $\bar{f}(\theta) = (1+\theta)^{3/2}$	3/2	$\theta_5(t) = \dfrac{t}{\sqrt{t^2+1}}$

On the other hand, suppose there exists an index i such that $\{x_i^k\}_{k=1}^{\infty}$ is not bounded below, i.e. $x_i^k \to -\infty$. However, due to

$$\phi_r(x_i^k, F_i(x^k)) \le \min\{x_i^k, F_i(x^k)\}$$

and Lemma 2.1(i), we have $\phi_r(x_i^k, F_i(x^k)) \to -\infty$. Therefore, Φ_r is coercive. The case when $F_i(x^k) \to -\infty$ for some i can be dealt with similarly. □

For monotone complementarity problems, we can actually approximate how far the approximated solution $x^{(r)}$ is from the optimal solution. We recall the following result from [18].

Lemma 4.2. *Suppose that $\theta \ge \theta_3$, where θ_3 is given in Table 1. Let $\{x^{(r)}\}$ be a sequence of solutions of $\Phi_r(x) = 0$, where $r \in (0, \bar{r})$ for some $\bar{r} > 0$. Then, the following holds.*

(i) $x_i^{(r)} F_i(x^{(r)}) \le r^2$ *for all* $i = 1, \dots, n$.

(ii) *Suppose there exists $h : [0,\infty) \to [0,\infty)$ such that $h(0) = 0$, $h(t) > 0$ when $t > 0$, and there exists $\varepsilon, \eta > 0$ such that $h : (0, \varepsilon) \to (0, \eta)$ is an increasing bijection and*

$$h(\|x - y\|) \le \langle x - y, F(x) - F(y)\rangle.$$

Then, the NCP(F) has a unique solution x^* and there exists $r_0 > 0$ such that for all $r \in (0, r_0)$,

$$\|x^* - x^{(r)}\| \le h^{-1}(nr^2).$$

Observe that the above result applies for all functions θ generated in Table 1 since $\theta_i \ge \theta_3$ for all $i \in \{1, \dots, 5\}$. Moreover, this lemma is a useful error estimate when dealing with monotone functions F. In fact, as the next result claims, it can also be extended to deal with uniform P-functions (in which case, the NCP solution is unique [1]).

Proposition 4.1. *Suppose that $\theta \ge \theta^3$, F is a uniform P-function with modulus $\kappa > 0$, and let x^* be the unique solution of the NCP(F). For any $\varepsilon > 0$, the globally asymptotically stable equilibrium point $x^{(r)}$ of (NN1) satisfies $\|x^{(r)} - x^*\| \le \varepsilon$ provided that $0 < r < \sqrt{\kappa}\varepsilon$. In particular, if F is strongly monotone with modulus $\mu > 0$, we can take $0 < r < \varepsilon\sqrt{\frac{\mu}{n}}$.*

Proof. If F is a uniform P-function with modulus $\kappa > 0$, then

$$\begin{aligned} 0 \le \kappa\|x^{(r)} - x\|^2 &\le \max_{1\le i\le n}(x_i^{(r)} - x_i^*)(F_i(x^{(r)}) - F_i(x^*)) \\ &= \max_{1\le i\le n} x_i^{(r)} F_i(x^{(r)}) - (x_i^{(r)} F_i(x^*) + x_i^* F_i(x^{(r)})) \\ &\le r^2, \end{aligned}$$

where we have used Lemma 4.2(i) and the fact that $x^{(r)}$ and $F(x^{(r)})$ both lie on $\mathbb{R}^n_+$. This establishes the result for the case when F is a uniform P-function. On the other hand, note that a strongly monotone function with modulus μ is a uniform P-function with modulus μ/n. This completes the proof. □

4.2. The second neural network

The first neural network (NN1) allows us to obtain approximate solutions of the NCP (2.1), with error bounds as given in Proposition 4.1. In practice, we may use (NN1) to proceed with Step 1 of Algorithm 1. A numerical implementation is

also described in Remark 4.1. In this section, we consider a neural network where the smoothing parameter r is considered as a function of time which decreases to zero. To this end, define the function $\Phi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n \times \mathbb{R}$ by

$$\Phi(w) = \begin{pmatrix} \Phi_r(x) \\ \alpha r \end{pmatrix}, \qquad \alpha > 0$$

where $w = (x, r) \in \mathbb{R}^n \times \mathbb{R}$, and let Ω be the set of all points in $\mathbb{R}^{n+1}$ where Φ is differentiable. Define $\Psi : \Omega \to \mathbb{R}_+$ by

$$\Psi(w) = \frac{1}{2}\|\Phi(w)\|^2 = \Psi_r(x) + \frac{1}{2}\alpha^2 r^2.$$

We consider the neural network

$$\frac{dw}{dt} = -\rho\nabla\Psi(w), \qquad w(0) = w^0 \in \Omega. \tag{NN2}$$

Applying the Chain Rule, we have

$$\nabla\Psi(w) = \nabla\Phi(w)\Phi(w), \quad \text{where} \quad \nabla\Phi(w) = \begin{pmatrix} \nabla\Phi_r(x) & 0 \\ \left[\frac{\partial}{\partial r}\Phi_r(x)\right]^T & \alpha \end{pmatrix}. \tag{4.5}$$

As the next proposition claims, neural network (NN2) has exponentially stable equilibrium point if $\nabla\Phi_0(x^*)$ is nonsingular, and thus $r = r(t)$ ultimately decreases to zero while $x = x(t)$ converges to a solution of (2.1).

Proposition 4.2. *Let $w^* = (x^*, 0) \in \Omega$ be an isolated equilibrium point of (NN2). Then, w^* is stable. Moreover, if $\nabla\Phi_0(x^*)$ is nonsingular, w^* is exponentially stable and $x^* \in \mathrm{SOL}(F)$.*

Proof. Stability of w^* follows by using Ψ as a Lyapunov function [33]. On the other hand, it is clear from (4.5) that $\nabla\Phi(w^*)$ is nonsingular whenever $\nabla\Phi_0(x^*)$ is. Since $\nabla\Psi(w^*) = \nabla\Phi(w^*)\Phi(w^*) = 0$, we know that $\Phi(w^*) = 0$. It follows from Lemma 2.1(ii) that x^* solves (2.1). The same arguments as those used in Corollary 4.1(ii) can be used to prove that w^* is exponentially stable. □

Theorem 4.3. *Let $w^* = (x^*, 0) \in \Omega$ be an isolated equilibrium point of (NN2) with x^* in the feasible region of NCP(F). Suppose θ is generated from (3.1) with $f(\theta) = \bar{f}(\theta)(1-\theta)^k$ where $k > 1$ and $\bar{f}$ is as described in Theorem 3.1. Suppose further that Φ is continuously differentiable on a neighborhood of w^*. Then w^* is exponentially stable and $x^* \in \mathrm{SOL}(F)$.*

Proof. It is enough to show, by Proposition 4.2, that $\nabla\Phi_0(x^*)$ is nonsingular. We have from Eqs. (3.1) and (4.2) that for any $s, t \geq 0$,

$$\nabla_a\phi_r(s,t) = \frac{f\left(\theta\left(\frac{s}{r}\right)\right)}{f\left(\theta\left(\theta^{-1}\left(\theta\left(\frac{s}{r}\right)\right) + \theta\left(\frac{t}{r}\right) - 1\right)\right)}.$$

Thus, we have

$$\begin{aligned}
\nabla_a\phi_r(s,t) &= \frac{f\left(\theta\left(\frac{s}{r}\right)\right)}{f\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right)} \\
&= \frac{\left(1-\theta\left(\frac{s}{r}\right)\right)^k \bar{f}\left(\theta\left(\frac{s}{r}\right)\right)}{\left(\left(1-\theta\left(\frac{s}{r}\right)\right) + \left(1-\theta\left(\frac{t}{r}\right)\right)\right)^k \bar{f}\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right)} \\
&= \left(1 + \frac{1-\theta\left(\frac{t}{r}\right)}{1-\theta\left(\frac{s}{r}\right)}\right)^{-k} \frac{\bar{f}\left(\theta\left(\frac{s}{r}\right)\right)}{\bar{f}\left(\theta\left(\frac{s}{r}\right) + \theta\left(\frac{t}{r}\right) - 1\right)}
\end{aligned}$$

If $s = t$, then it is clear that

$$\lim_{r\searrow 0} \nabla_a\phi_r(s,t) = \lim_{r\searrow 0} 2^{-k} \cdot \frac{\bar{f}\left(\theta\left(\frac{s}{r}\right)\right)}{\bar{f}\left(2\theta\left(\frac{s}{r}\right) - 1\right)} = 2^{-k}.$$

If $s < t$, then $s = at$ for some $a \in (0, 1)$. By Lemma 3.1,

$$\lim_{r\searrow 0} \nabla_a\phi_r(s,t) = \left(1 + a^{\frac{1}{k-1}}\right)^{-k} = \left(1 + \left(\frac{s}{t}\right)^{\frac{1}{k-1}}\right)^{-k}.$$

Similarly, when $s > t$, there exists $a \in (0, 1)$ such that $t = as$. Applying again Lemma 3.1, we obtain

$$\lim_{r\searrow 0} \nabla_a\phi_r(s,t) = \left(1 + a^{\frac{1}{1-k}}\right)^{-k} = \left(1 + \left(\frac{t}{s}\right)^{\frac{1}{1-k}}\right)^{-k}.$$

By symmetry, similar formulas can be obtained for $\nabla_b \phi_r(s,t)$. Note that all of the above limits are positive. By following the same arguments as in Theorem 4.1, taking the limit in (4.3) as $r \to 0$, and noting the continuous differentiability of Φ around w^*, we obtain the desired result. □

Observe that the preceding two results do not have any particular assumption on F. On the other hand, for the case when $k = 1$ and $s > t$, we have $\lim_{r \searrow 0} \nabla_a \phi_r(s,t) = 0$, whence, the matrix $B_r(x^*)$ in Eq. (4.3) may be singular when we let $r \to 0$. Consequently, we cannot proceed as in the preceding theorem to prove the nonsingularity of $\nabla \Phi_0(x^*)$. Nevertheless, we obtain a similar result when F is a P-function.

Theorem 4.4. *Let F be a P-function and let $w^* = (x^*, 0) \in \Omega$ be an isolated equilibrium point of (NN2) with x^* in the feasible region of NCP(F). Suppose θ is generated from (3.1) with $f(\theta) = \bar{f}(\theta)(1-\theta)$ where $\bar{f}$ is as described in Theorem 3.1. Suppose further that Φ is continuously differentiable on a neighborhood of w^*. Then, w^* is exponentially stable and $x^* \in \mathrm{SOL}(F)$.*

Proof. Following the same arguments from the proof of Theorem 4.3, we obtain that for any $s, t \geq 0$,

$$\lim_{r \searrow 0} \nabla_a \phi_r(s,t) = \begin{cases} 1 & \text{if } s < t \\ \frac{1}{2} & \text{if } s = t \\ 0 & \text{if } s > t \end{cases} \quad \text{and} \quad \lim_{r \searrow 0} \nabla_b \phi_r(s,t) = \begin{cases} 0 & \text{if } s < t \\ \frac{1}{2} & \text{if } s = t \\ 1 & \text{if } s > t. \end{cases} \tag{4.6}$$

We define three index sets that correspond to the three cases described in the above formulas:

$$I_1 = \{i \,:\, x_i^* < F_i(x^*)\}, \quad I_2 = \{i \,:\, x_i^* = F_i(x^*)\}, \quad \text{and} \quad I_3 = \{i \,:\, x_i^* > F_i(x^*)\}.$$

Let $A_r(x^*)$ and $B_r(x^*)$ be as defined in the proof of Theorem 4.1. Denote by $A_0(x^*)$ and $B_0(x^*)$ the corresponding limits of $A_r(x^*)$ and $B_r(x^*)$ as r decreases to zero, which are both diagonal matrices. Using (4.6), a straightforward calculation implies that the jth column of $A_0(x^*) + \nabla F(x^*) B_0(x^*)$ is given by

$$(A_0(x^*) + \nabla F(x^*) B_0(x^*))_{\cdot j} = \begin{cases} e_j & \text{if } j \in I_1 \\ \frac{1}{2} e_j + \frac{1}{2} (\nabla F(x^*))_{\cdot j} & \text{if } j \in I_2 \\ (\nabla F(x^*))_{\cdot j} & \text{if } j \in I_3, \end{cases}$$

where e_j is the standard unit vector in $\mathbb{R}^n$. From the above formula, it is then clear that

$$\det(A_0(x^*) + \nabla F(x^*) B_0(x^*)) = 2^{-|I_2|} \det(D + E), \tag{4.7}$$

where $D = \mathrm{diag}(v)$,

$$v_i = \begin{cases} 1 & \text{if } i \in I_1 \cup I_2 \\ 0 & \text{if } i \in I_3 \end{cases} \quad \text{and} \quad E_{ij} = \begin{cases} 0 & \text{if } j \in I_1 \\ (\nabla F(x^*))_{ij} & \text{if } j \in I_2 \cup I_3. \end{cases}$$

Using the same formula as (4.4), we have

$$\begin{aligned} \det(D+E) &= \sum_{\Lambda \subseteq \{1,\ldots,n\}} \det(D_\Lambda) \det(E_{\Lambda^c}) \\ &= \det(D_{I_1 \cup I_2}) \det(E_{I_3}) + \sum_{\Lambda \neq I_1 \cup I_2} \det(D_\Lambda) \det(E_{\Lambda^c}) \\ &= \det(E_{I_3}) + \sum_{\Lambda \neq I_1 \cup I_2} \det(D_\Lambda) \det(E_{\Lambda^c}). \end{aligned} \tag{4.8}$$

Since F is a P-function, any principal submatrix of $\nabla F(x^*)$ has a strictly positive determinant. Thus, $\det(E_{I_3}) > 0$ and $\det(E_{\Lambda^c}) \geq 0$ for any $\Lambda \subseteq \{1, \ldots, n\}$. Finally, since $\det(D_\Lambda) \geq 0$ for any $\Lambda \subseteq \{1, \ldots, n\}$, then we get from (4.8) that $\det(D+E) > 0$. Consequently, we have from (4.7) that $\det(\nabla \Phi_0(x^*)) > 0$, i.e. $\nabla \Phi_0(x^*)$ is nonsingular. This completes the proof. □

We now make some comparisons between the artificial neural networks (NN1) and (NN2). First, we note that the first neural network implemented via the procedure described in Remark 4.1 may be more robust than the second one, in the sense that convergence to solution is attained in the first neural network despite starting the algorithm with an initial condition x^0 far from an NCP solution. We illustrate this phenomenon in the next section (see Example 5.4). The drawback of the first neural network, however, is on designing a procedure to decrease the values of r when the sequence of neural networks is implemented. On the other hand, this is not required in the second neural network. The value of the smoothing parameter r ultimately decreases to zero because of the asymptotic stability of the solutions. Furthermore, convergence to equilibrium is often faster for the second neural network.

Table 2
Average convergence time of neural network (NN2) for linear complementarity problems.

Problem	Dimension	α	Average convergence time ($\times 10^{-2}$ s)				
	(n)		θ_1	θ_2	θ_3	θ_4	θ_5
LCP1	10	0.75	2.90	2.64	4.42	3.48	2.81
	25	0.75	2.96	2.67	4.83	3.65	2.87
	50	0.75	2.92	2.64	5.16	3.72	2.87
	75	0.75	3.00	2.90	5.50	4.00	3.00
	100	0.75	3.00	2.95	5.60	4.00	3.00
LCP2	10	0.75	2.44	1.79	5.24	3.73	2.58
	25	0.75	4.07	2.73	8.90	6.19	3.70
	50	0.75	6.82	4.33	14.98	10.27	5.60
	75	0.75	9.80	6.02	21.28	14.50	7.62
	100	0.75	12.53	7.65	27.38	18.58	9.52
LCP3	4	0.20	13.45	11.33	25.95	20.55	17.33

5. Numerical experiments

As mentioned in the introduction, the advantage of the neural network approach is that it provides real-time solutions via hardware implementation. On the other hand, to illustrate the applicability and efficiency of the neural network using the proposed θ functions, we provide in this section results of software implementations of the neural networks.

The main purpose of this section is to provide numerical simulations using the neural networks (NN1) and (NN2) to solve some standard test problems, involving linear and nonlinear complementarity problems. We also compare the performance of the different θ functions in Table 1 by means of performance profiles (see [36]) based on the convergence times of the neural networks.

We used Matlab's ordinary differential equation solver *ode23s* to perform the simulations. Since the parameter ρ is a time-scaling parameter, increasing its value gives better convergence rate. For the simulations, we used $\rho = 1000$. The initial value of the smoothing parameter r is set to $r^0 = 10$ when simulating both the neural networks. The simulation is stopped when at least one among the following conditions is attained: $\|\nabla\Psi(x^k)\| \leq 10^{-6}$ or $\|\Phi(x^k)\| \leq 10^{-9}$.

Example 5.1 (*Linear Complementarity Problems*)**.** We consider three standard linear complementarity problems (LCP), that is, $F : \mathbb{R}^n \to \mathbb{R}^n$ takes the form $F(x) = Ax + b$ where $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. The LCP is denoted by LCP(A, b)

(LCP1, [37]) $A = \begin{pmatrix} 1 & 2 & \cdots & 2 \\ 0 & 1 & \cdots & 2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$ is a upper triangular P-matrix and $b = (-1, \ldots, -1)^T$. This has a unique solution $x^* = (0, \ldots, 0, 1)^T$.

(LCP2, [38]) $A = \begin{pmatrix} 4 & 1 & 0 & \cdots & 0 \\ -2 & 4 & 1 & \cdots & 0 \\ 0 & -2 & 4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 4 \end{pmatrix}$ is a tridiagonal P-matrix and $b = (-1, \ldots, -1)^T$.

(LCP3, [24]) $A = \begin{pmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 2 \\ -1 & 1 & 2 & -2 \\ 1 & -2 & -2 & 2 \end{pmatrix}$ is a P_0-matrix and $b = (1, -1, 1, -1)^T$. The corresponding LCP has infinitely many solutions $x^* = (0, 0, k, k + 0.5)^T$, where $k \geq 0$.

Each of the above LCPs is solved by neural network (NN2) with varying dimensions and using 20 random initial points to illustrate the networks' local stability. The entries of each initial point are generated from the uniform distribution on [0, 20]. For all the LCPs considered, the neural network consistently converged to an NCP solution regardless of the initial condition used. We have verified that the convergence point is indeed a solution by checking that $x_i^k \geq 0$, $F_i(x^k) \geq 0$ and $|x_i^k F_i(x^k)| \leq 10^{-6}$ for all $i = 1, \ldots, n$. We report the average convergence time of the neural network in Table 2.

To compare the performance of $\theta_1, \ldots, \theta_5$ in terms of their convergence time, we present in Fig. 1 the performance profile for Example 5.1. From Fig. 1, we can see that θ_2, which belongs to $\mathscr{F}_1$, has the best performance among all the smoothing functions. Surprisingly, the function θ_5 from $\mathscr{F}_2$ was able to slightly outperform the function θ_1 from $\mathscr{F}_1$, which is one of the functions used in [18]. Among all five θ's, the function θ_3 from $\mathscr{F}_2$ has the worst performance. These observations suggest that the newly obtained functions, specifically θ_2 and θ_5, may have better convergence properties when dealing with P_0-LCPs by means of neural networks.

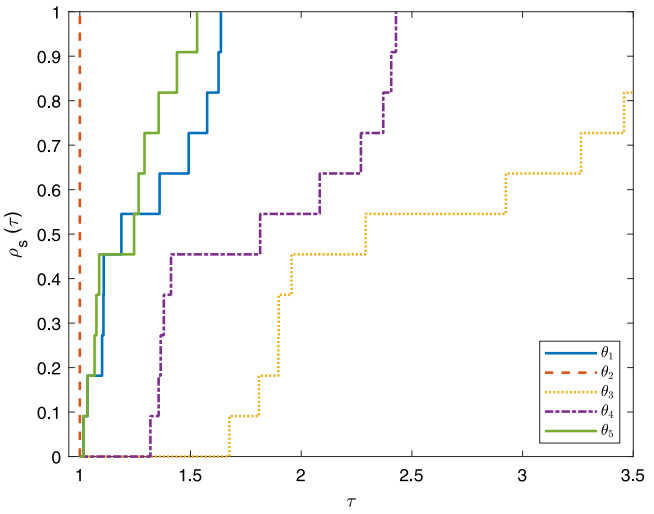

Fig. 1. Performance profile of convergence time for linear complementarity problems.

Example 5.2 (*Nonlinear Complementarity Problems*)**.** We consider monotone, P_0 and non-P_0 nonlinear functions $F : \mathbb{R}^n \to \mathbb{R}^n$, $F = (F_1, \ldots, F_n)$.

(NCP1, [17]) Let $F : \mathbb{R}^3 \to \mathbb{R}^3$ the strictly monotone function

$$F(x) = \begin{pmatrix} x_1 - 2 \\ x_2 - x_3 + x_2^3 + 3 \\ x_2 + x_3 + 2x_3^3 - 3 \end{pmatrix}.$$

The unique solution of NCP(F) is $x^* = (2, 0, 1)^T$.

(NCP2, [24]) Let $F_i(x) = -x_{i-1} + 2x_i - x_{i+1} + b_i(x) + c_i$ for $i = 1, \ldots, n$, where $x_0 = x_{n+1} = 0$, and let $b_i(x) = \arctan(x_i)$ and $c_i = i - \frac{n}{2}$. Here, F is a strongly monotone function.

(NCP3, [39]) Consider F as in NCP2 with $b_i(x) = \frac{1}{3}x_i^3$ and $c_i = (-1)^{i+1}$. This is another strongly monotone NCP.

(NCP4, [15]) Let $F(x) = p \odot \arctan(x) + (AA^T + B)x + q$, where $\odot$ denotes the Hadamard product and p, M, and q are generated as follows: The entries of $A \in \mathbb{R}^{n\times n}$, the skew-symmetric matrix $B \in \mathbb{R}^{n\times n}$, and the vector $p \in \mathbb{R}^n$ are uniformly generated from $(-4, 4)$. The vector $q \in \mathbb{R}^n$ is uniformly generated from $(0, 4)$.

(NCP5, [40]) We follow the construction of F used in [40]. Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a continuously differentiable function, and let $x^* = (0, 1, 0, 1, \ldots)^T \in \mathbb{R}^n$. Define $F : \mathbb{R}^n \to \mathbb{R}^n$ by

$$F_i(x) = \begin{cases} f_i(x) - f_i(x^*) + 1 & \text{if } i \text{ is odd} \\ f_i(x) - f_i(x^*) & \text{otherwise.} \end{cases}$$

It is clear that x^* is a nondegenerate solution of NCP(F). For this example, we take f from [41] given by

$$f_i(x) = \begin{cases} \sum_{j=1}^{n} x_j + x_i - (n+1) & \text{for } i = 1, \ldots, n-1 \\ \prod_{j=1}^{n} x_j - 1 & \text{for } i = n. \end{cases}$$

(NCP6, [42]) Let $F : \mathbb{R}^4 \to \mathbb{R}^4$ be given by

$$F(x) = \begin{pmatrix} 3x_1^2 + 2x_1x_2 + 2x_2^2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_1 + x_2^2 + 10x_3 + 2x_4 - 2 \\ 3x_1^2 + x_1x_2 + 2x_2^2 + 2x_3 + 9x_4 - 9 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

NCP(F) has two solutions: a degenerate solution $x^* = (\sqrt{6}/2, 0, 0, 1/2)^T$ and a non-degenerate solution $x^* = (1, 0, 3, 0)^T$.

Table 3
Average convergence time of neural network (NN2).

Problem	Dimension	α	Average convergence time ($\times 10^{-2}$ s)				
	(n)		θ_1	θ_2	θ_3	θ_4	θ_5
NCP1	3	0.50	3.83	3.45	7.10	5.23	4.33
NCP2	5	0.50	9.08	6.92	17.73	12.48	7.87
	10	0.30	48.35	34.53	103.82	70.15	39.93
	20	0.05	2174	1493	6956	3773	1871
NCP3	5	0.80	4.94	4.30	8.09	6.05	4.60
	10	0.80	8.52	6.50	17.27	11.79	7.20
	20	0.80	14.95	10.50	32.50	21.50	11.50
NCP4	5	0.50	5.37	5.21	7.49	6.13	5.30
	10	0.40	7.86	7.52	10.63	8.88	7.90
	20	0.30	11.30	11.00	17.95	13.85	12.00
NCP5	5	0.70	4.20	4.00	5.10	4.60	4.14
	10	0.65	4.01	3.71	5.21	4.50	4.01
	20	0.65	4.84	4.32	6.77	5.73	4.60
NCP6	4	0.20	20.32	21.28	20.14	19.05	19.65
NCP7	4	0.20	16.97	15.06	35.61	27.02	24.07
NCP8	5	1.00	0.60	0.50	1.38	1.00	0.90
	10	0.95	1.28	1.27	1.61	1.30	1.28

(NCP7, [42]) Let $F : \mathbb{R}^4 \to \mathbb{R}^4$ be given by

$$F(x) = \begin{pmatrix} 3x_1^2 + 2x_1x_2 + 2x_2^2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_1 + x_2^2 + 3x_3 + 2x_4 - 2 \\ 3x_1^2 + x_1x_2 + 2x_2^2 + 2x_3 + 3x_4 - 1 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

The solution is $x^* = (\sqrt{6}/2, 0, 0, 1/2)^T$, which is non-degenerate.

(NCP8, [43]) We consider the problem of finding the Nash–Cournot equilibrium of N-firm noncooperative games, as discussed in [43]. This involves solving NCP(F) with

$$F_i(x) = c_i + (L_i x_i)^{\frac{1}{\beta_i}} - \left(\frac{5000}{\sum_{j=1}^n x_j}\right)^{\frac{1}{\gamma}} + \frac{x_i}{\gamma \sum_{j=1}^n x_j}\left(\frac{5000}{\sum_{j=1}^n x_j}\right)^{\frac{1}{\gamma}}, \quad i = 1, \ldots, n.$$

We consider the case when $n = 5$ and $n = 10$. For $n = 5$, the parameters used are $c = (10, 8, 6, 4, 2)^T$, $L = (5, 5, 5, 5, 5)^T$, $\beta = (1.2, 1.1, 1, 0.9, 0.8)^T$ and $\gamma = 1.1$. The approximate solution to this NCP is $x^* = (15.4293, 12.4986, 9.6635, 7.1651, 5.1326)^T$. For $n = 10$, the parameters used are $c = (5, 3, 8, 5, 1, 3, 7, 4, 6, 3)^T$, $L = (10, \ldots, 10)^T$, β =(1.2,1,0.9,0.6,1.5,1,0.7,1.1, 0.95,0.75)^T and $\gamma = 1.2$. The approximate solution to this NCP is x^* =(7.4415,4.0978,2.5906, 0.9354,17.9490,4.0978,1.3047,5.5901,3.2222, 1.6771)^T.

Similarly, we solved the NCPs using neural network (NN2) through 20 random initial conditions and summarize the average convergence time of successful simulations in Table 3. Similar to Example 5.1, the neural network trajectories from all of the random starting points converged to a point in SOL(F). The performance profile for these NCP test problems is shown in Fig. 2, from which we can infer that functions from $\mathscr{F}_1$ dominate in terms of convergence time.

However, observe that θ_1 and θ_5 have almost the same performance in solving the NCPs. The same observations can be obtained from Fig. 3, which depicts the performance profile of all the test problems considered in Examples 5.1 and 5.2.

The above simulations show that functions from $\mathscr{F}_1$ have good convergence properties. Despite the better convergence time, however, one must carefully choose the parameter α if a class $\mathscr{F}_1$ function is used in order to avoid ill-conditioning. The parameter α plays a significant role in achieving successful simulations for (NN2) since this parameter controls the rate of decrease of the value of r. Theoretically, α can be set to any positive real number and we can still obtain stability as discussed in Section 3. However, from a practical point of view, a very large decrease in r may result in failed simulations as the trajectories get closer to the solution due to ill-conditioning effects when r approaches zero. In the numerical experiments, we found that a higher value of α can be used for θ's that belong to $\mathscr{F}_2$, since these functions approach 1 at a rate less than that of a function with exponential growth rate. For those that belong to $\mathscr{F}_1$, a lower value of α is often required to achieve successful simulations. As a result, we found that it is easier to control the parameter α when functions from $\mathscr{F}_2$ are used. Consequently, the convergence time can be improved and in fact, can be at par with (or better than) functions from $\mathscr{F}_1$ where a lower value of α is used. We illustrate this in the following example.

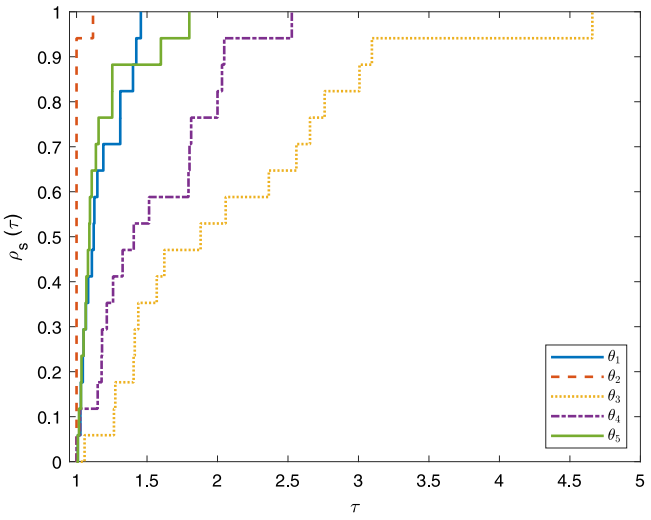

Fig. 2. Performance profile of convergence time for NCPs.

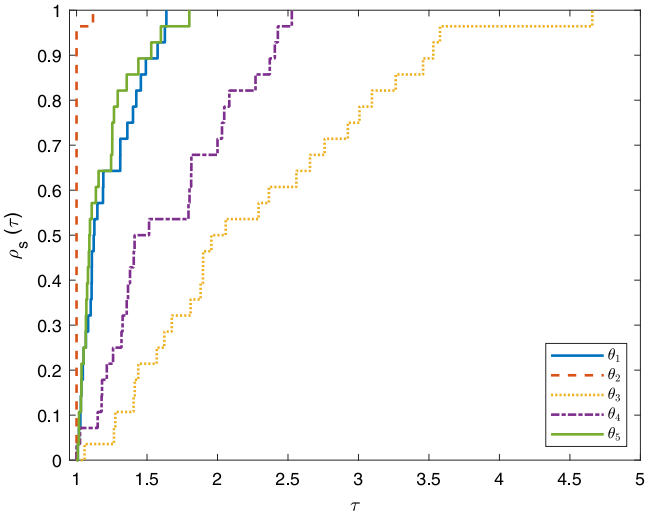

Fig. 3. Performance profile of convergence time for complementarity problems.

Example 5.3. Consider the trajectories $w(t) = (x(t), r(t))$ of neural network (NN2) for NCP7. Fig. 4 shows the convergence to zero of the error term $\|x(t) - x^*\|$ with $\alpha = 1$ for class $\mathscr{F}_2$ functions and $\alpha = 0.5$ for class $\mathscr{F}_1$ functions. This example shows that the convergence time of $\mathscr{F}_2$-functions can be improved by using a higher value of α. In particular, we see from this example that θ_3, θ_4 and θ_5 all have better convergence rates than the functions θ_1 and θ_2.

Our numerical experiments involving the above NCPs also suggest the insensitivity of the neural network (NN2) to initial conditions, especially when P_0-functions are involved. However, if F is not a P_0-function, the neural network may fail to converge to an NCP solution for some initial points and for some $\alpha > 0$. In such cases, we can implement neural network (NN1) via Remark 4.1, where the parameter r is reduced according to the simple rule $r^{k+1} = \beta r^k$, where $\beta \in (0, 1)$.

Example 5.4. We revisit NCP7 and use (NN2) equipped with $\mathscr{F}_2$ functions to solve the problem with initial condition $x^0 = (300, 600, 300, 150)^T$. From Figs. 5(i)–(iii), the trajectories of (NN2) converged to a stationary point which is not an optimal solution. Putting together the trajectories when a sequence of (NN1) is implemented, we obtain Figs. 5(iv)–(vi) which illustrate the convergence of (NN1) to the NCP solution. Observe that among the three functions, θ_5 has the fastest convergence time. Meanwhile, we show in Fig. 6 that the neural networks based on FB function [21] and the first symmetrization of the NR function [19] both fail to converge to the solution.

In the following example, we compare the neural network (NN2) with the neural networks in [20,21] based on the (generalized) Fischer–Burmeister functions.

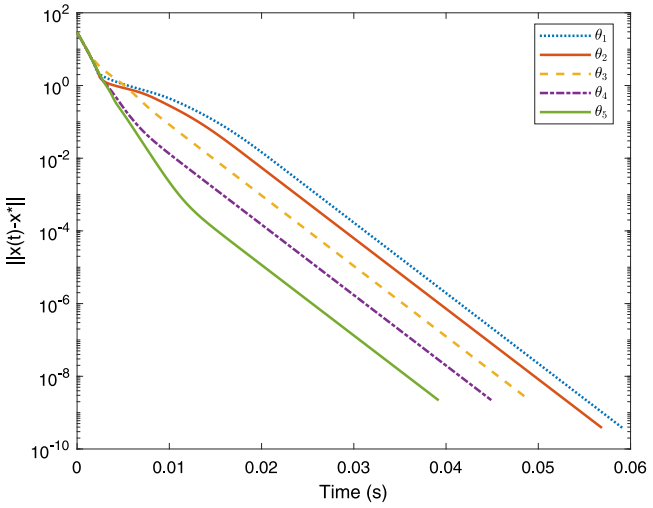

Fig. 4. Comparison of convergence rates of $\|x(t) - x^*\|$ for NCP7 using $\alpha = 0.5$ for functions from $\mathscr{F}_1$ (θ_1, θ_2) and $\alpha = 1$ for functions from $\mathscr{F}_2$ $(\theta_3, \theta_4, \theta_5)$, where $x^0 = (15, 15, 15, 15)^T$ and $r^0 = 10$.

Example 5.5 (*Comparison with Other Neural Networks*)**.** The performance profile shown in Fig. 7 depicts the comparison of convergence times of (NN2), FB-based NN and generalized FB-based NN when used to solve the LCPs and NCPs given in Examples 5.1 and 5.2. We solved all the problems using the neural networks starting from five random initial conditions, then obtain the average convergence time. For functions in class $\mathscr{F}_2$, we used a higher value of α to maximize the inherent properties of these functions in solving complementarity problems. The parameter values used in the simulations are specified in Table 4. From the performance profile, it can be observed that θ_5 has the best convergence time among all the functions, followed by θ_4 and θ_3. The FB and generalized FB neural networks have almost the same performance in solving the complementarity problems. Class $\mathscr{F}_1$ functions, on the other hand, have the worst performance.

As our last example, we illustrate a simple application of our neural network in solving an obstacle problem.

Example 5.6 (*Obstacle Problem [32]*)**.** In this problem, we find a curve joining the boundary of a domain with an obstacle g and minimal curvature f. Mathematically, this can be formulated as finding a function u such that

$$u(t) - g(t) \geq 0, \qquad \ddot{u}(t) - f(t) \geq 0 \qquad \text{and} \qquad \langle u(t) - g(t), \ddot{u}(t) - f(t)\rangle = 0$$

where ü denotes the second-order derivative of u. Using the second-order centered finite difference to approximate the second-order derivative of u, we get a discrete version on an equispaced grid $t_i = ih, i = 1, \ldots, n$, which can be formulated as an NCP: Find $\hat{u} \in \mathbb{R}^n$ such that

$$\hat{u} - \hat{g} \geq 0, \qquad D\hat{u} - \hat{f} \geq 0, \qquad \text{and} \qquad \langle \hat{u} - \hat{g}, D\hat{u} - \hat{f}\rangle = 0$$

where $\hat{f}, \hat{g} \in \mathbb{R}^n, \hat{f}_i = f(t_i), \hat{g}_i = g(t_i)$, and

$$D = \begin{pmatrix} \frac{2}{h^2} & -\frac{1}{h^2} & & \\ -\frac{1}{h^2} & \ddots & \ddots & \\ & \ddots & \ddots & -\frac{1}{h^2} \\ & & -\frac{1}{h^2} & \frac{2}{h^2} \end{pmatrix}.$$

Letting $x = \hat{u} - \hat{g}$, then the above problem is equivalent to NCP(F) with $F(x) = Dx + D\hat{g} - \hat{f}$. As in [32], we consider the obstacle function given by $g(t) = \max(0.8 - 20(t - 0.2)^2, \max(1 - 20(t - 0.75)^2, 1.2 - 30(t - 0.41)^2))$ and the minimum curvature required of the curve is $f(t) = 1$, and we use discretization with $n = 50$. In Fig. 8, we present the approximate curve obtained by (NN2) using the smoothing function θ_1. We note that exactly the same solution can be obtained using other theta functions $\theta_2, \ldots, \theta_5$, or using the FB-based or GFB-based neural networks.

To close this section, we summarize our observations from the above numerical experiments. We also mention some comparison of the present neural network with other networks used to deal with NCPs.

- For the same value of α, functions from class $\mathscr{F}_1$ have the fastest convergence time to solve complementarity problems. In particular, θ_2 performs best among all the five functions considered, while θ_3 has the worst performance.

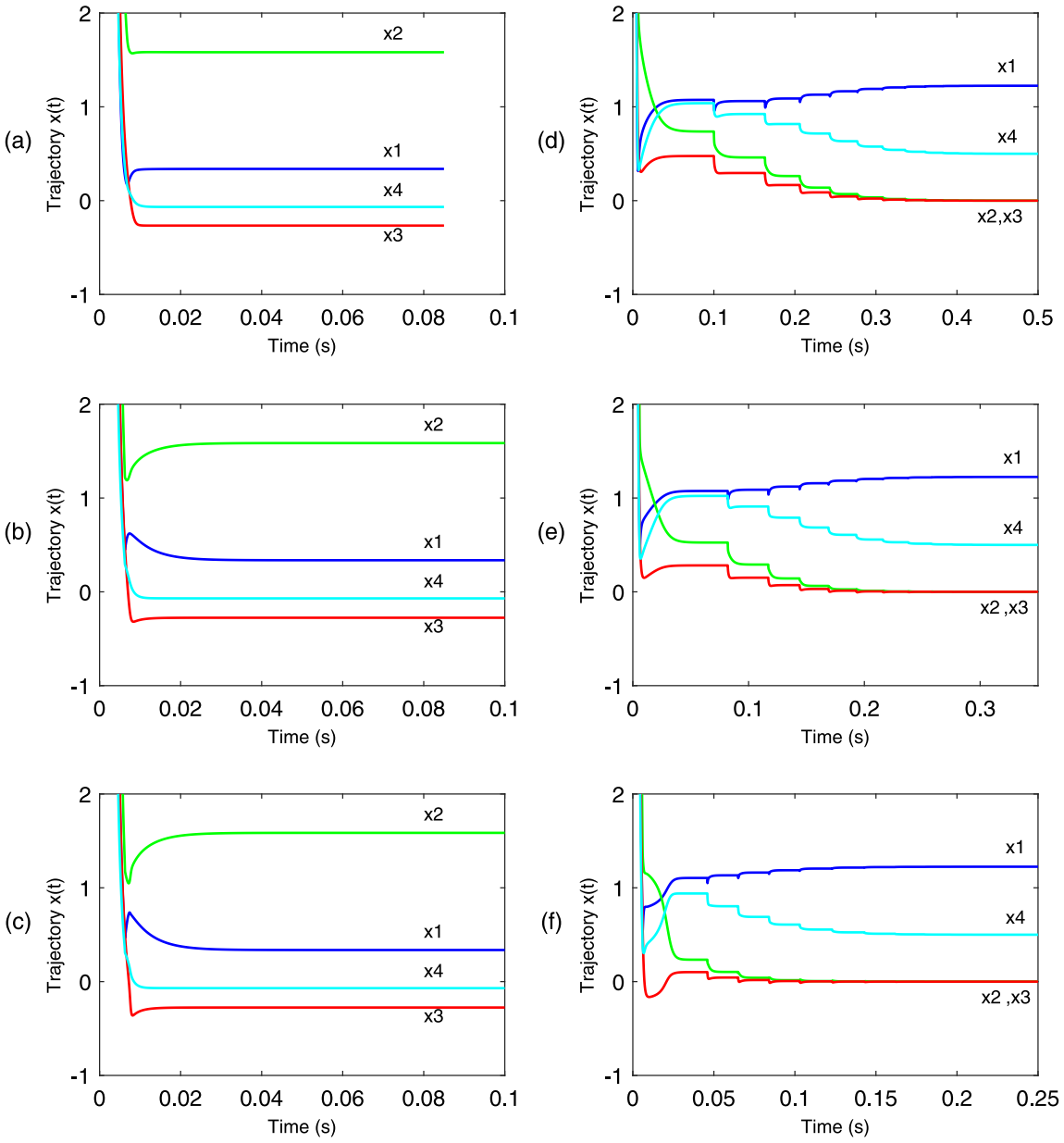

Fig. 5. Trajectories starting at $x^0 = (300, 600, 300, 150)^T$ of the neural network using (NN1) and (NN2) with functions from $\mathscr{F}_2$: θ_3, θ_4 and θ_5 were used for Figs. (i) & (iv), (ii) & (v) and (iii) & (vi), respectively. Figs. (i)–(iii) show the trajectories of (NN2) with $\alpha = 1$ which all converged to an equilibrium point which is not the NCP solution, while Figs. (iv)–(vi) show the trajectories of (NN1) with $\beta = 0.7$, which all converge to $x^* = (\sqrt{6}/2, 0, 0, 1/2)^T$.

- The performance of the neural networks can be improved by choosing a larger value of α, the parameter that controls the decrease of the perturbation parameter r. While this is theoretically acceptable, numerical experiments indicate some ill-conditioning problems when a very high value of α is used. Specifically, functions from $\mathscr{F}_1$ are more sensitive to larger values of α, while class $\mathscr{F}_2$ functions allow more flexibility. In fact, by choosing a higher value of α, functions from $\mathscr{F}_2$ can potentially outperform functions from class $\mathscr{F}_1$.
- The neural network (NN2) is not very sensitive to initial conditions, particularly when F is a P_0-function. However, Example 5.4 demonstrates that (NN2) may not converge to an NCP solution for some initial conditions when F is not a P_0-function. In this case, (NN1) is a good alternative but may take longer convergence time due to the sequence of neural networks that needed to be simulated.
- We see from Example 5.5 that functions from class $\mathscr{F}_2$ have great potential to outperform the well-known FB-based and generalized FB-based NNs.

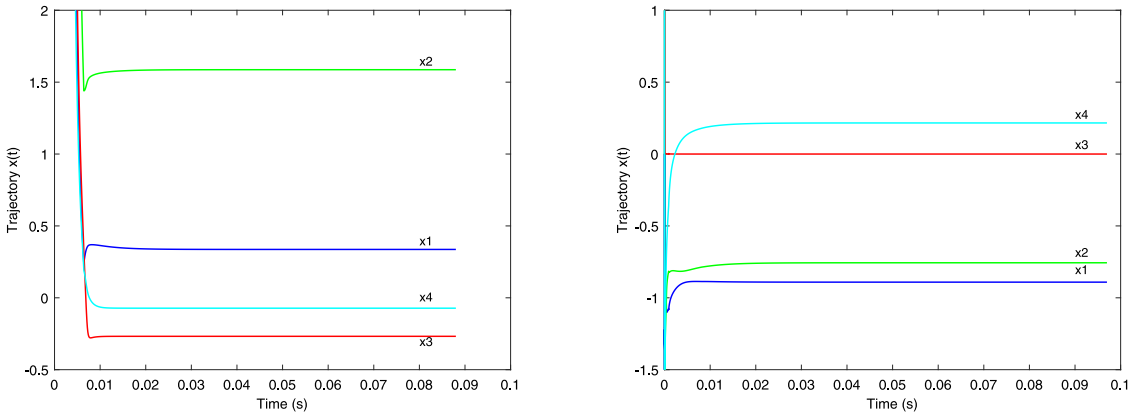

Fig. 6. Trajectories starting at $x^0 = (300, 600, 300, 150)^T$ of the neural network based on FB function [21] and the first symmetrization of the natural residual function [19].

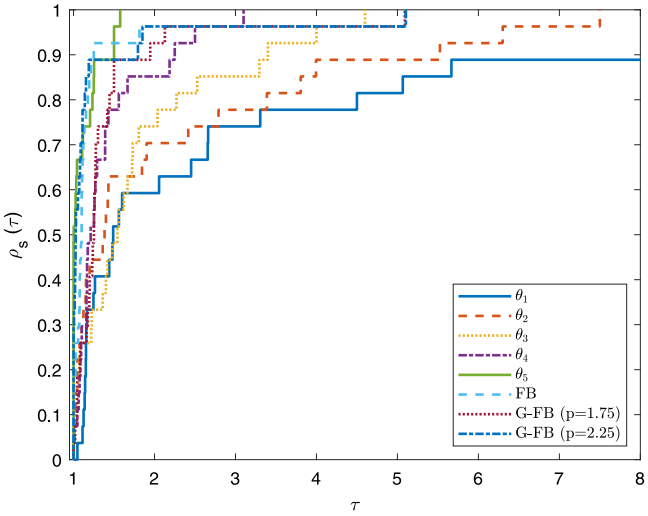

Fig. 7. Performance profile of convergence time of (NN2) and the neural networks based on FB and generalized FB function for solving complementarity problems.

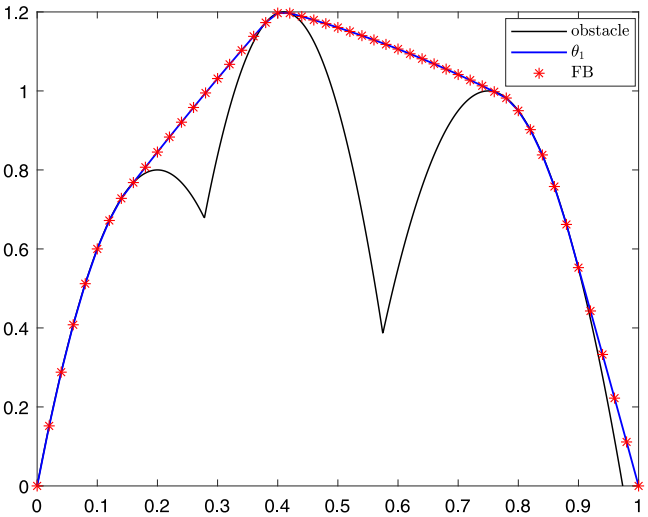

Fig. 8. Numerical solution of the obstacle problem using neural networks based on θ_1 and the Fischer–Burmeister function in [21].

Table 4
Average convergence time of neural network (NN2).

Problem	Dim	$(\alpha^{(1)}, \alpha^{(2)})$	Average convergence time ($\times 10^{-2}$ s)							
	(n)		θ_1	θ_2	θ_3	θ_4	θ_5	FB	GFB1[a]	GFB2[b]
LCP1	10	(0.75, 1.00)	2.94	2.66	3.60	2.94	2.54	2.88	3.18	2.72
	25	(0.75, 1.00)	2.88	2.64	3.72	2.94	2.52	2.96	3.28	2.74
	50	(0.75, 1.00)	2.98	2.68	4.04	3.08	2.62	2.86	3.08	2.70
	75	(0.75, 1.00)	3.00	3.00	4.50	3.30	2.60	3.00	3.30	3.00
	100	(0.75, 1.00)	3.00	2.90	4.50	3.40	2.70	3.00	3.40	3.00
LCP2	10	(0.75, 2.00)	2.38	1.74	1.30	0.84	0.72	0.86	1.02	0.80
	25	(0.75, 2.00)	4.08	2.74	1.82	1.20	0.72	0.90	1.04	0.82
	50	(0.75, 2.00)	6.90	4.42	2.72	1.80	1.00	0.88	1.02	0.80
	75	(0.75, 2.00)	9.90	6.30	4.00	2.50	1.50	1.10	1.50	1.00
	100	(0.75, 2.00)	12.50	7.50	4.60	3.10	1.50	1.10	1.50	1.10
LCP3	4	(0.40, 1.00)	6.40	5.60	5.60	4.30	4.00	4.30	4.80	4.10
NCP1	3	(0.50, 1.00)	3.90	3.40	3.40	2.90	2.50	2.80	2.90	2.70
NCP2	5	(0.50, 1.00)	9.20	6.90	7.70	5.40	3.80	4.20	4.80	3.90
	10	(0.30, 1.00)	47.60	34.10	21.90	19.60	17.90	18.40	19.40	17.90
	20	(0.05, 0.20)	2366	1582	728.7	649.0	576.0	467.0	476.0	467.0
NCP3	5	(0.80, 1.00)	4.90	4.30	6.80	5.26	4.20	4.50	5.02	4.28
	10	(0.80, 1.00)	8.50	6.50	13.40	9.20	6.02	6.38	7.30	5.90
	20	(0.80, 1.00)	14.60	10.10	23.40	15.52	8.90	7.64	8.74	7.10
NCP4	5	(0.50, 0.70)	5.88	5.66	5.00	4.94	4.84	4.72	4.92	4.82
	10	(0.40, 0.70)	11.90	11.50	8.60	9.74	8.04	8.50	8.66	8.48
	20	(0.30, 0.70)	14.00	13.00	9.80	10.20	10.40	9.40	20.00	17.40
NCP5	5	(0.70, 1.00)	4.40	4.20	7.24	5.84	6.62	7.62	8.18	7.54
	10	(0.65, 1.00)	4.10	3.60	4.34	3.92	3.68	3.58	3.72	3.56
	20	(0.65, 0.90)	4.83	4.35	5.20	4.48	4.08	4.00	4.14	3.92
NCP6	4	(0.20, 0.40)	20.34	21.32	7.68	7.64	8.52	38.95	38.90	39.00
NCP7	4	(0.20, 1.00)	16.74	14.86	3.72	4.80	4.48	4.52	4.66	4.42
NCP8	5	(1.00, 3.00)	0.526	0.478	0.448	0.420	0.416	0.426	0.434	0.422
	10	(0.95, 2.00)	1.214	1.212	1.120	1.148	1.126	1.0820	1.080	1.086

[a]Using $p = 1.75$.
[b]Using $p = 2.25$.

- Consider NCP(F) with $F : \mathbb{R}^3 \to \mathbb{R}^3$ given by $F(x) = (x_1, -x_2, -x_3)^T$. The neural network used in [28] does not approach the unique NCP solution $x^* = (0, 0, 0)^T$ as mentioned in [20]. This problem can be easily solved by our neural networks.

6. Conclusions

In this paper, we introduced two families of smoothing functions that can be used to solve nonlinear complementarity problems. Sufficient conditions on classifying the smoothing functions were also provided, thus extending and refining the results presented in [18]. Moreover, we presented a simple way to generate functions from these families. Such construction framework provides further insights on the relationship between the two families $\mathscr{F}_1$ and $\mathscr{F}_2$. Loosely speaking, we may view $\mathscr{F}_1$ as the limit case of the family $\mathscr{F}_2$. The construction method had also been useful in establishing some convergence results for the neural networks formulated based on the smoothing approach.

Two neural networks were constructed using the smooth perturbations of the natural residual function. The second neural network (NN2) was shown to be very efficient in solving several test problems. When class $\mathscr{F}_2$ functions are used in designing (NN2), we proved a strong result on the exponential stability of equilibrium points which corresponds to NCP solutions, where the only assumption on F is differentiability. For class $\mathscr{F}_1$ functions, a similar result also holds but under the hypothesis that F is a P-function. From a theoretical point of view, the neural network (NN2) has better convergence properties compared to some neural networks [24,28,29], which requires monotonicity to achieve Lyapunov stability. As was shown in our numerical experiments, (NN2) is efficient even in solving non-P_0-NCPs. Moreover, (NN2) is not sensitive to varying initial conditions, which can be encountered in some neural networks (for instance, [19,28]). On the other hand, (NN1) is even more robust compared to (NN2). We have established several sufficient conditions to achieve local and global asymptotic stability of (NN1), as well as exponential stability of the neural network. The implementation of (NN1) takes longer convergence time when compared to (NN2). However, this neural network is less sensitive to initial conditions and can achieve convergence to NCP solutions in cases where (NN2), FB-based, and GFB-based neural networks fail.

The neural network approach presented in this paper suggests how to choose an appropriate smoothing function. Class $\mathscr{F}_1$ functions, in general, provide faster convergence time for the neural networks. Intuitively, such is expected since class

$\mathscr{F}_1$ functions yield smoothing functions for the natural residual function. However, the disadvantage of these smoothing functions is their tendency to increase the propensity to encounter ill-conditioning effects. In such cases, the tuning parameter α can be set to a lower value to achieve successful simulations. On the other hand, class $\mathscr{F}_2$ functions have slower convergence time but are less susceptible to numerical difficulties involving ill-conditioning. Thus, the parameter α can be tuned to a higher value which results in faster convergence time. In effect, class $\mathscr{F}_2$ functions seem to be more preferable to be used in solving NCPs because of the flexibility of α-value for this class. In fact, the improved convergence time of class $\mathscr{F}_2$ functions when larger α-value is used often leads to better convergence time when compared to $\mathscr{F}_1$ functions. For other problems where ill-conditioning is encountered, we recommend to choose a smoothing function with higher value of k as defined in Theorem 3.1 in designing a neural network.

Because of the significance of α, further studies are required to obtain an adaptive form of this parameter, that is, a non-constant form of α can be considered. For instance, the numerical experiments suggest that α should be dependent on the current iterate x^k, the dimension n, and also on the smoothing function θ that is used. We leave this for our future research directions.

References

[1] F. Facchinei, J.-S. Pang, Finite-Dimensional Variational Inequalities and Complementarity Problems, Volumes I and II, Springer-Verlag, New York, 2003.
[2] M.C. Ferris, O.L. Mangasarian, J.-S. Pang, Complementarity: Applications, Algorithms and Extensions, Kluwer Academic Publishers, Dordrecht, 2001.
[3] M.C. Ferris, J.-S. Pang, Engineering and economic applications of complementarity problems, SIAM Rev. 39 (1997) 669–713.
[4] F. Facchinei, J. Soares, A new merit function for nonlinear complementarity problems and a related algorithm, SIAM J. Optim. 7 (1997) 225–247.
[5] C. Kanzow, Nonlinear complementarity as unconstrained optimization, J. Optim. Theory Appl. 88 (1996) 139–155.
[6] O.L. Mangasarian, M.V. Solodov, Nonlinear complementarity as unconstrained and constrained minimization, Math. Program. 62 (1993) 277–297.
[7] B. Xiao, P.T. Harker, A nonsmooth Newton method for variational inequalities, I: theory, Math. Program. Series A 65 (1994) 151–194.
[8] B. Xiao, P.T. Harker, A nonsmooth Newton method for variational inequalities, II: numerical results, Math. Program. Ser. A 65 (1994) 195–216.
[9] N. Yamashita, M. Fukushima, Modified Newton methods for solving a semismooth reformulation of monotone complementarity problems, Math. Program. 76 (1997) 469–491.
[10] Z.H. Huang, Locating a maximally complementary solutions of the monotone NCP by using non-interior-point smoothing algorithm, Math. Methods Oper. Res. 61 (2005) 41–55.
[11] D. Sun, A regularization Newton method for solving nonlinear complementarity problems, Appl. Math. Optim. 40 (1999) 315–339.
[12] P.T. Harker, J.-S. Pang, Finite dimensional variational inequality and nonlinear complementarity problem: a survey of theory, algorithms and applications, Math. Program. 48 (1990) 161–220.
[13] B. Chen, P.T. Harker, Smooth approximations to nonlinear complementarity problems, SIAM J. Optim. 7 (2) (1997) 403–420.
[14] C. Chen, O.L. Mangasarian, A class of smoothing functions for nonlinear and mixed complementarity problems, Comput. Optim. Appl. 5 (1996) 97–138.
[15] L. Fang, A new one-step smoothing Newton method for nonlinear complementarity problem with P_0-function, Appl. Math. Comput. 216 (2010) 1087–1095.
[16] H.D. Qi, L.Z. Liao, A smoothing Newton method for general nonlinear complementarity problems, Comput. Optim. Appl. 17 (2000) 231–253.
[17] H. Yu, D. Pu, Smoothing levenberg-marquardt method for general nonlinear complementarity problems under local error bound, Appl. Math. Model 35 (2011) 1337–1348.
[18] M. Haddou, P. Maheux, Smoothing methods for nonlinear complementarity problems, J. Optim. Theory Appl. 160 (2014) 711–729.
[19] J.H. Alcantara, J.-S. Chen, Neural networks based on three classes of NCP-functions for solving nonlinear complementarity problems, Neurocomputing 359 (2019) 102–113.
[20] J.-S. Chen, C.H. Ko, S.H. Pan, A neural network based on generalized Fischer-Burmeister function for nonlinear complementarity problems, Inform. Sci. 180 (2010) 697–711.
[21] L.Z. Liao, H.D. Qi, L. Qi, Solving nonlinear complementarity problems with neural networks: a reformulation method approach, J. Comput. Appl. Math. 131 (2001) 342–359.
[22] J.J. Hopfield, D.W. Tank, Neural computation of decision in optimization problems, Biol. Cybernet. 52 (1985) 141–152.
[23] D.W. Tank, J.J. Hopfield, Simple neural optimization networks: an a/d converter, signal decision circuit, and a linear programming circuit, IEEE Trans. Circuits Syst. 33 (1986) 533–541.
[24] C. Dang, Y. Leung, X. Gao, K. Chen, Neural networks for nonlinear and mixed complementarity problems and their applications, Neural Netw. 17 (2004) 271–283.
[25] X. Hu, J. Wang, Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network, IEEE Trans. Neural Netw. 17 (2006) 1487–1499.
[26] X. Hu, J. Wang, A recurrent neural network for solving a class of general variational inequalities, IEEE Trans. Syst. Man Cybern.-B 37 (2007) 528–539.
[27] M.P. Kennedy, L.O. Chua, Neural network for nonlinear programming, IEEE Trans. Circuits Syst. 35 (1988) 554–562.
[28] Y. Xia, H. Leung, J. Wang, A projection neural network and its application to constrained optimization problems, IEEE Trans. Circuits Syst. I 49 (2002) 447–458.
[29] Y. Xia, H. Leung, J. Wang, A general projection neural network for solving monotone variational inequalities and related optimization problems, IEEE Trans. Neural Netw. 15 (2004) 318–328.
[30] M. Yashtini, A. Malek, Solving complementarity and variational inequalities problems using neural networks, Appl. Math. Comput. 190 (2007) 216–230.
[31] M.S. Gowda, M.A. Tawhid, Existence and limiting behaviors of trajectories associated with P_0-equations, Comput. Optim. Appl. 12 (1999) 229–251.
[32] L. Abdallah, M. Haddou, T. Migot, Solving absolute value equation using complementarity and smoothing functions, J. Comput. Appl. Math. 327 (2018) 196–207.
[33] R.K. Miller, A.N. Michel, Ordinary Differential Equations, Academic Press, New York, 1982.
[34] H. Khalil, Nonlinear Systems, Prentice-Hall, Hoboken, N.J., 2002.
[35] R.W. Cottle, J.-S. Pang, R.E. Stone, The Linear Complementarity Problem, Academic Press, New York, 1982.

[36] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2002) 201–213.
[37] C. Kanzow, Some noninterior continuation methods for linear complementarity problems, SIAM J. Matrix Anal. Appl. 17 (1996) 851–868.
[38] B.H. Ahn, Iterative methods for linear complementarity problems with upperbounds on primary variables, Math. Program. 26 (1983) 295–315.
[39] C. Huang, S. Wang, A power penalty approach to a nonlinear complementarity problem, Oper. Res. Lett. 38 (2010) 72–76.
[40] M.A.G. Ruggiero, J.M. Martinez, S.A. Santos, Solving nonsmooth equations by means of quasi-Newton methods with globalization, in: Recent Advances in Nonsmooth Optimization, World Scientific, Singapore, 1995, pp. 121–140.
[41] E. Spedicato, Z. Huang, Numerical experience with Newton-like methods for nonlinear algebraic systems, Computing 58 (1997) 69–89.
[42] M. Kojima, S. Shindo, Extensions of Newton and quasi-Newton methods to systems of PC^1 equations, J. Oper. Res. Soc. Jpn. 29 (1986) 352–374.
[43] P.T. Harker, Accelerating the convergence of the diagonalization and projection algorithms for finite-dimensional variational inequalities, Math. Program. 48 (1990) 29–59.