Available online at www.tjnsa.com
J. Nonlinear Sci. Appl. 0 (2016), 00–00
Research Article

Discrete-time projection neural network methods for computing the solution of variational inequalities

Liping Zhang[a], Shu-Lin Wu[a,*]

[a]*School of Science, Sichuan University of Science and Engineering, Zigong 643000, China.*

Abstract

Neural networks are useful tools to solve mathematical and engineering problems. By using the implicit-explicit-θ method and the method proposed recently by Mohamad to discretize the continuous-time neural networks, we formulate two classes of discrete-time analogues to solve a system of variational inequalities. By adopting Lyapunov function and Razumikhin-type techniques, exponential stability of the discrete neural networks are established in terms of linear matrix inequalities (LMIs). Several numerical experiments are performed to compare the convergence rates of the proposed discrete neural networks and it is shown that: a) all of the discrete neural networks converge faster as the step size becomes larger; b) the discrete neural networks derived by the semi-implicit Euler method performs best.

Keywords: neural networks, linear matrix inequalities (LMIs), variational inequalities, discretization
2010 MSC: 34K45, 39A11, 92B20.

1. Introduction

Neural networks are very powerful tools to solve mathematical and engineering problems, such as signal processing, associative memory, pattern recognition, combination optimization and so on. In recent years, there are increasing interests that apply neural networks to solve variational inequality problem. In [27], Zeng and Liao studied the following variational inequality:

$$\textit{Finding an element } x \in \Omega \textit{ such that }: \ (u-x)^{\top}(Mx+q) \geq 0, \forall u \in \Omega, \tag{1.1}$$

where Ω is a nonempty bounded and closed convex subset, $M = (M_{ij})_{n\times n}$ is a real $n \times n$ matrix and $q = (q_1, q_2, \cdots, q_n)^{\top} \in R^n$. For (1.1), the authors in [24] suggests the following projection neural network

$$\frac{du}{dt} = -u + (E_n - M)F_{\Omega}(u) - q$$

*Corresponding author (Shu-Lin Wu)
Email address: wushulin_ylp@163.com (Shu-Lin Wu)

Received 00-00-00

where E_n denotes the $n \times n$ identity matrix and $F_\Omega : R^n \to \Omega$ is a projection operator:

$$F_\Omega(u) = \arg\min_{v\in\Omega} \|u - v\|,$$

with $\|\cdot\|$ being the l_2-norm in $\mathbb{R}^n$.

Other projection neural networks for solving monotone variational inequalities and constrained optimization problems can be found in [25, 6, 11, 21, 21] and the references therein. Particularly, Xia and Wang give a general projection neural network in [25], which includes existing neural networks for optimization, such as the primal-dual neural networks and the dual neural networks, as special cases. As pointed out by Hu and Wang [6], the projection neural networks can also be used to solve pseudomonotone variational inequalities and related pseudoconvex optimization problems.

Most recently, Lan and Cui [12] investigated the neural network method for solving the following variational inequalities:

$$\textit{Finding} \quad (x,y) \in \Omega_1 \times \Omega_2 \ \textit{such that}: \quad \begin{cases} (\alpha - x)^\top (x - y + Ny + p) \geq 0, \forall \alpha \in \Omega_1, \\ (\beta - y)^\top (y - x + Mx + q) \geq 0, \forall \beta \in \Omega_2, \end{cases} \tag{1.2}$$

where $\Omega_i \subset \mathbb{R}^n$ $(i = 1, 2)$ is a nonempty bounded and closed convex subset, $M = (M_{ij})_{n\times n}$ and $N = (N_{ij})_{n\times n}$ are real matrixes, $p = (p_1, p_2, \ldots, p_n)^\top, q = (q_1, q_2, \ldots, q_n)^\top$ are real vectors.

We note that the variational inequalities (1.2) have numerous applications such as economic equilibrium modeling, traffic networks equilibrium modeling, and structural analysis; see, for example, [8, 4]. In [12], the authors transmuted the solution of (1.2) to an equilibrium point of neural networks and analyzed the stability and convergence rate of the neural networks. The results presented in [12] generalize and improve the existing works in the literature. While most neural networks mentioned above are studied in continuous-time level, there exist at least two aspects that make the analysis of the discrete-time analogues important. First, to accelerate the process and reduce the cost of electric circuit design of the continuous-time neural networks, we need computer simulation to explore how the properties of the continuous-time neural networks depend on the involved parameters. To this end, discrete-time analogues are essentially important (see the same or similar opinion in [5, 14, 15, 17, 16, 18, 19, 26, 20, 22]). Second, by some suitable construction, a discrete-time neural network will approach to the same equilibrium point of its continuous-time counterpart, but the former is more flexible to be used, since it can be implemented conveniently by both computers and electric circuit. We note that, discrete-time neural networks can be implemented without electric circuits and just with a computer, since the final form is a difference equation and therefore the equilibrium point can be calculated step by step with a computer. Even for the implementation with electric circuit, the cost of discrete-time neural networks is less than that of the continuous-time ones, since for the latter we need (additionally) the device of digital integrator (or differentiator).

In this paper, we construct two types of discrete-time neural networks to solve the variational inequalities (1.2). We utilize the implicit-explicit-θ (IMEX-θ) method and the one proposed by Mohamad *et al.*[5, 14, 15, 17, 16, 18] to discretize the continuous-time neural networks introduced in [12], and then by using Lyapunov function and Razumikhin-type techniques we investigate exponential stability of the derived discrete-time analogues. The sufficient stability criteria are established in terms of linear matrix inequalities (LMIs), which can be solved numerically and very efficiently using the interior point algorithms [1].

The reminder of this paper is organized as follows: in Section 2, we introduce the discrete-time analogues discussed in this paper. In Section 3, two sufficient conditions for exponential stability are established for the discrete-time analogues. In Section 4, several numerical examples are given to show the usefulness of our results. The comparison between different discrete-time analogues with respect to convergence rates is also given in this section. Finally, Section 5 concludes the paper with some remarks.

2. Problem description and preliminaries

Let the subsets Ω_i in (1.2) take the following form:

$$\Omega_i = \left[a_1^{(i)}, b_1^{(i)}\right] \times \left[a_2^{(i)}, b_2^{(i)}\right] \times \cdots \times \left[a_n^{(i)}, b_n^{(i)}\right], i = 1, 2.$$

Then we define the projection functions $F_{\Omega_i}(s) = (F_{\Omega_i}(s_1), F_{\Omega_i}(s_2), \ldots, F_{\Omega_i}(s_n))^\top$ as

$$F_{\Omega_i}(s_j) = \begin{cases} b_j^{(i)}, & \text{if } s_j > b_j^{(i)}, \\ s_j, & \text{if } a_j^{(i)} \le s_j \le b_j^{(i)}, \\ a_j^{(i)}, & \text{if } s_j < a_j^{(i)}, \end{cases} \tag{2.1}$$

where $i = 1, 2$ and $j = 1, 2, \ldots, n$. It is easy to see that the function $F_{\Omega_i}(\cdot)$ is monotonically increasing and satisfies the following Lipschitz condition:

$$0 \le \frac{F_{\Omega_i}(s_1) - F_{\Omega_i}(s_2)}{s_1 - s_2} \le 1, \quad \forall s_1, s_2 \in \mathbb{R} \text{ and } s_1 \ne s_2. \tag{2.2}$$

2.1. Continuous-time neural network method for the variational inequalities

We first review an important lemma which transmutes the solution of the variational inequalities (1.2) to an equilibrium point of the neural networks.

Lemma 2.1 (**Lan and Cui [12]**). *The following statements are mutually equivalent:*

1. (x^*, y^*) *is a solution of the variational inequalities* (1.2)*;*
2. $x^* = F_{\Omega_1}(y^* - Ny^* - p), y^* = F_{\Omega_2}(x^* - Mx^* - q)$*;*
3. (x^*, y^*) *is the equilibrium point of the following neural networks:*

$$\begin{cases} \frac{dx(t)}{dt} = -x(t) + (I - N)F_{\Omega_2}(y(t)) - q, \\ \frac{dy(t)}{dt} = -y(t) + (I - M)F_{\Omega_1}(x(t)) - p, \end{cases} \tag{2.3}$$

where I denotes the $n \times n$ identity matrix and F_{Ω_i} $(i = 1, 2)$ are defined by (2.1).

The results of Lemma 2.1 indicate that obtaining the solution of the the variational inequalities (1.2) is equivalent to calculating the equilibrium point of neural network (2.3).

2.2. Formulation of discrete-time neural network method for the variational inequalities

A method which is widely used to derive a discrete-time neural network is to discretize the continuous-time counterpart. There exist many numerical schemes, such as Euler scheme, Runge–Kutta scheme, etc., that can be used to obtain discrete-time analogues of (2.3). As shown in [5, 14, 15, 17, 16, 18], the dynamical properties of different discrete-time analogues vary considerably. In this paper, it is the dynamical properties of converging exponentially towards the equilibrium point that are of great interest.

We begin our discussion by reformulating system (2.3) as the following approximation:

$$\begin{cases} x'(t) \approx -x(t) + (I - N)F_{\Omega_2}\left(y\left(\left[\frac{t}{h}\right]h\right)\right) - q, \\ y'(t) \approx -y(t) + (I - M)F_{\Omega_1}\left(x\left(\left[\frac{t}{h}\right]h\right)\right) - p, \end{cases} \tag{2.4}$$

where h is a fixed positive real number denoting a uniform discretization step size. For any real number r, $[r]$ denotes its integer part. There is no unique way to obtain a discrete-time analogue from (2.4). We first recall the idea proposed by Mohamad *et al.* [5, 14, 15, 17, 16, 18]. For this method, we first integrate (2.4) over $[nh, t]$ with $t < (n + 1)h$:

$$\begin{cases} e^t x(t) - e^{nh}x(nh) \approx \left(e^t - e^{nh}\right)\left[(I - N)F_{\Omega_2}\left(y\left(\left[\frac{t}{h}\right]h\right)\right) - q\right], \\ e^t y(t) - e^{nh}y(nh) \approx \left(e^t - e^{nh}\right)\left[(I - M)F_{\Omega_1}\left(x\left(\left[\frac{t}{h}\right]h\right)\right) - p\right]. \end{cases} \tag{2.5}$$

Clearly, for $t \in [nh, (n + 1)h)$ we have $\left[\frac{t}{h}\right] = n$. Therefore, by letting $t \to (n + 1)h$ in (2.5), we get the following discrete-time system

$$\begin{cases} x_{n+1} = e^{-h}x_n + \left(1 - e^{-h}\right)\left[(I - N)F_{\Omega_2}\left(y_n\right) - q\right], \\ y_{n+1} = e^{-h}y_n + \left(1 - e^{-h}\right)\left[(I - M)F_{\Omega_1}\left(x_n\right) - p\right]. \end{cases} \tag{2.6}$$

The discrete scheme (2.6) is called '*semi-exact*' formula throughout this paper.

We next apply the *implicit-explicit-θ* (IMEX-θ) method [9, 13] to construct another type of discrete-time analogues. The discrete-time analogues derived by applying the IMEX-θ method to (2.4) reads

$$\begin{cases} x_{n+1} = x_n + h\left[-\theta x_{n+1} - (1-\theta)x_n\right] + h\left[(I-N)F_{\Omega_2}(y_n) - q\right], \\ y_{n+1} = y_n + h\left[-\theta y_{n+1} - (1-\theta)y_n\right] + h\left[(I-M)F_{\Omega_1}(x_n) - p\right], \end{cases} \tag{2.7}$$

i.e.,

$$\begin{cases} x_{n+1} = \frac{1}{1+h\theta}\left[(1-h(1-\theta))x_n + h(I-N)F_{\Omega_2}(y_n) - hq\right], \\ y_{n+1} = \frac{1}{1+h\theta}\left[(1-h(1-\theta))y_n + h(I-M)F_{\Omega_1}(x_n) - hp\right], \end{cases} \tag{2.8}$$

where $\theta \in [0,1]$. We note that (2.7) includes a number of discrete-time neural networks by varying the parameter θ. To perform the stability analysis, we rewrite both (2.6) and (2.8) into a uniform form as

$$\begin{cases} x_{n+1} = ax_n + B_2 F_{\Omega_2}(y_n) - \tilde{q}, \\ y_{n+1} = ay_n + B_1 F_{\Omega_1}(x_n) - \tilde{p}, \end{cases} \tag{2.9a}$$

where the quantities a, B_1, B_2, $\tilde{p}$ and $\tilde{q}$ are given by

$$\begin{cases} a = e^{-h}, B_1 = \tilde{h}(I-M), B_2 = \tilde{h}(I-N), \tilde{p} = \tilde{h}p, \tilde{q} = \tilde{h}q \text{ with } \tilde{h} = (1-e^{-h}), & \text{semi-exact}, \\ a = \frac{1-h(1-\theta)}{1+h\theta}, B_1 = \frac{h(I-M)}{1+h\theta}, B_2 = \frac{h(I-N)}{1+h\theta}, \tilde{p} = \frac{h}{1+h\theta}p, \tilde{q} = \frac{h}{1+h\theta}q, & \text{IMEX-}\theta. \end{cases} \tag{2.9b}$$

Remark 2.2. The idea proposed by Mohamad *et al.*[5, 14, 15, 17, 16, 18] and the explicit Euler method (i.e., IMEX-θ method with $\theta = 0$) were used widely to construct discrete-time neural networks; see, e.g., [3, 5, 7, 10, 14, 15, 17, 16, 18, 19, 26, 20, 22, 23, 28] the references therein. However, the IMEX-θ method (except $\theta = 0$) is rarely used to construct discrete-time neural networks.

Clearly, the equilibrium point (x^*, y^*) of (2.3) satisfies the following algebraic equation

$$\begin{cases} x^* = (I-N)F_{\Omega_2}(y^*) - q, \\ y^* = (I-M)F_{\Omega_1}(x^*) - p, \end{cases} \tag{2.10}$$

which, together with (2.6) and (2.7), implies

$$\begin{cases} x^* = ax^* + B_2 F_{\Omega_2}(y^*) - \tilde{q}, \\ y^* = ay^* + B_1 F_{\Omega_1}(x^*) - \tilde{p}. \end{cases} \tag{2.11}$$

For notational convenience, we will shift the equilibrium point of system (2.9) to the origin. To this end, we make the transformation $u_n = x_n - x^*, v_n = y_n - y^*$, and then we obtain a representation of system (2.9) as

$$\begin{cases} u_{n+1} = au_n + B_2 \mathcal{F}_2(v_n), \\ v_{n+1} = av_n + B_1 \mathcal{F}_1(u_n), \end{cases} \tag{2.12}$$

where $\mathcal{F}_1(u_n) = F_{\Omega_1}(u_n + x^*) - F_{\Omega_1}(x^*)$ and $\mathcal{F}_2(v_n) = F_{\Omega_2}(v_n + y^*) - F_{\Omega_2}(y^*)$.

Therefore, to investigate the convergence of (x_n, y_n) to (x^*, y^*) it suffices to study the convergence of (u_n, v_n) to $(0,0)$. Moreover, from (2.2) we know that the functions $\mathcal{F}_{\Omega_i}$ satisfy the following conditions:

$$\mathcal{F}_i(0) = 0,\ 0 \le \frac{\mathcal{F}_i(s_1) - \mathcal{F}_i(s_2)}{s_1 - s_2} \le 1, \quad \forall s_1, s_2 \in \mathbb{R} \text{ and } s_1 \ne s_2. \tag{2.13}$$

Throughout this paper, we will use the notation $P > 0$ (or $P < 0$) to denote that P is a symmetric and positive definite (or negative definite) matrix. If P_1 and P_2 are symmetric matrices , then $P_1 > P_2$ (resp. $P_1 \ge P_2$) denotes that $P_1 - P_2$ is a positive definite (resp. positive semi-definite) matrix. For any matrix $P \in \mathbb{R}^{n\times n}$, let $\lambda_m(P)$ and $\lambda_M(P)$ denote the maximal and minimal eigenvalue of P, respectively. For any vector $z \in \mathbb{R}^n$ and matrix $P \in \mathbb{R}^{n\times n}$, $\|z\|$ denotes the Euclidean norm of z and $\|P\|$ denotes the induced norm of matrix P, that is $\|P\| = \sqrt{\lambda_M(P^\top P)}$.

Lemma 2.3 (Berman and Plemmons [2]). *For any symmetric matrix $P \in \mathbb{R}^{n\times n}$ it holds that*

$$\lambda_m(P)x^\top x \le x^\top Px \le \lambda_M(P)x^\top x,\ \forall x \in \mathbb{R}^n.$$

Definition 2.4. If there exist positive scalars $C > 0$ and $\gamma > 0$ such that

$$\sqrt{\|u_n\|^2 + \|v_n\|^2} \le Ce^{-\gamma n}\sqrt{\|u_0\|^2 + \|v_0\|^2},\quad n \ge 1, \tag{2.14}$$

the discrete-time system (2.12) is said to be globally exponentially stable with convergence rate γ, where $u_0 = x_0 - x^*$ and $v_0 = y_0 - y^*$ are the initial values of (2.12) and (x_0, y_0) are the initial values of (2.9).

3. Stability analysis

In this section, we give two criteria that guarantee the globally exponential convergence of the formulated discrete-time neural networks with uniform scheme (2.12).

Theorem 3.1. *Let λ_i be the maximal eigenvalue of the matrix $\begin{pmatrix} 0 & aB_i \\ aB_2^\top & B_i^\top B_i \end{pmatrix}$ with $i = 1, 2$. If the quantity $\mu = \max\{|a^2+\lambda_2|+|\lambda_1|, |a^2+\lambda_1|+|\lambda_2|\} < 1$, then the solutions $\{u_n\}$ and $\{v_n\}$ generated by formula (2.12) converges exponentially to zero and the exponential convergence rate is at least $\gamma = -\frac{\ln u}{2}$.*

Proof. Let $\mathcal{U}_n = u_n^\top u_n$ and $\mathcal{V}_n = v_n^\top v_n$. From (2.12), it is easy to get

$$\begin{aligned}
\mathcal{U}_{n+1} - \mathcal{U}_n &= (au_n + B_2\mathcal{F}_2(v_n))^\top (au_n + B_2\mathcal{F}_2(v_n)) - u_n^\top u_n \\
&= a^2u_n^\top u_n + 2au_n^\top B_2\mathcal{F}_2(v_n) + \mathcal{F}_2^\top(v_n)B_2^\top B_2\mathcal{F}_2(v_n) - u_2^\top u_n \\
&= (a^2-1)u_n^\top u_n + \begin{pmatrix} u_n \\ \mathcal{F}_2(v_n) \end{pmatrix}^\top \begin{pmatrix} 0 & aB_2 \\ aB_2^\top & B_2^\top B_2 \end{pmatrix} \begin{pmatrix} u_n \\ \mathcal{F}_2(v_n) \end{pmatrix} \\
&\le (a^2-1)\mathcal{U}_n + \lambda_2 \begin{pmatrix} u_n \\ \mathcal{F}_2(v_n) \end{pmatrix}^\top \begin{pmatrix} u_n \\ \mathcal{F}_2(v_n) \end{pmatrix} \\
&\le (a^2-1)\mathcal{U}_n + \lambda_2(\mathcal{U}_n + \mathcal{V}_n),
\end{aligned} \tag{3.1}$$

where in the first inequality we have used Lemma 2.2 and in the last inequality we have used the fact $\mathcal{F}_2(v_n)^\top\mathcal{F}_2(v_n) \le v_n^\top v_n$, which is a direct application of (2.13). Similar, we have the following result for $\mathcal{V}_n$:

$$\mathcal{V}_{n+1} - \mathcal{V}_n \le (a^2-1)\mathcal{V}_n + \lambda_1(\mathcal{V}_n + \mathcal{U}_n). \tag{3.2}$$

It then follows by combining (3.1) and (3.2) that

$$\begin{pmatrix} \mathcal{U}_{n+1} \\ \mathcal{V}_{n+1} \end{pmatrix} \le \Theta \begin{pmatrix} \mathcal{U}_n \\ \mathcal{V}_n \end{pmatrix}, \tag{3.3}$$

where $\Theta = \begin{pmatrix} a^2+\lambda_2 & \lambda_2 \\ \lambda_1 & a^2+\lambda_1 \end{pmatrix}$ and the inequality sign "$\le$" used here means less than or equal for each component of $\begin{pmatrix} \mathcal{U}_{n+1} \\ \mathcal{V}_{n+1} \end{pmatrix}$. It is easy to know that $\|\Theta\|_1 = \mu = \max\{|a^2+\lambda_2|+|\lambda_1|, |a^2+\lambda_1|+|\lambda_2|\}$. Therefore, $|\mathcal{U}_n| + |\mathcal{V}_n| \le \mu^n(|\mathcal{U}_0| + |\mathcal{V}_0|)$, which implies $\|u_n\|^2 + \|v_n\|^2 \le \mu^n(\|u_0\|^2 + \|v_0\|^2)$, i.e., $\sqrt{\|u_n\|^2 + \|v_n\|^2} \le e^{-(\frac{-\ln\mu}{2})n}(\|u_0\|^2 + \|v_0\|^2)$. □

We point out that, in practice it is difficult to compute the maximal eigenvalue of the symmetric matrix $\begin{pmatrix} 0 & aB_i \\ aB_2^\top & B_i^\top B_i \end{pmatrix}$ $(i = 1, 2)$ for very large scale problems. Besides, the assumption $\mu = \max\{|a^2+\lambda_2| + |\lambda_1|, |a^2+\lambda_1| + |\lambda_2|\} < 1$ is too restrictive. Therefore, the applicability of Theorem 3.1 may be limited, and a more practical criterion should be presented. To this end, we use the Lyapunov function $V(n) = e^{2\gamma n}\left(u_n^\top Pu_n + v_n^\top QV_n\right)$ to derive the stability conditions of discrete-time analogues (2.12).

Theorem 3.2. *If there exist symmetric matrices $P > 0, Q > 0$, positive diagonal matrices $D_i, E_i (i = 1, 2, 3)$ and scalar $\gamma > 0$ such that*

$$\Omega = \begin{bmatrix} \Omega_{11} & \Omega_{12} & 0 & \Omega_{14} \\ \star & \Omega_{22} & \Omega_{23} & 0 \\ \star & \star & \Omega_{33} & \Omega_{34} \\ \star & \star & \star & \Omega_{44} \end{bmatrix} < 0, \tag{3.4}$$

where $\star$ denotes the symmetric terms in a symmetric matrix and

$$\begin{aligned} \Omega_{11} &= \left(a^2 e^{2\gamma} - 1\right) P + 2D_1 + D_2, \\ \Omega_{12} &= D_3 - D_1, \\ \Omega_{14} &= a e^{2\gamma} P B_2, \\ \Omega_{22} &= e^{2\gamma} B_1^\top Q B_1 - D_2 - 2D_3, \\ \Omega_{23} &= a e^{2\gamma} Q B_1, \\ \Omega_{33} &= \left(a^2 e^{2\gamma} - 1\right) Q + 2E_1 + E_2, \\ \Omega_{34} &= E_3 - E_1, \\ \Omega_{44} &= e^{2\gamma} B_2^\top P B_2 - E_2 - 2E_3, \end{aligned}$$

then the solutions $\{u_n\}$ and $\{v_n\}$ generated by (2.12) satisfy

$$\sqrt{\|u_n\|^2 + \|v_n\|^2} \le \sqrt{\frac{\lambda_1}{\lambda_0}} e^{-\gamma n} \sqrt{\|u_0\|^2 + \|v_0\|^2}, n \ge 1, \tag{3.5}$$

where $\lambda_0 = \min\{\lambda_m(P), \lambda_m(Q)\}$ and $\lambda_1 = \max\{\lambda_M(P), \lambda_M(Q)\}$.

Proof. Let $\Delta V(n) = V(n+1) - V(n)$. Then, a routine calculation yields

$$\begin{aligned} \Delta V(n) &= e^{2\gamma(n+1)} \left(u_{n+1}^\top P u_{n+1} + v_{n+1}^\top Q V_{n+1}\right) - e^{2\gamma n} \left(u_n^\top P u_n + v_n^\top Q V_n\right) \\ &= e^{2\gamma n} \left(e^{2\gamma} \left[a u_n + B_2 \mathcal{F}_2 (v_n)\right]^\top P \left[a u_n + B_2 \mathcal{F}_2 (v_n)\right] - u_n^\top P u_n\right) + \\ &\quad e^{2\gamma n} \left(e^{2\gamma} \left[a v_n + B_1 \mathcal{F}_1 (u_n)\right]^\top Q \left[a v_n + B_1 \mathcal{F}_1 (u_n)\right] - v_n^\top Q v_n\right) \\ &= u_n^\top \left[\left(a^2 e^{2\gamma} - 1\right) P\right] u_n + 2u_n^\top \left[a e^{2\gamma} P B_2\right] \mathcal{F}_2 (v_n) + \mathcal{F}_2^\top (v_n) \left[e^{2\gamma} B_2^\top P B_2\right] \mathcal{F}_2 (v_n) + \\ &\quad v_n^\top \left[\left(a^2 e^{2\gamma} - 1\right) Q\right] v_n + 2v_n^\top \left[a e^{2\gamma} Q B_1\right] \mathcal{F}_1 (u_n) + \mathcal{F}_1^\top (u_n) \left[e^{2\gamma} B_1^\top Q B_1\right] \mathcal{F}_1 (u_n). \end{aligned} \tag{3.6}$$

Moreover, by condition (2.13) we have

$$\begin{aligned} &0 \le 2e^{2\gamma n} u_n^\top D_1 [u_n - \mathcal{F}_1(u_n)], \\ &0 \le e^{2\gamma n} \left[u_n + \mathcal{F}_1(u_n)\right]^\top D_2 [u_n - \mathcal{F}_1(u_n)], \\ &0 \le 2e^{2\gamma n} \mathcal{F}_1^\top (u_n) D_3 [u_n - \mathcal{F}_1(u_n)], \\ &0 \le 2e^{2\gamma n} v_n^\top E_1 [v_n - \mathcal{F}_2(v_n)], \\ &0 \le e^{2\gamma n} \left[v_n + \mathcal{F}_2(v_n)\right]^\top E_2 [v_n - \mathcal{F}_2(v_n)], \\ &0 \le 2e^{2\gamma n} \mathcal{F}_2^\top (v_n) E_3 [v_n - \mathcal{F}_2(v_n)], \end{aligned} \tag{3.7}$$

since D_i and E_i are positive diagonal matrices.

Let $W_n = \left(u_n^\top, \mathcal{F}_1^\top (u_n), v_n^\top, \mathcal{F}_2^\top (v_n)\right)^\top$. Then, from (3.6) and (3.7) we have

$$\Delta V(n) \le e^{2\gamma n} W_n^\top \Phi W_n^\top \le 0.$$

Therefore, $V(n+1) \leq V(n)$ and this together with Lemma 2.2 implies

$$V(n) \leq V(0) = \left(u_0^\top P u_0 + v_0^\top Q v_0\right) \leq \lambda_1(\|u_0\|^2 + \|v_0\|^2),$$
$$V(n) \geq e^{2\gamma n}\lambda_0\left(\|u_n\|^2 + \|v_n\|^2\right),$$

which gives (3.5). □

Remark 3.3. For given system parameters a, B_1 and B_2 and given exponential convergence rate γ, the LMI given in (3.4) can be calculated efficiently by using the interior point algorithms [1].

4. Numerical experiments

In this section, we provide numerical results to compare the discrete neural networks studied in this paper. We perform two types of comparisons: for a given step size h we compare the convergence rates of the discrete neural networks and for given exponential convergence rate we compare the threshold values of the step size h such that the discrete neural networks are still stable. Our model in this section is

$$\textit{Finding}\ \ (x,y) \in \Omega_1 \times \Omega_2\ \textit{such that}: \quad \begin{cases} (\alpha - x)^\top (x - y + Ny + p) \geq 0, \forall \alpha \in \Omega_1, \\ (\beta - y)^\top (y - x + Mx + q) \geq 0, \forall \beta \in \Omega_2, \end{cases} \tag{4.1}$$

where $\Omega_1 = [-2,3] \times [-3,4]$, $\Omega_2 = [-1,2] \times [1,4]$, $M = \begin{pmatrix} 0.68 & -0.75 \\ 1.05 & 0.68 \end{pmatrix}$, $N = \begin{pmatrix} 0.68 & -1.05 \\ 0.75 & 0.68 \end{pmatrix}$, $p = (1.1, 1)^\top$ and $q = (-2.5, 1.1)^\top$.

To solve the variational inequality (4.1) by the neural network method, we discretize the continuous-time neural network (2.3) by the semi-exact scheme (2.6) and the IMEX-θ scheme (2.8) with step size h. We shall test IMEX methods with $\theta = 0, 0.5$ and $\theta = 1$. For the variational inequality (4.1), with a sufficiently small step size, all of the four discrete neural networks converge to a solution

$$x^* = (0.38262790219848, -1.69397164577768)^\top, \quad y^* = (1.35196219437025, -2.04383022395726)^\top.$$

Example 4.1. We first investigate how the maximal exponential convergence rate (denoted by $\gamma_{\max}$ in this section) varies with respect to the step size h. By solving the LMI presented in Theorem 3.2, we list in Table 4.1 the maximal exponential convergence rate $\gamma_{\max}$ for several to different step sizes h. In all tables, we use 'S.-E.' to denote the semi-exact formula (2.6).

Table 4.1: $\gamma_{\max}$ corresponding to different step size h

Methods	$h = 0.01$	$h = 0.02$	$h = 0.04$	$h = 0.05$	$h = 0.10$	$h = 0.25$	$h = 0.40$
$\theta = 0$	0.000410	0.000816	0.001616	0.002009	0.003894	0.008520	0.010673
$\theta = \frac{1}{2}$	0.000408	0.000808	0.001585	0.001961	0.003721	0.007809	0.010121
$\theta = 1$	0.000406	0.000801	0.001555	0.001916	0.003562	0.007185	0.009313
S.-E.	0.000408	0.000808	0.001585	0.001961	0.003718	0.007781	0.010069

We see from Table 4.1 that the IMEX method with $\theta = 0$, i.e., the explicit Euler method slightly outperforms the other three methods. Moreover, for each method, as h varies from small to large, we see clearly from Table 4.1 that the convergence rate becomes better. The above observations have been illustrated in Figure 4.1, in which we plot the actually measured convergence rate of the four discretization methods with step size $h = 0.01, 0.05$ and 0.1. The measured convergence rate is defined as $\mathrm{Err}_n = \max\{\|x_n - x^*\|_\infty, \|y_n - y^*\|_\infty\}$.

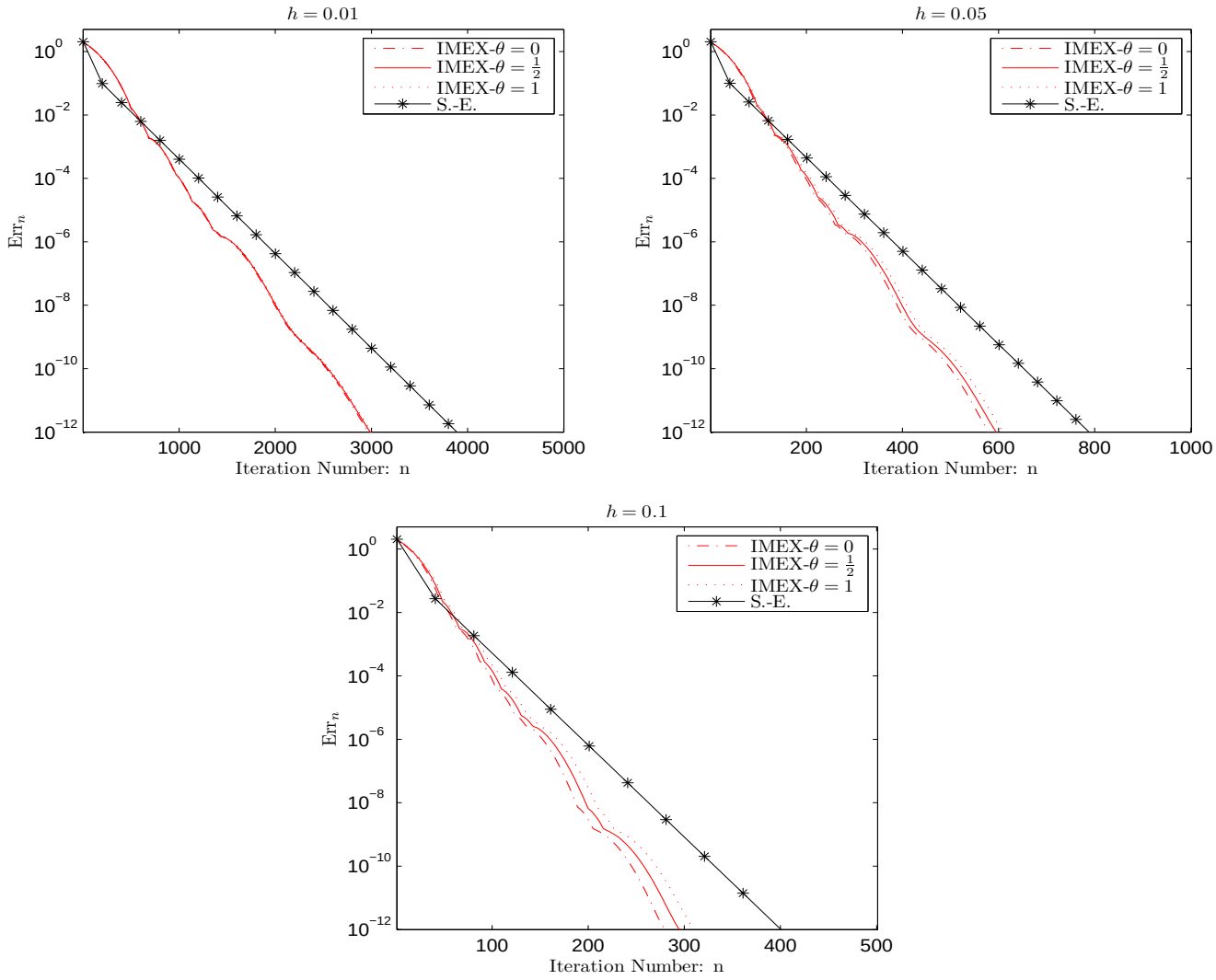

Figure 4.1: Convergence rates of the four numerical methods with step size $h = 0.01$ (left), 0.05 (right) and 0.1 (middle).

For these four methods, we see clearly that: compared to $h = 0.01$, each method possess 5 and 10 times improvement with respect to convergence rate when $h = 0.05$ and $h = 0.1$, respectively, and this confirms very well with the results shown in Table 4.1. Moreover, it is interesting to see in the three panels of Figure 4.1 that the discrete neural network formulated by the semi-exact method is obviously the *worst* one and this slightly contradicts the results given in Table 4.1, since the results in Table 4.1 imply almost equal convergence rates of the four methods. For this observation, we do not have a reasonable explanation at the moment, but this really encourages us to deeply investigate the difference between the IMEX-θ method and the semi-exact method in our future work.

Example 4.2. We next study how the threshold step size h varies with respect to different exponential convergence rate γ. In Table 4.2, we list the maximal step size (denoted by $h_{\max}$ hereafter) for to several different exponential convergence rates γ for each discrete-time neural networks. For each γ given in Table 4.2, the $h_{\max}$ is also calculated by solving LMI (3.4).

The results listed in Table 4.2 reveal that the discrete neural network formulated by the IMEX method with $\theta = 1$, i.e., the semi-implicit Euler method, can tolerate much larger step size. For example, under the condition that the solutions (x_n, y_n) converge to the equilibrium point (x^*, y^*) with rate $\gamma = 10^{-4}$, we can use a step size $h = 2.2373$ for the discrete neural network formulated by the semi-implicit Euler method,

Table 4.2: $h_{\max}$ corresponding to different exponential convergence rate γ

Methods	$\gamma = 10^{-4}$	$\gamma = 5 \times 10^{-4}$	$\gamma = 10^{-3}$	$\gamma = 5 \times 10^{-3}$	$\gamma = 10^{-2}$
$\theta = 0$	0.6911	0.6864	0.6804	0.6241	0.4919
$\theta = \frac{1}{2}$	1.0560	1.0452	1.0313	0.9071	0.6524
$\theta = 1$	2.2373	2.1892	2.1294	1.6600	0.9683
S.-E.	1.1747	1.1597	1.1408	0.9783	0.6772

while the step size can be only 0.6911, 1.0560 and 1.1747 for the neural networks formulated by the other three discretization methods. In Figure 4.2 and 4.3 we plot the discrete solutions $\{x_n\}$ and $\{y_n\}$ which are computed by the four discrete-time neural networks with step size $h = 2.23$, respectively. The results of these two figures show that the discrete neural network formulated by the semi-implicit Euler method has the best convergence rate, and the ones formulated by the IMEX method with $\theta = \frac{1}{2}$, i.e., the well know IMEX trapezoid formula, and the semi-exact method converge slowly , while the one formulated by the explicit Euler method diverges.

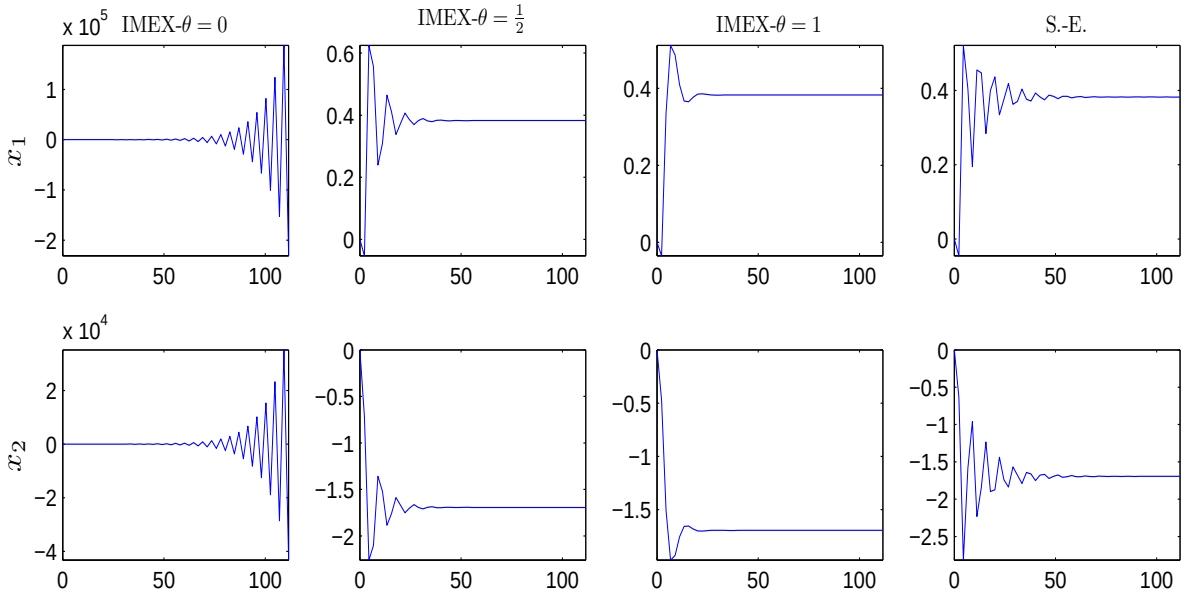

Figure 4.2: Behavior of the numerical solutions $\{x_{1,n}\}$ and $\{x_{2,n}\}$ generated by the four discrete-time neural networks with step size $h = 2.23$. From left to right: IMEX-($\theta = 0$), IMEX-($\theta = \frac{1}{2}$), IMEX-($\theta = 1$), and semi-exact.

Example 4.3. To finish this section, we compare the accuracy of the converged solution at the final time point generated by the discrete neural networks, formulated by the IMEX trapezoid formula, semi-implicit Euler method and the semi-exact method. To this end, we fix the time interval $t \in [0, 1000]$ and run the three discrete neural networks with different step size h. We remark that with step size h, a discrete neural network needs to run $\mathbf{N} = \frac{1000}{h}$ steps to arrive the end time point $T = 1000$. For a given step size h, we denote the discrete solution at the end point $T = 1000$ by $X^h_{\mathbf{N}}$ and $Y^h_{\mathbf{N}}$. And the error between $\left(X^h_{\mathbf{N}}, Y^h_{\mathbf{N}}\right)$ and (x^*, y^*) is defined by $\mathrm{Err}(h) = \max\left\{\|X^h_{\mathbf{N}} - x^*\|_\infty, \|Y^h_{\mathbf{N}} - y^*\|_\infty\right\}$. In Table 4.3, we list $\mathrm{Err}(h)$ of these three methods with respect to different step size h.

We see from Table 4.3 that the semi-implicit Euler method results in more accurate solutions than the other two methods. For example, if we want to calculate an approximation to (x^*, y^*) with a degree of accuracy $O\left(10^{-5}\right)$, the discrete neural network formulated by the IMEX trapezoid formula needs to runs

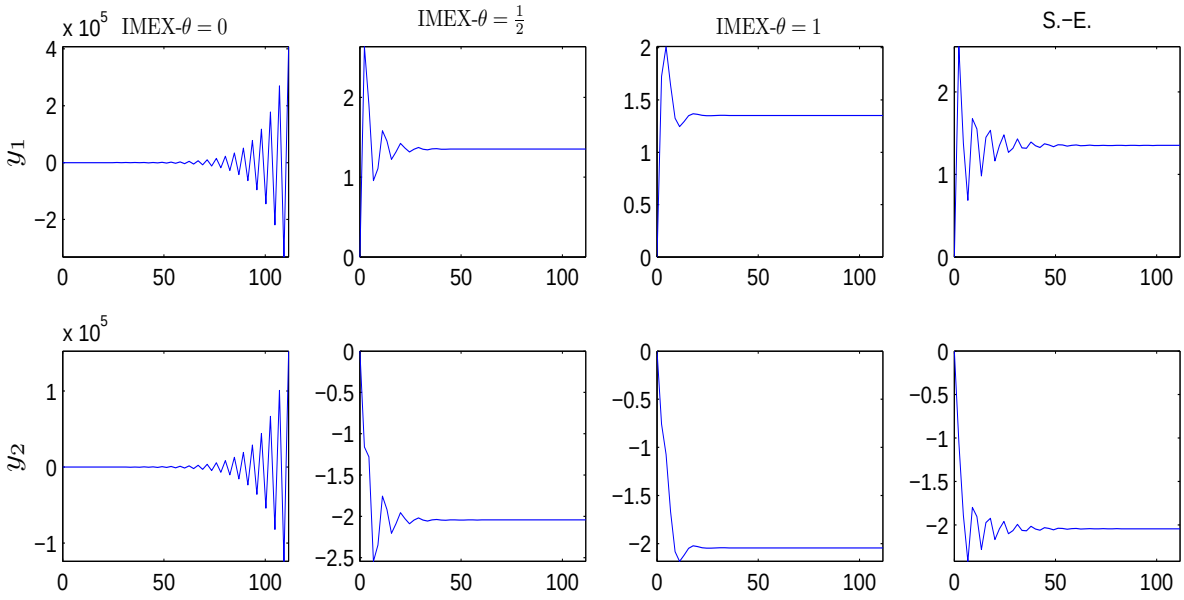

Figure 4.3: Behavior of solutions $\{y_{1,n}\}$ and $\{y_{2,n}\}$ for the four discrete-time neural networks with step size $h = 2.23$. From left to right: IMEX-($\theta = 0$), IMEX-($\theta = \frac{1}{2}$), IMEX-($\theta = 1$), and semi-exact.

Table 4.3: $\max\left\{\|X_{\mathbf{N}}^h - x^*\|_\infty, \|Y_{\mathbf{N}}^h - y^*\|_\infty\right\}$ corresponding to different step size h

Methods	$h = 2.5$	$h = 4.0$	$h = 5.0$	$h = 10$	$h = 20$	$h = 25$	$h = 40$
$\theta = \frac{1}{2}$	6.43e-15	9.28e-5	1.615956	10.39942	12.23441	22.6146	25.2876
$\theta = 1$	6.44e-15	6.66e-15	6.44e-15	6.22e-15	8.23e-10	7.81e-8	6.18e-5
S.-E.	6.43e-15	6.64e-15	6.22e-15	3.83e-10	2.07e-5	1.48e-4	0.00424

$\frac{1000}{4} = 250$ steps and the network formulated by the semi-exact method needs to run $\frac{10000}{20} = 50$ steps, while the one formulated by the semi-implicit Euler method needs only 25 steps. This means that the discrete neural network derived by using semi-implicit Euler method is more efficient.

5. Conclusions

We formulate two types of discrete-time projection neural networks to solve a system of variational inequalities. The discrete analogues are formulated by using the IMEX-θ method and the one introduced recently by Mohamad *et al.* [5, 14, 15, 17, 16, 18] to discretize the continuous-time neural network with a uniform step size h. Sufficient conditions which guarantee that the discrete neural networks converge exponentially to the equilibrium point are established in terms of LMIs. Numerical experiments are performed and the results confirm our theoretical predication very well.

Moreover, we find that as the step size varying from small to large, all of the discrete analogues converge faster to the solution of the variational inequalities, and for very large step size the semi-implicit Euler method is superior to the other discretization methods and therefore we suggest to using the semi-implicit Euler method to construct a discrete-time projection neural network in practice.

References

[1] S. Boyd, L. EI Ghaoui, E. Feron, V. Balakrishnan, *Linear matrix inequalities in systems and control theory*, Philadelphia (SIAM), PA, (1994). 1, 3.3

[2] A. Berman, R.J. Plemmons, *Nonnegative matrices in mathematical sciences*, Academic Press, New York, (1979). 2.3

[3] W. H. Chen, X. M. Lu, D. Y. Liang, *Global exponential stability for discrete-time neural networks with variable delays*, Phys. Lett. A, **358** (2006), 186-198. 2.2

[4] F. Facchinei, J. S. Pang, *Finite-dimensional variational inequalities and complementarity problems*, Springer-Verlag, New York, vol. I, II, (2003). 1

[5] K. Gopalsamy, *Stability of artificial neural networks with impulses*, Appl. Math. Comput., **154** (2003), 783-813. 1, 2.2, 2.2, 2.2, 5

[6] X.L. Hu, J. Wang, *Solving generally constrained generalized linear variational inequalities using the general projection neural networks*, IEEE Trans. Neural Networks, **18** (2007), 1697-1708. 1

[7] Z. Huang, X. Wang, F. Gao, *The existence and global attractivity of almost periodic sequence solution of discrete-time neural networks*, Phys. Lett. A (2006), 182-191. 2.2

[8] P.T. Harker, J.S. Pang, *Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications*, Math. Program., **48** (1990), 161-220. 1

[9] T. Koto, *Stability of IMEX Runge-Kutta methods for delay differential equations*, J. Comput. Appl. Math., **211** (2008), 201-212. 2.2

[10] J. L. Liang, J. D. Cao, *Exponential stability of continuous-time and discrete-time bidirectional associative memory networks with delays*, Chaos, Solitons and Fractals, **22** (2004), 773-785. 2.2

[11] Q. Liu, J. Wang, *A projection neural network for constrained quadratic minimax optimization*, IEEE Trans. Neural Networks and Learning Systems, **26** (2015), 2891-2900. 1

[12] H.Y. Lan, Y.S. Cui, *A neural network method for solving a system of linear variational inequalities*, Chaos Solitons and Fractals, **41** (2009), 1245-1252. 1, 1, 2.1

[13] T. Liu, X. Zhang, X. Gao, *Stability analysis for continuous-time and discrete-time genetic regulatory networks with delays*, Appl. Math. Comput., **274** (2016), 628-643. 2.2

[14] S. Mohamad, *Global exponential stability in continuous-time and discrete-time delayed bidirectional neural networks*, Physica D, **159** (2001), 233-251. 1, 2.2, 2.2, 2.2, 5

[15] S. Mohamad, *Global exponential stability in discrete-time analogues of delayed cellular neural networks*, J. Differential Equations Appl., **9** (2003), 559-575. 1, 2.2, 2.2, 2.2, 5

[16] S. Mohamad, K. Gopalsamy, *Exponential stability of continuous-time and discrete-time cellular neural networks with delays*, Appl. Math. Comput., **135** (2003), 17-38. 1, 2.2, 2.2, 2.2, 5

[17] S.Mohamad, A. G. Naim, *Discrete-time analogues of integrodifferential equations modelling bidirectional neural networks*, J. Comput. Appl. Math., **138** (2002), 1-20. 1, 2.2, 2.2, 2.2, 5

[18] S. Mohamad, K.Gopalsamy, *Dynamics of a class of discrete-time neural networks and their continuous-time counterparts*, Math. Comput. Simulat., **53** (2002), 1-39. 1, 2.2, 2.2, 2.2, 5

[19] S. Mohamad, *Exponential stability preservation in discrete-time analogues of artificial neural networks with distributed delays*, J. Comput. Appl. Math., **215** (2008), 270-287. 1, 2.2

[20] C.Y. Sun, C.B. Feng, *Exponential periodicity of continuous-time and discrete-time neural networks with delays*, Neural Processing Letters, **19** (2004), 131-146. 1, 2.2

[21] C. Sha, H. Zhao, T. Huang, W. Hu, *A projection neural network with time delays for solving linear variational inequality problems and its applications*, Circuits, Systems, and Signal Processing, **35** (2016), 2789C2809. 1

[22] C. Y. Sun, C. B. Feng, *Discrete-time analogues of integrodifferential equations modeling neural networks*, Phys. Lett. A, **334** (2005), 180-191. 1, 2.2

[23] L. Wu, L. Ju, L. Guo, *Dynamics of continuous-time neural networks and their discrete-time analogues with distributed delays*, Lecture Notes In Computer Science, **4491** (2007), 1054-1060. 2.2

[24] Y. Xia, *Further results on global convergence and stability of globally projected dynamical systems*, J. Optim. Theory Appl., **122** (2004), 627-649. 1

[25] Y. Xia, J. Wang, *A general projection neural network for solving monotone variational inequalities and related optimization problems*, IEEE Trans. Neural Networks, **15** (2004), 318-328. 1

[26] R. Yang, B. Wu, Y. Liu, *A halanay-type inequality approach to the stability analysis of discrete-time neural networks with delays*, Appl. Math. Comput., **265** (2015), 696-707. 1, 2.2

[27] Z. G. Zeng, X. X. Liao, *he solution to linear variational inequality by neural network*(in chinese), J. HuaZhong Univ. Sci. Technol. Nat. Sci., **20** (2002), 18-20. 1

[28] H. Zhang, L.S. Chen, *Asymptotic behavior of discrete solutions to delayed neural networks with impulses*, Neurocomput., **71** (2008), 1032-1038. 2.2