Journal of Global Optimization **19:** 363–381, 2001.

Stability Analysis of Gradient-Based Neural Networks for Optimization Problems ⋆

QIAOMING HAN[1], LI-ZHI LIAO[2], HOUDUO QI[3] and LIQUN QI[4]
[1]*School of Mathematics and Computer Science, Nanjing Normal University, Nanjing 210097, P.R. China;* [2]*Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Kowloon, Hong Kong (Corresponding author, e-mail: liliao@hkbu.edu.hk);* [3]*School of Mathematics, The University of New South Wales, Sydney, New South Wales 2052, Australia. (e-mail: hdqi@maths.unsw.edu.au);* [4]*Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hum, Kowloon, Hong Kong (e-mail: maqilq@polyu.edu.hk)*

(Received 23 May 2000; accepted in revised form 8 December 2000)

Abstract. The paper introduces a new approach to analyze the stability of neural network models without using any Lyapunov function. With the new approach, we investigate the stability properties of the general gradient-based neural network model for optimization problems. Our discussion includes both isolated equilibrium points and connected equilibrium sets which could be unbounded. For a general optimization problem, if the objective function is bounded below and its gradient is Lipschitz continuous, we prove that (a) any trajectory of the gradient-based neural network converges to an equilibrium point, and (b) the Lyapunov stability is equivalent to the asymptotical stability in the gradient-based neural networks. For a convex optimization problem, under the same assumptions, we show that any trajectory of gradient-based neural networks will converge to an asymptotically stable equilibrium point of the neural networks. For a general nonlinear objective function, we propose a refined gradient-based neural network, whose trajectory with any arbitrary initial point will converge to an equilibrium point, which satisfies the second order necessary optimality conditions for optimization problems. Promising simulation results of a refined gradient-based neural network on some problems are also reported.

Key words: Gradient-based neural network, Equilibrium point, Equilibrium set, Asymptotic stability, Exponential stability

1. Introduction

Optimization problems arise in almost every field of engineering, business and management sciences. In many engineering and scientific applications, the real-time solutions of optimization problems are demanded. However, traditional algorithms for digital computers may not be able to provide the solutions on-line. Therefore, the search for real-time on-line solutions in such cases becomes not only important but also essential. In 1980's, an attractive and very promising ap-

⋆ This research was supported in part by Australian Research Council, the Research Grant Council of Hong Kong, grants FRG/98-99/II-29 and FRG/99-00/II-23 from Hong Kong Baptist University, and the State Key Lab of Scientific and Engineering Computing in P. R. China.

proach was introduced to provide real-time solutions for optimization problems. The new approach is termed artificial neural network (**ANN**). Hopfield and Tank [12, 13, 28] initiated the recent study of neural networks in optimization. Generally speaking, ANN provides an alternative and attractive way for the solution of optimization problems. The significant and unique feature of ANN to optimization is the realization of simple and real-time hardware (or circuit) implementation. In other words, an electrical circuit can be constructed which generates the on-line solution of certain optimization problems.

The seminal work of Hopfield and Tank has inspired many researches to investigate various neural networks for solving linear and nonlinear programming problems. Numerous optimization neural networks have been developed, see [1–3, 5, 14–22, 24, 27, 30, 31, 33, 34]. From the optimization point of view, most of the existing ANN models for optimization problems could be divided into two classes. One class is the gradient-based neural network models, which are used for the unconstrained optimization problems. These problems normally come from a) some kind of transformations from the constrained minimization problems with penalty function methods [2, 15, 20, 22, 24]; and b) the complementarity problems with the NCP functions [17, 18]. The other class is the projective gradient based neural network models, which are derived from constrained minimization problems and complementarity problems with KKT-conditions [14, 19, 30, 31, 34]. [30, 31] present some neural networks for solving linear, quadratic and nonlinear convex programming problems, which are based on KKT system for optimization problems. Their models correspond to some variants of the projective gradient method for the complementarity problem and the variational inequality problem [7–11, 23]. It should be noted that for Lagrangian conditions with the nonnegative Lagrange multiplier, if the Lagrange multiplier is penalized or transformed into unrestricted case, the resulting neural network model belongs to the first class. On the other hand, if the nonnegative Lagrange multiplier is enforced by projection, it belongs to the second class.

In this paper, we pay our attention to the following unconstrained nonconvex optimization problem

$$\min E(x), \quad x \in R^n \tag{1}$$

with the general gradient-based motion equation, that is

$$\frac{dx(t)}{dt} = -Hg(x(t)), \quad x(t_0) = x_0 \in R^n, \tag{2}$$

where $E(x) : R^n \to R$ is the objective function or called the energy function, $g(x) : R^n \to R^n$ is the gradient of $E(x)$, $x_0 \in R^n$ is an arbitrary initial point and H is a constant symmetric positive definite matrix which can be viewed as a scaling matrix. For the case that H is a function of $x(t)$, various neural network models could be established. But in this paper, we only focus on the case that H is a constant positive definite matrix, and assume that $E(x)$ is smooth, i.e., $E(x) \in C^2(R^n)$.

Generally speaking, a neural network model consists of (a) an energy function, (b) a motion equation which is stable, and (c) the fact that the energy function should be monotonically nonincreasing along the solution of the motion equation. For simplicity, in the rest of the paper, we will call (2) as the general gradient-based neural network. The verification of (b) and (c) for the general gradient-based neural network (2) will be conducted later.

The stability analysis is a key issue in designing a neural network. However, the existing discussions on stability [1, 2, 15, 20, 21, 30, 34] are based on the Lyapunov's direct method which requires the existence of a Lyapunov function. Unfortunately, a Lyapunov function may not always exist, especially for constrained problems. This certainly limits the stability analysis for a general neural network model. In this paper, we introduce a new approach to address the stability issues of a general neural network model. The detailed discussion is given in Section 2. But first, let us review some of the classical results.

For the ordinary differential equation

$$\frac{dx(t)}{dt} = f(x), \tag{3}$$

we first state some classical results on the existence and uniqueness of the solution, and some stability definitions for the dynamic system (3) in [25, 29, 32].

THEOREM 1. *[32] Assume that f is a continuous function from R^n to R^n. Then for arbitrary $t_0 \geqslant 0$ and $x_0 \in R^n$ there exists a local solution $x(t)$ satisfying $x(t_0) = x_0, t \in [t_0, \tau)$ to (3) for some $\tau > t_0$. If furthermore f is locally Lipschitz continuous at x_0 then the solution is unique, and if f is Lipschitz continuous in R^n then τ can be extended to ∞.*

DEFINITION 1 (Equilibrium point). A point $x^* \in R^n$ is called an equilibrium point of (3) if $f(x^*) = 0$.

DEFINITION 2 (Stability in the sense of Lyapunov). Let $x(t)$ be the solution of (3). An isolated equilibrium point x^* is Lyapunov stable if for any $x_0 = x(t_0)$ and any scalar $\varepsilon > 0$ there exists a $\delta > 0$ such that if $\|x(t_0) - x^*\| < \delta$ then $\|x(t) - x^*\| < \varepsilon$ for $t \geqslant t_0$.

DEFINITION 3 (Asymptotic stability). An isolated equilibrium point x^* is said to be asymptotically stable if in addition to being a Lyapunov stable it has the property that $x(t) \to x^*$ as $t \to \infty$, if $\|x(t_0) - x^*\| < \delta$.

DEFINITION 4 (Exponential stability). An isolated equilibrium point x^* is exponentially stable for (3) if there exist $\omega < 0, \kappa > 0, \delta > 0$ such that any arbitrary solution $x(t)$ of (3), with the initial condition $x(t_0) = x_0$, $\|x(t_0) - x^*\| < \delta$, is defined on $[0, \infty)$ and satisfies

$$\|x(t) - x^*\| \leqslant \kappa e^{\omega t} \|x(t_0) - x^*\|, \quad t \geqslant t_0.$$

Note that above classical stabilities are only for isolated equilibrium points. Non-isolated equilibrium points could occur inevitably in the dynamic system for optimization problems. For example, if $E(x) = (x_1^2 + x_2^2 - 1)^2$, the points on the circle $x_1^2 + x_2^2 = 1$ are all the equilibrium points of (2). So it is necessary to extend the definitions of various stabilities from the isolated equilibrium points to non-isolated ones.

DEFINITION 5 (Connected equilibrium set). The subset Γ of the equilibrium set $S = \{x \in R^n : f(x) = 0\}$ of the dynamic system (3) is called a connected equilibrium set of S, if for any $x_1, x_2 \in \Gamma$ there exists a continuous curve in Γ connecting x_1 and x_2, and if there exists a positive number τ such that there is no other equilibrium point of (3) except Γ itself in the open neighborhood $B(\tau, \Gamma) = \{x \in R^n : d(x, \Gamma) < \tau\}$ of Γ, where $d(x, \Gamma) = \inf_{y \in \Gamma} \|x - y\|$ denotes the distance from x to Γ.

DEFINITION 6 (Stability for connected equilibrium set). Let $x(t)$ be the solution of (3). The connected equilibrium set Γ is stable if for any $x_0 = x(t_0)$ and any scalar $\varepsilon > 0$ there exists a $\delta > 0$ such that if $d(x(t_0), \Gamma) < \delta$ then $d(x(t), \Gamma) < \varepsilon$ for $t \geqslant t_0$.

DEFINITION 7 (Asymptotic stability for connected equilibrium set). The connected equilibrium set Γ is said to be asymptotically stable if in addition to being stable it has the property that $d(x(t), \Gamma) \to 0$ as $t \to \infty$, if $d(x(t_0), \Gamma) < \delta$.

DEFINITION 8 (Exponential stability for connected equilibrium set). The connected equilibrium set Γ is exponentially stable for (3) if there exist $\omega < 0, \kappa > 0, \delta > 0$ such that any arbitrary solution $x(t)$ of (3), with the initial condition $x(t_0) = x_0$, $d(x(t_0), \Gamma) < \delta$, is defined on $[0, \infty)$ and satisfies

$$d(x(t), \Gamma) \leqslant \kappa e^{\omega t} d(x(t_0), \Gamma), \quad t \geqslant t_0.$$

Note that there is a simple fact that the objective function $E(x)$ in (1) remains constant on the connected equilibrium set Γ of the system (2) since $g(x) = 0$ on Γ and Γ is connected.

The following Theorem 2 provides some interesting properties for autonomous system (3).

THEOREM 2 [25]. *(a) If $x(t), r_1 \leqslant t \leqslant r_2$ is a solution of (3), then for any real constant c the function $x_1(t) = x(t + c)$ is also a solution of (3). (b) Through any point passes at most one trajectory.*

Throughout this paper, we assume that the objective function $E(x)$ in (1) is bounded below and continuously differentiable. In addition, its gradient function $g(x)$ is Lipschitz continuous in R^n, i.e. there exists a constant L (> 0) such that

$$\|g(x_1) - g(x_2)\| \leqslant L\|x_1 - x_2\| \quad \text{for any } x_1, x_2 \in R^n. \tag{4}$$

Our study in this paper is mainly on the various stability properties of the gradient-based neural network (2) for the optimization problem (1). Our contributions consist of the following five parts.

- First, we introduce a new approach to study the stability issues for various motion equations.
- Second, under some mild assumptions, we prove that any trajectory of the gradient-based neural network (2) converges to an equilibrium point of (2) as $t \to \infty$.
- Third, we show that the Lyapunov stability is equivalent to the asymptotic stability in the gradient-based neural networks (2) for (1).
- As a direct corollary, for the convex objective function, we obtain that any trajectory of the gradient-based neural network (2) converges to an asymptotically stable equilibrium point under some mild conditions. These conditions are much weaker than those in Lyapunov's direct method or in the invariant set method [26, Chapter 3].
- Finally, for the general nonlinear objective function, we propose a refined gradient-based neural network, whose trajectory with any arbitrary initial point will converge to an equilibrium point, which satisfies the second order necessary optimality conditions for optimization problems.

The rest of the paper is organized as follows. In Section 2, detailed discussions on stability issues of the gradient-based neural network (2) for the optimization problem (1) are provided. A refined neural network model for the general optimization problem is provided in Section 3. Some stability issues are also addressed in this section. Section 4 is devoted to the simulation of the refined neural network on some problems. Promising simulation results are reported. Finally, some concluding remarks are drawn in Section 5.

2. Analysis for Gradient-Based Neural Networks

The analysis in this section is for $H = I$ in (2) for simplicity. For the case where H is symmetric positive definite matrix other than the identical matrix, the analysis is similar with the norm $\|x\|_{H^{-1}} = x^T H^{-1} x$ and the induced distance $d(x, y) = \|x - y\|_{H^{-1}}$ for any $x, y \in R^n$.

In the literature [1, 2, 15, 20, 21, 30, 34], the stability analysis for a motion equation is proved by Lyapunov's direct method which requires the existence of a Lyapunov function. But unfortunately, a Lyapunov function may not exist in general. To overcome this difficulty, we introduce a new approach to address stability issues of a motion equation without using any Lyapunov function. In addition, the approach does not require the prior information of the equilibrium point or set at all. The new approach only pays the attention to the energy function $E(x)$ and the motion equation. In the rest of this section, we will illustrate how this approach works for the general gradient-based neural network model (1)–(2).

To obtain our main results Theorem 4–Theorem 7, we need the following preliminaries.

THEOREM 3. *Let $g(x)$ in (2) be Lipschitz continuous in R^n and $g(x) \not\equiv 0$, then there is no periodic solution for the neural network system (2).*

Proof. From Theorem 1 there exists a unique solution $x(t)$ satisfying $x(t_0) = x_0$, $t \in [t_0, \infty)$. Suppose that there is a periodic solution $x(t)$ and the minimal period is $T > 0$, that is,

$$x(t) = x(t+T), \quad \forall\, t \in [t_0, \infty).$$

Since $g(x) \not\equiv 0$, there exists a $t_1 \in [0, \infty)$ such that $x(t_1) = x_1$, $g(x_1) \neq 0$, then $x(t_1 + T) = x_1$. Since $E(x(t))$ is nondecreasing on $[t_1, t_1 + T]$ from $\frac{dE(x(t))}{dt} = -g(x)^T g(x) \leqslant 0$, $t \in [t_1, t_1 + T]$, so $E(x(t))$ must be constant on $[t_1, t_1 + T]$, which is a contradiction since $\frac{dE(x(t))}{dt}|_{t=t_1} = -g(x_1)^T g(x_1) < 0$. □

From Theorem 2 and Theorem 3, we can conclude that the phase space of (2) can only consist of equilibrium points and nonintersecting trajectories. Moreover, every trajectory approaches to an equilibrium point, which is shown later (Theorem 4) without any assumption on the boundedness of level sets of the objective function. We extend R^n to include infinite point as an ordinary point, whose meaning is that for every $d \neq 0 \in R^n$, $x_0 + \alpha d$ approaches to it as $\alpha \to \infty$.

It is easy to see that there is a simple and important fact that every trajectory of the system (2) is orthogonal to contour curves of the objective function $E(x)$ in (1) if they intersect, because $E(x)$ is continuously differentiable and $E(x)$ remains constant on any contour.

LEMMA 1. *Let $x(t)$ be a trajectory of the system (2), Π is any contour of $E(x)$ in (1), then $x(t)$ is orthogonal to Π at their intersecting points.*

Proof. First, let us denote the phase plane of (2) as $x(t, s)$, where s represents the parameter along any contour of $E(x)$ in the phase plane of (2). Since $E(x) \in C^2(R^n)$, from Theorem 7.2 of [6, p. 25], we know that $x(t, s)$ has continuous partial derivative with respect to s. Then since $E(x(t, s))$ remains constant on its contour, i.e.

$$E(x(t,s)) = constant \quad \forall s,$$

we have

$$(\nabla_x E)^T \cdot \frac{dx}{ds} = 0 \quad \Longleftrightarrow \quad \left(\frac{dx}{dt}\right)^T \cdot \frac{dx}{ds} = 0.$$

This proves the lemma. □

In the proof of Theorem 4, we also need the following Barbalat lemma.

LEMMA 2 (Barbalat). [26] If a differentiable function $f(t)$ has a finite limit as $t \to \infty$, and $\frac{df(t)}{dt}$ is uniformly continuous, then $\frac{df(t)}{dt} \to 0$ as $t \to \infty$.

THEOREM 4. *If $E(x)$ is bounded below and its gradient $g(x)$ is Lipschitz continuous in R^n, then for any initial point x_0, the trajectory $x(t)$ of the system (2), satisfying $x(t_0) = x_0$, will converge to an equilibrium point of the neural network (2) as $t \to \infty$.*

Proof. Since $E(x)$ is bounded below and monotonically nonincreasing along the trajectory $x(t)$, it is easy to see that there exists a finite $E^* \in R$ such that $\lim_{t\to\infty} E(x(t)) = E^*$. Then we have that

$$E(x_0) - E^* = \int_{t_0}^{\infty} \|g(x(t))\|^2 dt, \tag{5}$$

so there exists at least a sequence $\{t_i\}$, satisfying $t_i < t_{i+1}$ and $\lim_{i\to\infty} t_i = \infty$, such that

$$\lim_{i\to\infty} \|g(x(t_i))\|^2 = 0. \tag{6}$$

In the following, we show that $\|g(x(t))\|$ is bounded in $[t_0, \infty)$ by contradiction. Assume that there is a sequence $\{s_i\}$, satisfying $s_i < s_{i+1}$ and $\lim_{i\to\infty} s_i = \infty$, such that

$$\|g(x(s_i))\| > i^2 \quad i = 1, 2, \cdots . \tag{7}$$

By extracting two subsequences if necessary from $\{t_i\}$ and $\{s_i\}$, respectively, we can assume that $s_i \in [t_i, t_{i+1}]$, $i = 1, 2, \cdots$. Without loss of generality, we assume that

$$\|g(x(s_i))\| = \max_{t\in[t_i, t_{i+1}]} \|g(x(t))\|. \tag{8}$$

From (6), for $\epsilon_0 = 0.1$, there exists a K_1 such that $\forall\, i > K_1$,

$$\|g(x(t_i))\| < \epsilon_0.$$

From (8) and the continuity of $g(x)$, for each $i > K_1$ there exists a $\mu_i \in (t_i, s_i)$ such that

$$\|g(x(\mu_i))\| = \epsilon_0 \quad \text{and} \quad \|g(x(t))\| \geqslant \epsilon_0 \;\forall t \in (\mu_i, s_i). \tag{9}$$

This together with (5) imply $s_i - \mu_i \to 0$ as $t \to \infty$. Then there exists a K_2 such that $\forall\, i > K_2$,

$$L(s_i - \mu_i) \leqslant 0.1, \tag{10}$$

where L is the Lipschitz constant defined in (4). From the continuity of $g(x)$, (8) and (10), we have that for $i > \max\{K_1, K_2\}$

$$\begin{aligned}\|g(x(s_i))\| - 0.1 &\leqslant \|g(x(s_i)) - g(x(\mu_i))\| \\ &\leqslant L\|x(s_i) - x(\mu_i)\| \\ &\leqslant L\|g(x(s_i))\|(s_i - \mu_i) \\ &\leqslant 0.1\|g(x(s_i))\|.\end{aligned} \tag{11}$$

This is a contradiction to (7). Therefore $\|g(x(t))\|$ is bounded in $[t_0, \infty\}$, that is, there exists an L_1 such that

$$\|g(x(t))\| \leqslant L_1 \ t \in [t_0, \infty). \tag{12}$$

It follows from (2), (4) and (12) that for any $t_1, t_2 \in [t_0, \infty)$

$$\begin{aligned}\left|\frac{dE(x(t_1))}{dt} - \frac{dE(x(t_2))}{dt}\right| &= |\|g(x(t_1))\|^2 - \|g(x(t_2))\|^2| \\ &\leqslant \|g(x(t_1)) + g(x(t_2))\| \cdot \|g(x(t_1)) - g(x(t_2))\| \\ &\leqslant 2L_1L\|x(t_1) - x(t_2)\| \\ &\leqslant 2L_1^2L|t_1 - t_2|,\end{aligned} \tag{13}$$

so $\frac{dE(x(t))}{dt}$ is uniformly continuous in $[t_0, \infty)$. From Lemma 2, we have

$$\lim_{t\to\infty} g(x(t)) = 0. \tag{14}$$

Next, we will prove that $\lim_{t\to\infty} x(t)$ exists for any initial condition $x(t_0) = x_0$, where $x(t)$ is the solution of the system (2). Furthermore, the limit point, say x^*, is an equilibrium point of the neural network (2).

First, we define

$$\Gamma = \{x^* : \exists \{t_i\}, \text{ such that } x(t_0) = x_0 \text{ and } \lim_{i\to\infty} x(t_i) = x^*\}$$

as the limit set of the trajectory $x(t)$ with $x(t_0) = x_0$, where $\{t_i\}$ is a sequence satisfying $t_i < t_{i+1}, i = 1, 2, \cdots$ and $t_i \to \infty$ as $i \to \infty$. Now we prove that Γ is a connected set by contradiction. Assume that Γ is not connected, then there exist two sets Γ_1, Γ_2, and two open sets Ω_1, Ω_2 such that $\Gamma_1, \Gamma_2 \subset \Gamma$, $\Gamma_1 \cup \Gamma_2 = \Gamma$, $\Gamma_1 \cap \Gamma_2 = \emptyset$, $\Omega_1 \cap \Omega_2 = \emptyset$, and $\Gamma_1 \subset \Omega_1, \Gamma_2 \subset \Omega_2$. We can also assume that at least one of the sets Γ_1, Γ_2 is bounded otherwise Γ_1 and Γ_2 are connected at infinite point. Suppose Γ_1 and Ω_1 are bounded. Therefore, there exists a bounded closed set D such that $\Omega_1 \subset D$. Considering two points $x^t \in \Gamma_1$ and $x^s \in \Gamma_2$, there are two sequences $\{t_i\}$ and $\{s_i\}$ with $t_i < s_i < t_{i+1}, i = 1, 2, \cdots$, $\lim_{i\to\infty} t_i = \infty$ and $\lim_{i\to\infty} s_i = \infty$ such that

$$\lim_{i\to\infty} x(t_i) = x^t, \quad \lim_{i\to\infty} x(s_i) = x^s.$$

As a result, there exist an $\epsilon > 0$, $B_\epsilon(x^t) \subset \Omega_1$, $B_\epsilon(x^s) \subset \Omega_2$, where $B_\epsilon(x)$ is a ball centered at x with radius ϵ, and a K such that $\forall\, i > K$,

$$x(t_i) \in B_\epsilon(x^t), \quad x(s_i) \in B_\epsilon(x^s).$$

From the continuity of $x(t)$, for each $i > K$, there exists a $\mu_i \in (t_i, s_i)$ such that $x(\mu_i) \notin \Omega_1$, $x(\mu_i) \notin \Omega_2$ and $x(\mu_i) \in D$. Then the closeness and boundedness of D imply that there exists a convergent subsequence $\{x(u_{i_k})\}$ such that $\lim_{k\to\infty} x(u_{i_k}) = x^\mu$. Since Ω_1 and Ω_2 are both open sets, then $x^\mu \notin \Omega_1$ and $x^\mu \notin \Omega_2$, so $x^\mu \notin \Gamma$, which is a contradiction to the definition of Γ. Therefore, Γ must be a connected set.

From the discussion at the beginning of the proof and the definition of Γ, it is easy to see that the value of $E(x(t))$ is the constant E^* on Γ, that is, Γ is a contour of $E(x(t))$. The trajectory $x(t)$ of the neural network (2) is orthogonal to Γ at the limit point (that is, every point on Γ) from Lemma 1. On the other hand, since Γ is the limit set of $x(t)$, the trajectory $x(t)$ must converge to Γ. This could be true only if Γ is a single point set. That is, $x(t)$ converges to a limit point, say x^*, as $t \to \infty$. And moreover (14) implies $g(x^*) = 0$. □

The result in Theorem 4 confirms that the energy function $E(x)$ in (1) and the motion equation, Equation (2) constitute a neural network model. The result in Theorem 4 also shows that any trajectory of the neural network (2) for the optimization problem (1) will converge to an equilibrium point of the neural network (2), regardless of the boundedness of level sets of the objective function and the isolation of the equilibrium point.

Because of $\frac{dE(x(t))}{dt} = -\|g(x(t))\|^2$, the energy function $E(x(t))$ in (1) is monotonically nonincreasing in t if x_0 is not an equilibrium point. This fact reveals that the energy function $E(x(t))$ does not preserve following the trajectory of the neural network system (2). Theorem 3 and Theorem 4 lead to the following surprising results.

THEOREM 5. *Assume that $E(x)$ in (1) is bounded below and its gradient $g(x)$ is Lipschitz continuous in R^n. Let Γ be a connected equilibrium set of the neural network (2), if Γ is stable, then it is asymptotically stable.*

Proof. From Definition 5, if Γ is a connected equilibrium set of the neural network (2), there exists a positive number τ such that there is no other equilibrium point of (3) except Γ itself in any open neighborhood $B(\tau, \Gamma) = \{x \in R^n : d(x, \Gamma) < \tau\}$ of Γ. Since Γ is stable, from Definition 6, if x_0 is in some small neighborhood of Γ, then the trajectory $x(t)$ satisfying $x(0) = x_0$ will stay in some neighborhood of Γ for any $t \geqslant 0$. From Theorem 4, $x(t)$ will approach to an equilibrium point of neural network (2), so $x(t)$ must approach to a point $x^* \in \Gamma$ as $t \to \infty$, that is, Γ is asymptotically stable from Definition 7. □

The result in Theorem 5 would ease many redundant discussions in proving both the stability and asymptotic stability in many neural networks for optimization problems. The asymptotic stability is particularly important for the neural network (2) of optimization problem (1), because only the point(s) in the asymptotically stable connected equilibrium set are the local optimal solution(s) of the problem (1). The following Theorem 6 shows the equivalence of the asymptotically stable connected equilibrium set of neural network (2) and the local optimal solution set of the problem (1).

THEOREM 6. *Assume that $E(x)$ in (1) is bounded below and its gradient $g(x)$ is Lipschitz continuous in R^n. Let Γ be a connected equilibrium set of the neural network (2). Then Γ is asymptotically stable if and only if each point of the set is a local optimal solution of the optimization problem (1).*

Proof. From Definition 7, the connected equilibrium set Γ is asymptotically stable if and only if any trajectory in all very small neighborhood of Γ will converge to Γ. The objective function $E(x(t))$ in (1) is nonincreasing along any trajectory of the neural network (2) as indicated in the proof of Theorem 3. In addition, $E(x)$ remains constant on Γ. So the conclusion is true. □

The following two examples illustrate the above results.

EXAMPLE 1.

$$\min E_1(x, y) = \frac{1}{2}(x + y - 1)^2. \tag{15}$$

The level sets of $E_1(x, y)$ are all unbounded unless it is empty, the neural network for (15) is

$$\begin{cases} \dfrac{dx}{dt} = -(x + y - 1), \\ \dfrac{dy}{dt} = -(x + y - 1). \end{cases} \tag{16}$$

The equilibrium set of (16) is the line $l_1 : x + y - 1 = 0$, which is connected and unbounded. The solutions to (16) are

$$x = \frac{1}{2}(1 + x_0 - y_0) + \frac{x_0 + y_0 - 1}{2}e^{-2t},$$
$$y = \frac{1}{2}(1 - x_0 + y_0) + \frac{x_0 + y_0 - 1}{2}e^{-2t}.$$

The trajectories are lines

$$l_2 : \quad y - x = y_0 - x_0.$$

It is obvious to see that every trajectory approaches to the equilibrium point $(\frac{1}{2}(1 + x_0 - y_0), \frac{1}{2}(1 - x_0 + y_0))$ as $t \to \infty$, which is on the line l_1, every trajectory l_2 is orthogonal to the line l_1, and $E_1(x, y) = 0$ on the line l_1.

EXAMPLE 2.

$$\min E_2(x) = x^2 e^{-x}. \tag{17}$$

The neural network for (17) is

$$\frac{dx(t)}{dt} = -xe^{-x}(2-x). \tag{18}$$

The equilibrium set of (18) is $\{0, 2, \infty\}$, and

$$\begin{aligned}
&\text{if } x_0 > 2, \quad x(t) \to \infty, \ t \to \infty,\\
&\text{if } x_0 < 2, \quad x(t) \to 0, \ t \to \infty,\\
&\text{if } x_0 = 2, \quad x(t) = 2, \ \forall\, t.
\end{aligned}$$

Theorem 4 and the above examples indicate that the infinite point can be viewed as an extended ordinary point. If the point is an isolated minimizer of the objective function as in Example 2, some trajectory will approach to it. If the point is not an isolated minimizer as in Example 1, there is no trajectory with a finite initial point can approach to it.

In fact, if the set R^n is relaxed by level sets $\{x : E(x) \leqslant E(x_0)\}$ for all x_0's, Theorem 1 and Theorem 4 still hold. For instance, in Example 2, $g(x)$ is not Lipschitz continuous in R^n, but Lipschitz continuous in any level set of $E(x)$, so the conclusions still hold as shown above.

At the end of this section, we provide some results for the convex optimization problem as the direct corollary of the above discussion. Some similar results have been discussed in previous works of the neural networks for convex optimization problems.

THEOREM 7. *(a) If $E(x)$ is convex, then the equilibrium set of the neural network (2) is connected and asymptotically stable.*

(b) If the convex objective function $E(x)$ is bounded below and continuously differentiable with Lipschitz continuous gradient, then every trajectory $x(t)$ will converge to an asymptotically stable equilibrium point of the neural network (2).

(c) If $E(x)$ is uniformly convex, then every trajectory of (2) will converge to an equilibrium point globally and exponentially.

Note that there is no any assumption on the boundedness of level sets of $E(x)$ or the isolation of the equilibrium point in Theorem 7. These assumptions are always required in previous works of neural networks for convex optimization problems.

3. A Refined Neural Network

For the general nonlinear objective function, some results for convex optimization problems as mentioned in Theorem 7 may not hold. However, motivated by the

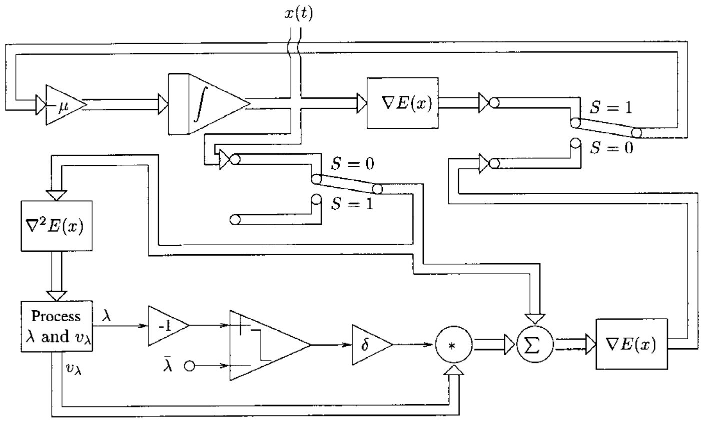

Figure 1. The block diagram of the refined neural network model.

work of [4] where eigenvalues and eigenvectors of a matrix can be computed by neural networks, we propose a refined gradient-based neural network model for the optimization problem (1) in this section.

The essence of the refined neural network model is the following. First, by using the gradient-based neural network (2), the trajectory of the neural network will converge to an equilibrium point of the neural network. Second, by using the neural network of [4], the minimum eigenvalue, say λ, of the Hessian matrix at the equilibrium point can be computed. If $\lambda \geqslant 0$, the model will output the current equilibrium point and stop the simulation. The obtained point satisfies the second order necessary optimality conditions for the optimization problem (1). Otherwise $\lambda < 0$, the trajectory is perturbed along the eigenvector corresponding to λ. With the new initial point, the gradient-based neural network is implemented once again. The process in the second part may be repeated a few times until an equilibrium point with positive semi-definite Hessian is reached. In the whole process, the reduction of the energy function $E(x)$ is guaranteed. Summarizing the above discussions, we provide a refined neural network model for the optimization problem (1) as follows.

A refined neural network model:

Step 0. Given an initial point x_0, a very small positive number $\bar{\lambda}$, and a small positive number δ.

Step 1. Implement the neural network (2) with the initial point x_0 to reach an equilibrium point, say x^*.

Step 2. Use the neural network in [4] to compute the minimum eigenvalue λ and its corresponding eigenvector v_λ of the Hessian matrix H^* of $E(x)$ at x^*. If $\lambda \geqslant -\bar{\lambda}$, stop. Otherwise, set $x_0 = x^* + \delta v_\lambda$ and go to Step 1.

A block diagram of the refined neural network is shown in Figure 1, where $H = \mu I$ and μ (> 0) is a scaling parameter. The MOS switch (see Appendix B, [5]) used

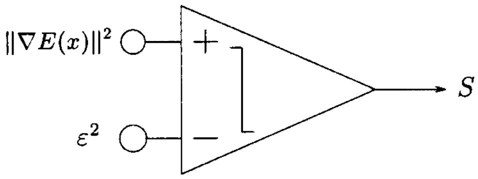

Figure 2. Computation of S in the MOS switch of Figure 1.

in Figure 1 is shown in Figure 2, where ε is used to control the stopping of Step 1, for details, see Section 4. The output of Figure 2, S, equals 1 if $\|\nabla_x E(x(t))\| \leqslant \varepsilon$, equals zero otherwise.

From Theorem 4, the refined neural network is convergent, which is summarized in the following theorem.

THEOREM 8. *If $E(x)$ is bounded below and its gradient $g(x)$ is Lipschitz continuous in R^n, then for any initial point x_0, the trajectory $x(t)$ of the refined neural network, satisfying $x(t_0) = x_0$, will converge to an equilibrium point of the neural network as $t \to \infty$. In addition, the equilibrium point satisfies the second order necessary optimality conditions for the optimization problem.*

4. Simulation

In this section, the refined neural network in Section 3 is simulated on two problems. Our simulations are conducted with Matlab version 5.2. The numerical ordinary differential equation solver used in all simulations is ode23s. The matrix H in (2) is set to μI, and the scaling parameter μ is fixed at $\mu = 1000$ in all simulations. The stopping criterion is

$$\|\nabla E(x(t))\| \leqslant \varepsilon = 10^{-6}. \tag{19}$$

We use t_f to denote the final time when (19) is satisfied. In Step 2 of the refined neural network, we choose $\bar{\lambda} = 10^{-8}$ and $\delta = 0.1$.

PROBLEM 1.

$$\min\ E(x) = (x_1^3 - 3x_1)^2 + x_2^2.$$

The corresponding dynamic system is

$$\begin{cases} \dfrac{dx_1}{dt} = -6x_1(x_1^2 - 1)(x_1^2 - 3), \\ \dfrac{dx_2}{dt} = -2x_2. \end{cases}$$

From the definition of $E(x)$, we can see that it has the following 5 stationary points, $(0, 0)^T$, $(\pm\sqrt{3}, 0)^T$, $(\pm 1, 0)$. Among these stationary points, three of them,

Table 1. Simulation results of Problem 1

Initial point	t_f	$\nabla E(x)$	$E(x)$	Limit point
$(1, 1)^T$	0.0107	5.87e-07	2.43e-15	(1.73205, 0)
$(-1, -1)^T$	0.0079	5.51e-07	9.48e-15	(−3.04e-08, 3.43e-08)
$(2, 2)^T$	0.0076	6.11e-07	9.35e-14	(1.73205, 0)
$(-2, -2)^T$	0.0076	6.11e-07	9.35e-14	(-1.73205, 0)

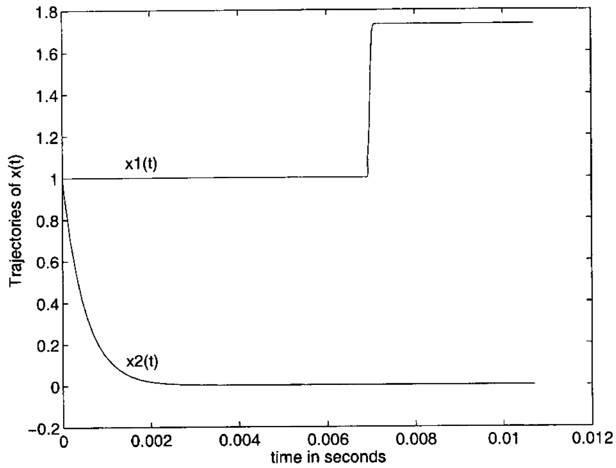

Figure 3. The evolution of $x(t)$ in Problem 1 with $x_0 = (1, 1)$.

$(0, 0)^T$, and $(\pm\sqrt{3}, 0)^T$ are local optimal solutions. For initial points of $(1, 1)^T$ and $(-1, -1)^T$, Step 1 of the refined neural network drives the trajectories of the system to points $(1, 0)^T$ and $(-1, 0)^T$, respectively, which are stationary points but not local optimal solutions of the optimization problem. While Step 2 continues the simulation until the system reaches local optimal solutions (see Figures 3 and 4). For the other two initial points of $(2, 2)^T$ and $(-2, -2)^T$, the trajectories of the system converge to local optimal solutions directly. These simulation results are summarized in Table 1. While the trajectories of $x(t)$ for the four different starting points are illustrated in Figures 3–6.

PROBLEM 2.

$$\min \; E(x) = (x_1^2 - x_2^2 + 1)^2.$$

The corresponding dynamic system is

$$\begin{cases} \dfrac{dx_1}{dt} = -4x_1(x_1^2 - x_2^2 + 1), \\ \dfrac{dx_2}{dt} = 4x_2(x_1^2 - x_2^2 + 1). \end{cases}$$

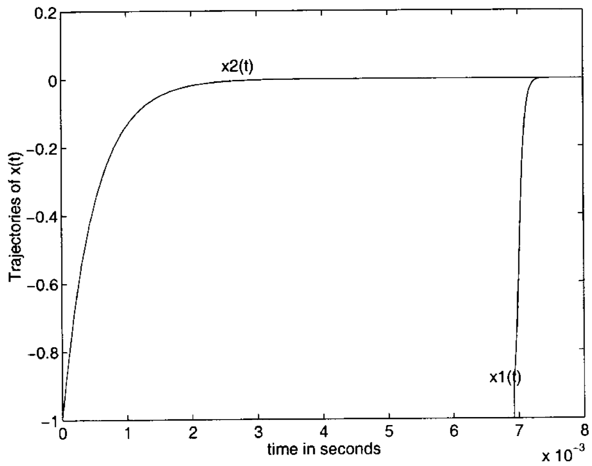

Figure 4. The evolution of $x(t)$ in Problem 1 with $x_0 = (-1, -1)$.

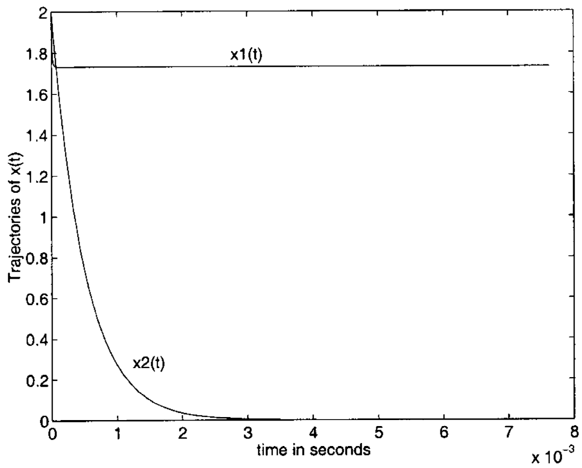

Figure 5. The evolution of $x(t)$ in Problem 1 with $x_0 = (2, 2)$.

Obviously in Problem 2, the stationary points are $(0, 0)^T$ and $(x_1, x_2)^T$ satisfying $x_1^2 - x_2^2 = -1$. Local optimal solutions for Problem 2 are $(x_1^*, x_2^*)^T$ satisfying $(x_1^*)^2 - (x_2^*)^2 = -1$. For initial points of $(0.1, 0)^T$ and $(-0.1, 0)^T$, Step 1 of the refined neural network drives the trajectories of the system to point $(0, 0)^T$ which is a stationary point but not a local optimal solution. While Step 2 continues the simulation until the trajectories reach local optimal solutions (see Figures 7 and 8). For the other initial point of $(1, -1)^T$, the trajectory of the system converges to a local optimal solution directly. These simulation results are summarized in Table 2. The trajectories of $x(t)$ for the three different starting points are illustrated in Figures 7–9. In Problem 1, the equilibrium points are all isolated. While in Problem 2 the equilibrium set is connected and unbounded. The simulation results of both problems confirm the results of Theorem 8.

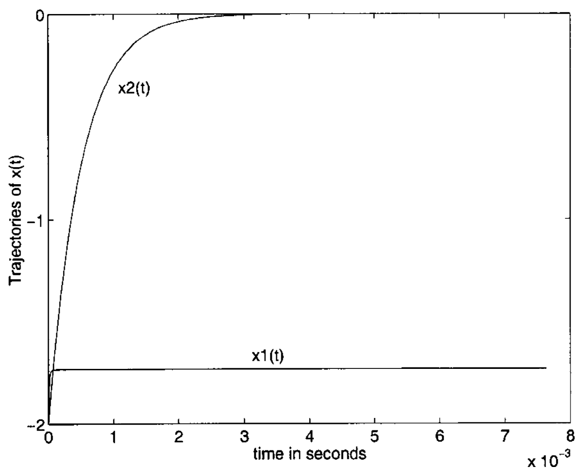

Figure 6. The evolution of $x(t)$ in Problem 1 with $x_0 = (-2, -2)$.

Table 2. Simulation results of Problem 2

Initial point	t_f	$\nabla E(x)$	$E(x)$	Limit point
$(0.1, 0)^T$	0.0306	6.73e-08	2.83e-16	(0, 1)
$(-0.1, 0)^T$	0.0306	6.73e-08	2.83e-16	(0, 1)
$(1, -1)^T$	0.0183	6.45e-08	1.16e-16	$(0.786212, -1.27206)$

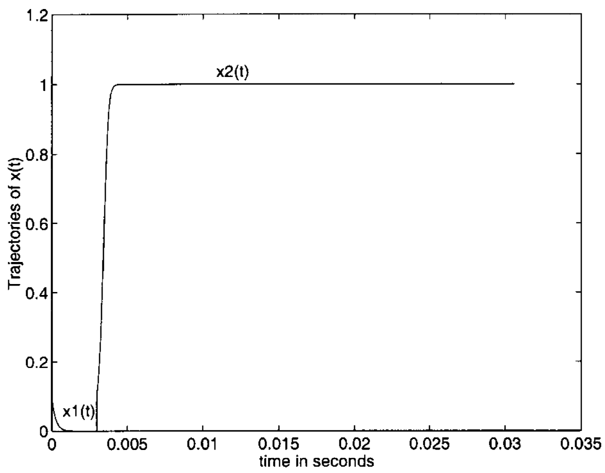

Figure 7. The evolution of $x(t)$ in Problem 2 with $x_0 = (0.1, 0)$.

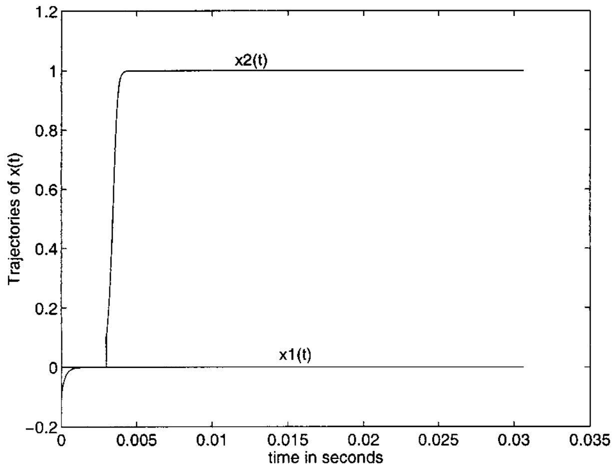

Figure 8. The evolution of $x(t)$ in Problem 2 with $x_0 = (-0.1, 0)$.

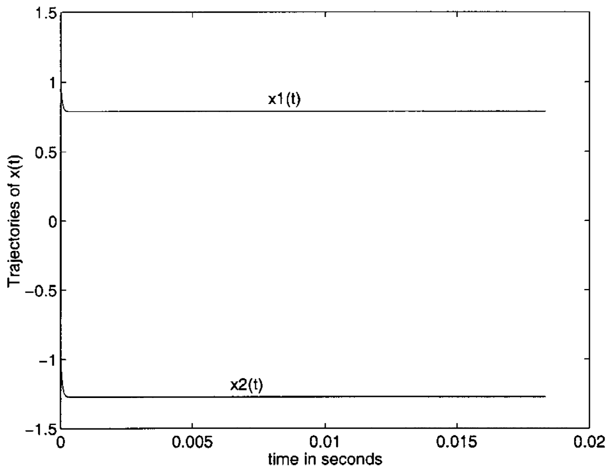

Figure 9. The evolution of $x(t)$ in Problem 2 with $x_0 = (1, -1)$.

5. Concluding Remarks

Our research in this paper focuses on the gradient-based neural network (2) for the optimization problem (1), which is the most commonly used neural network model for optimization problems. First, we extend some definitions of isolated equilibrium points and the associated stabilities to the case of any connected equilibrium set which could be unbounded. With these more general definitions, we have obtained the following results:

- First, a new approach is introduced to analyze stability properties of various neural network models. The new approach does not require the existence of any Lyapunov function. The use of our new approach in stability analysis is illustrated in Section 2.

- For gradient-based neural networks (2), if $E(x)$ is bounded below and its gradient $g(x)$ is Lipschitz continuous, then any trajectory $x(t)$ of the system (2) will converge to an equilibrium point of the neural network (2).
- The Lyapunov stability is equivalent to the asymptotically stability in gradient-based neural networks for optimization problems.
- For convex optimization problems, the equilibrium set is connected and asymptotically stable, and that any trajectory of gradient-based neural networks will converge to an asymptotically stable equilibrium point if the objective function is bounded below and has Lipschitz continuous gradient. Our results require weaker assumptions than the Lyapunov's direct method where the equilibrium point must be isolated or the invariant set method where level sets of the objective function must be bounded. In fact, our assumptions are necessary, because the bounded below assumption on $E(x)$ ensures that the optimization problem (1) is meanful and Lipschitz continuity of $g(x)$ guarantees the existence and uniqueness of the solution of the neural network (2).
- For the general nonlinear objective function, a refined gradient-based neural network model is established, whose trajectory with any arbitrary initial point will converge to an equilibrium point, which satisfies the second order necessary optimality conditions for optimization problems.

We believe that similar conclusions would hold for projective gradient based neural networks for optimization problems.

References

1. Bouzerdoum, A. and Pattison, T.R. (1993), Neural network for quadratic optimization with bound constraints, *IEEE Trans. Neural Networks* 4: 293–304.
2. Chen Y.H. and Fang, S.C. (1998), Solving convex programming problem with equality constraints by neural networks, *Computers Math. Appli.* 36: 41–68.
3. Chua L.O. and Lin, G.N. (1984), Nonlinear programming without computation, *IEEE Trans. Circuits Syst.* 31: 182–188.
4. Cichocki, A. and Unbehauen, R. (1992), Neural networks for computing eigenvalues and eigenvectors, *Biological Cybernetics* 68: 155–164.
5. Cichocki, A. and Unbehauen, R. (1993), *Neural Networks for Optimization and Signal Processing.* Wiley, Chichester, U.K.
6. Coddington E.A. and Levinson, N. (1955), *Theory of Ordinary Differential Equations*, McGraw-Hill, New York.
7. Goldstein, A.A. (1964), Convex programming in Hilbert space, *Bulletin of American Mathematical Society* 70: 709–710.
8. He, B.S. (1992), A projection and contraction method for a class of linear complementary problem and its application in convex quadratic programming, *Applied Mathematics and Optimization* 25: 247–262.

9. He, B.S. (1994), A new method for a class of linear variational inequalities, *Mathematical Programming* 66: 137–144.
10. He, B.S. (1994), Solving a class of linear projection equations, *Numerische Mathematik* 68: 71–80.
11. He, B.S. (1997), A class of projection and contraction methods for monotone variational inequalities, *Applied Mathematics and Optimization* 35: 69–76.
12. Hopfield, J.J. (1984), Neurons with graded response have collective computational ability, in *Proc. Natl. Acad. Sci., USA 81*, pp. 3088–3092.
13. Hopfield, J.J. and Tank, D.W. (1985), Neural computation of decisions in optimization problems, *Biol. Cybern.* 52: 141–152.
14. Hou, Z.-G., Wu, C.-P. and Bao, P. (1998), A neural network for hierarchical optimization of nonlinear large-scale systems, *International Journal of Systems Science* 29(2): 159–166.
15. Kennedy M.P. and Chua, L.O. (1988), Neural networks for nonlinear programming, *IEEE Trans. Circuits Syst.* 35: 554–562.
16. Liao L.-Z. and Dai, Y. (1999), A time delay neural network model for unconstrained nonconvex optimization, (manuscript).
17. Liao L.-Z. and Qi, H. (1999), A neural network for the linear complementarity problem, *Math. Comput. Modelling* 29(3): 9–18.
18. Liao, L.-Z., Qi, H. and Qi, L. Solving nonlinear complementarity problems with neural networks: A reformulation method approach, (to appear in *JCAM*).
19. Liang, X.B. and Wang, J. (1999), A recurrent neural network for optimizing a continuously differentiable function with bound constraints, (manuscript).
20. Lillo, W.E., Loh, M.H., Hui, S. and Zak, S.H. (1993), On solving constrained optimization problems with neural networks: A penalty method approach, *IEEE Trans. Neural Networks* 4: 931–940.
21. Maa, C.Y. and Shanblatt, M.A. (1992), Linear and quadratic programming neural network analysis, *IEEE Trans. Neural Networks* 3: 580–594.
22. Maa, C.Y. and Shanblatt, M.A. (1992), A two-phase optimization neural network, *IEEE Trans. Neural Networks* 3: 1003–1009.
23. Polyak, B.T. (1966), Constrained minimization problems, *USSR Computational Mathematics and Mathematical Physics* 6: 1–50.
24. Rodríguez-Vázquez, A., Domínguez-Castro, R., Rueda, A., Huertas, J.L. and Sánchez-Sinencio, E. (1990), Nonlinear switch-capacitor 'neural' networks for optimization problems, *IEEE Trans. Circuits Syst.* 37: 384–398.
25. Sánchez, D.A. (1968), *Ordinary Differential Equations and Stability Theory: An Introduction*. W.H. Freeman and Company, San Francisco.
26. Slotine, J.J.E. and Li, W. (1991), *Applied Nonlinear Control.* Prentice Hall, Englewood Cliffs.
27. Sudharsanan, S. and Sundareshan, M. (1991), Exponential stability and a systematic synthesis of a neural network for quadratic minimization, *Neural Networks* 4: 599–613.
28. Tank, D.W. and Hopfield, J.J. (1986), Simple neural optimization networks: An A/D convert, signal decision circuit, and a linear programming circuit, *IEEE Trans. Circuits Syst.* 33: 533–541.
29. Williems, J.L. (1970), *Stability Theory of Dynamical Systems*, Nelson.
30. Wu, X., Xia, Y., Li, J. and Chen, W.K. (1996), A high performance neural network for solving linear and quadratic programming problems, *IEEE Trans. Neural Networks* 7: 643–651.
31. Xia, Y. and Wang, J. (1998), A general methodology for designing globally convergent optimization neural networks, *IEEE Trans. Neural Networks* 9: 1331–1343.
32. Zabczyk, J. (1992), *Mathematical Control Theory: An Introduction*, Birkhauser, Boston.
33. Zak, S.H., Upatising, V. and Hui, S. (1995), Solving linear programming problems with neural networks: a comparative study, *IEEE Trans. Neural Networks* 6: 94–104.
34. Zhang, S. and Constantinides, A.G. (1992), Lagrange programming neural network, *IEEE Trans. Circuits Syst.* 39: 441–452.