Proceedings of International Joint Conference on Neural Networks, Atlanta, Georgia, USA, June 14-19, 2009

Solving Convex Optimization Problems Using Recurrent Neural Networks in Finite Time

Long Cheng, Zeng-Guang Hou, Noriyasu Homma, Min Tan, and Madam M. Gupta

***Abstract*— A recurrent neural network is proposed to deal with the convex optimization problem. By employing a specific nonlinear unit, the proposed neural network is proved to be convergent to the optimal solution in finite time, which increases the computation efficiency dramatically. Compared with most of existing stability conditions, i.e., asymptotical stability and exponential stability, the obtained finite-time stability result is more attractive, and therefore could be considered as a useful supplement to the current literature. In addition, a switching structure is suggested to further speed up the neural network convergence. Moreover, by using the penalty function method, the proposed neural network can be extended straightforwardly to solving the constrained optimization problem. Finally, the satisfactory performance of the proposed approach is illustrated by two simulation examples.**

I. Introduction

Recently, various engineering problems, such as the robot control, control system design, signal and image processing, and pattern recognition, can be converted into the optimization problems. However, traditional numerical algorithms for solving the optimization problems may encounter serious speed bottleneck due to the serial nature of the digital computer employed. Therefore, how to solve the optimization problem in real time has been studied intensively. One promising approach is to employ the artificial neural network based on the hardware implementation.

In the past two decades, recurrent neural networks for the optimization purpose have been widely investigated. Since the seminal work of Tank and Hopfield [1], many Hopfield-type recurrent neural network models have been proposed to resolve various kinds of optimization problems [2]–[16]. Kennedy and Chua presented a nonlinear programming neural network based on the penalty method [2]. A two-layer Lagrange neural network was designed to deal with the optimization problem with equality constraints [3]. In [4], Xia proposed a primal-dual neural network for the linear and quadratic programming problems. In [5], Xia *et al.,* also proposed a projection neural network to solve the nonlinear optimization problem with the box-type or sphere-type inequality constraints. For the nonlinear optimization problem with general convex inequality constraints and affine equality constraints, the novel neural network models were studied in [6], [7], which represented the state of the art in this area. In [8], Hu and Wang further exploited the capability of projection neural network for the pseudoconvex optimization problem. Meanwhile, time delays in the signal communication among the neurons were taken into account, and the delayed projection neural network was proved to be still applicable to the nonlinear optimization problem [9]. By using the non-smooth analysis and the theory of differential inclusion, Forti *et al.* presented a generalized neural network for solving non-smooth nonlinear programming problems based on the gradient method [10]. In [11], Cheng *et al.* also generalized the projection neural network to solve the optimization problem with the non-smooth objective function and constraints. When the neural network is utilized to solve the quadratic programming problem, its architecture complexity and the implementation cost can be reduced. The early attempt was the dual neural network [12]. By eliminating the Lagrange multiplier associated with the equality constraint, the dual neural network could be further simplified [13], [14]. By employing the discontinuous hard-limiting activation function, a one-layer neural network was suggested whose number of neurons was equal to the dimension of the equality constraints [15]. The interested readers are referred to [16] for the fundamental idea of the neural-network-based optimization solver.

In parallel with the theory research of neural networks, several successful applications can also be founded, such as the support vector machine learning [17], the constrained model predictive control [18], and the hierarchical control of interconnected dynamic systems [19].

Inspired by the above discussions, a recurrent neural network is proposed to solve the convex optimization problem in this paper. It is noted that most of existing neural networks for the optimization purpose are only asymptotically stable or exponentially stable. The proposed neural network with a specific nonlinear unit is demonstrated to be convergent to its equilibrium point in finite time. Therefore, the obtained result can be considered as a useful supplement to the current literature. Extensions have been done to speed up the neural network convergence, and to solve the optimization problem with the linear equality and nonlinear inequality constraints. At last, simulation results are given to show the desired performance of the proposed neural network and substantiate the theoretical analysis.

The remainder of this paper is organized as follows. Section II introduces the problem formulation and some preliminary results. Section III gives the recurrent neural network

Long Cheng, Zeng-Guang Hou, Min Tan are with the the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. Emails: chenglong@compsys.ia.ac.cn; hou@compsys.ia.ac.cn; tan@compsys.ia.ac.cn).

Noriyasu Homma is with School of Health Sciences, Faculty of Medicine, Tohoku University, 2-1 Seiryo-machi, Aoba-ku, Sendai, Japan 980-8575. Email: homma@abe.ecei.tohoku.ac.jp.

Madam M. Gupta is with the Intelligent Systems Research Laboratory, College of Engineering, University of Saskatchewan, Saskatoon, Saskatchewan, S7N 5A9, Canada. Email: madan.gupta@usask.ca.

model. Section IV analyzes the finite-time convergence of the proposed neural network. Section V provides extensions to speed up the network convergence, and to solve the constrained optimization problem. Illustrative examples are shown and discussed in Section VI. Section VII concludes this paper with final remarks.

II. Preliminaries

First, the following unconstrained nonlinear convex optimization problem is considered

$$\min \ f(x), \tag{1}$$

where $x = (x_1, x_2, \cdots, x_n)^T \in \mathbb{R}^n$, and $f(x) : \mathbb{R}^n \to \mathbb{R}$ is the twice differentiable objective function.

Partial differentials are used throughout this paper. For the sake of discussion, let the gradient vector and Hessian matrix of $f(x)$ be, respectively, defined as

$$\frac{\partial f(x)}{\partial x} \triangleq \left(\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \cdots, \frac{\partial f(x)}{\partial x_n}\right)^T \in \mathbb{R}^n,$$

$$\frac{\partial^2 f(x)}{\partial x^2} \triangleq \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1 \partial x_1}, & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2}, & \cdots, & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1}, & \frac{\partial^2 f(x)}{\partial x_2 \partial x_2}, & \cdots, & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \cdots, & \cdots, & \ddots, & \cdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1}, & \frac{\partial^2 f(x)}{\partial x_n \partial x_2}, & \cdots, & \frac{\partial^2 f(x)}{\partial x_n \partial x_n} \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

Definition 1: A twice differentiable function $f(x)$ is called strongly convex on $\mathbb{R}^n$ if there exists $\beta > 0$ such that $\forall x \in \mathbb{R}^n$, $x^T \frac{\partial^2 f(x)}{\partial x^2} x \geq \beta x^T x$.

Apparently, if Q is a positive definite matrix, then the quadratic objective function $f(x) = \frac{1}{2} x^T Q x + b^T x + c$ is strongly convex.

The following nonlinear function will be used in the neural network dynamical equation.

$$\mathrm{sig}(r)^\alpha \triangleq (\mathrm{sign}(r_1)|r_1|^{\alpha_1}, \mathrm{sign}(r_2)|r_2|^{\alpha_2}, \cdots, \mathrm{sign}(r_n)|r_n|^{\alpha_n})^T, \tag{2}$$

where $r = (r_1, r_2, \cdots, r_n)^T \in \mathbb{R}^n$, $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n)^T \in \mathbb{R}^n$, $\alpha_i > 0$, $\forall i = 1, 2, \cdots, n$, and $\mathrm{sign}(\cdot)$ is the sign function defined by

$$\mathrm{sign}(r_i) = \begin{cases} 1 & \text{if } r_i > 0 \\ 0 & \text{if } r_i = 0 \\ -1 & \text{if } r_i < 0 \end{cases}, \quad i = 1, 2, \cdots, n.$$

The following two lemmas are useful to analyze the neural network stability.

Lemma 1: Let $V(t) : [0, \infty) \to [0, \infty)$ be the differentiable function ($V(t)$ is right-side differentiable at zero). If

$$\frac{dV(t)}{dt} \leq -kV^\gamma(t), \tag{3}$$

where $k > 0$ and $0 < \gamma < 1$, then

$$V(t) \equiv 0, \quad \forall t \geq t^* = \frac{V^{1-\alpha}(0)}{k(1-\alpha)}. \tag{4}$$

Proof: See the proof of Theorem 4.2 in [20]. ■

Lemma 2: Let $\zeta_1, \zeta_2, \cdots, \zeta_n \geq 0$ and $0 < \alpha \leq 1$, then

$$\left(\sum_{i=1}^n \zeta_i\right)^\alpha \leq \sum_{i=1}^n \zeta_i^\alpha. \tag{5}$$

Proof: See the proof of Lemma 2 in [21]. ■

III. Recurrent Neural Network Model

Recently, the parallel optimization capability of recurrent neural networks is recognized by researchers in many different disciplines. The essence of this computing approach is that: by setting up a neural circuit which is determined by a set of differential equations, for some initial network state, the neural networks will eventually evolve to an equilibrium state and this state will coincide with the optimal solution to the original problem.

In the literature, the following gradient-based neural network can be used to solve the convex optimization problem defined by (1),

$$\frac{dx}{dt} = -\Lambda \frac{\partial f(x)}{\partial x}, \tag{6}$$

where $\Lambda = \mathrm{diag}\{\mu_1, \mu_2, \cdots, \mu_n\}$, and $\mu_i > 0$, $\forall i = 1, 2, \cdots, n$. Λ controls the convergence rate of the neural network.

According to the methodology provided in [22], the neural network defined by (6) is asymptotically stable at its equilibrium point x^*, $\left.\frac{\partial f(x)}{\partial x}\right|_{x=x^*} = 0$. By the first-order optimality condition of convex function, x^* is the the optimal solution to (1).

In this paper, a novel recurrent neural network is proposed for the optimization problem defined by (1). This neural network can be convergent to its equilibrium point in finite time. The dynamical equation of the proposed neural network is stated as follows

$$\frac{dx}{dt} = -\Lambda\, \mathrm{sig}\left(\frac{\partial f(x)}{\partial x}\right)^\alpha. \tag{7}$$

It is easy to see that the right-hand of (7) is continuous at every point in $\mathbb{R}^n$, but is not Lipschitz continuous at the equilibrium point x^*, which gives the chance of having a different stability result compared with (6).

The block diagram of the proposed neural network is shown in Fig. 1.

IV. Stability Analysis

Theorem 1: If the objective function $f(x)$ in the optimization problem defined by (1) is strongly convex, then the trajectory of the proposed neural network defined by (7) is limited in a compact set. Furthermore, the trajectory of the proposed neural network will be convergent to the optimal solution of (1) in finite time.

Proof: Firstly, construct the following Lyapunov-type function

$$V_1(t) = f(x). \tag{8}$$

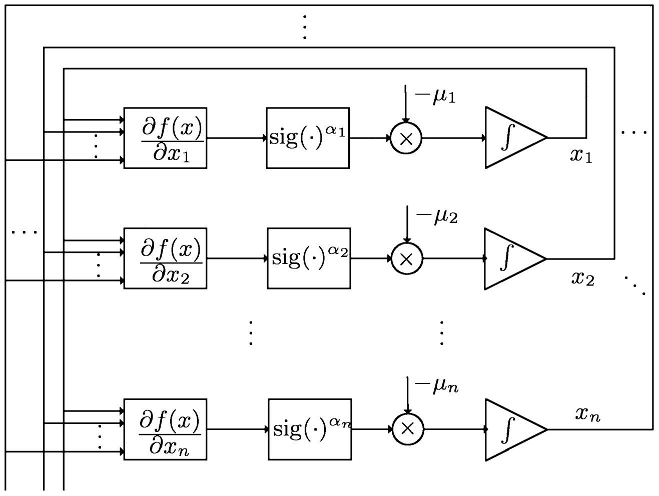
$\frac{\partial f(x)}{\partial x_1}$
$\mathrm{sig}(\cdot)^{\alpha_1}$
$-\mu_1$
x_1
$\frac{\partial f(x)}{\partial x_2}$
$\mathrm{sig}(\cdot)^{\alpha_2}$
$-\mu_2$
x_2
$\frac{\partial f(x)}{\partial x_n}$
$\mathrm{sig}(\cdot)^{\alpha_n}$
$-\mu_n$
x_n

Fig. 1. Block diagram of the proposed recurrent neural network.

Differentiating $V_1(t)$ with respect to time obtains

$$\begin{aligned}\frac{dV_1(t)}{dt} &= \sum_{i=1}^{n} \frac{\partial f(x)}{\partial x_i}\frac{dx_i}{dt} \\ &= -\sum_{i=1}^{n} \mu_i \frac{\partial f(x)}{\partial x_i} \operatorname{sign}\left(\frac{\partial f(x)}{\partial x_i}\right)\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i} \\ &= -\mu_0 \sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i+1} \\ &\leq 0. \end{aligned} \tag{9}$$

Therefore, the $f(x)$ is non-increasing with respect to time. Then, $f(x(t)) \leq f(x(0))$. Since $f(x)$ is a strongly convex function, it is easy to see that $f(x)$ is radially unbounded and has a lower bound. Therefore, it can be obtained that the trajectory of the proposed neural network, $x(t)$, is limited in a compact set.

Secondly, it is demonstrated that the proposed neural network will be convergent to its equilibrium point in finite time. Construct the following Lyapunov-type function

$$V_2(t) = \sum_{i=1}^{n} \frac{1}{\mu_i(\alpha_i+1)}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i+1}. \tag{10}$$

Differentiating $V_2(t)$ with respect to time obtains

$$\begin{aligned}\frac{dV_2(t)}{dt} &= \sum_{i=1}^{n} \frac{1}{\mu_i}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i} \operatorname{sign}\left(\frac{\partial f(x)}{\partial x_i}\right)\frac{d\left(\frac{\partial f(x)}{\partial x_i}\right)}{dt} \\ &= -\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i} \operatorname{sign}\left(\frac{\partial f(x)}{\partial x_i}\right) \\ &\quad \times \left(\sum_{j=1}^{n} \frac{\partial f^2(x)}{\partial x_i \partial x_j}\left|\frac{\partial f(x)}{\partial x_j}\right|^{\alpha_j} \operatorname{sign}\left(\frac{\partial f(x)}{\partial x_j}\right)\right) \\ &= -\operatorname{sig}^T\left(\frac{\partial f(x)}{\partial x}\right)^{\alpha} \frac{\partial^2 f(x)}{\partial x^2} \operatorname{sig}\left(\frac{\partial f(x)}{\partial x}\right)^{\alpha} \\ &\leq 0. \end{aligned} \tag{11}$$

Therefore, $V_2(t)$ is non-increasing with respect to time. Then it can be obtained that

$$\forall t > 0,\ i = 1, 2, \cdots, n, \quad \left|\frac{\partial f(x)}{\partial x_i}\right| \leq (\mu_i(\alpha_i+1)V_2(0))^{\frac{1}{\alpha_i+1}} \tag{12}$$

Let $\alpha_0 = \max_i \alpha_i$. Since $f(x)$ is a twice-differentiable strongly convex function, by (11), it follows that

$$\begin{aligned}\frac{dV_2(t)}{dt} &\leq -\beta \operatorname{sig}^T\left(\frac{\partial f(x)}{\partial x}\right)^{\alpha} \operatorname{sig}\left(\frac{\partial f(x)}{\partial x}\right)^{\alpha} \\ &= -\beta \frac{\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{2\alpha_i}}{(V_2(t))^{\frac{2\alpha_0}{1+\alpha_0}}}(V_2(t))^{\frac{2\alpha_0}{1+\alpha_0}}. \end{aligned} \tag{13}$$

According to Lemma 2 and $\frac{2\alpha_0}{1+\alpha_0} < 1$, it can be obtained that

$$\begin{aligned}\frac{\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{2\alpha_i}}{(V_2(t))^{\frac{2\alpha_0}{1+\alpha_0}}} &= \frac{\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{2\alpha_i}}{\left(\sum_{i=1}^{n}\frac{1}{\mu_i(\alpha_i+1)}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\alpha_i+1}\right)^{\frac{2\alpha_0}{1+\alpha_0}}} \\ &\geq \frac{\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{2\alpha_i}}{\left(\sum_{i=1}^{n}\left(\frac{1}{\mu_i(\alpha_i+1)}\right)^{\frac{2\alpha_0}{1+\alpha_0}}\left|\frac{\partial f(x)}{\partial x_i}\right|^{\frac{2\alpha_0(\alpha_i+1)}{1+\alpha_0}}\right)} \\ &\geq \frac{\left|\frac{\partial f(x)}{\partial x_s}\right|^{2\alpha_s}}{\left(\sum_{i=1}^{n}\left(\frac{1}{\mu_i(\alpha_i+1)}\right)^{\frac{2\alpha_0}{1+\alpha_0}}\right)\left|\frac{\partial f(x)}{\partial x_s}\right|^{\frac{2\alpha_0(\alpha_s+1)}{1+\alpha_0}}} \\ &= \frac{\left|\frac{\partial f(x)}{\partial x_s}\right|^{2\alpha_s-\frac{2\alpha_0(\alpha_s+1)}{1+\alpha_0}}}{\left(\sum_{i=1}^{n}\left(\frac{1}{\mu_i(\alpha_i+1)}\right)^{\frac{2\alpha_0}{1+\alpha_0}}\right)}, \end{aligned} \tag{14}$$

where $s = \arg_i \max_i \left|\frac{\partial f(x)}{\partial x_i}\right|^{\frac{2\alpha_0(1+\alpha_i)}{1+\alpha_0}}$.

Because $0 < \alpha_s \leq \alpha_0 < 1$, it can be proved that

$$2\alpha_s - \frac{2\alpha_0(\alpha_s+1)}{1+\alpha_0} = \frac{2(\alpha_s-\alpha_0)}{1+\alpha_0} \leq 0.$$

Then, according to (12), it follows that

$$\left|\frac{\partial f(x)}{\partial x_s}\right|^{\frac{2(\alpha_s-\alpha_0)}{1+\alpha_0}} \geq (\mu_s(\alpha_s+1)V_2(0))^{\frac{2(\alpha_s-\alpha_0)}{(1+\alpha_0)(1+\alpha_s)}}.$$

Let

$$K = \frac{1}{\left(\sum_{i=1}^{n}\left(\frac{1}{\mu_i(\alpha_i+1)}\right)^{\frac{2\alpha_0}{1+\alpha_0}}\right)} \times \min_i\left((\mu_i(\alpha_i+1)V_2(0))^{\frac{1}{\alpha_i+1}(2\alpha_i-\frac{2\alpha_0(\alpha_i+1)}{1+\alpha_0})}\right),$$

then

$$\frac{\sum_{i=1}^{n}\left|\frac{\partial f(x)}{\partial x_i}\right|^{2\alpha_i}}{(V_2(t))^{\frac{2\alpha_0}{1+\alpha_0}}} \geq K.$$

Therefore, by (13), it follows that

$$\frac{dV_2(t)}{dt} \leq -\beta K (V_2(t))^{\frac{2\alpha_0}{1+\alpha_0}}. \tag{15}$$

According to Lemma 1, it can be seen that $V_2(t)$ will be zero after $t^* = \frac{(1+\alpha_0)(V_2(0))^{\frac{1-\alpha_0}{1+\alpha_0}}}{\beta K(1-\alpha_0)}$. It is easy to see that $V_2(t) = 0$ implies that $\frac{\partial f(x)}{\partial x_i} = 0$, $i = 1, 2 \cdots, n$, which is the optimal solution of the optimization problem defined by (1). Hence, the trajectory of the proposed neural network is convergent to the optimal solution in finite time. ■

Remark 1: It is noted that most of existing neural networks for the optimization purpose are only asymptotically stable or exponentially stable. There are few results on the finite-time convergence of the recurrent neural network. In [6], the neural network can be convergent to the optimal solution in finite time, however, the scale parameter μ_i has to be a very large value. The neural network in this paper does not have this limitation. In [10], the neural network is capable of solving the optimization problem in finite time, however, the neural network contains the discontinuous activation function which is hard to be implemented in practice. In this paper, the right side of the dynamical equation of the proposed neural network is continuous. It is also noted that, a similar neural network structure is proposed in [23], however, there is no theoretical analysis on the network convergence. Therefore, the proposed neural network is a useful supplement to the current literature.

V. Extensions

According to the analysis in Section IV, the convergence time of the proposed neural network is closely related to the initial value of $V_2(t)$. If $V_2(0)$ is very large, then the neural network will converge slowly. In order to speed up the network convergence, the following switching structure is proposed. When $f(x)$ is strongly convex, it can be proved that the neural network defined by (6) is exponentially stable at its equilibrium point. Construct the Lyapunov-type function

$$V_3(t) = \frac{1}{2}\left(\frac{\partial f(x)}{\partial x}\right)^T \Lambda \frac{\partial f(x)}{\partial x}.$$

Differentiating $V_3(t)$ along the trajectory of (6) obtains that

$$\dot{V}_3(t) \leq -2\eta V_3(t),$$

where $\eta = \frac{\beta\mu_{\min}^2}{\mu_{\max}}$, $\mu_{\min} = \min_i \mu_i$ and $\mu_{\max} = \max_i \mu_i$. Hence,

$$V_3(t) \leq V_3(0)e^{-2\eta t}.$$

Therefore, it is obvious that the neural network defined by (6) has a good convergence speed when $\left\|\frac{\partial f(x)}{\partial x}\right\|$ is large, and the proposed neural network defined by (7) converges fast when $\left\|\frac{\partial f(x)}{\partial x}\right\|$ is small. Thus, by combining the two neural networks together will speed up the neural network convergence. For example, the neural network has the following dynamical equation

$$\frac{dx}{dt} = \begin{cases} -\Lambda\frac{\partial f(x)}{\partial x}, & t < t^s, \\ -\Lambda \operatorname{sig}\left(\frac{\partial f(x)}{\partial x}\right)^\alpha, & t \geq t^s, \end{cases} \tag{16}$$

where t^s is the first time when $\left\|\frac{\partial f(x)}{\partial x}\right\|_2^2 \leq \epsilon$, and ϵ is a given positive constant. Because the combined neural network only switches its structure once, it still preserves the finite-time convergence property.

In the following section, the proposed neural network is extended to solving the constrained optimization problem. Consider

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & g(x) \leq 0, \\ & h(x) = 0, \end{aligned} \tag{17}$$

where $f(x)$ is the strongly convex objective function; $g(x) = (g_1(x), g_2(x), \cdots, g_m(x))^T : \mathbb{R}^n \to \mathbb{R}^m$ is the inequality constraint, $g_i(x)$ is a differentiable convex function on $\mathbb{R}^n$; and $h(x) = (h_1(x), h_2(x), \cdots, h_r(x))^T : \mathbb{R}^n \to \mathbb{R}^r$, $h_i(x)$ is the affine function on $\mathbb{R}^n$.

By using the penalty method [24], the above constrained optimization problem can be converted into the following unconstrained optimization problem:

$$\min \quad E(x) = f(x) + \frac{\rho}{2}\left\{\sum_{i=1}^{m}\left(g_i^+(x)\right)^2 + \sum_{i=1}^{r}(h_i(x))^2\right\}, \tag{18}$$

where ρ is a positive penalty parameter, and $g_i^+(x) = \max\{0, g_i(x)\}$.

Then the following neural network can be used to solve (17).

$$\frac{dx}{dt} = \begin{cases} -\Lambda\left(\frac{\partial f(x)}{\partial x} + \rho\left(\nabla g(x)g^+(x) + \nabla h(x)h(x)\right)\right), \\ -\Lambda \operatorname{sig}\left(\frac{\partial f(x)}{\partial x} + \rho\left(\nabla g(x)g^+(x) + \nabla h(x)h(x)\right)\right)^\alpha, \end{cases} \tag{19}$$

where $\nabla g(x) = \left(\frac{\partial g_1(x)}{\partial x}, \frac{\partial g_2(x)}{\partial x}, \cdots, \frac{\partial g_m(x)}{\partial x}\right) \in \mathbb{R}^{n\times m}$, $\nabla h(x) = \left(\frac{\partial h_1(x)}{\partial x}, \frac{\partial h_2(x)}{\partial x}, \cdots, \frac{\partial h_r(x)}{\partial x}\right) \in \mathbb{R}^{n\times r}$, and $g^+(x) = \left(g_1^+(x), g_2^+(x), \cdots, g_m^+(x)\right)^T$.

According to the similar analysis in Section IV, the neural network defined by (19) can be convergent to its equilibrium point x^* in finite time. And x^* can approximate the optimal solution of (17) by the results given in [24].

VI. Simulation Examples

In this section, two examples are given to illustrate both the theoretical analysis and the performance of the proposed neural network. All the neural network dynamical equations are simulated by the fourth-order Runge-Kutta method.

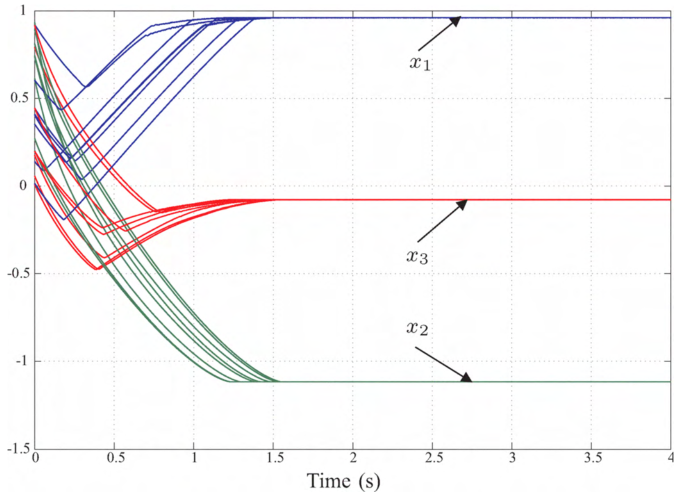

Fig. 2. The transient behavior of the neural network defined by (7) with 8 random initial points for solving (20).

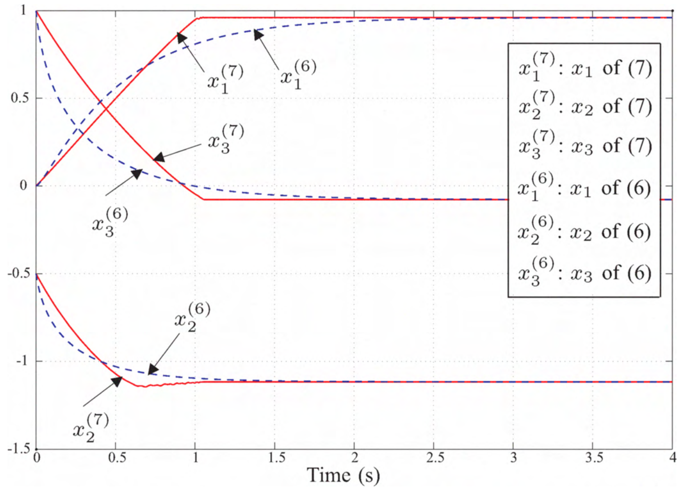

Fig. 3. Performance comparison between the neural network defined by (7) and neural network defined by (6) for solving (20).

A. Example One

Consider the following unconstrained optimization problem

$$\min \quad f(x) = (x_1-1)^2+(x_2+1)^2+e^{x_3^2}+e^{x_1+3x_2+2x_3}. \tag{20}$$

It is easy to check that the objective function is strongly convex. This problem has a unique solution $x^* = (0.9608, -1.1174, -0.0778)^T$, and $f(x^*) = 1.0097$. Then the proposed neural network defined by (7) can be employed to solve the above problem. Let $\alpha = (0.1, 0.3, 0.2)^T$ and Λ be the unity matrix. The neural network is simulated with eight initial points randomly distributed in $[0, 1]$. According to the simulation result, all trajectories are convergent to the equilibrium point x^* in finite time. Figure 2 shows the transient behavior of the neural network defined by (7).

In order to illustrate the intuition that the proposed neural network defined by (7) has a better performance than the neural network defined by (6) when $\left\|\frac{\partial f(x)}{\partial x}\right\|$ is small, the following comparison is made. Two neural networks defined by (7) and (6) are simulated with the same initial point $x(0) = (0, -0.5, 1)^T$, respectively. Let Λ be the unity matrix for both neural networks. And let $\alpha = (0.1, 0.3, 0.2)^T$. Figure 3 shows the transient behaviors of two neural networks. According to the simulation result, the proposed neural network defined by (7) reaches the equilibrium point before the neural network defined by (6).

When $\left\|\frac{\partial f(x)}{\partial x}\right\|$ is large, by using the combined neural network defined by (16), the satisfactory performance can also be obtained. Figure 4 shows the transient behaviors of two neural networks defined by (16) and (6). The neural network configuration is small as the above one. The network initial state is $(4, -4, 2)^T$, and $\epsilon = 2$. By the simulation result, it can be seen that the neural network defined by (16) can inherit both the advantages of neural networks defined by (6) and (7).

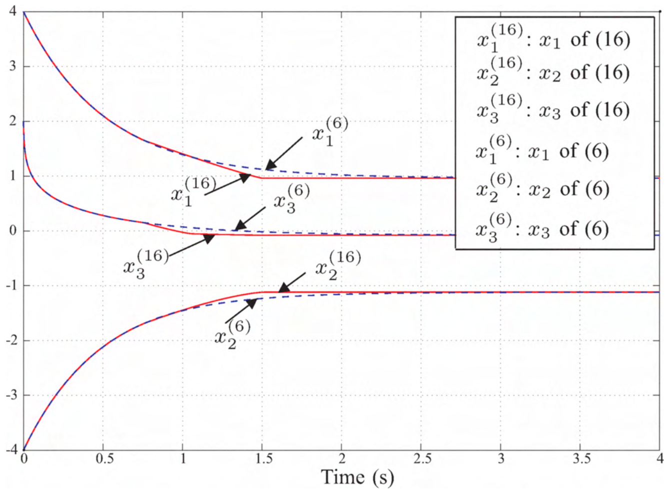

Fig. 4. Performance comparison between the neural network defined by (16) and neural network defined by (6) for solving (20).

B. Example Two

Consider the following constrained optimization problem with the same objective function of Example One.

$$\begin{aligned} \min \quad & f(x) = (x_1-1)^2+(x_2+1)^2+e^{x_3^2}+e^{x_1+3x_2+2x_3}, \\ s.t. \quad & x_1+x_2+x_3 = 1, \\ & x_1 \le 0.3, \\ & x_2 \ge 0, \\ & x_3 \ge 0.6. \end{aligned} \tag{21}$$

This problem has a unique optimal solution $x^* = (0.3, 0, 0.7)^T$, and $f(x^*) = 8.5963$. The proposed neural network defined by (19) is used to solve this problem. Here, the neural network parameters are set to be: $\alpha = (0.1, 0.3, 0.2)^T$, $\epsilon = 100$, $\rho = 10^4$, and Λ is the unity matrix. Figure 5 shows the profile of the trajectory of the proposed neural network. It is easy to see that the neural network trajectory reaches the equilibrium point in finite time. The

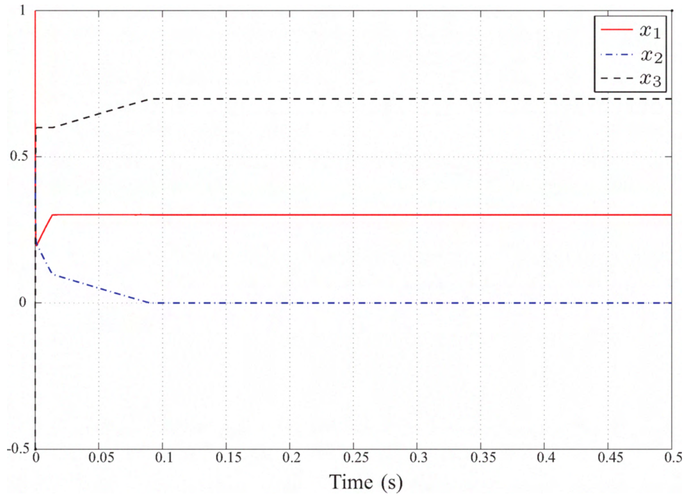

Fig. 5. Profile of the trajectory of the neural network defined by (19) for solving (21).

equilibrium point is $x^a = (0.3009, -0.005, 0.6983)^T$ which is very close to the optimal solution x^*. Hence, the proposed neural network is illustrated to have the capability of solving the constrained optimization problem.

VII. Conclusions

This paper proposes a recurrent neural network for solving optimization problems. The proposed neural network could be convergent to the optimal solution in finite time, which is more attractive than the existing asymptotically stable or exponentially stable results. In addition, by switching the neural network structure, the neural network convergence can be further speeded up. Furthermore, by using the penalty function method, the proposed neural network can be extended straightforwardly for the constrained optimization case. At last, the satisfactory performance of the proposed approach is demonstrated by simulation examples.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grants 60775043, 60725309, and 60621001).

References

[1] D.W. Tank and J.J. Hopfield, "Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Transcations on Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

[2] M.P. Kennedy and L.O. Chua, "Neural networks for nonlinear programming," *IEEE Transcations on Circuits and Systems*, vol. 35, no. 5, pp. 554–562, 1988.

[3] S. Zhang and A. G. Constantinides, "Lagrange programming neural networks," *IEEE Transcations on Circuits and Systems*, vol. 39, no. 7, pp. 441–452, 1992.

[4] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, pp. 1544–1547, 1996.

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

[6] Y. Xia, "An extened projection neural network for conostrained optimization," *Neural Computation*, vol. 16, no. 4, pp. 863–883, 2004.

[7] X. Gao, L. Liao, and L. Qi, "A novel neural network for variational inequalities with linear and nonlinear constraints", *IEEE Transactions on Neural Networks*, vol. 16, no. 6, pp. 1305–1317, 2005.

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

[9] L. Cheng, Z.-G. Hou, and M. Tan, "A neutral-type delayed projection neural network for solving nonlinear variational inequalities," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, no. 8, pp. 806–810, 2008.

[10] M. Forti, P. Nistri, and M. Quincampoix, "Generalized neural network for nonsmooth nonlinear programming problems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 9, pp. 1741–754, 2004.

[11] L. Cheng, Z.-G. Hou, M. Tan, X.Q. Wang, Z.S. Zhao, and S.Q. Hu, "A recurrent neural network for non-smooth nonlinear programming problems", in *Proceedings of International Joint Conference on Neural Networks*, Orlando, Florida, USA, Auguest, 2007, pp. 596–601.

[12] Y. Zhang and J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Physics Letters A*, vol. 298, no. 4, pp. 271–278, 2002.

[13] S. Liu and J. Wang, "A simplified dual neural network for quadratic programming with its KWTA application," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1500–1510, 2006.

[14] L. Cheng, Z.-G. Hou, M. Tan, and X.Q. Wang,, "A simplified recurrent neural network for solving nonlinear variational inequalities," in *Proceedings of International Joint Conference on Neural Networks*, Hongkong, June, 2008, pp. 104–109.

[15] Q. Liu and J. Wang, "A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 558-570, 2008.

[16] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*. Chichester, U.K.: Wiley, 1993.

[17] R. Perfetti and E. Ricci, "Analog neural network for support vector machine learning," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 1085–1091, 2006.

[18] L. Cheng, Z.-G. Hou, and M. Tan, "Constrained multi-variable generalized predictive control using a dual neural network," *Neural Computing and Applications*, vol. 16, no. 6, pp. 505–512, 2007.

[19] Z.-G. Hou, M.M. Gupta, P.N. Nikiforuk, M. Tan, and L. Cheng, "A recurrent neural network for hierarchical control of interconnected dynamic systems," *IEEE Transactions on Neural Networks*, vol. 18, no. 2, pp. 466–481, 2007.

[20] S.P. Bhat and D.S. Bernstein, "Finite-time stability of continuous autonomous systems," *SIAM Journal of Control and Optimization*, vol. 38, no. 3, pp. 751–766, 2000.

[21] F. Xiao, L. Wang, and Y. Jia, "Fast information sharing in networks of autonomous agents," in *Proceedings of American Control Conference*, Seattle, Washington, USA, June, 2008, pp. 4388–4393.

[22] Y. Xia and J. Wang, "A general methodology for designing globally convergent optimization neural networks," *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1331–1343, 1998.

[23] Z.-G. Hou, F.-S. Jing, M. Tan, "Using a nonlinear mechanism to speed up the neural optimization processes," in *Proceedings of IEEE International Symposium on Intelligent Control*, Vancouver, Canada, October, 2002, pp. 868–873.

[24] C. Y. Maa and M. A. Shanblatt, "Linear and quadratic programming neural network analysis," *IEEE Transactions on Neural Networks*, vol. 3, no. 4, pp. 580–594, Jul. 1992.