

An Alternative Recurrent Neural Network for Solving Variational Inequalities and Related Optimization Problems

Xiaolin Hu, *Member, IEEE*, and Bo Zhang

Abstract—There exist many recurrent neural networks for solving optimization-related problems. In this paper, we present a method for deriving such networks from existing ones by changing connections between computing blocks. Although the dynamic systems may become much different, some distinguished properties may be retained. One example is discussed to solve variational inequalities and related optimization problems with mixed linear and nonlinear constraints. A new network is obtained from two classical models by this means, and its performance is comparable to its predecessors. Thus, an alternative choice for circuits implementation is offered to accomplish such computing tasks.

Index Terms—Asymptotic stability, global convergence, linear programming (LP), optimization, quadratic programming (QP), recurrent neural network (RNN), variational inequality.

I. INTRODUCTION

In recent years, much interest has emerged for designing recurrent neural networks (RNNs) for solving optimization-related problems (for example, see [1]–[20] and the references therein). Such neurodynamic systems have found many applications in signal processing [21] and intelligent control [22]–[25]. However, most of these networks were individually developed, and a general framework about how new models can be designed was seldom discussed, except in [2], to the best of the authors’ knowledge. In this paper, we aim at giving a novel perspective on the RNN design methodology. The idea is to obtain new networks by changing connections in existing networks. The new networks may have distinct advantages in some aspects and therefore offer different choices for circuits implementation. The idea will be made clear by showing how a new model for solving variational inequalities and related optimization problems can be derived from two classical networks. We hope that this method can be generalized to other existing RNNs (e.g., [7], [13], and [17]) and stimulate the emergence of various groups of new networks.

Consider solving the following nonlinear variational inequality [26]. Find a vector $x^* \in \Omega$ such that

$$(x - x^*)^T F(x^*) \geq 0 \quad \forall x \in \Omega \quad (1)$$

where

$$\Omega = \{x \in X | g(x) \leq 0, Ax = b\}.$$

Manuscript received December 21, 2008; revised April 8, 2009. First published August 4, 2009; current version published November 18, 2009. This work was supported in part by the National Natural Science Foundation of China under Grants 60805023, 60621062, and 60605003, the National Key Foundation R&D Project under Grants 2003CB317007, 2004CB318108, and 2007CB311003, the China Postdoctoral Science Foundation under Grants 20080430032 and 200801072, and the Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList). This paper was recommended by Associate Editor J. Wang.

The authors are with the State Key Laboratory of Intelligent Technology & Systems, TNList, and Department of Computer Science & Technology, Tsinghua University, Beijing 100084, China (e-mail: xiaolin.hu@gmail.com; dcszb@tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TSMCB.2009.2025700

In addition, $x = (x_1, \dots, x_n)^T \in \mathbb{R}^n$, $F: \mathbb{R}^n \rightarrow \mathbb{R}^n$ is an n -dimensional vector-valued function, $g(x): \mathbb{R}^n \rightarrow \mathbb{R}^m$ is an m -dimensional vector-valued function, $A \in \mathbb{R}^{r \times n}$, $b \in \mathbb{R}^r$ are constants, and X is a nonempty box set defined as $X = \{x \in \mathbb{R}^n | l_i \leq x_i \leq h_i, i = 1, \dots, n\}$ (some l_i ’s can be $-\infty$, and some h_i ’s can be $+\infty$). The function $F(x)$ is continuous on \mathbb{R}^n , whereas the functions $g_1(x), \dots, g_m(x)$ are convex on X and are continuously differentiable on an open set \tilde{X} that contains X .

Note that, if a continuously differentiable scalar function $f(x)$ is pseudoconvex on Ω , then the following optimization problem is equivalent to the variational inequality (1), with $F = \nabla f$, i.e., the gradient of f [26]:

$$\text{minimize } f(x) \quad \text{subject to } x \in \Omega. \quad (2)$$

There are two representatives of the state-of-the-art RNNs for solving the variational inequality (1), as proposed in [4] and [5], respectively:

$$\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\lambda \begin{pmatrix} x - \mathcal{P}_X(x - \alpha(F(x) + \nabla g(x)^T y - A^T z)) \\ y - \tilde{y} \\ Ax - b \end{pmatrix} \quad (3)$$

$$\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\lambda \begin{pmatrix} 2(x - v) \\ y - \tilde{y} \\ Ax - b \end{pmatrix} \quad (4)$$

where $\tilde{y} = (y + g(x))^+$, $v = \mathcal{P}_X(x - \alpha(F(x) + \nabla g(x)^T \tilde{y} - A^T(z - Ax + b)))$, $\lambda > 0$, $\alpha > 0$, $\nabla g(x) = (\nabla g_1(x), \dots, \nabla g_m(x))^T$, and $\mathcal{P}_X(\cdot)$ and $(\cdot)^+$ are two projection operators (see [4] for definitions). Note that α in (3) and (4) was set to one in [4] and [5]. However, it is easy to verify that, when $\alpha \neq 1$, the stability results of the two networks in [4] and [5] are still valid. Moreover, it was found that tuning this parameter can offer advantages to RNN (4) [8].

The block diagrams of the two networks are depicted in Fig. 1. It is shown that the types and the numbers of basic elements in the two networks are nearly the same. The only difference is that the model (4) entails $2r$ more connections and one more summator than (3) [see Fig. 1(b), bottom right]. Nevertheless, both models have $O(n^2 + mn + nr)$ connections; thus, when the scale of the problem gets large, the $2r$ additional connections in (4) do not much affect the complexity. In this sense, in general, hardware realizations of the two models entail equal effort.

II. MAIN RESULTS

In this section, we present a new RNN for solving (1), which is obtained from RNNs (3) or (4) by changing some connections in their diagrams. Then, we will discuss a special case of the proposed network for solving linear programming problems (LPs) and quadratic programming problems (QPs).

A. Architecture

The new model can be obtained by changing the starting point of the first dashed line in Fig. 1(a) (counted in a top–down manner and with the same convention in what follows) from $S1$ to $S1'$ and the starting point of the second dashed line from $S2$ to $S2'$. Meanwhile, the scaling factors of the first and third integrators need to be changed

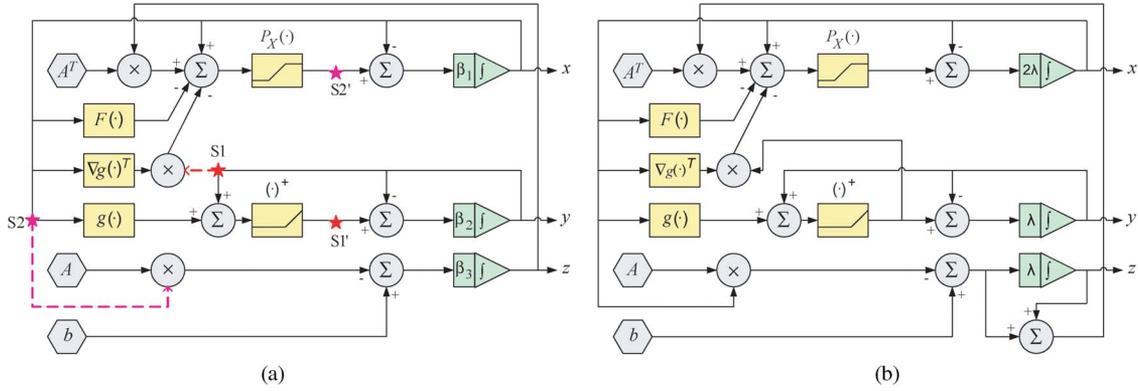


Fig. 1. Block diagrams of RNNs (3)–(5). In (a), if the two dashed lines are connected (i.e., starting from points S1 and S2, respectively) and $\beta_1 = \beta_2 = \beta_3 = \lambda$, then the diagram depicts network (3); if the starting points of the two dashed lines are, respectively, changed to points S1' and S2', whereas $\beta_1 = \beta_3 = 2\lambda$, $\beta_2 = \lambda$, then the diagram depicts network (5). For clarity, the scaling factor α in these networks is set to 1. (a) RNN (3) or (5). (b) RNN (4).

from λ to 2λ for analysis. Then, the dynamic equations of the model become

$$\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\lambda \begin{pmatrix} 2(x - \tilde{x}) \\ y - \tilde{y} \\ 2(A\tilde{x} - b) \end{pmatrix} \quad (5)$$

where $\tilde{x} = \mathcal{P}_X(x - \alpha(F(x) + \nabla g(x)^T \tilde{y} - A^T z))$, and the other notations are the same as in (3) and (4).

Note that this model can also be obtained from (4) by changing some connections in Fig. 1(b). In addition, it is easy to see that RNNs (3) and (4) can be transformed into each other this way. In this sense, the three models share the same structural complexity.

B. Stability Analysis

We now show that the new model is also comparable to (3) and (4) in terms of performance to solve the variational inequality (1).

Some notations and assumptions are first introduced. \hat{X} denotes an open set that contains X , $\|\cdot\|$ denotes the Euclidean norm, and U^* denotes the equilibrium set of RNN (5). Throughout this paper, it is assumed that (1) has at least one finite solution and satisfies the Slater condition, i.e., there exists a point x^\dagger such that $g(x^\dagger) < 0$, $Ax^\dagger = b$, and $l < x^\dagger < h$.

Definition 1: F is said to be a gradient mapping on a set C if there exists a differentiable function $f : C \rightarrow \mathbb{R}$ such that $\nabla f(x) = F(x)$ for any $x \in C$.

Throughout this paper, it is assumed that $F(x)$ and $\nabla g(x)$ are locally Lipschitz continuous on an open set \hat{X} that contains X . If $F(x)$ and $\nabla g(x)$ are continuously differentiable on \hat{X} , then this assumption holds. The following two assumptions will be referred to in later discussion.

A1. $F(x)$ is co-coercive at one finite solution x^* of (1) with respect to X with modulus $\nu > 0$, i.e., $(x - x^*)^T (F(x) - F(x^*)) \geq \nu \|F(x) - F(x^*)\|^2$ for any $x \in X$.

A2. $F(x)$ is monotone at one finite solution x^* of (1) with respect to X , i.e., $(x - x^*)^T (F(x) - F(x^*)) \geq 0$ for any $x \in X$.

Note that **A1** implies **A2** but not vice versa. If $F(x)$ is monotone on X [26], then **A2** holds; however, the converse is not true. If a scalar function $f(x)$ is continuously differentiable on \hat{X} , then it is convex on X if and only if $\nabla f(x)$ is monotone on X [26].

First, it is easy to verify the following property of the new model by considering the Karush–Kuhn–Tucker conditions (see [4]).

Theorem 1: A point x^* is a solution of (1) if and only if there exist y^* and z^* such that $((x^*)^T, (y^*)^T, (z^*)^T)^T$ is an equilibrium point of (5).

It is assumed that (1) has at least one finite solution; thus, according to Theorem, 1 there exists at least one finite point in U^* .

The following three lemmas can similarly be established as in [5, Lemma 3], [8, Lemma 4], and [5, Theorem 2], respectively. For brevity, the details are omitted.

Lemma 1: Consider the following function:

$$V_1(u) = \alpha (\phi(u) - \phi(u^*) - (u - u^*)^T \nabla \phi(u^*)) + \frac{1}{2} (\|x - x^*\|^2 + \alpha \|y - y^*\|^2 + \alpha \|z - z^*\|^2) \quad (6)$$

where $\alpha > 0$, $u = ((x)^T, (y)^T, (z)^T)^T$, $u^* = ((x^*)^T, (y^*)^T, (z^*)^T)^T \in U^*$ is a finite point, and $\phi(u) = \|\tilde{y}\|^2/2$. It has the following three properties.

- 1) $V_1(u)$ is convex on $X \times \mathbb{R}^{m+r}$ and is continuously differentiable on $\hat{X} \times \mathbb{R}^{m+r}$.
- 2) $V_1(u) \geq \min(1, \alpha) \|u - u^*\|^2/2$ for all $u \in X \times \mathbb{R}^{m+r}$.
- 3) $\nabla V_1(u)^T G(u) \geq 2\|x - \tilde{x}\|^2 + \alpha \|y - \tilde{y}\|^2 + 2\alpha(\tilde{x} - x^*)^T (F(x) - F(x^*))$ for all $u \in X \times \mathbb{R}^{m+r}$, where

$$G(u) = \begin{pmatrix} 2(x - \tilde{x}) \\ y - \tilde{y} \\ 2(A\tilde{x} - b) \end{pmatrix}.$$

Lemma 2: Assume that F is a gradient mapping on X . Consider the following function:

$$V_2(u) = \alpha (\psi(u) - \psi(u^*) - (u - u^*)^T \nabla \psi(u^*)) + \frac{1}{2} (\|x - x^*\|^2 + \alpha \|y - y^*\|^2 + \alpha \|z - z^*\|^2) \quad (7)$$

where $\psi(u) = f(x) + \|\tilde{y}\|^2/2$, with $\nabla f(x) = F(x)$, and the other notations are the same as in Lemma 1. It has the following three properties.

- 1) $V_2(u)$ is convex on $X \times \mathbb{R}^{m+r}$ and is continuously differentiable on $\hat{X} \times \mathbb{R}^{m+r}$.
- 2) $V_2(u) \geq \min(1, \alpha) \|u - u^*\|^2/2$ for all $u \in X \times \mathbb{R}^{m+r}$.
- 3) $\nabla V_2(u)^T G(u) \geq 2\|x - \tilde{x}\|^2 + \alpha \|y - \tilde{y}\|^2 + 2\alpha(x - x^*)^T \times (F(x) - F(x^*))$ for all $u \in X \times \mathbb{R}^{m+r}$, where $G(u)$ is defined in Lemma 1.

Lemma 3: Suppose that **A1** holds. Then, for any initial point, $u(t_0) = (x(t_0)^T, y(t_0)^T, z(t_0)^T)^T \in X \times \mathbb{R}^{m+r}$, RNN (5), with $\alpha < 4\nu$, has a unique bounded continuous solution $x(t)$ for all $t \geq t_0$, and $x(t)$ stays in X forever.

Now, we are in a position to state the main results of the proposed RNN.

Theorem 2: Suppose that **A1** holds. Then, RNN (5) is stable in the sense of Lyapunov, with $\alpha < 4\nu$, and its trajectory $u(t)$ converges to U^* with any $u(t_0) \in X \times \mathbb{R}^{m+r}$. Furthermore, if **A1** holds at any solution of (1), then the trajectory converges to an exact point in U^* . In particular, if there is only one point in U^* , the RNN is asymptotically stable.

Proof: Consider the function $V_1(u(t))$ as defined in (6), where x^* satisfies **A1**. According to Lemmas 1 and 3, for any initial point $u(t_0) \in X \times \mathbb{R}^{m+r}$, we have

$$\frac{dV_1(u(t))}{dt} \leq -\lambda \left(2 - \frac{\alpha}{2\nu}\right) \|x - \tilde{x}\|^2 - \lambda\alpha \|y - \tilde{y}\|^2 \leq 0 \quad \forall t \geq t_0$$

by noticing

$$\begin{aligned} & (\tilde{x} - x^*)^T (F(x) - F(x^*)) \\ &= (\tilde{x} - x + x - x^*)^T (F(x) - F(x^*)) \\ &\geq -\|x - \tilde{x}\| \|F(x) - F(x^*)\| + \nu \|F(x) - F(x^*)\|^2 \\ &\geq -\frac{1}{4\nu} \|\tilde{x} - x\|^2. \end{aligned}$$

Therefore, the neural network is stable in the sense of Lyapunov. According to the LaSalle invariance principle, $u(t)$ converges to the largest invariant set \mathcal{M} in $\{u \in \mathbb{R}^{n+m+r} | dV_1(u)/dt = 0\}$. In what follows, we show that $\mathcal{M} = U^*$. Clearly, any point in U^* also belongs to \mathcal{M} . Consider any point $u \in \mathcal{M}$. We have $dV_1/dt = 0$; thus, $x = \tilde{x}$, and $y = \tilde{y}$ from the aforementioned equation, which implies that $dx/dt = -2\lambda(x - \tilde{x}) = 0$ and $dy/dt = -\lambda(y - \tilde{y}) = 0$. It follows that x is in a steady state (a constant) and so is \tilde{x} . Denote $A\tilde{x} - b$ by c , where c is a constant. If $c \neq 0$, then $dz/dt = -2\lambda c$, and $z \rightarrow \infty$ when $t \rightarrow +\infty$, which contradicts the boundedness of $u(t)$. Consequently, $c = 0$, and $dz/dt = 0$. It follows that $u \in U^*$, and hence, $\mathcal{M} = U^*$.

Because $u(t)$ is bounded over $[t_0, +\infty)$, there exists a convergent subsequence $t_0 < \dots < t_n < t_{n+1} < \dots$ such that

$$\lim_{k \rightarrow +\infty} u(t_k) = \hat{u} \quad \hat{u} \in U^*.$$

Define another Lyapunov function $\hat{V}(u)$ to be the same as $V_1(u)$ in (6), except that u^* in $V_1(u)$ is replaced with \hat{u} . If **A1** holds at any solution of (1), it is easy to see that $\hat{V}(u)$ decreases along the trajectory of (5) and satisfies $\hat{V}(\hat{u}) = 0$. Therefore, for any $\varepsilon > 0$, there exists $q > 0$ such that, for all $t \geq t_q$

$$\min(1, \alpha) \|u(t) - \hat{u}\|^2 / 2 \leq \hat{V}(u(t)) \leq \hat{V}(u(t_q)) < \varepsilon$$

i.e., $\lim_{t \rightarrow +\infty} u(t) = \hat{u}$.

In particular, if U^* contains a unique point, based on the aforementioned analysis, the neural network is asymptotically stable. The proof is completed. ■

The following theorem reveals that, when F is a gradient mapping, the convergence of the proposed RNN need not impose restrictions on the scaling factor α .

Theorem 3: Suppose that **A2** holds. If F is a gradient mapping on X , then RNN (5) is stable in the sense of Lyapunov, and its trajectory $u(t)$ converges to U^* with any $u(t_0) \in X \times \mathbb{R}^{m+r}$. If, in addition, **A2** holds at any solution of (1), then $u(t)$ converges to an exact point

in U^* . In particular, if there is only one point in U^* , the RNN is asymptotically stable.

Proof: The results can be established by defining a Lyapunov function $V_2(u)$ in Lemma 2, in which u^* satisfies **A2** and following a similar reasoning procedure for Theorem 2. ■

C. Comparisons With Existing Models

We now compare the theoretically guaranteed performances of the new model (5) with its predecessors (3) and (4). First, by comparing Theorems 2 and 3 in this paper with [5, Theorem 3] and [8, Theorems 3 and 4], it is shown that the stability results of (4) and (5) are exactly the same. Some major results of (3) are listed in Theorem 4, which are collected from [4], [16], and [27]. In these results, $g_j(x)$ ($j = 1, \dots, m$) is not assumed to be convex on X in general.

Theorem 4: Suppose that $F(x)$ and $\nabla g(x)$ are locally Lipschitz continuous in \mathbb{R}^n . If any of the following statements is true, then the state trajectory of RNN (3), with $u(t_0) \in X \times \mathbb{R}_+^m \times \mathbb{R}^r$, where \mathbb{R}_+^m denotes the nonnegative quadrant of \mathbb{R}^m , converges to an exact solution of (1).

- 1) $F(x)$ and $\nabla g_j(x)$ ($j = 1, \dots, m$) are monotone on X , and $\nabla F(x^*)$ is positive definite.
- 2) $F(x)$ and $\nabla g_j(x)$ ($j = 1, \dots, m$) are monotone on X , and there exists a nonnegative vector $p = (p_1, \dots, p_m)^T \leq y^*$ such that $\nabla F(x) + \sum_{j=1}^m \nabla^2 g_j(x) p_j$ is positive definite on X ; $y(t_0) \geq p$.
- 3) $F(x) + \nabla g(x)^T y(t)$ is monotone on X , and $\nabla F(x^*)$ is positive definite.
- 4) $\nabla F(x) + \sum_{j=1}^m \nabla^2 g_j(x) y_j(t)$ is positive definite on X .
- 5) $F(x)$ is a gradient mapping on X , and $\nabla F(x) + \sum_{j=1}^m \nabla^2 g_j(x) y_j$ is positive semidefinite on $X \times Y$, where $Y = \{y_\zeta = (1 - \zeta)y^* + \zeta y(t) \in \mathbb{R}^m | 0 \leq \zeta \leq 1\}$, and $\nabla F(x^*) + \sum_{j=1}^m \nabla^2 g_j(x^*) y_j^*$ is positive definite.

In these results, x^* denotes a finite solution of (1), y^* denotes the corresponding Lagrange multiplier (see Theorem 1), and $y(t)$ is a part of the state trajectory $u(t)$ of the network.

From Theorem 4, it can be seen that a notable merit of (3) refers to its capability to solve some nonmonotone variational inequalities (1) and nonconvex optimization problems (2). However, none of the results guarantees that the network can solve degenerate convex optimization problems, including LP problems (see Examples 2 and 3 in Section III); in contrast, both RNNs (4) and (5) have this capability. One of the demerits of RNNs (4) and (5), however, is that, if F is not a gradient mapping, an upper bound restriction on the scaling factor α has to be imposed, which is not required by (3). In summary, with regard to the stability results, there is no simple conclusion as to which model (3)–(5) is the best.

D. Special Case

Consider solving the following QP problem:

$$\begin{aligned} & \text{minimize} && \frac{1}{2} x^T Q x + p^T x \\ & \text{subject to} && A x = b \quad C x \leq d \quad x \in X \end{aligned} \quad (8)$$

where $Q = Q^T \in \mathbb{R}^{n \times n}$, $p \in \mathbb{R}^n$, $C \in \mathbb{R}^{m \times n}$, $d \in \mathbb{R}^m$, and the other notations are the same as in (1). If $Q = 0$, the problem degenerates to an LP problem. The architectures of the three networks (3)–(5) for solving the problem are illustrated in Fig. 2. In contrast to Fig. 1, scalar forms of the dynamic equations are depicted in the figure, where $\{q_{ij}\}_{n \times n} = Q$, $\{a_{ij}\}_{r \times n} = A$, $\{c_{ij}\}_{m \times n} = C$, $\{p_i\}_{n \times 1} = p$, $\{b_i\}_{r \times 1} = b$, and $\{d_i\}_{m \times 1} = d$.

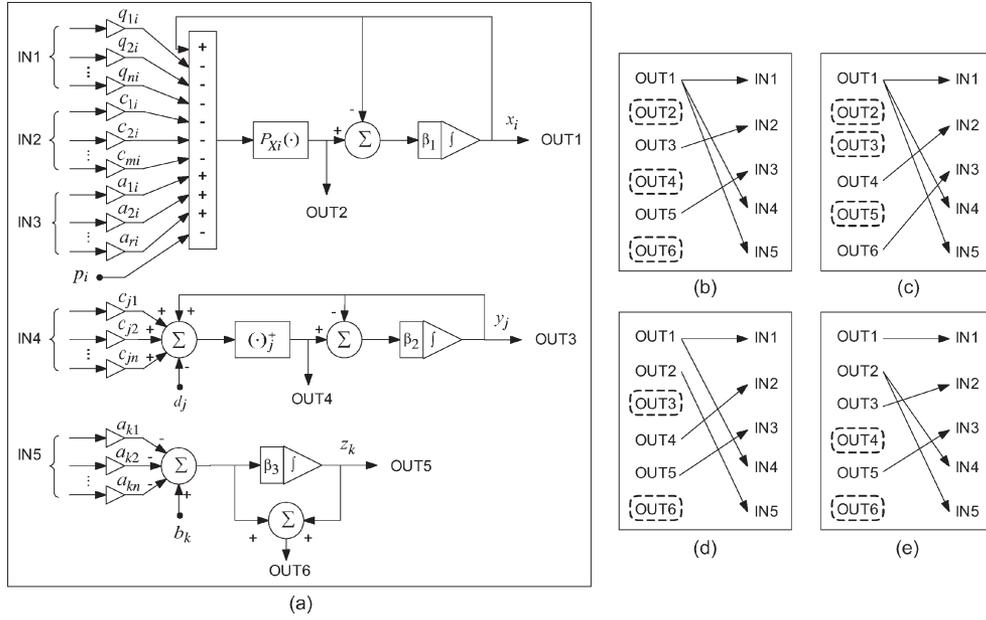


Fig. 2. Block diagrams of the RNNs (3)–(5) and (9) for solving (8). (a) Basic blocks with their input and output ports, where $i = 1, \dots, n$, $j = 1, \dots, m$, and $k = 1, \dots, r$. (b)–(e) Blocks should be connected to constitute RNNs (3)–(5) and (9), respectively. The output ports that are surrounded by dashed rectangles do not need to be connected to other ports in the corresponding configuration. The gain factors differ in these configurations: (b) $\beta_1 = \beta_2 = \beta_3 = \lambda$, (c) $\beta_1 = 2\lambda$, $\beta_2 = \beta_3 = \lambda$, (d) $\beta_1 = \beta_3 = 2\lambda$, $\beta_2 = \lambda$, and (e) $\beta_1 = \lambda$, $\beta_2 = \beta_3 = 2\lambda$. For clarity, the scaling factor α in the networks is set to one.

Recently, we have devised another RNN for solving (8), which was also obtained by changing some connections in RNNs (3) and (4) [20]. The dynamic equations are given as follows:

$$\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\lambda \begin{pmatrix} x - \bar{x} \\ 2(y - (y + C\bar{x} - d)^+) \\ 2(A\bar{x} - b) \end{pmatrix} \quad (9)$$

where $\bar{x} = \mathcal{P}_X(x - Qx - p - C^T y + A^T z)$, and $\lambda > 0$. The architecture of the network is also illustrated in Fig. 2.

In Fig. 2, it is shown that switching between these four networks entails simply changing several gain factors and starting positions of some connections, which implies that circuits implementations of the networks entail the same number of computing elements. The only difference is that RNN (4) entails an additional summation operation [i.e., the summator just before the port “OUT6” in Fig. 2(a)] compared with other networks. With regard to the performance for solving (8), if Q is positive definite, then all of the four networks can converge to a solution of the problem from any initial point; if Q is only positive semidefinite (e.g., $Q = 0$), however, only RNNs (4), (5), and (9) are guaranteed to solve the problem. The stability results of the networks are formally stated as follows, which directly follow from Theorems 3, 4, [5, Theorem 3], and [20, Theorem 2].

Corollary 1:

- 1) If Q is positive definite, then RNN (3) for solving (8) is stable in the sense of Lyapunov, and the trajectory $u(t)$ converges to an exact point in U^* with any $u(t_0) \in X \times \mathbb{R}_+^m \times \mathbb{R}^r$. In particular, if there is only one point in U^* , the RNN is asymptotically stable.
- 2) If Q is positive semidefinite, then RNNs (4), (5), and (9) for solving (8) are stable in the sense of Lyapunov, and the trajectories $u(t)$ converge to an exact point in U^* with any $u(t_0) \in \mathbb{R}^{n+m+r}$. In particular, if there is only one point in U^* , the RNNs are asymptotically stable.

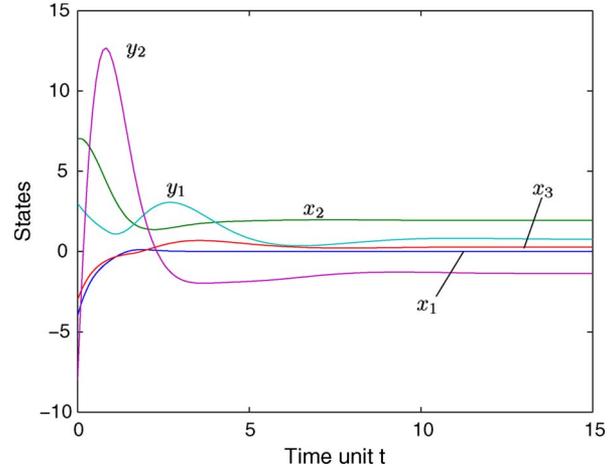


Fig. 3. State trajectories of RNN (5), with $\alpha = 0.03$, $\lambda = 1$, and $u(0) = (-4, 7, -3, 3, -8)^T$ in Example 1.

III. ILLUSTRATIVE EXAMPLES

We then illustrate the theoretical results of the proposed RNN by a few examples. The numerical simulations were conducted in MATLAB.

Example 1: Consider solving the variational inequality (1) with $F = Mx + p$, where

$$M = \begin{pmatrix} 2 & -2 & 1 \\ 0 & 4 & 2 \\ 1 & -1 & 5 \end{pmatrix} \quad p = \begin{pmatrix} 8 \\ 0 \\ 5 \end{pmatrix}.$$

In addition, $X = \mathbb{R}_+^3$, $g(x) = x_1^3 - 2x_2 - 4x_3 + 5$, $A = (2, -5, -1)$, and $b = -10$. It is easy to check that $g(x)$ is convex on $X = \mathbb{R}_+^3$ and **A1** holds at any solution of the problem. In addition, ν can be chosen as $\theta_{\min}/\theta_{\max} = 1.254/31.835 = 0.039$, where θ_{\min} and θ_{\max} are, respectively, the minimum eigenvalue of $(M + M^T)/2$ and

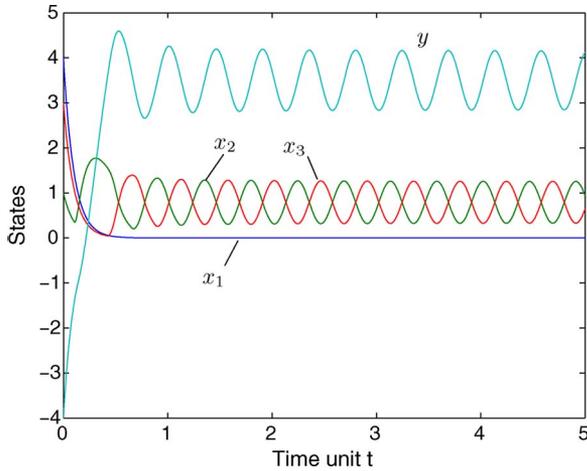


Fig. 4. State trajectories of RNN (3), with $\alpha = \lambda = 1$ and $u(0) = (4, 1, 3, -4)^T$ in Example 2.

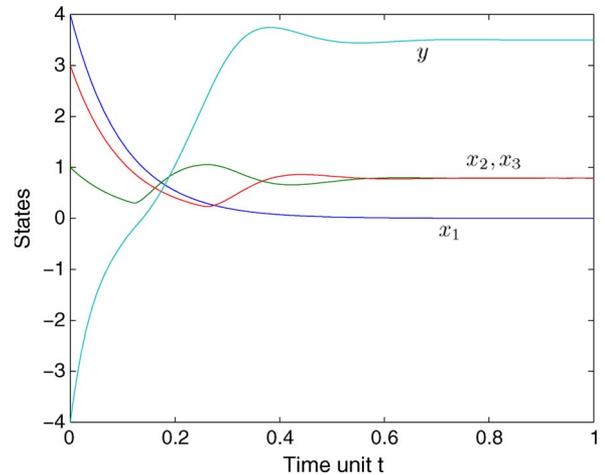


Fig. 5. State trajectories of RNN (5), with $\alpha = \lambda = 1$ and $u(0) = (4, 1, 3, -4)^T$ in Example 2.

the maximum eigenvalue of $M^T M$. According to Theorem 2, RNN (5) should converge to U^* , with $\alpha < 0.039$, starting from any initial point $u(0) \in \mathbb{R}_+^3 \times \mathbb{R}^2$. All simulations verified this fact and indicated that there is only one point in U^* , i.e., $u^* = (0.0000, 1.9445, 0.2788, 0.7652, -1.3656)^T$. In addition, note that, even with $x(0) \notin X$, the RNN still always converged to u^* . Fig. 3 shows such an example. This phenomenon indicates that the stability results in this paper leave space for improvement.

Example 2: Consider the following problem:

$$\begin{aligned} & \text{minimize} && f(x) = \frac{1}{3}(x_1 + x_2 + x_3)^3 - 6x_2 + x_3 \\ & \text{subject to} && x_1 + x_2 - x_3 \leq 0 \quad 0 \leq x \leq 5. \end{aligned}$$

It is easy to check that $\nabla^2 f(x)$ is always positive semidefinite only on \mathbb{R}_+^3 which implies that f is convex but not strictly convex on \mathbb{R}_+^3 . RNN (3) may not solve the problem. The experiments verified this point. Fig. 4 shows a simulation, from which it can be seen that some components of the state trajectory $u(t)$ oscillate and do not converge to steady values. On the contrary, RNN (5) can solve the problem. According to Theorem 3, RNN (5), with $F = \nabla f$, should converge to a solution from any $u(0) \in [0, 5]^3 \times \mathbb{R}$. The simulation results are consistent with the prediction. Fig. 5 depicts the trajectory-evolving process in one of many runs. The trajectories converge to the unique equilibrium point, i.e., $u^* = (0.0000, 0.7906, 0.7906, 3.5000)^T$. The same solution was obtained using RNN (4).

Example 3: To show that RNN (3) may fail on some LP problems, we consider solving a special case of (8), in which $Q = p = C = d = b = 0$, $X = \mathbb{R}$, and $A = 1$. Obviously, the correct solution

should be $x^* = 0$. If we let $\lambda = \alpha = 1$, the dynamic equations of (3) become

$$\frac{d}{dt} \begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} z \\ -x \end{pmatrix}.$$

The equations are simple; thus, the solution can analytically be obtained as follows:

$$\begin{cases} x(t) = x_0 \cos t + z_0 \sin t \\ z(t) = -x_0 \sin t + z_0 \cos t \end{cases}$$

where $(x_0, z_0)^T$ is the initial state. Clearly, if $x_0 \neq 0$, $z_0 \neq 0$, the solution will never converge to a steady state. Similarly, we can write down the simplified dynamic equations (4), (5), and (9) for solving the problem as

$$\begin{aligned} \frac{d}{dt} \begin{pmatrix} x \\ z \end{pmatrix} &= \begin{pmatrix} 2z - 2x \\ -x \end{pmatrix} \\ \frac{d}{dt} \begin{pmatrix} x \\ z \end{pmatrix} &= \begin{pmatrix} 2z \\ -2x - 2z \end{pmatrix} \\ \frac{d}{dt} \begin{pmatrix} x \\ z \end{pmatrix} &= \begin{pmatrix} z \\ -2x - 2z \end{pmatrix} \end{aligned}$$

and derive their closed-form solutions, respectively, as in the equation shown at the bottom of the page. Obviously, from any initial state, the three networks will all converge to $(0, 0)^T$, which corresponds to the correct solution of the problem.

$$\begin{aligned} \begin{pmatrix} x(t) \\ z(t) \end{pmatrix} &= \begin{pmatrix} e^{-t}(\cos t - \sin t) & 2e^{-t} \sin t \\ -e^{-t} \sin t & e^{-t}(\cos t + \sin t) \end{pmatrix} \begin{pmatrix} x_0 \\ z_0 \end{pmatrix} \\ \begin{pmatrix} x(t) \\ z(t) \end{pmatrix} &= \begin{pmatrix} \frac{1}{3}e^{-t} (3 \cos(\sqrt{3}t) + \sqrt{3} \sin(\sqrt{3}t)) & \frac{2}{3}\sqrt{3}e^{-t} \sin(\sqrt{3}t) \\ -\frac{2}{3}\sqrt{3}e^{-t} \sin(\sqrt{3}t) & \frac{1}{3}e^{-t} (3 \cos(\sqrt{3}t) - \sqrt{3} \sin(\sqrt{3}t)) \end{pmatrix} \begin{pmatrix} x_0 \\ z_0 \end{pmatrix} \\ \begin{pmatrix} x(t) \\ z(t) \end{pmatrix} &= \begin{pmatrix} e^{-t}(\cos t + \sin t) & e^{-t} \sin t \\ -2e^{-t} \sin t & e^{-t}(\cos t - \sin t) \end{pmatrix} \begin{pmatrix} x_0 \\ z_0 \end{pmatrix} \end{aligned}$$

IV. DISCUSSION

The aim of this paper has been to provide a novel perspective for designing RNNs for solving optimization and related problems: new salient models can be obtained by changing connections in existing ones without resorting to additional computing units. One example network has been presented, which is derived from two classical models [4], [5] for solving variational inequalities. It has been shown to be comparable to the two models in terms of both structure and performance for solving such problems. Therefore, it enriches the family of RNNs for solving related problems and offers flexibility to circuits practitioners for hardware implementation. Recently, through this method, another model has been derived from the two models [4], [5] for solving LP and QP problems [20]. However, the generalization of that model to solve non-LP problems or variational inequalities encountered difficulty in stability analysis. The invention in this paper therefore outperforms that model.

Aside from [20], we can find other RNNs for solving optimization-related problems, which can also be deemed as derived from others by changing connections (sometimes with a little modification to computing units). For instance, the model in [3] for LP and QP can be regarded as obtained from the model in [4] (see [20] for the detailed discussion). Therefore, this idea seems to represent a general methodology for designing groups of new RNNs. However, how could we know which connections should be changed? Unfortunately, it is hard to answer this question at this stage; only some hints are available. Any useful model has to converge to its equilibrium state, which must have a good correspondence with the solutions of the problem; thus, we can always equate the right-hand sides of the dynamic equations to zeros and find equivalent terms in this state. Exchanging these terms in dynamic equations may not alter the equilibrium points of the system, and new models can be figured out. The following two issues remain to be examined.

- Can this term-exchanging manipulation be realized by switching connections only?
- Are the stability results still true or even better?

It is often easier to answer the first question than the second one. In answering the second question, we may get hints from theoretical analysis procedures for existing models similar to what we have done in this paper.

REFERENCES

- [1] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1544–1547, Nov. 1996.
- [2] Y. Xia and J. Wang, "A general methodology for designing globally convergent optimization neural networks," *IEEE Trans. Neural Netw.*, vol. 9, no. 6, pp. 1331–1343, Nov. 1998.
- [3] Q. Tao, J. Cao, and D. Sun, "A simple and high-performance neural network for quadratic programming problems," *Appl. Math. Comput.*, vol. 124, no. 2, pp. 251–260, Nov. 2001.
- [4] Y. Xia, "An extended projection neural network for constrained optimization," *Neural Comput.*, vol. 16, no. 4, pp. 863–883, Apr. 2004.
- [5] X. Gao, "A novel neural network for nonlinear convex programming," *IEEE Trans. Neural Netw.*, vol. 15, no. 3, pp. 613–621, May 2004.
- [6] M. Forti, P. Nistri, and M. Quincampoix, "Generalized neural network for nonsmooth nonlinear programming problems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 9, pp. 1741–1754, Sep. 2004.
- [7] X. Gao, L. Liao, and L. Qi, "A novel neural network for variational inequalities with linear and nonlinear constraints," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1305–1317, Nov. 2005.
- [8] X. Gao and L. Du, "A neural network with finite-time convergence for a class of variational inequalities," in *Proc. 4th Int. Conf. Intell. Comput.*, Kunming, China, Aug. 2006, vol. 4113, pp. 32–41.
- [9] Y. Yang and J. Cao, "Solving quadratic programming problems by delayed projection neural network," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1630–1634, Nov. 2006.
- [10] Y. Yang and J. Cao, "A delayed neural network method for solving convex optimization problems," *Int. J. Neural Syst.*, vol. 16, no. 4, pp. 295–303, Aug. 2006.
- [11] X. Hu and J. Wang, "Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1487–1499, Nov. 2006.
- [12] H. Ghasabi-Oskoei and N. Mahdavi-Amiri, "An efficient simplified neural network for solving linear and quadratic programming problems," *Appl. Math. Comput.*, vol. 175, no. 1, pp. 452–464, Apr. 2006.
- [13] R. Perfetti and E. Ricci, "Analog neural network for support vector machine learning," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 1085–1091, Jul. 2006.
- [14] X. Hu and J. Wang, "Design of general projection neural networks for solving monotone linear variational inequalities and linear and quadratic optimization problems," *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, vol. 37, no. 5, pp. 1414–1421, Oct. 2007.
- [15] X. Hu and J. Wang, "Solving generally constrained generalized linear variational inequalities using the general projection neural networks," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1697–1708, Nov. 2007.
- [16] Y. Xia, G. Feng, and M. Kamel, "Development and analysis of a neural dynamical approach to nonlinear programming problems," *IEEE Trans. Autom. Control*, vol. 52, no. 11, pp. 2154–2159, Nov. 2007.
- [17] Y. Xia and G. Feng, "A new neural network for solving nonlinear projection equations," *Neural Netw.*, vol. 20, no. 5, pp. 577–589, Jul. 2007.
- [18] Q. Liu and J. Wang, "A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming," *IEEE Trans. Neural Netw.*, vol. 19, no. 4, pp. 558–570, Apr. 2008.
- [19] L. Cheng, Z. G. Hou, and M. Tan, "A neutral-type delayed-projection neural network for solving nonlinear variational inequalities," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 55, no. 8, pp. 806–810, Aug. 2008.
- [20] X. Hu and B. Zhang, "A new recurrent neural network for solving convex quadratic programming problems with an application to the k -winners-take-all problem," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 654–664, Apr. 2009.
- [21] Y. Xia and M. S. Kamel, "A generalized least absolute deviation method for parameter estimation of autoregressive signals," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 107–118, Jan. 2008.
- [22] Y. Zhang, S. S. Ge, and T. H. Lee, "A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators," *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, vol. 34, no. 5, pp. 2126–2132, Oct. 2004.
- [23] Z. Hou, L. Cheng, and M. Tan, "Coordination of two redundant robots using a dual neural network," in *Proc. IJCNN*, Vancouver, BC, Canada, Jul. 2006, pp. 4187–4192.
- [24] Z. Hou, M. Gupta, P. Nikiforuk, M. Tan, and L. Cheng, "A recurrent neural network for hierarchical control of interconnected dynamic systems," *IEEE Trans. Neural Netw.*, vol. 18, no. 2, pp. 466–481, Mar. 2007.
- [25] Y. Pan and J. Wang, "Two neural network approaches to model predictive control," in *Proc. Amer. Control Conf.*, Seattle, WA, Jun. 2008, pp. 1685–1690.
- [26] F. Facchinei and J. S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, vol. I/II. New York: Springer-Verlag, 2003.
- [27] Y. Xia and G. Feng, "On the convergence conditions of an extended projection neural network," *Neural Comput.*, vol. 17, no. 3, pp. 515–525, Mar. 2005.