Neural Networks 55 (2014) 20–29

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

A collective neurodynamic optimization approach to bound-constrained nonconvex optimization☆

Zheng Yan [a], Jun Wang [a,b,*], Guocheng Li [c]

[a] Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong
[b] School of Control Science and Engineering, Dalian University of Technology, Dalian, Liaoning, China
[c] Department of Mathematics, Beijing Information Science and Technology University, Beijing, China

ARTICLE INFO

Article history:
Received 15 February 2014
Received in revised form 10 March 2014
Accepted 13 March 2014
Available online 28 March 2014

Keywords:
Collective neurodynamic optimization
Recurrent neural network
Nonconvex optimization

ABSTRACT

This paper presents a novel collective neurodynamic optimization method for solving nonconvex optimization problems with bound constraints. First, it is proved that a one-layer projection neural network has a property that its equilibria are in one-to-one correspondence with the Karush–Kuhn–Tucker points of the constrained optimization problem. Next, a collective neurodynamic optimization approach is developed by utilizing a group of recurrent neural networks in framework of particle swarm optimization by emulating the paradigm of brainstorming. Each recurrent neural network carries out precise constrained local search according to its own neurodynamic equations. By iteratively improving the solution quality of each recurrent neural network using the information of locally best known solution and globally best known solution, the group can obtain the global optimal solution to a nonconvex optimization problem. The advantages of the proposed collective neurodynamic optimization approach over evolutionary approaches lie in its constraint handling ability and real-time computational efficiency. The effectiveness and characteristics of the proposed approach are illustrated by using many multimodal benchmark functions.

1. Introduction

Optimization problems are omnipresent in scientific and engineering applications including system modeling and control, signal processing, computer vision, pattern recognition, machine learning, and so on (e.g., see Boyd and Vandenberghe (2004), Hyvärinen and Oja (2000), Vapnik (2000)). As many real-world optimization problems become increasingly complex and fall into nonconvex optimization, computing global optimal solutions in real time using traditional numerical optimization techniques is computationally demanding. Developing better and efficient optimization methods is always desirable. A promising approach to real-time optimization is neurodynamic optimization where recurrent neural networks (RNNs) serve as parallel computational models for optimization problems solving (Xia & Wang, 1999). The essence of neurodynamic optimization lies in its inherent nature of parallel and distributed information processing and the availability of hardware implementation.

Since the pioneering work of Hopfield and Tank (1985) on using a neural network for solving the Traveling-Salesmen Problem, neurodynamic optimization has received a great deal of attention over the past three decades. Many researchers investigated alternative neurodynamic optimization models for solving various linear and nonlinear optimization problems. For example, Kennedy and Chua (1988) presented a recurrent neural network for nonlinear optimization by utilizing finite penalty parameter method to compute approximate optimal solutions. Zhang and Constantinides (1992) proposed a two-layer Lagrangian neural network to deal with the optimization problems with equality constraints. Wang (1994) proposed a deterministic annealing neural network for solving convex programming problems. Wang (1996) presented a recurrent neural network for solving the shortest path problem. Wang (1997) presented a primal–dual neural network for the zero–one integer linear programming. In recent years, many recurrent neural

☆ The work described in the paper was in part supported by the Research Grants Council, University Grants Committee, Hong Kong under Grant CUHK416812E; and by the National Natural Science Foundation of China under Grant 61273307.

* Corresponding author at: Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong. Tel.: +852 39438472.

E-mail addresses: zyan@mae.cuhk.edu.hk (Z. Yan), jwang@mae.cuhk.edu.hk, junwang111@gmail.com (J. Wang), xyliguocheng@sohu.com (G. Li).

http://dx.doi.org/10.1016/j.neunet.2014.03.006
0893-6080/

network models with simple model complexity and good convergence properties have been developed for linear programming (Liu & Wang, 2008a, 2011a), quadratic programming (Gao & Liao, 2010; Hu & Wang, 2008; Liu & Wang, 2006, 2008b; Xia, Feng, and Wang, 2004), variational inequalities (Hu & Wang, 2006, 2007), general convex nonlinear programming (Xia, Feng, & Wang, 2008; Xia, Leung, & Wang, 2002; Xia & Wang, 2004a, 2004b, 2005), pseudoconvex optimization (Guo, Liu, & Wang, 2011; Liu, Guo, & Wang, 2012), and nonsmooth optimization (Bian & Xue, 2009; Cheng et al., 2011; Forti, Nistri, & Quincampoix, 2004; Liu & Wang, 2011b, 2013). Most of the designed models are based on some classic optimality equations and conditions (Liao, Qi, & Qi, 2004). For example, Xia et al. (2002) applied a projection method for RNN design. Liu and Wang (2006) used duality properties. Liu and Wang (2008b) was based on the Karush–Kuhn–Tucker (KKT) conditions. Liu and Wang (2011a) applied saddle point theorems and obtained the finite-time convergence property. Cheng et al. (2011) applied the Lagrangian method for solving nonsmooth convex optimization problems. Hosseini, Wang, and Hosseini (2013), Liu and Wang (2011b), and Li, Yan, and Wang (2014) used nonsmooth penalty function methods for solving some generalized convex nonsmooth optimization problems.

While neurodynamic optimization approaches with individual RNNs have achieved great successes, they would reach their solvability limits at constrained optimization problems with unimodal objective functions exclusively and are impotent for global optimization with multimodal objective functions. When dealing with general nonconvex optimization problems, the dynamic behaviors of a recurrent neural network could change drastically and become unpredictable. In parallel to neurodynamic optimization research, population-based evolutionary computation approaches emerged as a branch of popular meta-heuristic methods for global and multi-objective optimization in recent years. Evolutionary optimization algorithms are stochastic, heuristic, discrete-time, and multiple-state in their nature. In particular, particle swarm optimization (PSO) is a global optimization method introduced by Kennedy and Eberhart that mimics swarm behaviors such as birds flocking and fish schooling (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995). In recent years, a number of variant PSO algorithms have been proposed for the purpose of accelerating convergence speed and avoiding premature convergence. For example, Liang, Qin, Suganthan, and Baskar (2006) proposed a comprehensive PSO for discouraging premature convergence. Zhan, Zhang and Chung (2009) proposed an adaptive PSO for better search efficiency. Zhan, Zhang, Yi, and Shi (2011) proposed an orthogonal learning PSO for better solution quality and stronger robustness. Owing to its ease of implementation and high efficiency, PSO has been widely adopted and successfully applied for many optimization problems (Chen et al., 2010; Duan, Luo, Shi, & Ma, 2013; Eberhart & Shi, 2007).

Despite of their capabilities of global search, PSO algorithms are deficient in precise local search and constraint handling. In contrast, based on optimization and dynamic systems theories, neurodynamic optimization approaches are competent for constrained local search, but incapable of global search in the presence of nonconvexity. It would be a good idea to combine the two types of optimization methods for constrained global optimization. In this paper, a collective neurodynamic optimization approach in framework of PSO is proposed for optimization problems with bound constraints. Inspired by brainstorming, a number of RNNs are exploited in a cooperative way to tackle constrained nonconvex optimization problems. Each neural network carries out local search according to its own neurodynamics and converges to a candidate solution. The movements of the neural networks are guided by individual best known solution as well as the best known solution of the entire group. By iteratively improving the initial conditions and the converged solutions, the neural network group is expected to discover the global optimal solution. The proposed approach can be viewed as an emulation for the brainstorming process of human beings, which offers a new paradigm for real-time optimization.

The rest of this paper is organized as follows. In Section 2, a neural network model is reviewed. In Section 3, the optimality and convergence of the neural network are investigated. In Section 4, a collective neurodynamic optimization approach is developed. In Section 5, simulation results on benchmark problems are provided. Finally, Section 6 concludes this paper.

2. Problem formulation and model description

Consider an optimization problem with bound constraints

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad x \in \Omega \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)^T \in \Re^n$ is the decision variable, $f : \Re^n \to \Re$ is an objective function, Ω is a nonempty and closed convex set in $\Re^n$. In this paper, Ω is assumed to be a box set.

Remark 1. Consider a general constrained optimization problem given by

$$\begin{aligned} &\text{minimize} \quad \tilde{f}(x) \\ &\text{subject to} \quad g_i(x) \le 0, i = 1, \ldots, m, \\ &x \in \Omega, \end{aligned} \tag{2}$$

where $g_i : \Re^n \to \Re, i = 1, \ldots, m$, are inequality constraints. One common approach to deal with the constrained optimization problem (2) is to introduce a penalty term into the objective function to penalize constraint violations (Fiacco & McCormick, 1990). The transformed objective function can be described as

$$f(x) = \tilde{f}(x) + \gamma\phi(g(x)), \tag{3}$$

where $\phi(g(x)) \ge 0$ is a real-valued function which imposes a penalty on constraint violation controlled by a penalty parameter γ. The penalty function method enables the solution to the optimization problem (2) to be obtained by solving the optimization problem (1) with a suitably defined exact penalty function. An exact penalty function has a property such that there exists a finite penalty parameter for which a solution to the penalized problem is a solution to the corresponding constrained problem. As a result, there is an incentive to develop a reliable and efficient approach to the optimization problem (1).

Xia et al. (2002) presented a projection based neural network for solving convex optimization problems subject to box constraints. The neural network model does not have any design parameter and hence it is more convenient for implementation. The neural network model is briefly reviewed here.

The one-layer projection neural network for the optimization problem (1) is described by the following dynamical equation:

$$\dot{x}(t) = -x(t) + P_\Omega(x(t) - \nabla f(x(t))), \tag{4}$$

where $x \in \Re^n$ is the state vector of the neural network, which corresponds to the decision vector in (1), ∇f is the gradient of f, and P_Ω is a projection operator defined as

$$P_\Omega(u) = \arg\min_{v\in\Omega} ||u - v||. \tag{5}$$

Computing the projection of a point onto a convex set is generally nontrivial. However, if Ω is a box set or a sphere set, the projection is well defined. When Ω is a box set, i.e., $\Omega = \{u \in \Re^n : l_i \le u_i \le h_i, i = 1, \ldots, n\}$, P_Ω is defined as

$$P_\Omega(u_i) = \begin{cases} l_i, & u_i < l_i; \\ u_i, & l_i \le u_i \le h_i; \\ h_i, & u_i > h_i. \end{cases} \tag{6}$$

When Ω is a sphere set, i.e., $\Omega = \{u \in \Re^n : ||u - s|| \leq r, s \in \Re^n, r > 0\}$, P_Ω is defined as

$$P_\Omega(u_i) = \begin{cases} u, & ||u - s|| \leq r; \\ s + \dfrac{r(u-s)}{||u-s||}, & ||u - s|| > r. \end{cases} \tag{7}$$

The model (4) has a one-layer structure with the number of neurons equal to the number of decision variables in (1). It has the lowest model complexity among existing neurodynamic optimization models.

3. Theoretical analysis

Definition 1 (*KKT Point*)**.** Let $f : \Re^n \to \Re$ and $g_i : \Re^n \to \Re$, $i = 1, \ldots, m$ be differentiable functions. Consider the problem to minimize $f(x)$ subject to $g(x) \leq 0$. If a point $(\bar{x}, \bar{\mu}) \in \Re^n \times \Re^m$ satisfies the conditions:

$$\nabla f(\bar{x}) + \bar{\mu}^T \nabla g(\bar{x}) = 0,$$
$$\bar{\mu} \geq 0,$$
$$g(\bar{x}) \leq 0,$$
$$\bar{\mu}^T g(\bar{x}) = 0,$$

then $\bar{x}$ is said to be a KKT point.

According to the convergence analysis in Xia et al. (2002), under the condition that the objective function $f(x)$ in (1) is convex or pseudoconvex on the feasible region Ω, the one-layer projection neural network can globally converge to the exact optimal solution to (1). However, when the objective function $f(x)$ is nonconvex on Ω, global convergence property cannot be obtained. As the KKT point provides first order necessary conditions for a solution in nonlinear programming to be optimal, one way to obtain the local optima of a nonconvex optimization problem is to solve the KKT equations (Bazaraa, Sherali, & Shetty, 2013). In this section, we investigate on using the neural network model (4) to search for KKT points of a nonconvex optimization problem. To obtain meaning solutions to a nonconvex optimization, two main issues to be addressed. The first one is the optimality of the neural network, i.e., correspondence between its equilibria and the KKT points needs to be ensured. The second one is the convergence of the neural network model. These two issues will be discussed in detail in this section.

Theorem 1. *Assume that the objective function $f(x)$ in (1) is continuous differentiable, and the feasible region Ω is a box set. The equilibria of the one-layer projection neural network are in one-to-one correspondence to the Karush–Kuhn–Tucker points of the nonconvex optimization problem (1).*

Proof. Based on the KKT conditions, $\bar{x}$ is a KKT point if and only if

$$\frac{\partial f}{\partial \bar{x}_i} \begin{cases} \geq 0, & \bar{x}_i = l_i; \\ = 0, & \bar{x}_i \in (l_i, h_i); \\ \leq 0, & \bar{x}_i = h_i. \end{cases} \tag{8}$$

Conditions (8) can be equivalently written as the following projection equation (Bertsekas, 1989)

$$P_\Omega(\bar{x} - \nabla f(\bar{x})) = \bar{x}. \tag{9}$$

In view of (4), $\bar{x}$ is an equilibrium state if and only if

$$\bar{x} = P_\Omega(\bar{x}) - \nabla f(P_\Omega(\bar{x})). \tag{10}$$

By comparing (9) and (10), it is straightforward to conclude that every equilibrium state of (4) corresponds to a KKT point of (1), and vice versa. □

Denote $\mathcal{M}$ as the equilibria set of the neural network (4)

$$\mathcal{M} = \{\bar{x} \in \Re^n : \bar{x} = P_\Omega(\bar{x} - \nabla f(\bar{x}))\}. \tag{11}$$

According to Theorem 1, $\mathcal{M}$ is also the KKT-point set.

Definition 2. The state vector of the neural network in (4) is said to be convergent to the KKT-point set $\mathcal{M}$ if for any initial value $x_0 = x(t_0) \in \Omega$,

$$\lim_{t \to \infty} \text{dist}(x(t), \mathcal{M}) = 0, \tag{12}$$

where $\text{dist}(x(t), \mathcal{M})$ is the distance between $x(t)$ and $\mathcal{M}$ defined as

$$\text{dist}(x(t), \mathcal{M}) = \min_{u \in \mathcal{M}} ||x - u||.$$

To show the convergence property of the projection neural network, the following lemma is needed.

Lemma 1 (*Bertsekas, 1989*)**.** *Let $\Omega \subseteq \Re^n$ be a closed convex set. $P_\Omega(\cdot)$ has the following properties: (a) $\forall z \in \Re^n$, $y \in \Omega$, $\langle z - P_\Omega(z), y - P_\Omega(z)\rangle \leq 0$, where $\langle \cdot, \cdot \rangle$ denotes the inner product operator. (b) $\|P_\Omega(y) - P_\Omega(z)\| \leq \|y - z\|$, $y \in \Re^n$, $z \in \Re^n$.*

Theorem 2. *Assume that the objective function $f(x)$ in (1) is continuous differentiable, and the feasible region Ω is a box set. For any initial value $x(t_0) = x_0 \in \Omega$, the state vector of the one-layer projection neural network is convergent to the KKT-point set $\mathcal{M}$.*

Proof. Let $x(t)$ be a state trajectory of (4). Set $H(x) = \frac{1}{2}\text{dist}^2(x, \Omega)$, then

$$\begin{aligned} \dot{H}(x(t)) &= \langle x(t) - P_\Omega(x(t)), \dot{x}(t)\rangle \\ &= \langle x(t) - P_\Omega(x(t)), -x(t) + P_\Omega(x(t) - \nabla f(x(t)))\rangle \\ &= \langle x(t) - P_\Omega(x(t)), -(x(t)) - P_\Omega(x(t)) \\ &\quad + P_\Omega(x(t) - \nabla f(x(t))) - P_\Omega(x(t))\rangle \\ &= \langle x(t) - P_\Omega(x(t)), -(x(t)) - P_\Omega(x(t))\rangle \\ &\quad + \langle x(t) - P_\Omega(x(t)), P_\Omega(x(t) - \nabla f(x(t))) - P_\Omega(x(t))\rangle \\ &= -\langle x(t) - P_\Omega(x(t)), x(t) - P_\Omega(x(t))\rangle \\ &\quad + \langle x(t) - P_\Omega(x(t)), P_\Omega(x(t) - \nabla f(x(t))) - P_\Omega(x(t))\rangle. \end{aligned} \tag{13}$$

As the first term $-\langle x(t) - P_\Omega(x(t)), x(t) - P_\Omega(x(t))\rangle = -\|x(t) - P_\Omega(x(t))\|_2^2$ and the second term above $\langle x(t) - P_\Omega(x(t)), P_\Omega(x(t) - \nabla f(x(t))) - P_\Omega(x(t))\rangle \leq 0$ according to Lemma 1, it follows that

$$\begin{aligned} \dot{H}(x(t)) &\leq -\|x(t) - P_\Omega(x(t))\|_2^2 \\ &= -2H(x(t)). \end{aligned}$$

It follows that $\text{dist}(x(t), \Omega) \leq \text{dist}(x_0, \Omega)e^{-t}$. When $x_0 \in \Omega$, $\text{dist}(x(t), \Omega) = 0$. Therefore, $x(t) \in \Omega$, $\forall t$. As Ω is a bounded convex set, $x(t)$ is bounded. Also, $x(t)$ is well defined on $t \in [0, +\infty)$.

Next, we proceed to show that $\lim_{t \to +\infty} \dot{x}(t) = 0$.

$$\forall t \in [0, +\infty), \quad \frac{d}{dt} f(x(t)) = \langle \nabla f(x(t)), \dot{x}(t)\rangle. \tag{14}$$

Since Ω is convex and $x(t) \in \Omega$, $\langle x(t) - P_\Omega(x(t) - \nabla f(x(t))), x(t) - \nabla f(x(t)) - P_\Omega(x(t) - \nabla f(x(t)))\rangle \leq 0$. It follows that $\langle -\dot{x}(t), -\dot{x}(t) - \nabla f(x(t))\rangle \leq 0$. As a result,

$$\frac{d}{dt} f(x(t)) \leq -\|\dot{x}(t)\|^2. \tag{15}$$

Integrating (15) over $[0, T]$, it is obtained $f(x_0) - f(x(T)) \geq \int_0^T \|\dot{x}(t)\|^2 dt$. Since f is continuous and $x(t)$ is bounded, $f(x(t))$ is bounded. In view of (15), $\lim_{T \to +\infty} f(x(T))$ exists. Moreover, $\int_0^{+\infty} \|\dot{x}(t)\|^2 dt \leq f(x_0) - \lim_{T \to +\infty} f(x(T))$. As $\int_0^{+\infty} \|\dot{x}(t)\|^2 dt$

is convergent, $\dot{x}(t)$ is bounded, $x(t)$ is uniformly continuous, it is concluded that $\lim_{t\to+\infty}\dot{x}(t)=0$.

Now we shall show that any cluster point of $x(t)$ is an equilibrium of (4). Assume $\hat{x}$ is the limit of $x(t)$, then there exists a sequence $\{t_n\}$ in $[0,+\infty)$ such that $\lim_{n\to+\infty}t_n=+\infty$ and $\lim_{n\to+\infty}x(t_n)=\hat{x}$. According to (4),

$$\dot{x}(t_n)=-x(t_n)+P_{\Omega}(x(t_n)-\nabla f(x(t_n))). \tag{16}$$

Taking limits on both sides of (16), it is obtained $\hat{x}=P_{\Omega}(\hat{x}-\nabla f(\hat{x}))$. Therefore, $\hat{x}$ is an equilibrium of (4). In addition, it follows that $\lim_{t\to\infty}\mathrm{dist}(x(t),\mathcal{M})=0$. □

4. Collective neurodynamic optimization

Inspired by the paradigm of brainstorming where each member in a group makes spontaneous contribution of ideas, a group of recurrent neural networks are exploited collectively for searching the global optimal solutions to bounded constrained nonconvex optimization problems. The proposed collective neurodynamic optimization can be viewed as a combination of particle swarm optimization (PSO) and neurodynamic optimization.

In a PSO algorithm, each particle in a swarm represents a potential solution. Each particle adapts its search patterns by learning from experiences of itself and its neighbors. Each particle has a fitness value and a velocity to adjust its search direction. The swarm is expected to move to the global optimal solution through an iterative learning and movement process.

Denote $u_i=(u_{i1},\ldots,u_{in})^T\in\Re^n$ as the position of the ith particle and $v_i=(v_{i1},\ldots,v_{in})^T\in\Re^n$ as the velocity of ith particle, where n is the dimension of each particle. Define $pbest_i=(pbest_{i1},\ldots,pbest_{in})^T\in\Re^n$ as the best previous position yielding the best the fitness value for the ith particle, and $gbest=(gbest_i,\ldots,gbest_n)^T\in\Re^n$ as the best position of the swarm. The movement of the ith particle in the swarm is updated as follows

$$v_i\leftarrow v_i+c_1r_1(pbest_i-u_i)+c_2r_2(gbest-u_i), \tag{17}$$
$$u_i\leftarrow u_i+v_i, \tag{18}$$

where c_1 and c_2 are two weighting parameters, r_1 and r_2 are two random numbers lie in $[0,1]$. The movement of the swarm will stop if the number of iterations reaches a predefined maximum number or *gbest* stops improving.

Despite of its global search capability, PSO is deficient for precise local search and constraints handling. To overcome these shortcomings and improve the computational efficiency, we propose to use the one-layer projection neural network to serve as an individual agent. As a result, each agent in the group can perform precise local search due to the correspondence between its equilibria and KKT points. The advantages of neurodynamic optimization and PSO can be combined together. Let N be the number of RNNs in the group. At a iteration step k, $k=0,1,\ldots,k_{\max}$ where $k_{\max}$ is the maximum iteration step, denote $x^i\in\Re^n$ as the state variable of the *ith* neural network and $\bar{x}^i$ as the equilibrium point of the *ith* neurodynamic equation. The principles of the collective neurodynamic optimization are summarized as follows.

- *Initialization:* For each recurrent neural network $i=1,\ldots,N$, do
  1. Initialize x^i with a uniformly distributed random vector in the search space.
  2. Initialize the individual best solution $x_p^i\leftarrow x^i$.
  3. Initialize the global best solution $x_g^*\leftarrow x_p^j: f(x_p^j)\le f(x_p^i)$, $\forall i=1,\ldots,N$.
- *Main loop:* Repeat:
  1. Each RNN carries out constrained local search guided by its dynamical equation
  $$\epsilon_i\frac{dx^i}{dt}=-x^i+P_{\Omega}(x^i)-\nabla f(P_{\Omega}(x^i)), \tag{19}$$
  to obtain $\bar{x}^i$.
  2. If $f(\bar{x}^i)<f(x_p^i)$, then update the individual best solution as $x_p^i\leftarrow\bar{y}^i$.
  3. If $f(x_p^i)<f(x_g^*)$, then update the global best solution as $x_g^*\leftarrow x_p^i$.
  4. Update the starting point of each recurrent neural network according to
  $$x^i\leftarrow x^i+c_0(\bar{x}^i-x^i)+c_1r_1(x_p^i-x^i)+c_2r_2(x_g^*-x^i). \tag{20}$$
- Termination: Terminate the iteration process if one or more of the following conditions are met:
  1. The predefined maximum number of iteration $k_{\max}$ is reached.
  2. x_g^* stops improving for five consecutive iterations, i.e., $||x_g^*(k)-x_g^*(k-1)||_2\le\varepsilon$ where ε is a sufficient small positive number.
  3. A solution with adequate cost function value is found, i.e., $|f(x_g^*)-f_d|\le\varepsilon$, where f_d is a desired cost function value. This criteria is used for problems with a priori knowledge on the optimal solution.

Remark 2. While PSO and other evolutionary algorithms need to search for the global optima among infinite number of solution candidates in the whole feasible region, the collective neurodynamic optimization approach searches among finite number of candidates (i.e., local minima, local maxima, and saddle points) only, where the local maxima would happen only if an initial point is a local maximum at the first iteration. As a result, the efficiency can be significantly increased. The collective neurodynamic optimization approach is able to obtain the global optimal solutions almost surely provided that the sufficient number of neural networks are applied and sufficient number of iterations are executed.

Fig. 1 shows the complete flowchart of the proposed neurodynamic optimization approach. Similar to a PSO algorithm, (20) reflects both the experiential knowledge of each individual agent and socially exchanged knowledge of the whole population. c_0 is an inertial weight. c_1 and c_2 are positive coefficients used to scale the contributions of individual think and social cooperation respectively. r_1 and r_2 are random variables used to introduce stochastic elements to the algorithm. Moreover, since a neural network is a brain-like computational model, the cooperative principles of neural networks can be viewed as an emulation of brainstorming process. Typically, a brainstorming process consists of the following steps:

Step 1. Get together a brainstorming group of people with diverse background;
Step 2. Each person generate ideas according to one's own knowledge and experiences;
Step 3. Evaluate those ideas according to specific criterion;
Step 4. Group members collect the evaluation feedback and exchange ideas to generate more ideas;
Step 5. Pick up better ideas generated in Step 4 to do evaluation;
Step 6. Repeat Steps 3–5 until the group reached a consensus on the solution quality;

Specifically, Steps 1–3 are emulated by the initialization, and Steps 4–6 are emulated by the main loop.

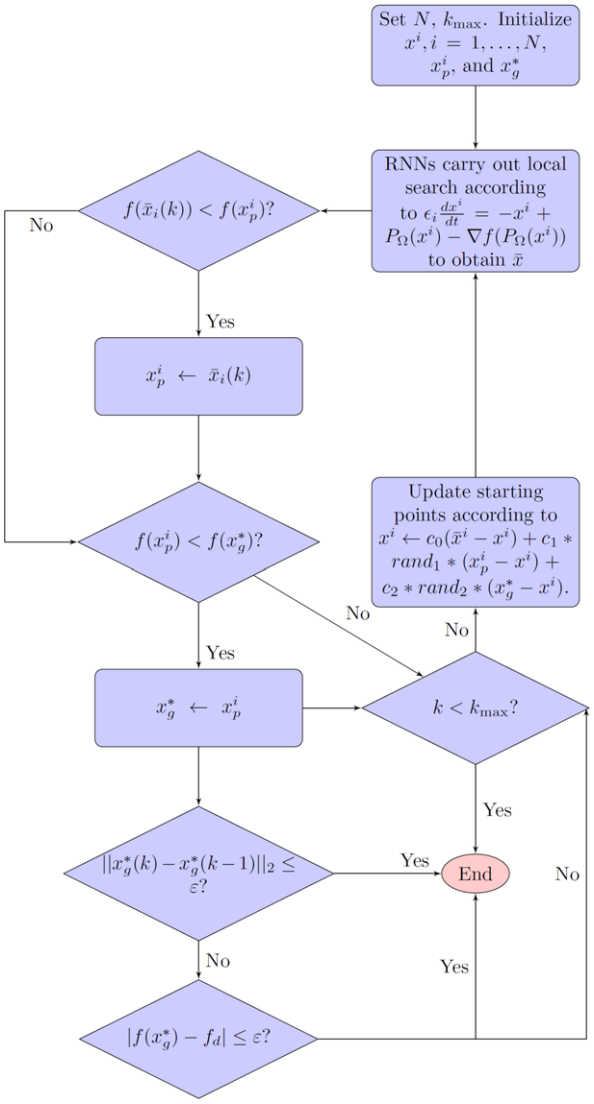

Fig. 1. Flowchart of the collective neurodynamic optimization approach.

5. Simulation results

To test the performance, seven multimodal benchmark problems (Das, Maity, Qu, & Nagaratnam, 2011) are solved using the proposed collective neurodynamic optimization approach.

Example 1. Consider the Six-Hump Camel Back function

$$f(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (4x_2^2 - 4)x_2^2,$$

$$\Omega = \{x : -1.9 \leq x_1 \leq 1.9, -1.1 \leq x_2 \leq 1.1\}. \quad (21)$$

As shown in Fig. 2, there are two global minima and four additional local minima in Six-Hump Camel Back function:

- $f(-0.0898, 0.7127) = -1.0316$,
- $f(0.0898, -0.7127) = -1.0316$,
- $f(-1.7036, 0.7961) = -0.2155$,
- $f(1.7036, -0.7961) = -0.2155$.
- $f(-1.6071, -0.5687) = 2.1043$,
- $f(1.6071, 0.5687) = 2.1043$.

As stated in Theorems 1 and 2, the one-layer projection neural network can converge to a KKT point from any initial state. However, the global optima cannot be guaranteed by a single neural network. Let $\epsilon = 10^{-6}$ in the simulation. Fig. 3 depicts the transient behaviors of a one-layer projection neural network with 10 random initial conditions. To obtain the global optimal solutions, the proposed collective neurodynamic optimization approach is applied. Let the number of neural networks be 10 in the simulation. Fig. 4 shows the two-dimensional phase plot for (x_1, x_2) for each neural network. In the first iteration, each neural network converges to a local minimum. In the second iteration, all neural networks converge to one of the global minima. The proposed collective neurodynamic optimization approach can obtain the global optimal solutions in microsecond time scale.

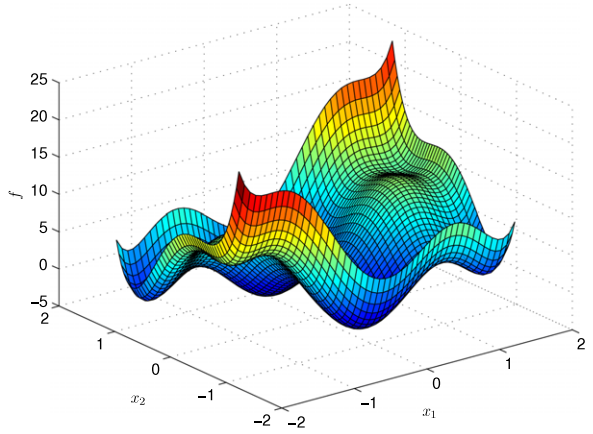

Fig. 2. Six-Hump Camel Back function.

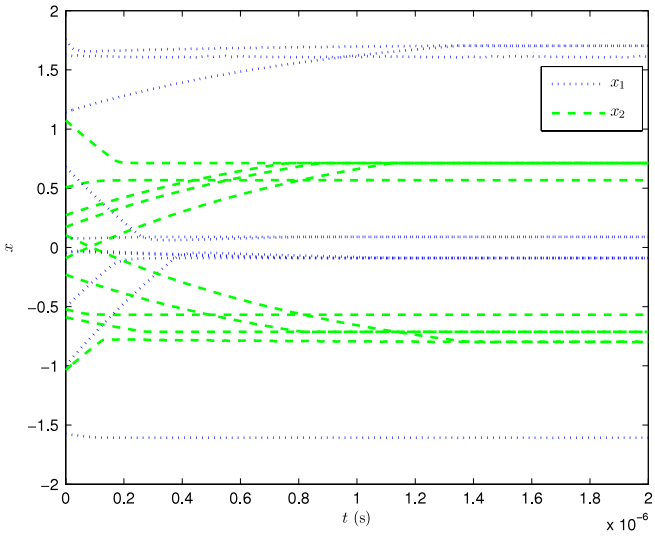

Fig. 3. Transient behaviors for Six-Hump Camel Back function.

Example 2. Consider Himmelblau's function

$$f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2,$$

$$\Omega = \{x : -6 \leq x_i \leq 6, i = 1, 2\}. \quad (22)$$

Himmelblau's function is shown in Fig. 5 and it has four identical local minima

- $f(3.0, 2.0) = 0.0$,
- $f(-2.8051, 3.1313) = 0.0$,
- $f(-3.7793, -3.2832) = 0.0$,
- $f(3.5844, -1.8481) = 0.0$.

One interesting property of Himmelblau's function is that every KKT point is a global minimum. Therefore, the one-layer projection neural network can always obtain a global optimal solution. Fig. 6

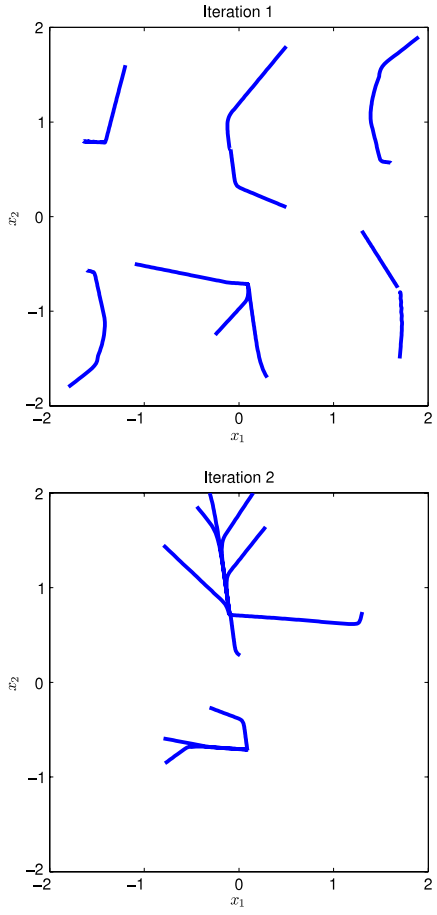

Fig. 4. Two-dimensional phase plot for Six-Hump Camel Back function.

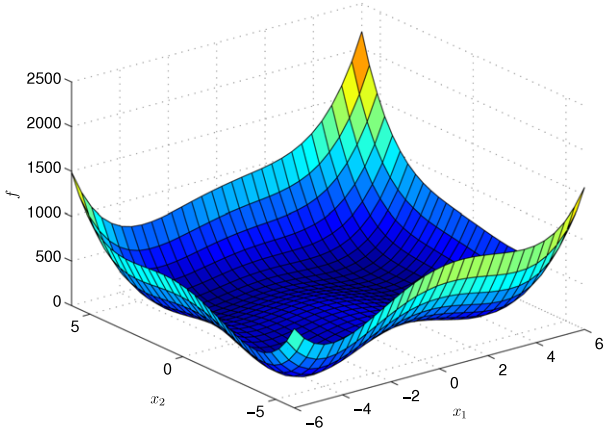

Fig. 5. Himmelblau's function.

depicts the transient behaviors of the neural network from 10 random initial conditions. Fig. 7 depicts the two-dimensional phase plot for (x_1, x_2) with 50 random initial conditions. As it is shown that the convergence to its equilibrium is within a few microseconds, the one-layer projection neural network is potentially competent for real-time optimization.

Example 3. Consider Rosenbrock's function

$$f(x) = \sum_{i=1}^{n-1}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2),$$

$$\Omega = \{x : -2.048 \le x_i \le 2.048, i = 1, \ldots, n\}. \quad (23)$$

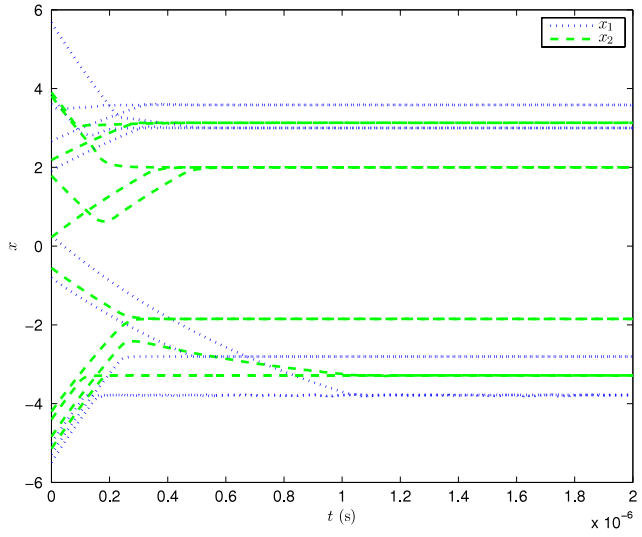

Fig. 6. Transient behaviors for Himmelblau's function.

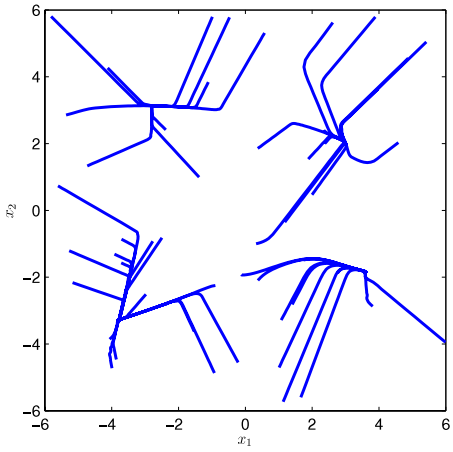

Fig. 7. Two dimensional phase plot for Himmelblau's function.

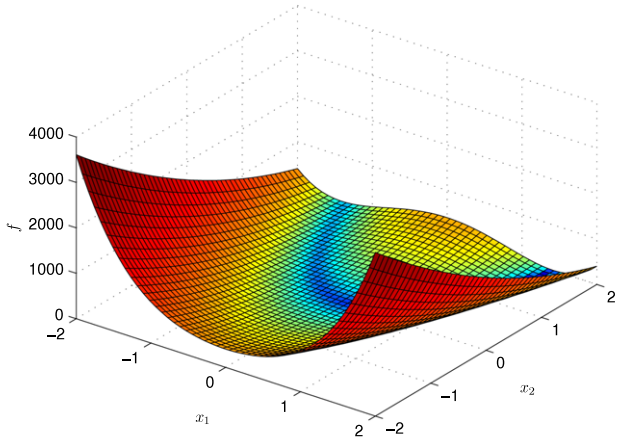

Fig. 8. Rosenbrock's function.

A 2-D Rosenbrock's function is shown in Fig. 8. Rosenbrock's function has exactly one minimum for $n \le 3$ and two minima for $4 \le n \le 7$. In the simulation, we set $n = 5$. In this case, the two minima are given as follows (Shang & Qiu, 2006)

- $f(1, 1, \ldots, 1) = 0$,
- $f(-0.9621, 0.9357, 0.8807, 0.7779, 0.6051) = 3.9308$.

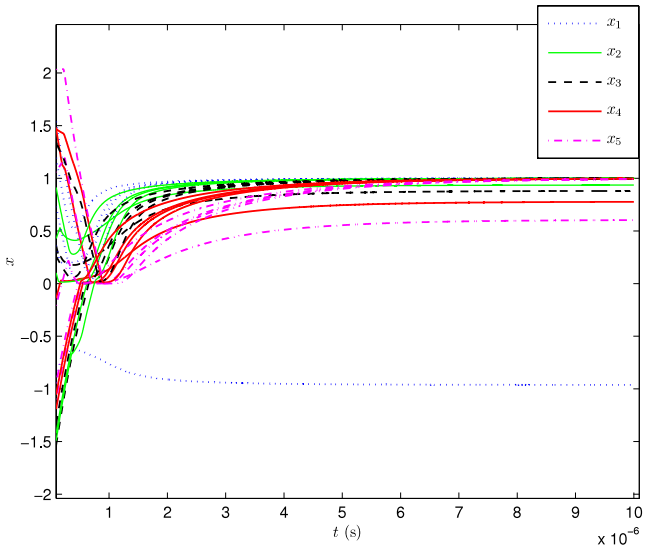

Fig. 9. Transient behaviors for Rosenbrock's function.

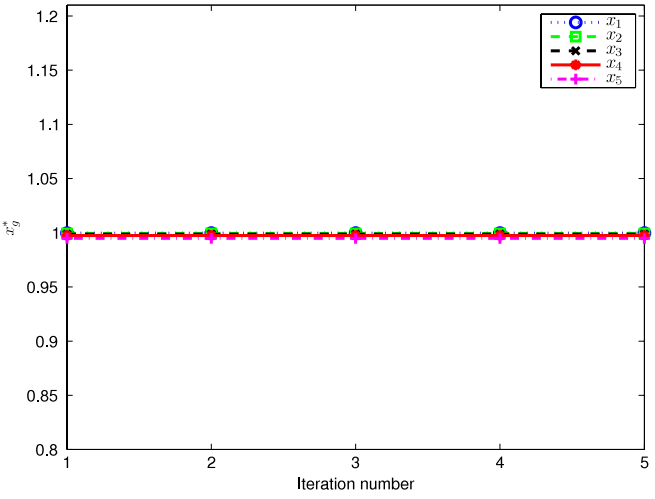

Fig. 10. Evolution of x_g^* for Rosenbrock's function.

Fig. 9 depicts the transient behaviors of the neural network. It can be seen that the neural network converges to a KKT point within 10 μs. The proposed collective neurodynamic optimization approach is expected to escape the local minima regardless of the initial condition. Let the number of neural networks be 5 in the simulation. Fig. 10 shows the evolution of x_g^*. The proposed approach is competent for obtaining the global optimal solution in real time.

Example 4. Consider Ackley's function

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e,$$

$$\Omega = \{x : -32.768 \le x_i \le 32.768,\ i = 1, \ldots, n\}. \quad (24)$$

As shown in Fig. 11, Ackley's function has one narrow global optimum basin and many minor local optima. The unique global minimum of Ackley's function is $f(0, \ldots, 0) = 0$. Let the number of neural networks be 15 in the simulation. Fig. 12 depicts the transient behaviors of a single neural network with five random

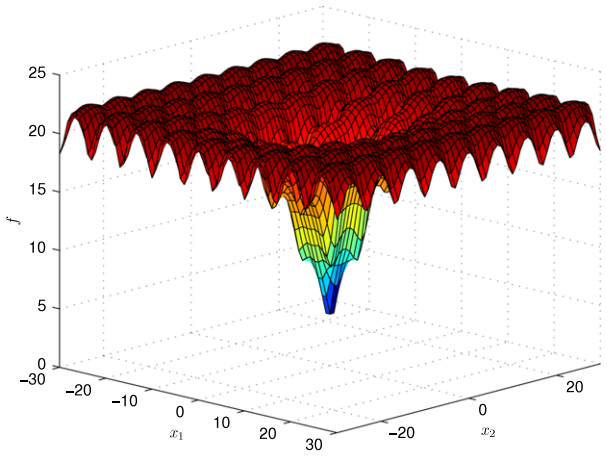

Fig. 11. Ackley's function.

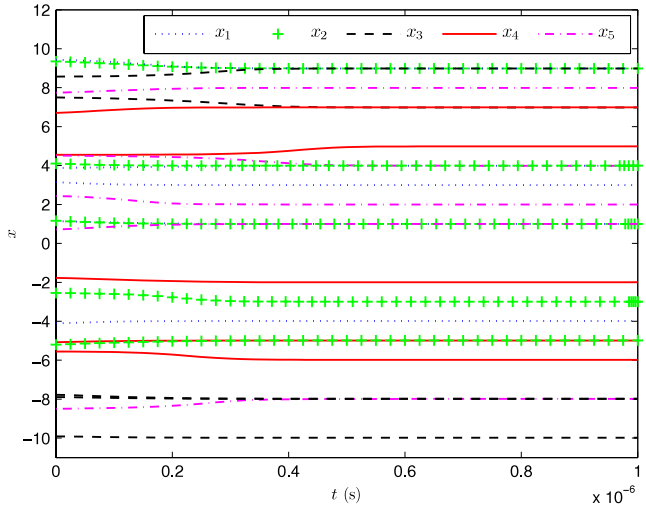

Fig. 12. Transient behaviors for Ackley's function.

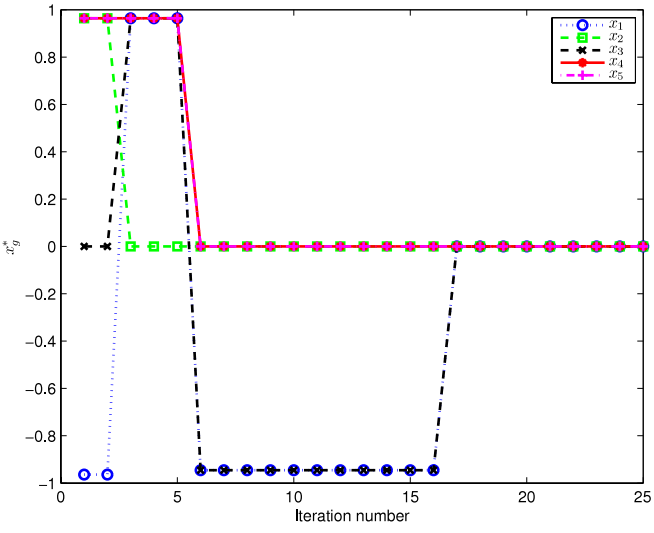

Fig. 13. Evolution of x_g^* for Ackley's function.

initial conditions. Fig. 13 depicts the evolution of x_g^*. It is shown that the proposed collective neurodynamic optimization approach obtains the global optimal solution after 17 iterations.

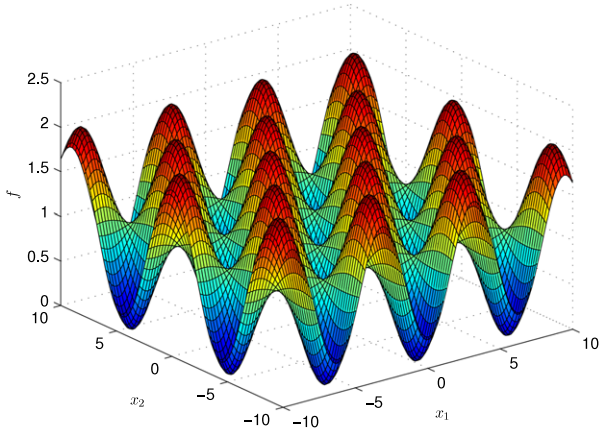

Fig. 14. Griewank's function.

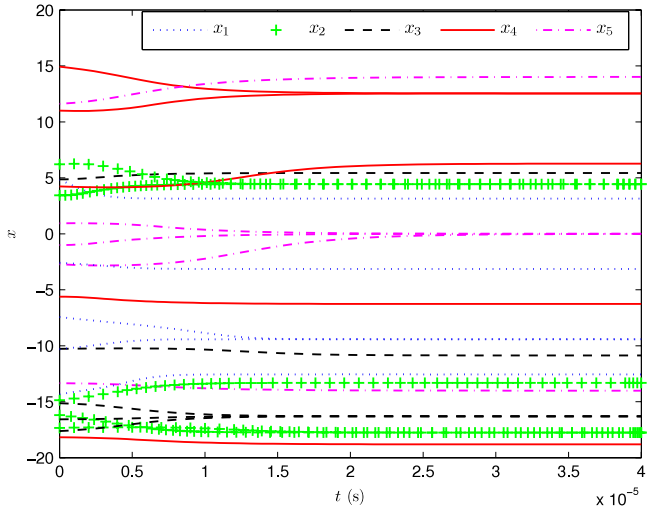

Fig. 15. Transient behaviors for Griewank's function.

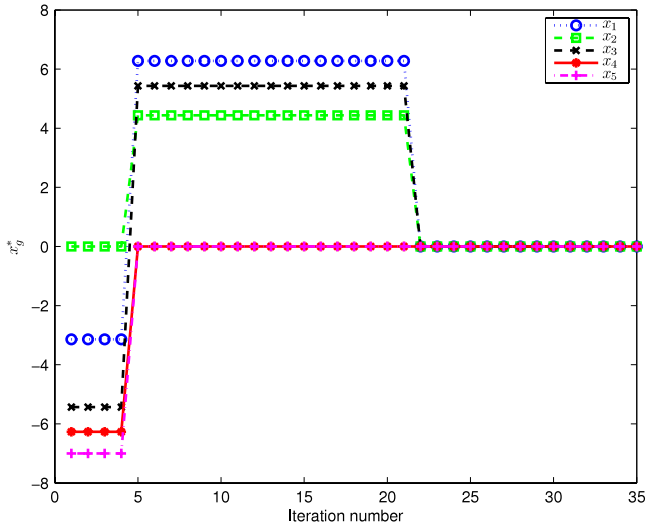

Fig. 16. Evolution of x_g^* for Griewank's function.

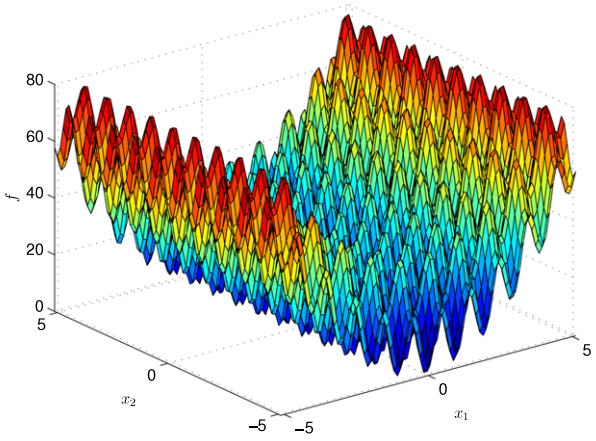

Fig. 17. Rastrigin's function.

Example 5. Consider Griewank's function

$$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$

$$\Omega = \{x : -600 \le x_i \le 600,\ i = 1, \ldots, n\}. \quad (25)$$

Griewank's function has a $\prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$ term causing linkages among variables, thereby making it difficult to reach the global minimum, (e.g., see Fig. 14 for $n = 2$). An interesting characteristic of Griewank's function is it is more difficult for lower dimensions than higher dimensions Whitley, Rana, Dzubera, and Mathias (1996). Therefore, we set $n = 5$ in the simulation. The unique global minimum of Griewank's function is $f(0, \ldots, 0) = 0$. Let the number of neural networks be 20 in the simulation. Fig. 15 depicts the transient behaviors of a single neural network model with 5 random initial conditions. The state variables can converge to equilibria within a microsecond. Fig. 16 depicts the evolution of x_g^*. The global optimal solution is obtained after 21 iterations.

Example 6. Consider Rastrigin's function

$$f(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10),$$

$$\Omega = \{x : -5.12 \le x_i \le 5.12,\ i = 1, \ldots, n\}. \quad (26)$$

Rastrigin's function is a complex multimodal function with a large number of local minima (e.g., see Fig. 17 for $n = 2$). The unique global minimum of Rastrigin's function is $f(0, \ldots, 0) = 0$. Let the number of neural networks be 15 in the simulation. Fig. 18 depicts the transient behaviors of a single neural network with 5 random initial vectors. The neural network converges to its equilibria within a microsecond. Fig. 19 shows the evolution of x_g^* by applying the proposed collective neurodynamic optimization approach. The global optima can be obtained after 8 iterations.

Example 7. Consider Schwefel's function

$$f(x) = 418.9829 * n - \sum_{i=1}^{n} x_i \sin(|x|^{\frac{1}{2}}),$$

$$\Omega = \{x : -500 \le x_i \le 500,\ i = 1, \ldots, n\}. \quad (27)$$

Schwefel's function is deceptive in that the global minimum is far away from the next best local minima. Therefore, search algorithms are potentially prone to converge to the wrong direction. Schwefel's function has a large number of local minima (e.g., see Fig. 20 for $n = 2$), and has a unique global minimum given by $f(420.9687, \ldots, 420.9687) = 0$.

Let the number of neural networks be 15 in the simulation. Fig. 21 depicts the transient behaviors of a single neural network with 10 random initial vectors. The neural network converges to

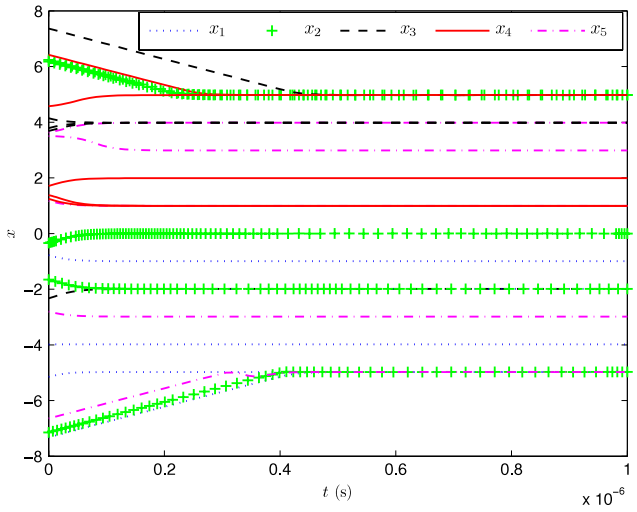

Fig. 18. Transient behaviors of a single neural network for Rastrigin's function.

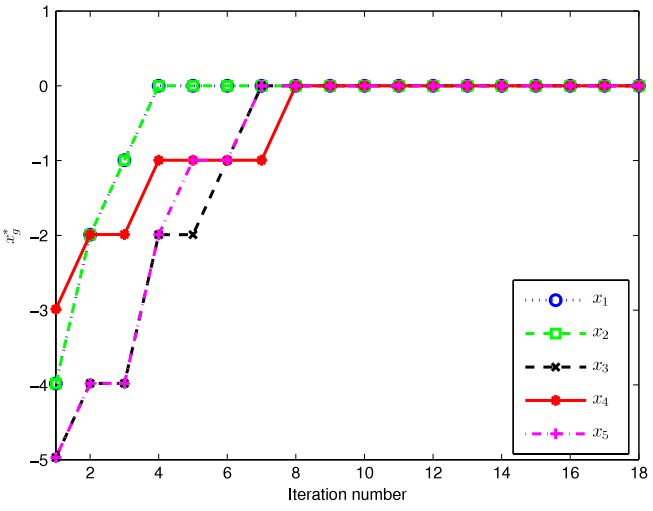

Fig. 19. Evolution of x_g^* for Rastrigin's function.

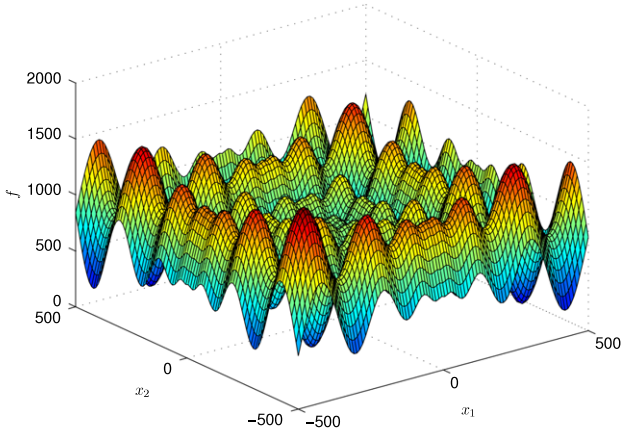

Fig. 20. Schwefel's function.

its equilibria, but it cannot be guaranteed to converge to the global optima. Fig. 22 shows the evolution of x_g^* by applying the proposed collective neurodynamic optimization approach. The global optima is reached. It can be seen that the collective neurodynamic optimization approach is powerful for real-time optimization.

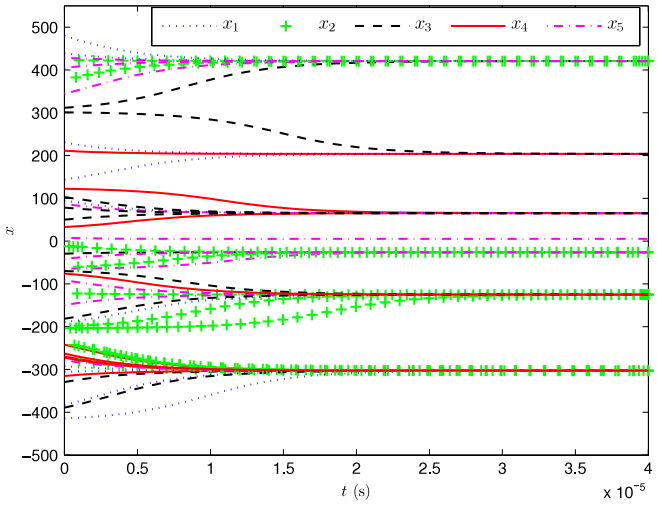

Fig. 21. Transient behaviors of a single neural network for Schwefel's function.

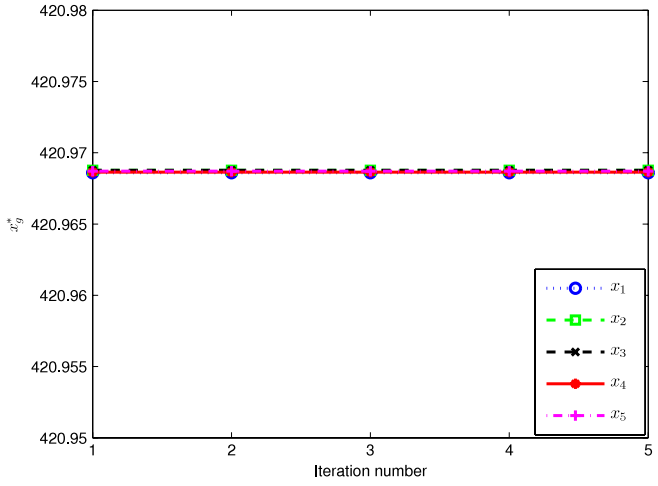

Fig. 22. Evolution of x_g^* for Schwefel's function.

6. Conclusions

This paper presents a novel collective neurodynamic optimization approach to nonconvex optimization problems with bound constraints. The proposed approach can be viewed as an emulation of brainstorming process, and it is implemented in framework of particle swarm optimization. A group of one-layer projection neural networks are exploited in a cooperative way to search for global optimal solutions. Each neural network carries out precise local search according to its own dynamic equations. The whole group iteratively improves the solution quality through cooperative information exchange between each member. One appealing property of the applied neural network model is that its equilibria are in one-to-one correspondence with the Karush–Kuhn–Tucker points of the optimization problem. The proposed approach results in superior performances on benchmark multimodal optimization problems. The collective neurodynamic optimization approach is most suitable for real-time optimization as it is exceptionally computationally efficient.

References

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (2013). *Nonlinear programming: theory and algorithms* (3rd ed.). John Wiley & Sons.
Bertsekas, D. P. (1989). *Parallel and distributed computation: numerical methods*. Englewood Cliffs, NJ: Prentice-Hall.

Bian, W., & Xue, X. (2009). Subgradient-based neural networks for nonsmooth nonconvex optimization problems. *IEEE Transactions on Neural Networks*, *20*, 1024–1038.
Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.
Cheng, L., Hou, Z., Lin, Y., Tan, M., Zhang, W. C., & Wu, F. (2011). Recurrent neural network for nonsmooth convex optimization problems with applications to the identification of genetic regulatory networks. *IEEE Transactions on Neural Networks*, *22*, 714–726.
Chen, W., Zhang, J., Chung, H., Zhong, W., Wu, W., & Shi, Y. (2010). A novel set-based particle swarm optimization method for discrete optimization problems. *IEEE Transactions on Evolutionary Computation*, *14*, 278–300.
Das, S., Maity, S., Qu, B., & Nagaratnam, S. (2011). Real-parameter evolutionary multimodal optimization a survey of the state-of-the-art. *Swarm and Evolutionary Computation*, *1*, 71–88.
Duan, H., Luo, Q., Shi, Y., & Ma, G. (2013). Hybrid particle swarm optimization and genetic algorithm for multi-uav formation reconfiguration. *IEEE Computational Intelligence Magazine*, *8*, 16–27.
Eberhart, R.C., & Kennedy, J. 1995. A new optimizer using particle swarm theory. In *Proc. 6th int. symp. micromachine human sci.* (pp. 39–43), Nagoya, Japan.
Eberhart, R. C., & Shi, Y. (2007). *Computational intelligence: concepts to implementations*. Morgan Kaufmann Publishers.
Fiacco, A. V., & McCormick, G. P. (1990). *Nonlinear programming: sequential unconstrained minimization techniques, Vol. 4*. SIAM.
Forti, M., Nistri, P., & Quincampoix, M. (2004). Generalized neural network for nonsmooth nonlinear programming problems. *IEEE Transactions on Circuits and Systems: Part I*, *51*, 1741–1754.
Gao, X. B., & Liao, L. Z. (2010). A new one-layer neural network for linear and quadratic programming. *IEEE Transactions on Neural Networks*, *21*, 918–929.
Guo, Z., Liu, Q., & Wang, J. (2011). A one-layer recurrent neural network for pseudoconvex optimization subject to linear equality constraints. *IEEE Transactions on Neural Networks*, *22*, 1892–1900.
Hopfield, J. J., & Tank, D. W. (1985). Neural computation of decisions in optimization problems. *Biological Cybernetics*, *52*, 141–152.
Hosseini, A., Wang, J., & Hosseini, S. M. (2013). A recurrent neural network for solving a class of generalized convex optimization problems. *Neural Networks*, *44*, 78–86.
Hu, X., & Wang, J. (2006). Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network. *IEEE Transactions on Neural Networks*, *17*, 1487–1499.
Hu, X., & Wang, J. (2007). Solving generally constrained generalized linear variational inequalities using the general projection neural networks. *IEEE Transactions on Neural Networks*, *18*, 1697–1708.
Hu, X., & Wang, J. (2008). An improved dual neural network for solving a class of quadratic programming problems and its *k*-winners-take-all application. *IEEE Transactions on Neural Networks*, *19*, 2022–2031.
Hyvärinen, A., & Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural Networks*, *13*, 411–430.
Kennedy, M. P., & Chua, L. O. (1988). Neural networks for nonlinear programming. *IEEE Transactions on Circuits and Systems*, *35*, 554–562.
Kennedy, J., & Eberhart, C. 1995. Particle swarm optimization. In *Proc. IEEE int. conf. neural networks* (pp. 1942–1948).
Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation*, *10*, 281–295.
Liao, L. Z., Qi, H. D., & Qi, L. Q. (2004). Neurodynamical optimization. *Journal of Global Optimization*, *28*, 175–195.
Liu, Q., Guo, Z., & Wang, J. (2012). A one-layer recurrent neural network for constrained pseudoconvex optimization and its application for dynamic portfolio optimization. *Neural Networks*, *26*, 99–109.
Liu, S., & Wang, J. (2006). A simplified dual neural network for quadratic programming with its KWTA application. *IEEE Transactions on Neural Networks*, *17*, 1500–1510.
Liu, Q., & Wang, J. (2008a). A one-layer recurrent neural network with a discontinuous activation function for linear programming. *Neural Computation*, *20*, 1366–1383.
Liu, Q., & Wang, J. (2008b). A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming. *IEEE Transactions on Neural Networks*, *19*, 558–570.
Liu, Q., & Wang, J. (2011a). Finite-time convergent recurrent neural network with a hard-limiting activation function for constrained optimization with piecewise-linear objective functions. *IEEE Transactions on Neural Networks*, *22*, 601–613.
Liu, Q., & Wang, J. (2011b). A one-layer recurrent neural network for constrained nonsmooth optimization. *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, *40*.
Liu, Q., & Wang, J. (2013). A one-layer projection neural network for nonsmooth optimization subject to linear equalities and bound constraints. *IEEE Transactions on Neural Networks and Learning Systems*, *24*, 812–824.
Li, G., Yan, Z., & Wang, J. (2014). A one-layer recurrent neural network for constrained nonsmooth invex optimization. *Neural Networks*, *50*, 79–89.
Shang, Y., & Qiu, Y. (2006). A note on the extended Rosenbrock function. *Evolutionary Computation*, *14*, 119–126.
Vapnik, V. (2000). *The nature of statistical learning theory*. Springer.
Wang, J. (1994). A deterministic annealing neural network for convex programming. *Neural Networks*, *7*, 629–641.
Wang, J. (1996). A recurrent neural network for solving the shortest path problem. *IEEE Transactions on Circuits and Systems: Part I*, *43*, 482–486.
Wang, J. (1997). Primal and dual assignment networks. *IEEE Transactions on Neural Networks*, *8*, 784–790.
Whitley, D., Rana, S., Dzubera, J., & Mathias, K. E. (1996). Evaluating evolutionary algorithms. *Artificial Intelligence*, *85*, 245–276.
Xia, Y., Feng, G., & Wang, J. (2004). A recurrent neural network with exponential convergence for solving convex quadratic program and linear piecewise equations. *Neural Networks*, *17*, 1003–1015.
Xia, Y., Feng, G., & Wang, J. (2008). A novel recurrent neural network for solving nonlinear optimization problems with inequality constraints. *IEEE Transactions on Neural Networks*, *19*, 1340–1353.
Xia, Y., Leung, H., & Wang, J. (2002). A projection neural network and its application to constrained optimization problems. *IEEE Transactions on Circuits and Systems: Part I*, *49*, 447–458.
Xia, Y., & Wang, J. (1999). Recurrent neural networks for optimization: the state of the art. In L. R. Medsker, & L. C. Jain (Eds.), *Recurrent neural networks: design and applications* (pp. 13–45). Boca Raton: CRC Press.
Xia, Y., & Wang, J. (2004a). A general projection neural network for solving optimization and related problems. *IEEE Transactions on Neural Networks*, *15*, 318–328.
Xia, Y., & Wang, J. (2004b). A recurrent neural network for nonlinear convex optimization subject to nonlinear inequality constraints. *IEEE Transactions on Circuits and Systems: Part I*, *51*, 1385–1394.
Xia, Y., & Wang, J. (2005). A recurrent neural network for solving nonlinear convex programs subject to linear constraints. *IEEE Transactions on Neural Networks*, *16*, 379–386.
Zhang, S., & Constantinides, A. G. (1992). Lagrange programming neural network. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, *39*, 441–452.
Zhan, Z., Zhang, J., Li, Y., & Chung, H. (2009). Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, *39*, 1362–1381.
Zhan, Z., Zhang, J., Yi, J., & Shi, Y. (2011). Orthogonal learning particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *15*, 832–847.