A General Methodology for Designing Globally Convergent Optimization Neural Networks

Youshen Xia, Member, IEEE, and Jun Wang, Senior Member, IEEE

Abstract—In this paper, we present a general methodology for designing optimization neural networks. We prove that the neural networks constructed by using the proposed method are guaranteed to be globally convergent to solutions of problems with bounded or unbounded solution sets, in contrast with the gradient methods whose convergence is not guaranteed. We show that the proposed method contains both the gradient methods and nongradient methods employed in existing optimization neural networks as special cases. Based on the theoretical results of the proposed method, we study the convergence and stability of general gradient models in case of unisolated solutions. Using the proposed method, we derive some new neural network models for a very large class of optimization problems, in which the equilibrium points correspond to exact solutions and there is no variable parameter. Finally, some numerical examples show the effectiveness of the method.

Index Terms—Design methodology, global convergence, optimization, recurrent neural networks.

I. INTRODUCTION

Optimization problems arise in a wide variety of scientific and engineering applications including signal processing, system identification, filter design, function approximation, regression analysis, and so on. In many engineering and scientific applications, the real-time solutions of optimization problems are widely required. However, traditional algorithms for digital computers may not be efficient since the computing time required for a solution is greatly dependent on the dimension and structure of the problems. One possible and very promising approach to real-time optimization is to apply artificial neural networks. Because of the inherent massive parallelism, the neural-network approach can solve optimization problems in running times at the orders of magnitude much faster than the most popular optimization algorithms executed on general-purpose digital computers. Hopfield and Tank [1], [2] first proposed a neural network for solving linear programming problems. Their seminal work has inspired many researchers to investigate alternative neural networks for solving linear and nonlinear programming problems. Many optimization neural networks have been developed [3]–[18]. For example, Kennedy and Chua [3] proposed a neural network for solving nonlinear programming problems, which employs both gradient method and penalty function method and its equilibrium points correspond to approximate optimal solutions only. Applying the gradient method and switched-capacitor technology, Rodriguez-Vazquez et al. [5] proposed a class of neural networks for solving optimization problems, in which their design does not require the calculation of a penalty parameter. Using gradient and projection methods [19], Bouzerdoum and Pattison [8] presented a neural network for solving quadratic optimization problems with bounded variables only, which constitutes a generalization of the network described by Sudharsanan and Sundareshan [7]. This neural network is exponentially stable to unique optimal solution through the appropriate choice of both the self-feedback and lateral connection matrices and under the strict convexity assumption. Based on dual and projection methods [20]–[23] Xia and Xia et al. [10]–[14] presented several neural networks for solving linear and quadratic programming problems with nonunique solutions, which are proved to be globally convergent to exact solutions, and in which there is no variable parameter to tune. Therefore, some of them are better than others in network computation and implementation. Besides some considerations in hardware implementation, a neural network with a good computational performance should satisfy threefold. First, the global convergence of the neural networks with an arbitrarily given initial state should be guaranteed. Second, the network design preferably contains no variable parameter. Third, the equilibrium points of the network should correspond to the exact or approximate solution. From a mathematical point of view, these characteristics are relevant to the optimization techniques employed for deriving optimization neural networks models. Prevalent and conventional methods are gradient methods which involve constructing an appropriate computational energy function for the optimization problem and designing a neural network model which performs some form of gradient descent on that function. The gradient methods have an advantage in that neural network models may be defined directly using the derivative of the energy function. But its shortcoming is that the convergence is not guaranteed, especially in the case of unbounded solution sets. Therefore, a rigorous analysis of a resulting network must be considered [12], [24]. In [10], [11], [13], and [14], some globally convergent neural networks with nongradient methods are presented. However, there is no deterministic procedure to be used directly to construct neural networks for solving other optimization problems. We thus feel that it is important to investigate deterministic methods such that designed neural networks are globally convergent and can solve problems with bounded or unbounded solution sets.

Manuscript received November 2, 1996; revised November 3, 1997 and September 9, 1998. This study was supported in part by a grant from the Hong Kong Research Grants Council (CUHK 381/96E).

The authors are with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong.

Publisher Item Identifier S 1045-9227(98)08810-9.
In this paper, we propose a general methodology for designing globally convergent optimization neural networks as an alternative to existing ones. The advantage of the proposed method is fourfold. First, the present method guarantees that any derived network is Lyapunov stable and globally convergent, and thus the complex stability analysis for each resulting network can be relaxed. Second, the neural network constructed by using the proposed method can cope with problems with unbounded solution sets. Third, the gradient methods and nongradient methods employed in existing optimization neural networks fall into the special cases of the proposed method, and thus the proposed method has more generality. Fourth, because the derived neural network is nonuniquc for the same optimization problem, it may provide more alternative neural networks for a well hardware implementation.

A neural network can operate in either continuous time or discrete time form. A continuous-time neural network described by a set of ordinary differential equations enables us to solve optimization problems in real time due to the massively parallel operations of the computing units and due to its real-time convergence rate. In comparison, discrete-time models can be considered as special cases of discretization of continuous-time models. Thus, in this paper we consider continuous-time neural networks only.

This paper is divided into seven sections. In Section II, we introduce some mathematical preliminaries on which the development and usage of the proposed method is based. In Section III, we describe the proposed method, and show its theory and characteristics. In Section IV, using the proposed method, we analyze the global convergence and stability of the general gradient models. In Section V, using the proposed method, we derive some new optimization neural networks for solving a very large class of optimization problems. In Section VI, three illustrative examples are discussed. Conclusions are found in Section VII.

II. PRELIMINARIES

This section provides the necessary mathematical backgrounds which are used to study the proposed method and its usage.

We are concerned with a general nonlinear program (P) of the following form:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad g(x) \leq 0 \\
& \quad h(x) = 0
\end{align*}
\]

where \( x \in \mathbb{R}^n, f: \mathbb{R}^n \to \mathbb{R}, g = [g_1, \ldots, g_m]^T: \mathbb{R}^n \to \mathbb{R}^m \) is an \( m \)-dimensional vector-valued continuous function of \( n \) variables, and \( h = [h_1, \ldots, h_r]^T: \mathbb{R}^n \to \mathbb{R}^r \) is an \( r \)-dimensional vector-valued continuous function of \( n \) variables. P is said to be a convex program (CP) if \( f \) and \( g_j \)'s are convex functions on \( \mathbb{R}^n \) and \( h_i \)'s are affine functions on \( \mathbb{R}^n \). A vector \( x \) is called a feasible solution to \( P \) if and only if \( x \) satisfies the \( r + m \) constraints of \( P \). The feasible solution \( x \) is said to be a regular point if the gradients \( \nabla g_j(x), \nabla h_j(x), 1 \leq j \leq r, i \in I = \{i | g_i(x) = 0\} \), are linearly independent.

We also consider the variational inequality problem \( VI(U, \Omega_0) \) of finding \( x^* \in \Omega_0 \) satisfying

\[
U(x^*)^T (x - x^*) \geq 0, \quad \forall x \in \Omega_0
\]

where \( \Omega_0 \subset \mathbb{R}^n \) is a closed convex set and \( U \) is a continuous function on \( \Omega_0 \). This problem is related to \( P \). For example, it contains CP as a special case [23]. Moreover, this problem has many important applications such as equilibrium models arising in fields of economics and transportation science, etc., [25], [26].

The following theorem contains a basic result concerning optimal solutions to \( P \).

1) **Saddle Point Theorem**: If there exists a point \((x^*, y^*, z^*)\) such that for all \( x \in \mathbb{R}^n \), and all \((y, z) \in \mathbb{R}^{m+r} \) with \( y \geq 0 \)

\[
L(x^*, y, z) \leq L(x^*, y^*, z^*) \leq L(x, y^*, z^*)
\]

then \( x^* \) is an optimal solution to \( P \), where \( L(x, y, z) = f(x) + y^T g(x) + z^T h(x) \) is referred to as a Lagrange function. 

Proof: See [27].

When \( f, g, \) and \( h \) are continuously differentiable, and there is a regular point \( \hat{x} \) which is feasible, from (3) it follows that \((x^*, y^*, z^*)\) satisfies the Kuhn–Tucker conditions below

\[
y \geq 0, \quad g(x) \leq 0, \quad h(x) = 0 \\
\nabla f(x) + \nabla g(x)y + \nabla h(x)z = 0, \quad y^T g(x) = 0
\]

where \( \nabla g(x) = (\nabla g_1(x), \ldots, \nabla g_m(x)), \quad \nabla h(x) = (\nabla h_1(x), \ldots, \nabla h_r(x)) \), and \( \nabla f(x), \nabla g_j(x), \nabla h_i(x) \) is the gradient of \( f, g_j, h_i \), respectively. If \( P \) is the optimization problem with equality constraints only, then the Kuhn–Tucker conditions become that \( h(x) = 0 \) and

\[
\nabla f(x) + \nabla h(x)z = 0.
\]

The Kuhn–Tucker conditions (4), generally speaking, are only the necessary conditions for \( x^* \) to be an optimal solution to \( P \), but the conditions (3) are sufficient conditions. Under the convexity assumptions of the functions, condition (3) or (4) is both sufficient and necessary. Moreover, \( x^* \) is a solution to CP if and only if \( x^* \) is a solution to the variational inequality problems \( VI(U, \Omega_0) \) where \( \Omega_0 = \{x \in \mathbb{R}^n | g_i(x) \leq 0, h_i(x) = 0\} \) is a closed convex set and \( U(x) = \nabla f(x) \).

A basic result concerning the problem \( VI(U, \Omega_0) \) is the following theorem.

2) **Projection Theorem**: \( x^* \) is a solution to \( VI(U, \Omega_0) \) if and only if \( x^* \) satisfies

\[
R_{\Omega_0}(x - \alpha U(x)) = x
\]

where \( \alpha \) is any a positive constant, and \( R_{\Omega_0}: \mathbb{R}^n \to \Omega_0 \) is a projection operator which is defined by

\[
R_{\Omega_0}(x) = \arg \min_{v \in \Omega_0} ||x - v||.
\]

Proof: See [23].

For the convenience of later discussions, some definitions and lemmas need to be introduced.
A vector \( x^* \) is said to be an isolated solution to \( P \) if there is a neighborhood of \( x^* \) which contains no other solutions to \( P \). Otherwise, \( x^* \) is an unisolated solution to \( P \).

If \( \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \), then any nonempty set of the form \( L(r) = \{ x \in \Omega | \phi(x) \leq r, r \in \mathbb{R} \} \) is a level set of \( \phi \).

Let \( x(t) \) be a solution of the system \( \dot{x} = f(x) \). The equilibrium point \( x^* \), that is \( f(x^*) = 0 \), is said to be stable in the sense of Lyapunov (that is, the system is Lyapunov stable at \( x^* \)) if for any \( x_0 = x(t_0) \) and scalar \( \epsilon > 0 \) there exists \( \delta > 0 \) such that if \( \| x(t_0) - x^* \| < \delta \), then \( \| x(t) - x^* \| < \epsilon \) for all \( t \geq 0 \). A system is said to be asymptotically stable at \( x^* \) if it is stable at \( x^* \) and there exists \( \delta > 0 \) such that if \( \| x(t_0) - x^* \| < \delta \), then \( x(t) \rightarrow x^* \) as \( t \rightarrow \infty \).

A neural network is said to be globally convergent if for any initial point taken in domain every trajectory of the corresponding dynamic system converges an equilibrium point which depends on the initial state of the trajectory. It is easy to see that if the equilibrium point is stable, then the neural network is asymptotically stable at the equilibrium point.

The mapping \( U : \Omega_0 \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n \) is Lipschitz continuous with constant \( \mathbb{L} \) on the set \( \Omega_0 \) if, for each pair of points \( x, y \in \Omega_0 \), we have

\[
\| U(x) - U(y) \| \leq \mathbb{L} \| x - y \|.
\]

The mapping is said to be locally Lipschitz continuous on \( \mathbb{R}^n \) if each point of \( \mathbb{R}^n \) has a neighborhood \( D_\delta \) such that the above inequality holds for each pair of points \( x, y \in D_\delta \).

The following two lemmas are relevant to the global convergence of neural networks.

**Lemma 1:** Let \( \phi : \Omega_1 \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \), where \( \Omega_1 \) is unbounded. Then all level sets of \( \phi \) are bounded if and only if \( \lim_{\| x \| \rightarrow \infty} \phi(x) = +\infty \) whenever \( x \in D \) and \( \lim_{\| x \| \rightarrow \infty} \| x \| = +\infty \).

**Proof:** See [28].

**Lemma 2:** Assume that \( \phi : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \) is continuous on the closed set \( \Omega \). Then \( \phi \) has a bounded level set if and only if the sets of global minimizers of \( \phi \) is nonempty and bounded.

**Proof:** See [28].

The following lemma provides a basis for model design.

**Lemma 3:** Assume that \( \phi \) is a continuously differentiable function. Then \( \phi \) is convex on the closed convex set \( \Omega_0 \) if and only if

\[
\phi(y) - \phi(x) \geq \nabla \phi(x)^T (y - x), \quad \forall x, y \in \Omega.
\]

**Proof:** See [28].

**Lemma 4:** Assume that \( \Omega_0 \) is a closed convex set. Then \( (v - P_{\Omega_0}(v))^T (P_{\Omega_0}(v) - u) \geq 0 \), \( v \in \mathbb{R}^n, u \in \Omega_0 \).

**Proof:** See [23].

### III. METHODOLOGY

In the present section, we describe the general method for designing optimization neural networks, and then show its theoretical significance.

#### A. Description of the Method

To formulate an optimization problem in terms of a neural network, there exist two types of methods. One approach commonly used in developing an optimization neural network is to first convert the constrained optimization problem into an associated unconstrained optimization problem, and then design a neural network that solves the unconstrained problem with gradient methods. Another approach [10], [11], [13], [14] is to construct a set of differential equations such that their equilibrium points correspond to the desired solutions and then find an appropriate Lyapunov function such that all trajectory of the systems converges to some equilibrium points. Combining the above two types of the methods, we propose a more general method. That is, a neural network is constructed by using the procedure below:

**Step 1:** Find a continuous function \( \Phi : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \) such that its minima correspond to the exact or approximate solutions to \( P \), where \( \Omega = \{ u = (u_1, \ldots, u_{n+1})^T | u_i \text{ or part satisfies some interval constraints} \} \).

**Step 2:** Construct a continuous vector valued function \( F : \Omega \subseteq \mathbb{R}^{n+1} \rightarrow \mathbb{R}^{n+1} \) such that \( F(u) = 0 \) and \( \Phi(u) \) satisfy

1. \( (u - u_0)^T F(u) \leq -\alpha(\Phi(u) - \Phi(u_0)) \), \( \forall u \in \Omega \);
2. \( \Phi(u) \) satisfies local Lipschitz conditions, \( \alpha > 0 \) and fixed.

**Step 3:** Let the neural-network model for solving \( P \) be represented by the following dynamic systems:

\[
\frac{du}{dt} = \Delta F(u), \quad u \in \Omega \tag{7}
\]

where \( \Delta = \text{diag}(\mu_1, \ldots, \mu_{n+1}) \), and \( \mu_i > 0 \) which is to scale the convergence rate of (7).

**Step 4:** Based on the systems in (7), design the neural-network architecture for solving \( P \).

A block diagram of the neural network is shown in Fig. 1, where the projection operator \( P_{\Omega}(u) \) enforces state vector \( u_L \) in \( \Omega \), which is defined by \( P_{\Omega}(u) = \{ P_{\Omega}(u_1), \ldots, P_{\Omega}(u_{n+1}) \}^T \) and for \( i \in L - I \), \( P_{\Omega}(u_i) \leq u_i \); for \( i \in I \), \( P_{\Omega}(u_i) \geq u_i \)

\[
P_{\Omega}(u_i) = \begin{cases} u_i & u_i < d_i \\ d_i & d_i \leq u_i \leq h_i \\ u_i & u_i > h_i \end{cases}
\]

where \( L = \{ 1, \ldots, n+l \} \) and \( I \subseteq L \).
In the aforementioned method, the first step is to establish a computational energy function such that the lowest energy state corresponds to the desired solution. Toward this, the basic approach is to transform the constrained problem \( P \) into an unconstrained problem, for which we need to find some nonlinear inequalities and equations through the saddle point theorem, the Kuhn–Tucker conditions (2) and (3), projection theorem, and some equivalent results concerning optimal solutions to \( P \) such as (5) and (6) \cite{29}, \cite{30}. The second step is to establish the relation between equilibrium points of the system in (7) and the lowest energy state and to ensure that the network trajectory globally converges to stable equilibrium, which possibly uses analytical properties of the function \cite{28}, the properties of the projection operator \cite{23}, and some optimization technique \cite{20}–\cite{23}, \cite{29}–\cite{33}.

**Remark 1:** The proposed method may be generalized to design neural networks for solving other computational problems such as \( VI(U, \Omega) \), etc.

**B. Theoretical Significance of the Method**

Having described the proposed method, we need to show its significance in theory. Its significance in term of methodology will be discussed after two sections.

**Theorem 1:** Any neural network derived from the proposed method is Lyapunov stable and globally convergent to an exact or approximate solution to \( P \).

**Proof:** Without loss of generality we assume that the set of minimizers of \( P \) is unbounded, and thus the set of global minimizers of \( \Phi \) is unbounded.

First, we know from the first step of the method that an exact or approximate solution to \( P \) corresponds to a minimizer of \( \Phi \), and from the second step that \( F(u) = 0 \) if and only if \( u \) is a minimizer of \( \Phi \). Thus it follows that \( F(u) = 0 \) if and only if \( u \) is an exact or an approximate solution to \( P \). That is, the equilibrium points of the system in (7) correspond to exact or approximate solutions to \( P \).

Next, by the existence theory of ordinary differential equations \cite{34} we see that for any an initial point taken in \( \Omega \) there exists an unique and continuous solution \( u(t) \subset \Omega \) for the systems in (7) over \([t_0, T]\) since the function \( F(u) \) satisfies local Lipschitz conditions.

Now, we consider the positive definite function

\[
V(u) = \frac{1}{2}||\Lambda_1(u - u^*)||_2^2, \quad u \in \Omega
\]  

(8)

where \( \Lambda_1 = \text{diag}(\mu_1^{-1/2}, \ldots, \mu_r^{-1/2}) \) and \( u^* \) is a fixed minimizer of \( \Phi \). From condition 2) we have

\[
\frac{d}{dt} V(u) = \frac{dV}{du} \frac{du}{dt} = (u - u^*)^T \Lambda_1^2 \nabla F(u) = (u - u^*)^T F(u) \leq -\alpha(\Phi(u) - \Phi(u^*)) \leq 0.
\]  

(9)

Thus

\[
||u(t) - u^*||_2 \leq \beta ||u(t_0) - u^*||_2 \quad \forall t \in [t_0, T)
\]

where \( \beta \) is a positive constant. Then the solution \( u(t) \) is bounded on \([t_0, T)\), and thus \( T = \infty \). Moreover, the system in (7) is Lyapunov stable at each equilibrium point.

On the other hand, since \( \lim_{k \to \infty} V(u^k) = +\infty \) whenever the sequence \( u^k \subset \Omega \) and \( \lim_{k \to \infty} ||u^k|| = +\infty \), by Lemma 2 we see that all level sets of \( V \) are bounded though all level sets of \( \Phi \) are unbounded, thus \( \Omega = \{u \in \Omega | V(u) \leq V(u^0)\} \) is bounded. Because \( V(u) \) is continuously differentiable on the compact set \( \Omega \) and \( \{u(t) | t \geq t_0 \} \subset \Omega \), it follows from the LaSalle’s invariance principle \cite{34} that trajectories \( u(t) \) converge to \( \Sigma \), the largest invariant subset of the following set:

\[
E = \left\{ u \in \Omega \left| \frac{dV}{du} = 0 \right. \right\}.
\]

Finally, let \( \lim_{k \to \infty} u(t_k) = \hat{u} \), then \( \hat{u} \in \Omega^* \). Therefore, for \( \forall \epsilon > 0 \) there exists \( q > 0 \) such that

\[
||\Lambda_1(u(t_k) - \hat{u})|| < \epsilon \quad k \geq q.
\]

Note that (9) holds for each \( u^* \in \Omega^* \), then \( ||\Lambda_1(u(t) - \hat{u})|| \) is decreasing as \( t \to \infty \). It follows that

\[
||\Lambda_1(u(t) - \hat{u})||_2 \leq ||\Lambda_1(u(t_q) - \hat{u})||_2 < \epsilon \quad t \geq t_q
\]

then

\[
\lim_{t \to \infty} ||\Lambda_1(u(t) - \hat{u})||_2 = 0.
\]

So

\[
\lim_{t \to \infty} u(t) = \hat{u}.
\]
Remark 2: Theorem 1 guarantees that any neural network designed by using the proposed method is Lyapunov stable and globally convergent. Thus an stability and convergence analysis for each resulting network may not be needed.

C. Features of the Method

Besides Remarks 1 and 2, we know that the neural networks derived from the proposed method can cope with problems with bounded or unbounded solution sets. In contrast to the analytical results of the existing neural networks, a feasible set or an isolated solution is assumed [6]–[18], [24], [35], [36], [40]. Next, from the description of method we see that for a given optimization problem the functions $\Phi(u)$ and $F(u)$ satisfying the first step and condition 1) in the second step are not unique. From Lemma 3 it follows that $\Phi(u)$ and $F(u) = -\nabla \Phi(u)$ always satisfy condition 2) in the second step when $\Phi(u)$ is continuously differentiable and convex on $\Omega$. Moreover, since the derived neural network model for solving an optimization problem may not be unique, we may choose as much as possible the model with few neurons, few interconnections, and simple hardware, for a good circuit implementation. More importantly, the proposed method generalizes both the gradient methods and nongradient methods employed in existing optimization neural networks. For example, consider the following linear programming problems (LP’s):

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Dx = b, \quad x \geq 0
\end{align*}
\] (10)

where $D \in \mathbb{R}^{m \times m}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. By the Kuhn–Tucker conditions we know that $x^*$ is an optimal solution to LP if and only if there exists $y^* \in \mathbb{R}^m$ such that $(x^*, y^*)$ satisfies

\[
\begin{align*}
Dx = b \\
(x + D^T y - c)^+ = x.
\end{align*}
\] (11)

Based on (11), we can construct functions

\[
\Phi(u) = ||Dx - b||^2_2 + ||(x + D^T y - c)^+ - x||^2_2
\] (12)

and

\[
F(u) = -\begin{cases} 
D^T(Dx - b) - r(D^T y - c) \\
rD[(x + D^T y - c)^+-b]
\end{cases}
\] (13)

where $r = \frac{||(x + D^T y - c)^+ - x||^2_2}{x^T(x^*) + \cdots + x_{n}^T(x^*)} - \max \{x_i, 0\}$, and $\Omega = \{(x, y) \in \mathbb{R}^{m+n} | x \geq 0\}$. It is easy to see that $\Phi(u)$ and $F(u)$ satisfy the first step and condition 1) and also condition 2) in the second step (see [10]). Let $\alpha = 1$, then the existing model [10] for solving LP is derived from the neural-network model described by (7). The model is not a gradient model since $F(u) \neq -\nabla \Phi(u)$. To construct a gradient model, we consider the equivalent Kuhn–Tucker conditions

\[
\begin{align*}
Dx = b, & \quad D^T y \leq c \\
c^T x - b^T y = 0, & \quad x \geq 0.
\end{align*}
\] (14)

Based on the new conditions we may construct the new functions

\[
\begin{align*}
\Phi(u) &= \frac{1}{2}||Dx - b||^2_2 + \frac{1}{2}((c^T x - b^T y)^2 \\
&\quad + \frac{1}{2}((D^T y - c)^+||^2_2 + \frac{1}{2}||(-x)^+||^2_2
\end{align*}
\] (15)

and

\[
\begin{align*}
F(u) &= -\nabla \Phi(u) \\
&\quad = \begin{cases} 
(c^T x - b^T y)x + D^T(Dx - b) - (-x)^+ \\
-(c^T x - b^T y)b + D(D^T y - c)^+ 
\end{cases}
\] (16)

where $[x] = (x_1, \cdots, x_n)^T$ and $\Omega = \{(x, y) \in \mathbb{R}^{n+m}\}$. Clearly, $\Phi(u)$ and $F(u)$ satisfy the first step and condition 1) in the second step. Moreover, they satisfy also condition 2) in the second step because of the convexity of $\Phi$. Let $\alpha = 1$, then the existing model [13] for solving LP can also be derived from the neural-network model defined by (7).

Besides LP problems, similarly there exist the functions $\Phi(u)$ and $F(u)$ satisfy conditions 1) and 2) in the existing neural networks [11], [13] for solving quadratic programming problems (QP) and extended LP problems [15]. Furthermore, in the next section we will see that there exist $\Phi(u)$ and $F(u)$ satisfy conditions 1) and 2) in general gradient models. Therefore, we can conclude that the gradient and nongradient methods employed in existing neural networks are special cases of the proposed method.

Remark 3: From Theorem 1 it follows that the existing model [10]–[14] can solve LP and QP problems with unbounded solution sets.

IV. GRADIENT MODELS

As an important application of the proposed general method, in this section, we analyze the stability and convergence of the general gradient models.

A. The Case of the Unconstrained Optimization Problems

Consider the following optimization problem:

\[
\begin{align*}
\text{minimize} & \quad E(x) \\
\text{subject to} & \quad x \in \mathbb{R}^n
\end{align*}
\] (17)

where $E: \mathbb{R}^n \rightarrow \mathbb{R}$ is a continuously differentiable function. It is well known that the gradient methods transform the minimization problem into an associated system of ordinary differential equations

\[
\frac{dx}{dt} = -\Lambda \nabla E(x), \quad x \in \mathbb{R}^n
\] (18)

where $\Lambda$ is defined as in (7) and $\nabla E(x)$ is the gradient of $E(x)$. In order to show that the above system of differential equations is Lyapunov stable, many researchers [4], [13], [17]–[19] define the function $V(x) = E(x) - E(x^*)$ where $x^*$ is a global minimizer of $E$, then the function $V(x)$ is always bounded from below and satisfies the following relation:

\[
\frac{dV}{dt} = \frac{dE}{dt} = \nabla E(x)^T \frac{dx}{dt} = -\nabla E(x)^T \Lambda \nabla E(x) \leq 0
\]
with
\[ \frac{dV}{dt} = 0 \iff \frac{dx}{dt} = 0 \]
since \( \Lambda \) is a positive definite matrix. If \( x^* \) is unique global minimizer of \( E \), then \( V(x) \) is positive definite in some neighborhood about \( x^* \). Thus the system in (18) is asymptotically stable at \( x^* \). If \( x^* \) is not unique global minimizer of \( E \) (that is, in each neighborhood of \( x^* \) there exists another point \( x^{**} \) which is global minimizer of \( E \) also), then \( V(x) \) is not positive definite in any neighborhood about the point \( x^* \). Thus the Lyapunov theorem can not conclude that the system in (18) is stable at nonisolated minimizer of \( E \). Moreover, from the above relation it is not guaranteed that the trajectories of the system can converge to stable equilibrium point (see [35] for an example). Therefore, it is necessary to give a complete result concerning the stability and convergence of the systems in (18). Using relation between the proposed method and the gradient methods we can get the general result below.

**Theorem 2:** Assume that the function \( E(x) \) is continuously differentiable and convex on \( R^n \). Then the neural network of (18) is Lyapunov stable and globally convergent to an optimal solution to problem (17).

**Proof:** In order to use the conclusion of Theorem 1 we need to show that the gradient model can be constructed by the proposed method. Since the function \( E(x) \) is continuously differentiable and convex on \( R^n \), \( x \) is a local minimizer of \( E \) if and only if \( x \) is a global minimizer of \( E \) and from Lemma 3 we have
\[ E(y) - E(x) \geq \nabla E(x)^T(y - x), \quad \forall x, y \in R^n. \]
Replacing \( y \) by \( x^* \) which is a global minimizer of \( E \), then
\[ E(x^*) - E(x) \geq \nabla E(x)^T(x^* - x), \quad \forall x \in R^n. \]
Thus
\[ (x - x^*)^T(-\nabla E(x)) \leq -(E(x) - E(x^*)), \quad \forall x \in R^n. \]
Let \( \alpha = 1, \Phi(x) = E(x), \) and \( F(x) = -\nabla E(x) \). Since
\[ \nabla E(x) = 0 \text{ if and only if } x \text{ is a global minimizer of } E \]
(i.e., a global minimizer of \( \Phi \)), and thus the functions \( \Phi(x) \) and \( F(x) \) satisfy both the first step and the second step in the proposed method. That is, the gradient model can be derived from the model (7). It follows by Theorem 1 that the neural network of (18) is Lyapunov stable and globally convergent to an optimal solution to the problem (17).

**Remark 4:** Theorem 2 gives a general result of both the stability and global convergence of the gradient model in (18) in the case of bounded or unbounded solution sets, in contrast to the existing results [35] in which unique or an isolated solution is assumed. Moreover, the results of Theorem 2 expand further the existing ones [12] since in that the conditions that \( E(x^*) = 0 \) is required and the proof is not given in the case of unbounded solution sets.

When \( E(x) \) is not convex on \( R^n \), \( E(x) \) is often convex at some region or neighborhood of minimizers of \( E \). For example, let \( E(x) = \ln(1 + x^2) - 2, x \in R \), then \( E(x) \) is not convex on \( R \) but is convex at the region \((-1, 1) \) of unique minimizer of \( E \), which is \( x^* = 0 \). Therefore, if we choose the

corollary 2: Assume that \( P \) is convex program. Then for any fixed \( s > 0 \), the neural network of (20) is Lyapunov stable and globally convergent to a minimizer of \( E_s \). Moreover, the minimizer is an approximate solution to \( P \) when \( s \) is sufficiently large.
Proof: For any fixed \( s > 0 \), because \( f(x) \), \( g_i(x) \), and \( h_j(x) \) are continuously differentiable and convex on \( R^n \), the function \( E_s(x) \) is also. Using Theorem 2 and the penalty function theorem [27], we can obtain the proof of Corollary 2.

Remark 6: The result of Corollary 2 has the practical importance in two aspects. One is that a very large penalty parameter is difficult to implement in a circuit. Another is that for a large optimization problem, assuming a bounded solution set needs to add many bounding constraints into the original problem.

Remark 7: In the case of sufficiently large penalty parameters, the set of minimizers of \( E_s(x) \) is unbounded also when the set of minimizers of CP is unbounded. Thus all level sets of the function \( E_s(x) \) are unbounded. Using the Corollary 2 and the penalty function Theorem, we can obtain the proof of Corollary 3.

V. NEW MODELS

To illustrate further the advantage of the proposed method in model design, in this section, we derive some new neural-network models by use of the proposed method for a large class of optimization problems, in which equilibrium points give exact solutions and there is no variable parameter in the models.

We see easily from the proposed method that the key is that \( u^* \) is a minimizer of \( \Phi \) if and only if \( u^* \) corresponds to an exact solution to \( P \) since \( u^* \) is a minimizer of \( \Phi \) if and only if \( F(u^*) = 0 \). To do this, previously mentioned results concerning optimal solutions to \( P \) need to be used. In addition, to find the function \( F(u) \) such that \( F(u) \) and \( \Phi(u) \) satisfy conditions 1) and 2), we usually use gradient methods when \( \Phi(u) \) is a continuously differentiable function and use the property of projection operator and some optimality conditions when \( \Phi(u) \) is only continuous.

A. Models for Solving \( P \)

From the Kuhn–Tucker conditions we see that the following function:

\[
\Phi(u) = \frac{1}{3} \sum_{i=1}^{m} [g_i^+(x)]^3 + \frac{1}{2} \| \nabla f(x) + \nabla g(x)y + \nabla h(x)z \|_2^2 + \frac{1}{2} \| y^T g(x) \|_2^2 + \frac{1}{2} \| [h(x)]_2 \|_2^2
\]

(23)

ensures that \( u^* = (x^*, y^*, z^*) \) is a minimizer of \( \Phi \) if and only if \( u^* \) is a Kuhn–Tucker point. Let the gradient model for solving \( P \) be

\[
\frac{du}{dt} = \Lambda F(u) = -\Lambda \nabla \Phi(u)
\]

where \( u \in \Omega = \{ (x, y, z) \in R^{m+n+p} \mid y \geq 0 \} \), then by Corollary 3 we get easily the following result concerning stability and convergence of the gradient model.

Corollary 4: Assume that \( f(x) \), \( g(x) \), and \( h(x) \) are twice continuously differentiable and convex on \( R^n \), then the neural network described above is Lyapunov stable and is at least locally convergent to an optimal solution to the problem \( P \).

Note that if the above mentioned function \( \Phi(u) \) is convex on \( R^{m+n+p} \), then the conclusion of Corollary 4 holds globally over \( R^n \). For example, we consider the following quadratic program with equality constraints:

\[
\text{minimize} \quad \frac{1}{2} x^T Ax + a^T x
\]

subject to \( Dx = b \)

(24)

where \( D \in R^{n \times m}, b \in R^m, a \in R^p, \) and \( A \in R^{m \times n} \) is positive semidefinite. By the Kuhn–Tucker condition (4) we know that \( (x^*, y^*) \) is a Kuhn–Tucker point if and only if \( u^* = (x^*, y^*) \) is a minimizer of the following function:

\[
\Phi(u) = \frac{1}{2} || Dx - b ||_2^2 + \frac{1}{2} || Ax + DT y + a^T ||_2^2
\]

(25)

Let \( F(u) = -\nabla \Phi(u) \), then gradient model for solving (24) is

\[
\frac{du}{dt} = -\Lambda \left\{ D^T (Dx - b) + AT^T (Ax + DT y + a) \right\}
\]

\[
D^T (Ax + Dy + a)
\]

(26)

where \( \Lambda \) is defined as in (7) and \( u \in \Omega = R^{m+n} \). Then \( F(u) \) and \( \Phi(u) \) satisfy conditions 1) and 2) since \( \Phi(u) \) is continuously differentiable and convex on \( \Omega \). So from Corollary 4 we see that the neural network for solving (24) is Lyapunov stable and globally convergent to an optimal solution to the problem (24). In contrast to the existing second-order neural network [16] for solving (24), which solves (24) with an unique solution only and has a complex model structure.
B. Models for Solving Nonlinear Inequality and Linear Equality Systems

\begin{equation}
\begin{cases}
g(x) \leq 0 \\
Ax = b
\end{cases}
\end{equation}

where \( g(x) \) is defined in (1) and is continuously differentiable, \( A \in \mathbb{R}^{n \times m} \), and \( b \in \mathbb{R}^n \). This problem arises in numerous fields, e.g., in optimization with linear and nonlinear constraints which is solved by interior point methods [37] as a preliminary step. To solve (27), we choose the following functions:

\begin{equation}
\Phi(x) = \frac{1}{2}||Ax - b||^2 + \frac{1}{2}||g(x)||^2
\end{equation}

and

\begin{equation}
F(x) = -AT(Ax - b) - \nabla g(x)g(x)^T
\end{equation}

where \( \nabla g(x) \) and \( g(x)^T \) is defined in (20). Let the new model for solving (27) be the system in (7), then we have the result concerning its stability and convergence as follows.

**Corollary 5:** Assume that \( \Omega^* = \{x \in \mathbb{R}^n | x \text{ satisfies (27)} \} \) is a nonempty set. Then the neural network described by the system in (7) is Lyapunov stable and globally convergent to an exact solution to (27).

**Proof:** First, we know easily that \( x^* \in \Omega^* \) if and only if \( \Phi(x) = 0 \) and \( F(x) = -\nabla \Phi(x) \). Therefore, functions \( \Phi(x) \) and \( F(x) \) satisfy the first step and condition 1) in the second step. We next see that \( \Phi(x) \) is continuously differentiable and convex on \( \Omega^* \) since \( g(x) \) is continuously differentiable and convex on \( \mathbb{R}^n \). It follows by Theorem 2 that Corollary 5 holds. Note that the new model can solve linear inequality and equality systems. In contrast to existing neural networks [15], which solve only linear inequalities.

C. Models for Solving Linear Complementary Problems

We consider a class of linear complementary problems below: Find a vector \( x \in \mathbb{R}^l \) such that

\begin{equation}
\begin{cases}
x_i(Mx + q) = 0, & (Mx + q)_i \geq 0, \quad x_i \geq 0 \quad \forall i \in I \\
(Mx + q)_j = 0, & \forall j \in L - I
\end{cases}
\end{equation}

where \( L = 1, \ldots, l \), \( I \subset L \), \( q \in \mathbb{R}^l \), and \( M \in \mathbb{R}^{l \times l} \) is a positive semidefinite matrix. The problem (30) has been recognized as a unifying description of a wide class of problems including sets of piecewise linear equations, fixed point problems and bimatrix equilibrium points [38]. In electrical engineering applications, (30) is used for the analysis and modeling of piecewise linear resistive circuits [25]. According to [39], \( x^* \) is a solution to (30) if and only if \( x^* \) satisfies the following equations:

\end{equation}

\begin{equation}
P_\Omega(x - \omega U(x)) = x
\end{equation}

where \( \Omega = \{x \in \mathbb{R}^l | x_i \geq 0, \forall i \in I \} \) and \( P_\Omega(x) \) denotes the projection onto the set \( \Omega \). Thus we construct the functions

\begin{equation}
\Phi(x) = ||P_\Omega(x - Mx - q) - x||^2
\end{equation}

and

\begin{equation}
F(x) = (I + M^T)(P_\Omega(x - Mx - q) - x).
\end{equation}

Let the new model for solving (30) be the system in (7), then the result concerning global stability and convergence is as follows.

**Corollary 6:** Assume that \( \Omega^* = \{x \in \mathbb{R}^l | x \text{ satisfies (30)} \} \) is a nonempty set. Then the new neural network is Lyapunov stable and globally convergent to a solution to (30).

**Proof:** The proof is given in the Appendix.

**Remark 8:** Friesz et al. proposed the dynamic system below [40]

\begin{equation}
\frac{dx}{dt} = (P_\Omega(x - Mx - q) - x).
\end{equation}

Yet, its stability and convergence cannot be guaranteed.

**Remark 9:** A general linear and convex quadratic programming problem can be transformed to the above problem [30]. So the new model can solve general linear and quadratic programming problems as well. In addition, the new neural network can be implemented by using simple hardware without any analog multiplier for variables.

D. Models for Solving Monotone Variational Inequalities

We consider the following monotone variational inequality problem \( VI(U, \Omega) \) of finding \( x^* \in \Omega \) satisfying

\begin{equation}
U(x^*)^T(x - x^*) \geq 0, \quad \forall x \in \Omega
\end{equation}

where \( \Omega = \{x \in \mathbb{R}^l | \text{some elements of } x \text{ satisfy some interval constraints} \} \) and \( U \) is a continuous and monotone function on \( \Omega \). For the problem (34), we define the function

\begin{equation}
\Phi(x) = ||P_\Omega(x - \omega U(x)) - x||^2
\end{equation}

and let [31]

\begin{equation}
F(x) = -\{z + \omega U(P_\Omega(z)) - P_\Omega(z)\}
\end{equation}

where \( z = x - \omega U(x) \), \( P_\Omega(z) \) denotes the projection onto set \( \Omega \) and \( \omega \in (0, +\infty) \). Note that \( x \) is a solution to \( VI(U, \Omega) \) if and only if for any \( \alpha > 0 \), \( u \) is a fixed point of the projection map

\begin{equation}
P_\Omega(x - \alpha U(x)) = x
\end{equation}

Thus if \( F(x) = 0 \), then

\begin{equation}
P_\Omega(z - \alpha U[P_\Omega(z)]) = z
\end{equation}

where \( z = x - \alpha U(x) \), and hence

\begin{equation}
P_\Omega(P_\Omega(z) - \alpha U[P_\Omega(z)]) = P_\Omega(z)
\end{equation}

So \( P_\Omega(z) \) is a solution of \( VI(U, \Omega) \). Then \( x = P_\Omega(z) \), and thus \( x \) solves \( VI(U, \Omega) \). Conversely, if \( x \) is a solution of \( VI(U, \Omega) \), then \( x = P_\Omega(z) \), thus \( U[P_\Omega(z)] = U(x) \), and hence \( F(x) = 0 \). So \( x \) is a solution satisfying \( F(x) = 0 \) if and only if \( x \) is a solution of \( VI(U, \Omega) \).

Now, assume that \( U(x) \) satisfies

\begin{equation}
(x - z)^T(U(x) - U(z)) \leq \lambda ||x - z||^2, \quad \forall x, z \in \mathbb{R}^l
\end{equation}

(For example, we take \( \lambda \) to be the Lipschitz constant of \( U \)), and let the new model for solving (34) be the system in (7), then the result concerning its global stability and convergence as is follows.
Corollary 7: Assume that $\Omega^\ast = \{x \in \mathbb{R}^d | x \text{ satisfies (34)}\}$ is a nonempty set and $\alpha < 1/\lambda$. Then the new neural network is Lyapunov stable and globally convergent to an exact solution to (34).

Proof: The proof is given in the Appendix.

Remark 10: If all bounds are infinite, then $VI(U, \Omega)$ is reduced to the problem of finding a solution of the system of equations $U(x) = 0$. If $\Omega = \{x \in \mathbb{R}^d | x \geq 0\}$, then $VI(U, \Omega)$ is just the well-known nonlinear complementarity problem

$$x \geq 0, \quad U(x) \geq 0, \quad x^T U(x) = 0.$$  

Moreover, there are a number of optimization problems which is this special class of the variational inequalities. For example, when the set $\Omega$ is polyhedral

$$\Omega = \{Bx = d, x \geq 0\}$$

where $B$ is an $m \times I$ matrix, then there exist $y \in \mathbb{R}^m$, using linear programming duality, such that $x$ solves $VI(U, \Omega)$ if and only if $(x, y)$ satisfies

$$U(x) - B^T y \geq 0, \quad x^T (U(x) - B^T y) = 0$$

$$Bx = d, \quad x \geq 0$$

which is easily seen to be equivalent to the monotone variational inequality $VI(G, \Omega)$ where the set $\Omega = \{(x, y) | x \geq 0, y \text{ unconstrained}\}$ and

$$G(x, y) = \left\{ \begin{array}{c} U(x) - B^T y \\ Bx = d \end{array} \right\}.$$  

Therefore, despite the particular structure of the feasible set $\Omega$, $VI(U, \Omega)$ is a very general problem.

Remark 10: The conclusion of Corollary 7 holds for any closed convex set.

VI. SIMULATION RESULTS

In this section, we simulate the effectiveness of the proposed method through three illustrative examples.

Example 1: This example illustrates the global convergence of a neural network for solving the problem with an unbounded solution set. Consider the following quadratic optimization problem:

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T Ax + c^T x \\
\text{subject to} & \quad D x \geq b, \quad x \geq 0
\end{align*}$$

where

$$A = \begin{bmatrix} 2 & -2 & 0 \\ -2 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \quad c = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix},$$

$$D = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad b = [2].$$

This problem has an unbounded solution set $\Omega^\ast = \{(x_1, x_2, 0.5) | x_1 - x_2 = 2\}$. It is easy to see that its dual problem is

$$\begin{align*}
\text{maximize} & \quad b^T y - \frac{1}{2} x^T Ax \\
\text{subject to} & \quad D^T y - Ax \leq c, \quad y \geq 0
\end{align*}$$

where $x = (x_1, x_2, x_3)^T$ and $y \in \mathbb{R}$. From Lagrangian duality [30], one can see that $x^\ast$, $y^\ast$ is an optimal solution to (37) and (38), respectively, if and only if $(x^\ast, y^\ast)$ satisfies

$$c + A^T x - D^T y \geq 0, \quad x \geq 0, \quad x^T (c + A^T x - D^T y) = 0$$

$$D x - b \geq 0, \quad y \geq 0, \quad y^T (D x - b) = 0$$

which can define the problem (30) as a special case, where

$$M = \begin{bmatrix} A & -D^T \\ D & 0 \end{bmatrix}, \quad q = \begin{bmatrix} c \\ -b \end{bmatrix}$$

and $\Omega = \{(z, x, y) \in \mathbb{R}^d | z \geq 0\}$. Notice that matrix $M$ is positive semidefinite since $z^T M z = x^T A x \geq 0$. Thus, we solve the above problem (37) by using the proposed new model

$$\frac{dx}{dt} = (I + M^T) (z - M z - q)^+ - z$$

where $(z)^+ = \max\{(x_1)^+, (x_2)^+, (x_3)^+, (y)^+\}$, $(y)^+ = \max\{0, y\}$, $(x_1)^+ = \max\{0, x_1\}$, and

$$M = \begin{bmatrix} 2 & -2 & 0 & -1 \\ -2 & 2 & 0 & 1 \\ 0 & 0 & 2 & 0 \\ 1 & -1 & 0 & -2 \end{bmatrix}, \quad q = \begin{bmatrix} 0 \\ 0 \\ -1 \\ -2 \end{bmatrix}.$$  

Fig. 2(a)–(c) shows that the trajectories of the new model with three different initial points can converge to three different solutions, respectively.

Example 2: This example illustrates the convergence of both the gradient model and nongradient model for solving a nonconvex problem. Consider the following quadratic program:

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T Ax + c^T x \\
\text{subject to} & \quad 0 \leq x_i \leq 1, \quad (i = 1, 2, 3)
\end{align*}$$

where

$$A = \begin{bmatrix} 1 & 1.5 & 0 \\ 1.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} -5 \\ -5 \\ 0 \end{bmatrix}.$$  

Since the matrix has a negative eigenvalue $-0.5$, this problem is not convex. But it has an optimal solution $(1, 1, 0)^T$. By the Karush–Kuhn–Tuker conditions, $x^\ast$ is an optimal solution to (40) if and only if $x^\ast$ satisfies

$$(x - x^\ast)^T (Ax^\ast + c) \geq 0, \quad \forall x \in \Omega$$

which is equivalent to (31), where $\Omega = \{0 \leq x_i \leq 1, i = 1, 2, 3\}$. So we use the proposed new system

$$\frac{dx}{dt} = (I + M^T) (P_2 (x - Ax - c) - x)$$

to solve (40). Fig. 3(a) and (b) shows the trajectories of the system with two different initial points converges the optimal solution $(1, 1, 0)^T$. If we solve (40) by using the gradient model

$$\frac{dx}{dt} = -(Ax + c)$$

then its trajectories converge a point $[2, 2, 0]^T$ which is not an optimal solution or is divergent as different initial points as shown in Fig. 3(c).
Example 3: This example illustrates the convergence of the new model for solving nonlinear equations. Consider the nonlinear equations below

\[
\begin{align*}
U_1(x_1, x_2) &= x_1^2 - |x_2| = 0 \\
U_2(x_1, x_2) &= x_1^2 - x_2 = 0.
\end{align*}
\]

From Remark 10, we see that the new model for solving the nonlinear equations is

\[
\frac{dx}{dt} = -\alpha U(x - \alpha U(x))
\]

where \(U(x) = (U_1(x_1, x_2), U_2(x_1, x_2))^T\) and \(\alpha\) is a positive constant. Although the above problem does satisfy the monotone condition, we still obtain its solutions by using the new model. Let \(\alpha = 0.1\), Fig. 4(a) and (b) shows that the trajectories of the model with two different initial points converges an solution \((1, 1)^T\).

VII. CONCLUSIONS

We have proposed a general methodology for designing optimization neural networks. We have proved that the neural networks constructed by using the proposed method are Lyapunov stable and globally convergent to an exact or approximate solution to optimization problems. Thus the method can relax the complex analysis concerning convergence and stability for each resulting network model. The results from the present method show that the neural networks designed can solve problems with a bounded or unbounded solution set, in comparison with the gradient methods in which a bounded feasible set or an isolated solution is assumed. In addition, the proposed method may be generalized to be used to design neural networks for solving other computational problems replacing the optimization problems, and thus has more wide applicability.

More importantly, we have shown that both the gradient methods and nongradient methods employed in existing
optimization neural networks fall into the special cases of the proposed method. Based on the results of the proposed method, we analyze the stability and convergence of the general gradient models. For unconstrained optimization problems, the gradient models are Lyapunov stable and globally convergent to an optimal solution to the convex problems, and are Lyapunov stable and at least locally convergent to an optimal solution to the nonconvex problem. For constrained optimization problems, the gradient models with a penalty parameter are Lyapunov stable and globally convergent to an approximate solution to CP, and are Lyapunov stable and at least locally convergent to an approximate solution to P. Thus we give a complete proof concerning the stability and convergence of Kennedy–Chua network and improve further the results given by Maa and Shanblatt in that the conditions of both a sufficiently large penalty parameters and a bounded solution set are removed. Using the proposed method we derive some new neural-network models for solving a very large class of optimization problems, in which an equilibrium point corresponds to an exact solution and there is no variable parameter in the models. Furthermore, since \( \Phi(u) \) and \( F(u) \) satisfying conditions 1) and 2) are not unique, the derived neural network model for solving the same optimization problem is not unique, and thus we may choose as much as possible the network model with fewer neurons, fewer interconnections, and simpler hardware, for an easier circuit implementation. The proposed method is important for the analysis and the design of the globally convergent neural networks.

**APPENDIX**

**Proof of Corollary 6:** We see first that \( x^* \in \Omega^* \) if and only if \( \Phi(u^*) = 0 \). Therefore, \( \Phi(u) \) and \( F(u) \) satisfy the first step and condition 1) in the second step. Similar to the proof of [21], by the fact that \( \Pi(u - Mu - q) \) is the projection of...
Since $x^*$ is a solution of (30), then
\[
\{P_2(x - Mx - q) - x^*\}^T\{Mx^* + q\} \geq 0, \quad \forall x \in \mathbb{R}^d.
\]
Adding the two resulting inequalities yields
\[
\{x^* - P_2(x - Mx - q)\}^T
\cdot \{Mx - Mx^* + P_2(x - Mx - q) - x\} \geq 0.
\]
Adding the two resulting inequalities yields
\[
(x^* - x)^T M(x^* - x)
\leq (x - x^*)^T(I + M^T)(x - P_2(x - Mx - q))
\leq ||x - P_2(x - Mx - q)||_2^2.
\]
Note that $(x - x^*)^T M(x - x^*) \geq 0$. Thus, we have
\[
(x - x^*)^T(I + M^T)(x - P_2(x - Mx - q))
\leq ||x - P_2(x - Mx - q)||_2^2.
\]
Thus
\[
(x - x^*)^T F(x) \leq -(\Phi(x) - \Phi(x^*))
\]
So $\Phi(x)$ and $F(x)$ satisfy conditions 1) and 2). By Theorem 1 we can obtain the proof of Corollary 6.

**Proof of Corollary 7:** For any fixed $\alpha > 0$, we see that $u^* \in \Omega^*$ if and only if $\Phi(u^*) = 0$. Therefore, $\Phi(u)$ and $F(u)$ satisfy the first step and condition 1) in the second step. Similarly to a technique of the proof in [32], by properties of the projection operator we have for $\forall x \in \mathbb{R}^d$
\[
\{y - P_2(x - \omega U(x))\}^T
\cdot \{\omega U(x) + P_2(x - \omega U(x)) - x\} \geq 0, \quad \forall y \in \Omega.
\]
Let $x^* \in \Omega^*$, then
\[
U(x^*)^T(y - x^*) \geq 0, \quad \forall y \in \Omega.
\]
Taking $y = x^*$ in the first inequality and taking $y = P_2(x - \omega U(x))$ in the second inequality and then adding the two resulting inequalities yields
\[
0 \leq \{x^* - P_2(x - \alpha U(x))\}^T
\cdot \{P_2(x - \alpha U(x)) + \alpha U(x) - x - \alpha U(x^*)\}
\]
than
\[
0 \leq \alpha\{x^* - P_2(x - \alpha U(x))\}^T
\cdot \{U[P_2(x - \alpha U(x))] - U(x^*)\}
\leq ||P_2(x - \alpha U(x)) - x||_2^2
\]
\[
+ \alpha\{x - P_2(x - \alpha U(x))\}^T
\cdot \{U(x) - U[P_2(x - \alpha U(x))]\}
\leq ||x - P_2(x - \alpha U(x)) - x||_2^2
\]
\[
+ \alpha\{x - P_2(x - \alpha U(x))\}^T
\cdot \{P_2(x - \alpha U(x)) - x\}
\]
\[
\leq (x - x^*)^T(\alpha U[P_2(x)] + P_2(x) - x)
\leq ||P_2(x - \alpha U(x)) - x||_2^2
\]
\[
+ \alpha\{x - P_2(x - \alpha U(x))\}^T
\cdot \{U(x) - U[P_2(x - \alpha U(x))]\}
\]
where the last inequality follows from the monotone property of $U$. Using (35) we get
\[
(x^* - x)^T \{P_2(x) - \omega U(x)\} 
\geq (1 - \alpha\lambda)||P_2(x - \alpha U(x)) - x||_2^2
\]
thus
\[
(x - x^*)^T F(x) \leq -(1 - \alpha\lambda)\Phi(x)
\]
So $\Phi(x)$ and $F(x)$ satisfy conditions 1) and 2) since $1 - \alpha\lambda > 0$. According to Theorem 1, we can obtain the proof of Corollary 7.
REFERENCES


Youshen Xia (M’96) received the B.S. and M.S. degrees in computational mathematics from Nanjing University, China, respectively, in 1982 and 1989. Presently, he is working on the Ph.D. degree in the Department of Mechanical and Automation Engineering at the Chinese University of Hong Kong. In 1995, he was an Associate Professor with the Department of Mathematics at Nanjing University of Posts and Telecommunication in China. His research interests include computational mathematics, neural networks, and control theory.

Jun Wang (S’89–M’90–SM’93) received the B.S. degree in electrical engineering and the M.S. degree in systems engineering from Dalian Institute of Technology (now Dalian University of Technology), Dalian, China. He received the Ph.D. degree in systems engineering from Case Western Reserve University, Cleveland, OH. He was an Associate Professor at the University of North Dakota, Grand Forks. He is currently an Associate Professor in the Department of Mechanical and Automation Engineering at the Chinese University of Hong Kong (CUHK). He authored and coauthored numerous papers in more than 20 journals, several edited books, and many conference proceedings. Dr. Wang is a Senior Member of the Institute of Industrial Engineers (II) and a member of the International Neural Network Society (INNS).