We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

13

Neurodynamic Optimization: Towards Nonconvexity

Xiaolin Hu
Tsinghua National Lab for Information Science and Technology,
State Key Lab of Intelligent Technology and Systems,
Department of Computer Science and Technology,
Tsinghua University, Beijing 100084,
China

1. Introduction

Optimization is a ubiquitous phenomenon in nature and an important tool in engineering. As the counterparts of biological neural systems, properly designed artificial neural networks can serve as goal-seeking computational models for solving various optimization problems in many applications. In many engineering applications such as optimal control and signal processing, obtaining real-time locally optimal solutions is more important than taking time to search for globally optimal solutions. In such applications, recurrent neural networks are usually more competent than numerical optimization methods because of the inherent parallel nature.

Since the seminal work of Tank and Hopfield in 1980s (Hopfield & Tank, 1986; Tank & Hopfield, 1986), recurrent neural networks for solving optimization problems have attracted much attention. In the past twenty years, many models have been developed for solving convex optimization problems, from the earlier proposals including the penalty method based neural network (Kennedy & Chua, 1988), the switched-capacitor neural network (Rodriǵuez-Vázquez et al., 1990) and the deterministic annealing neural network (Wang, 1994), to the latest development including (Xia, 2004; Gao, 2004; Gao et al., 2005; Hu & Wang, 2007b; Hu & Wang, 2007c; Hu & Wang, 2008). These latest models have a common characteristic: they were all formulated based on optimality conditions of the problems and therefore their equilibria correspond exactly to the solutions of the problems. In addition, for ensuring this correspondence, in contrast to many earlier proposals such as the penalty-based neural network (Kennedy & Chua, 1988), there is no need to let any parameter go infinity. More importantly, if these neural networks are applied to solve nonconvex optimization problems, this nice property will be retained in the sense of critical points instead of global optima, e.g., Karush-Kuhn-Tucker (KKT) points (i.e., the equilibria will correspond no longer to the global optima but to these critical points). Naturally, one will ask if these models are suitable for searching for critical points, especially local optima, of general nonconvex optimization problems.

Unfortunately, there is no guarantee that these optimality-conditions-based neural networks can be directly adopted to solve nonconvex optimization problems. In designing recurrent neural networks for optimization, letting the equilibria correspond to solutions is just one

Open Access Database www.i-techonline.com

Source: Recurrent Neural Networks, Book edited by: Xiaolin Hu and P. Balasubramaniam, ISBN 978-953-7619-08-4, pp. 400, September 2008, I-Tech, Vienna, Austria

issue. The other issue that cannot be neglected is to ensure the stability of the networks at these equilibria. In fact, if the above mentioned neural networks are directly applied to nonconvex problems, their dynamic behaviors could change drastically and become unpredictable. This is not like the circumstance of extending penalty-based neural networks for constrained convex optimization to solve constrained nonconvex problems. In that case, the performances of the networks for solving nonconvex problems can be predicted easily based on their performances in solving convex counterparts, e.g., if a network is previously globally convergent to some points, then it is locally convergent to these points now.

So far, no much achievement in this direction has been obtained yet. In the chapter, I will review some recent progress made by us along this route. Our primary aim is to design locally or globally convergent recurrent neural networks (1) for solving special nonconvex optimization problems whose local minima are also global, and (2) for seeking Karush-Kuhn-Tucker points of general nonconvex optimization problems. The two issues are presented in Section 3 and Section 4, respectively, after a brief introduction of some preliminaries in Section 2. Section 5 summarizes the findings and discusses several possible future directions related to this topic.

2. Preliminaries

Throughout the chapter, without specifications, the following notations are adopted. $\Re^n$ denotes the n dimensional real space and $\Re^n_+$ denotes its nonnegative quadrant. If a function $g : \Re^n \to \Re$, then $\nabla g \in \Re^n$ stands for its gradient and $\nabla^2 g \in \Re^{n\times n}$ stands for its Hessian matrix. If $g(x, y) : \Re^n \times \Re^m \to \Re$, then $\nabla_x g(x,y) \in \Re^n$ and $\nabla_{xx} g(x,y) \in \Re^{n\times n}$ are viewed as respectively the gradient and Hessian matrix of g with respect to x. If a function $G : \Re^n \to \Re^m$, $\nabla G \in \Re^{m\times n}$ stands for its Jacobian matrix. The transpose of a real matrix A is denoted by A^T. A square matrix A is said to be positive definite (positive semidefinite), denoted by $A > 0$ ($A \geq 0$), if $x^TAx > 0$ ($x^TAx \geq 0$) $\forall x \neq 0$. $\|x\| = \sqrt{\sum_{i=1}^n x_i^2}$ denotes the L_2 norm of a vector x.

In many recurrent neural networks, the following projection operator is used as their activation functions

$$P_\Omega(x) = \arg\min_{y\in\Omega} \|x - y\|, \tag{1}$$

where Ω is a closed convex set and "arg" stands for the solution of the minimization problem adhering to it. In general, computing the projection of a point onto a convex set Ω is itself an optimization problem (see (Hu & Wang, 2008) for a neurodynamic solution to such a problem). But if Ω is a box set or a sphere set, the calculation is straight forward. For instance, if $\Omega = \{x \in \Re^n | l_i \leq x_i \leq u_i, \forall i = 1, ..., n\}$, then $P_\Omega(x) = (P_\Omega(x_1), \ldots, P_\Omega(x_n))^T$ and

$$P_\Omega(x_i) = \begin{cases} l_i, & x_i < l_i, \\ x_i, & l_i \leq x_i \leq u_i, \\ u_i, & x_i > u_i. \end{cases} \tag{2}$$

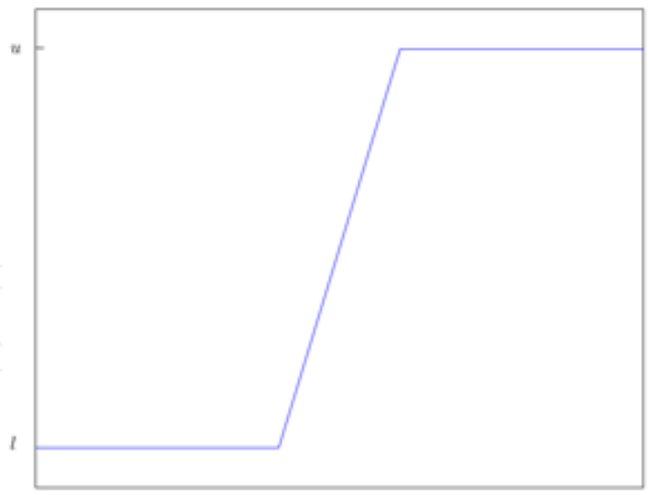


Figure 1. Projection operator in one dimensional case. Reprint of Fig. 1.3 in (Hu, 2007).

Note u_i might be $+\infty$ and l_i might be $-\infty$. Fig. 1 illustrates this operator in one dimensional case, which is somewhat similar in shape to the sigmoid activation function in the Hopfield neural network (cf. Fig. 3(A) in (Hopfield & Tank, 1986)). In particular, if $l = 0$ and $u = \infty$, the operator becomes $P_{\Re^n_+}(x)$. To simplify the notation, in this case it is written as x^+. And the definition can be simplified as $x^+ = (x_1^+, \ldots, x_n^+)^T$ with $x_i^+ = \max(x_i, 0)$.

For another instance, if $\Omega = \{x \in \Re^n | \|x - c\| \leq r, r > 0\}$ where $c \in \Re^n$ and $r \in \Re$ are two constants. Then

$$P_\Omega(x) = \begin{cases} x, & \|x - c\| \leq r, \\ c + \frac{x-c}{\|x-c\|}, & \|x - c\| > r. \end{cases}$$

Definition 1 (Lipschitz Continuity) *A function* F: $\Re^n \to \Re^n$ *is said to be Lipschitz continuous with constant* L *on a set* D *if, for every pair of points* $x, y \in D$,

$$\|F(x) - F(y)\| \leq L\|x - y\|.$$

F is said to be locally Lipschitz continuous on D if each point of D has a neighborhood $D_0 \subset D$ *such that the above inequality holds for every pair of points* $x, y \in D_0$.

Definition 2 (Convexity) *A function* $f : \Re^n \to \Re$ *is convex over a convex set D if for all* $x, y \in D$, *and* $0 < \alpha < 1$

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha) f(y).$$

f(x) is strictly convex on D *if above strict inequality holds whenever* $x \neq y$.

Lemma 1 *A differentiable function* $f : \Re^n \to \Re$ *is convex on a convex set D if and only if for every pair of distinct points* $x, y \in D$,

$$f(y) \geq f(x) + \nabla f(x)^T (y - x).$$

f(x) is strictly convex if and only if above strict inequality holds whenever $x \neq y$.

3. Solving pseudoconvex optimization problems

In this section we consider solving the following problem

$$\min f(x) \quad \text{s.t. } x \in \Omega \tag{3}$$

where $f : \Re^n \to \Re$ is a differentiable nonconvex function and $\Omega \subseteq \Re^n$ is a box set or sphere set defined in Section 2.

To pave the way for discussion, some additional definitions are needed.

Definition 3 (Pseudoconvexity) *A differentiable function $f : \Re^n \to \Re$ is pseudoconvex on a convex set D if for every pair of distinct points $x, y \in D$,*

$$\nabla f(x)^T (y - x) \geq 0 \Longrightarrow f(y) \geq f(x).$$

f is strictly pseudoconvex on D if for every pair of distinct points $x, y \in D$,

$$\nabla f(x)^T (y - x) \geq 0 \Longrightarrow f(y) > f(x),$$

and strongly pseudoconvex on D if there exists a constant $\beta > 0$ such that for every pair of points $x, y \in D$,

$$\nabla f(x)^T (y - x) \geq 0 \Longrightarrow f(y) \geq f(x) + \beta \|y - x\|^2.$$

Definition 4 (Pseudomonotonicity) *A function $F: \Re^n \to \Re^n$ is pseudomonotone on a convex set D if, for every pair of distinct points $x, y \in D$,*

$$F(x)^T (y - x) \geq 0 \Longrightarrow F(y)^T (y - x) \geq 0.$$

F is strictly pseudomonotone on D if, for every pair of distinct points $x, y \in D$,

$$F(x)^T (y - x) \geq 0 \Longrightarrow F(y)^T (y - x) > 0,$$

and strongly pseudomonotone on D if there exists a constant $\gamma > 0$ such that for every pair of points $x, y \in D$,

$$F(x)^T (y - x) \geq 0 \Longrightarrow F(y)^T (y - x) \geq \gamma \|x - y\|^2.$$

It is shown in (Karamardian & Schaible, 1990) that a differentiable function is pseudoconvex and strictly pseudoconvex if and only if its gradient is a pseudomonotone and strictly pseudomonotone mapping, respectively. Moreover, if its gradient is strongly pseudomonotone, the function is strongly pseudoconvex; but the converse is not true (Hadjisavvas & Schaible, 1993).

Lemma 2 *Suppose a differentiable function $f : \Re^n \to \Re$ is pseudoconvex on $\Omega \subset \Re^n$. Then a point $x^* \in \Omega$ satisfies*

$$\nabla f(x^*)^T (x - x^*) \geq 0 \quad \forall x \in \Omega$$

if and only if x^* is a minimum of $f(x)$ in Ω.

One of the important classes of pseudoconvex optimization problems are the quadratic fractional problems in the following form:

$$\begin{aligned} \min \quad & f(x) = \frac{x^T Q x + a^T x + a_0}{b^T x + b_0} \\ \text{s.t.} \quad & x \in \Re^n \in \mathcal{X} = \{x | b^T x + b_0 > 0\}, \end{aligned} \tag{4}$$

where Q is an $n \times n$ symmetric matrix, $a, b \in \Re^n$, $a_0, b_0 \in \Re$. It is well known (e.g., Avriel et al., 1988) that f is pseudoconvex on $\mathcal{X}$ when $Q \geq 0$. Conditions for f being pseudoconvex on $\mathcal{X}$ when Q is not positive semidefinite are discussed in (Cambini et al., 2002). Specially, when $b = 0$, problem (4) reduces to the classic quadratic programming problem, and when $Q = 0$ it reduces to the so-called *linear fractional problem*, which is of course pseudoconvex on $\mathcal{X}$ (Bazaraa et al., 1993).

Throughout this section, $f(x)$ in (3) is assumed to be pseudoconvex on Ω and ∇f is assumed to be Lipschitz continuous on Ω. Note that if f is twice continuously differentiable in an open set containing Ω, then the latter assumption is satisfied automatically.

3.1 Two-layer projection neural network

Consider a recurrent neural network for solving (3) whose dynamics is governed by

$$\frac{dx}{dt} = \lambda\{-x + P_\Omega(x - \alpha F(x)) + \alpha F(x) - \alpha F[P_\Omega(x - \alpha F(x))]\}, \tag{5}$$

where $\lambda > 0$ and $\alpha > 0$ are two scaling factors, $P_\Omega : \Re^n \to \Omega$ is the projection operator defined in section 2, and $F(x)$ stands for $\nabla f(x)$. The architecture of the network is illustrated in Fig. 2. In contrast to the projection neural network, which has a one-layer structure and will be discussed in next subsection, for convenience, the above network is termed *two-layer projection neural network* or TLPNN for short in the chapter.

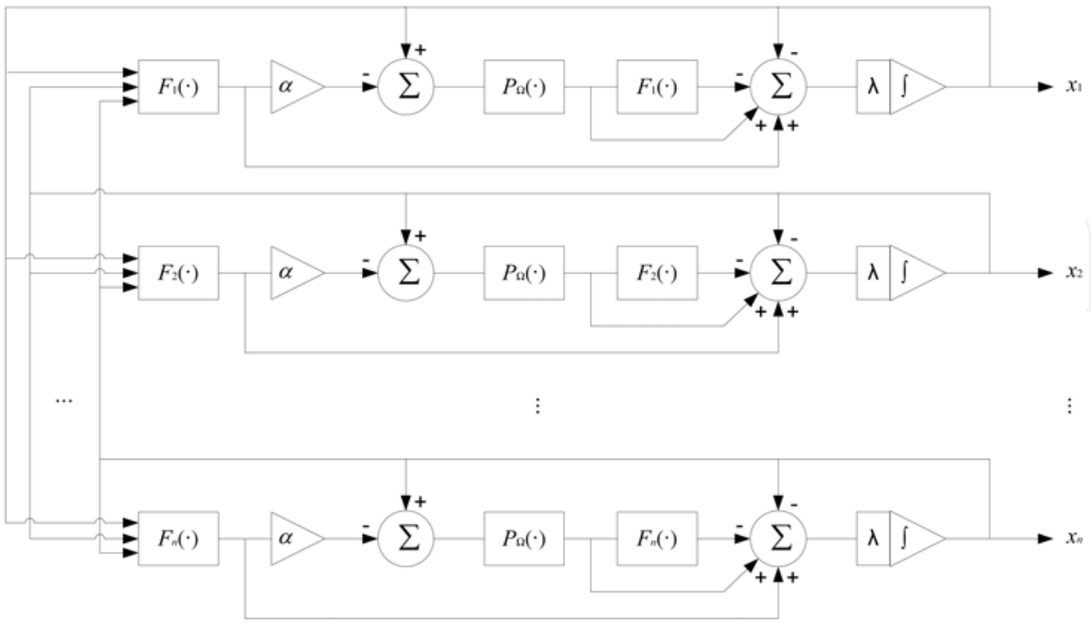


Figure 2. Architecture of the TLPNN (5). Reprint of Fig. 2.1 in (Hu, 2007).

It is proved in (Xia & Wang, 1998) that $x^* \in \Omega$ is a solution of (3) if and only if it is an equilibrium point of the neural network (5). The dynamic behavior of the system was first discussed in (Xia & Wang, 1998), and later in (Xia & Wang, 2000) with different convexity assumptions. In (Hu & Wang, 2006a) we have shown that the corresponding results are still valid when the neural network is employed to solve pseudoconvex optimization problems in the form of (3) (of course with some additional assumptions). The results are contained in the following theorem, which is a restatement of Theorems 2 and 3 in (Hu & Wang, 2006a).

Theorem 1 *Assume that $\nabla f(x)$ is Lipschitz continuous in $\Re^n$ with a constant L.*

- *The TLPNN is globally convergent to a solution of* (3) *with $\alpha < 1/L$. In particular, if* (3) *has a unique solution, the neural network is globally asymptotically stable.*
- *If $\nabla f(x)$ is strongly pseudomonotone on Ω with constant γ, where $\gamma > 4L$, then the TLPNN is globally exponentially stable with $\alpha < (\gamma - 4L)/\gamma L$.*

Remark 1 *Note that the Lipschitz continuity of $\nabla f(x)$ in $\Re^n$ is a stronger condition than the Lipschitz continuity in Ω.*

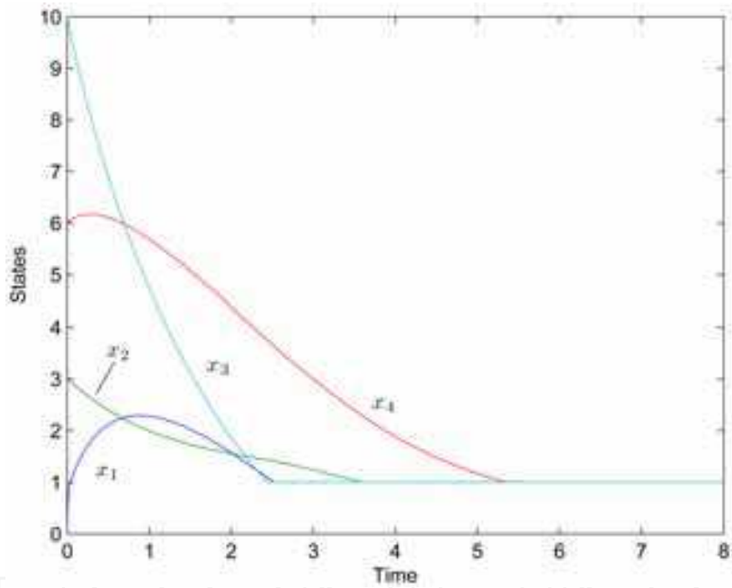


Figure 3. Transient behavior of the TLPNN (5) in Example 1.

Example 1 *We now use the TLPNN to solve a quadratic fractional programming problem* (4) *with*

$$Q = \begin{pmatrix} 5 & -1 & 2 & 0 \\ -1 & 5 & -1 & 3 \\ 2 & -1 & 3 & 0 \\ 0 & 3 & 0 & 5 \end{pmatrix}, \quad a = \begin{pmatrix} 1 \\ -2 \\ -2 \\ 1 \end{pmatrix}, \quad b = \begin{pmatrix} 2 \\ 1 \\ 1 \\ 0 \end{pmatrix},$$
$$a_0 = -2, \quad b_0 = 4.$$

It is easily verified that Q is symmetric and positive definite in $\Re^4$, and consequently f is pseudoconvex on $\mathcal{X} = \{x \in \Re^4 | b^T x + b^0 > 0\}$. We minimize f over $\Omega = \{x \in \Re^4 | 1 \le x_i \le 10, i = 1,\ldots, 4\} \subset \mathcal{X}$ by using the TLPNN with

$$F(x) = \frac{(b^T x + b_0)(2Qx + a) - b(x^T Qx + a^T x + a_0)}{(b^T x + b_0)^2}.$$

This problem has a unique solution $x^* = (1, 1, 1, 1)^T$ *in* Ω. *Simulations show that the TLPNN* (5) is *globally asymptotically stable at* x^* *with any initial point if* α *is appropriately selected. For instance, Fig. 3 shows that the trajectories of the neural network with* $\lambda = 100$, $\alpha = 0.01$ *and the initial point* $x_0 = (0, 3, 6, 10)^T$ *converge to* x^*.

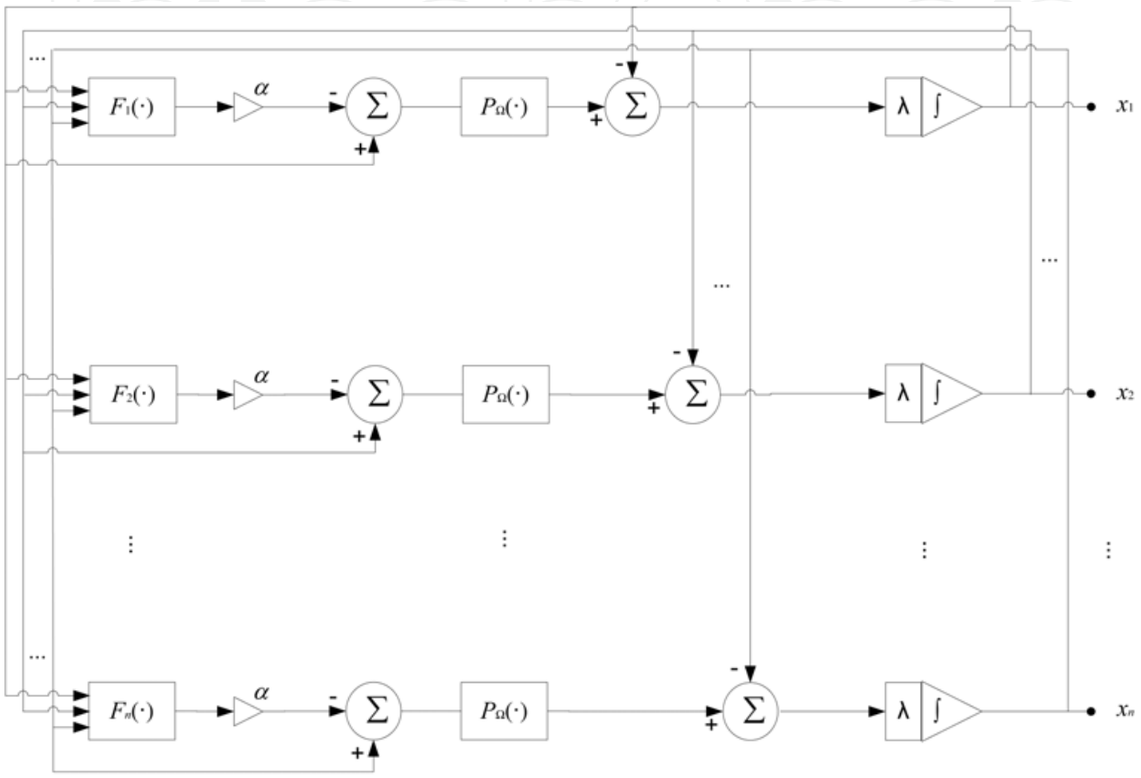


Figure 4. Architecture of the PNN (6). Reprint of Fig. 3.1 in (Hu, 2007).

3.2 One-layer projection neural network

Consider a simpler neural network, called the *projection neural network* or PNN, for solving problem (3) whose dynamic behavior is governed by the following equation

$$\frac{dx}{dt} = \lambda\{-x + P_\Omega(x - \alpha F(x))\}, \tag{6}$$

where the notations are the same as in (5). According to (Kinderlehrer & Stampcchia, 1980), x^* is a solution of (3) if and only if it is an equilibrium point of the PNN. One of the merits of this neural network is its simplicity compared with the TLPNN. The architecture of the network is illustrated in Fig. 4. Its stability results were presented in (Hu & Wang, 2006b, Corollary 3) which is restated as follows.

Theorem 2 *Assume that* $f(x)$ *is twice continuously differentiable on an open set containing* Ω. *Then the PNN* (6) *is stable in the sense of Lyapunov and globally convergent to a solution of* (3). *Moreover,*

- *If rf is strongly pseudomonotone on* Ω *and there exists* $\delta > 0$ *such that* $f(x) \le \delta \|x - x^*\|^2$, *where* x^* *is the unique solution of (3), then the neural network is globally exponentially stable;*
- *If* ∇f *is strongly pseudomonotone on* Ω *and* $\|\nabla f(x)\|$ *has an upper bound on* Ω, *then the neural network is globally asymptotically stable at the unique solution of* (3), *while the convergence rate is upper bounded by*

$$\|x(t) - x^*\| \le \sqrt{\frac{1}{a + b(t - t_0)}} \quad \forall t \ge t_0,$$

where a, b are two positive constants.

Example 2 *We now use the PNN to solve the pseudoconvex optimization problem in Example 1. Simulations show that the PNN* (6) *is globally asymptotically stable at* x^* *with any* α, λ *and any initial point. For instance, Fig.* 5 *shows that the trajectories of the neural network with* $\lambda = \alpha = 1$ *and the initial point* $x_0 = (0, 3, 6, 10)^T$ *converge to* x^*.

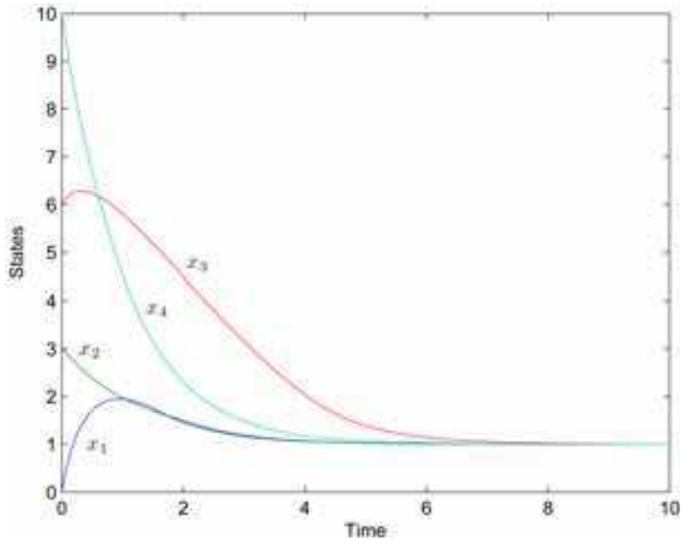


Figure 5. Transient behavior of the PNN (6) in Example 2. Reprint of Fig. 6 in (Hu & Wang, 2006b).

4. Solving general nonconvex optimization problems

Pseudoconvex optimization problems in the form of (3) represent a very special case in the family of nonconvex optimization problems. In this section let's consider solving the following generally constrained nonconvex optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \le 0, \quad x \in \mathcal{X} \end{aligned} \tag{7}$$

where $f : \Re^n \to \Re$, $g(x) = [g_1(x),\ldots,g_m(x)]^T$ is an m-dimensional vector-valued function of n variables, and $\mathcal{X}$ is a box set or a sphere set defined in Section 2. In what follows, the

functions $f, g_1(x),\ldots,g_m(x)$ are assumed to be twice continuously differentiable. If all functions $f(x)$ and $g_j(x)$ are convex over $\Re^n$, the problem is called a *convex optimization problem*; otherwise, it is called a *nonconvex optimization problem*, which is what we are interested in here. Equation (7) represents a wide variety of optimization problems. For example, it is well known that if a problem has equality constraints $h(x) = 0$, then this constraint can be expressed as $h(x) \le 0$ and $-h(x) \le 0$.

For solving general nonconvex optimization problems (including pseudoconvex optimization problems (3) where Ω is a general convex set instead of box set or sphere set), no much progress has been made in the neural network community. This is mainly due to the difficulty in characterizing global optimality of nonconvex optimization problems by means of explicit equations. From the optimization context, it is known that under fairly mild conditions an optimum of the problem must be a Karush-Kuhn-Tucker (KKT) point, while the KKT points are easier to characterize. In terms of developing neural networks for global optimization, it is very hard to find global optima at the very beginning; and a more attainable goal at present is to design neural networks for seeking local optima first with the aid of KKT conditions.

To pave the way for discussion, some additional notations and definitions are needed in this section. In what follows, let $I = \{1, \ldots, n\}$, $J = \{1, \ldots, m\}$. If $u \in \Re^n$, then $u^p = (u_1^p, \ldots, u_n^p)^T$ where p is an integer; $\Gamma(u) = \mathrm{diag}(u_1,\ldots, u_n)$. $\mathrm{int}\mathcal{S}$ denotes the interior of a set $\mathcal{S}$.

Definition 5 *A solution x satisfying the constraints in* (7) *is called a feasible solution. A feasible solution x is said to be a regular point if the gradients of $g_j(x)$, $\nabla g_j(x)$, $\forall j \in \{ j \in J \mid g_j(x) = 0\}$, are linearly independent.*

Definition 6 *A point x^* is said to be a strict minimum of the problem in* (7) *if $f(x^*) < f(x)$, $\forall x \in K(x^*) \cap S$, where $K(x^*)$ is a neighborhood of x^* and S is the feasible region of the problem.*

According to (Kinderlehrer & Stampcchia, 1980), the Karush-Kuhn-Tucker (KKT) condition (Bazaraa et al., 1993) for problem (7) can be expressed as

$$\begin{cases} P_{\mathcal{X}}(x - \alpha(\nabla f(x) + \nabla g(x)y)) = x, \\ (y + \alpha g(x))^+ = y, \end{cases} \tag{8}$$

where $\alpha > 0$, $y \in \Re^m$ and $\nabla g(x) = (\nabla g_1(x), ..., \nabla g_m(x))$.

The classical Lagrangian function associated with problem (7) is defined as

$$L(x, y) = f(x) + \sum_{j=1}^{m} y_j g_j(x). \tag{9}$$

Note that the Hessian of the Lagrangian function is calculated as

$$\nabla_{xx} L(x, y) = \nabla_{xx} f(x) + \sum_{j=1}^{m} y_j \nabla_{xx} g_j(x). \tag{10}$$

Lemma 3 (Second-order sufficiency conditions (Bazaraa et al., 1993)) *Suppose that x^* is a feasible point to problem (7) and $x^* \in int\ \mathcal{X}$. If there exists $y^* \in \Re^m$, such that (x^*, y^*) is a KKT point pair and the Hessian matrix $\nabla_{xx}L(x^*, y^*)$ in* (10) *is positive definite on the tangent subspace:*

$$M(x^*) = \{d \in \Re^n | d^T \nabla g_j(x^*) = 0, \forall j \in J(x^*)\},$$

where J(*x**) *is defined by*

$$J(x^*) = \{j \in J | y_j^* > 0\}, \tag{11}$$

then *x**is a strict minimum point of problem (7).

In what follows, let $\Omega = \mathcal{X} \times \Re_+^n$ and Ω* denote the KKT point set of (7) or the solution set of (8).

4.1 Local convergence of the extended projection neural network

In a series of papers (Xia & Wang, 2004; Xia, 2004; Xia & Feng, 2005; Xia et al., 2007), a recurrent neural network, termed extended projection neural network (or EPNN for short), was developed for solving the convex optimization problems in the form of (7) with the following dynamical equation:

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} P_{\mathcal{X}}(x - \alpha(\nabla f(x) + \nabla g(x)y)) - x \\ (y + \alpha g(x))^+ - y \end{pmatrix}, \tag{12}$$

where $\alpha > 0$. According to the projection formulation (8), the equilibria of the above EPNN correspond to the KKT points of problem (7) exactly. If problem (7) is convex, then the KKT points correspond to the global optima, and the EPNN solves the problem. One wonders what will happen if (12) is used to solve a nonconvex program in the form of (7). Contrary to our expectation, in the nonconvex case, the EPNN can not be guaranteed to converge to any KKT point (which may not correspond to a global optimum), even locally, as will be shown by numerical examples later on. It is thus demanded to find some necessary and/or sufficient conditions that guarantee the local convergence of the neural network. The following theorem provides such a set of sufficient conditions, which is an improved version of Theorem 9.4 in (Hu, 2007).

Theorem 3 *Let x* be a feasible and regular point of problem* (7)*, and u* =* $((x^*)^T, (y^*)^T)^T$ *be the corresponding KKT point of the problem. If the Hessian matrix* $\nabla_{xx}L(x^*; y^*)$ *in* (10) *is positive definite, then the EPNN* (12) *is asymptotically stable at u*, and x* is a strict local minimum of the problem.*

Remark 2 In *Theorem 9.4 of (Hu, 2007) there is an additional requirement on u*: it should satisfy the second-order sufficiency conditions in Lemma 3. This requirement is actually unnecessary as it can be covered by the positive definiteness of* $\nabla_{xx}L(x^*; y^*)$.

4.2 Augmented Lagrange networks

Theorem 3 reveals that if the Hessian matrix of the Lagrangian function is positive definite at a local minimum solution, the EPNN (12) may be locally convergent to that local optimum. But in many cases, this condition fails to exist. Fortunately, there exist ways to generate this condition, and one popular technique is to utilize the augmented Lagrangian functions (Li & Sun, 2000).

In 1992, Zhang and Constantinides proposed a neural network based on the augmented Lagrangian function for seeking local minima of the following equality constrained optimization problem (Zhang & Constantinides, 1992):

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

where $f : \Re^n \to \Re$, $h : \Re^n \to \Re^m$ and both f and h are assumed twice continuously differentiable. The dynamic equation of the network is as follows

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \nabla h(x)y + c\nabla h(x)h(x)) \\ h(x) \end{pmatrix},$$

where $c > 0$ is a control parameter. Under the second-order sufficiency conditions, the neural network can be shown convergent to local minima with appropriate choice of c. The disadvantage of the neural network lies in that it handles equality constraints only. Though in theory inequality constraints can be converted to equality constraints by introducing slack variables, the dimension of the neural network will inevitably increase, which is usually not deemed a good strategy in terms of model complexity.

An alternative extension of the neural network in (Zhang & Constantinides, 1992) for handling inequality constraints in (7) directly can be found in (Huang, 2005) and its dynamic system is as follows (the bound constraint $x \in \mathcal{X}$ is not considered explicitly in that paper):

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \nabla g(x)y^2 + c\nabla g(x)\Gamma(y^2)g(x)) \\ 2\Gamma(y)g(x) \end{pmatrix}.$$

The local convergence of the neural network to its equilibrium set, denoted by $\hat{\Omega}^e$, was proved by using the linearization techniques, and moreover, $\Omega^* \subset \hat{\Omega}^e$. However, it is clear that $\hat{\Omega}^e \neq \Omega^*$. For example, any critical point x of the objective function, which makes $\nabla f(x) = 0$, and $y = 0$ constitute an equilibrium point of the neural network, but in rare cases such an equilibrium corresponds to a KKT point.

Other augmented Lagrangians associated with problem (7) could be tested from the viewpoint of recurrent neural networks. But whether a particular Lagrangian is suited for the design of recurrent neural networks does not have a straightforward answer. For example, the *essentially quadratic augmented Lagrangian* discussed in (Sun et al., 2005) might be a candidate, but its Hessian matrix is not continuous which lays difficulties for analyzing the convergence of the resulting neural networks. On the other hand, the *exponential-type augmented Lagrangian* does have a continuous Hessian matrix, but as the reformulation raises the constraints to exponents of some exponential functions which causes numerical difficulties, that method rarely works in practice. In what follows, we discuss about two promising augmented Lagrangians without these difficulties. For convenience, the resulting neural networks are termed *Augmented Lagrange Networks*.

4.2.1 Partial *p*-power augmented Lagrangian

Problem (7) can be written as

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & \hat{g}(x) \leq b, \quad x \in \mathcal{X} \end{aligned} \tag{13}$$

where $\hat{g}(x) = g(x) + b$. Consider the partial p-power transformation of (13):

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & [\hat{g}_j(x)]^p \le [b_j]^p, \quad j \in J, \\ & x \in \mathcal{X} \end{aligned} \tag{14}$$

with $p \ge 1$. If we assume that $b_1,\ldots,b_m$ are positive constants and $g_1(x),\ldots,g_m(x)$ are nonnegative over $\mathcal{X}$, then problem (13) is equivalent to (14). This assumption does not impose a strict restriction on problem (13) as we can always apply some suitable equivalent transformation on the problem if necessary. Correspondingly, the standard Lagrangian function of problem (14) is defined as

$$L_p(x,y) = f(x) + \sum_{j=1}^{m} y_j([\hat{g}_j(x)]^p - [b_j]^p),$$

where $y_j \ge 0;\ j \in J$, which can be regarded as an augmented Lagrangian function associated with the original problem (7). Then, from (12), the neural network for solving (14) becomes

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} P_{\mathcal{X}}(x - \alpha(\nabla f(x) + \nabla \tilde{g}(x)y)) - x \\ (y + \alpha \tilde{g}(x))^+ - y \end{pmatrix}, \tag{15}$$

where $\tilde{g}(x) = [\hat{g}(x)]^p - [b]^p$.It is easy to calculate

$$\nabla \tilde{g}(x) = p\left([\hat{g}_1(x)]^{p-1}\nabla \hat{g}_1(x), ..., [\hat{g}_m(x)]^{p-1}\nabla \hat{g}_m(x)\right).$$

Problem (14) is termed *partial p-power transformation* of the problem (13) (Li & Sun, 2000). The following lemma reveals one of the advantages of the transformation.

Lemma 4 (Li & Sun, 2000) *Let x^* be a local optimal solution of* (13) *and $x^* \in int\mathcal{X}$. Assume that x^* is a regular point and satisfies the second-order sufficiency conditions. If $J(x^*) \ne \varnothing$*in (11), then *there exists a $q > 0$ such that the Hessian of the partial p-power Lagrangian function, $\nabla_{xx}L_p(x^*, y^*)$, is positive definite when $p > q$.*

Hence we have the following stability results about neural network (15), which follows from Theorem 3 and Lemma 4.

Theorem 4 *Let x^* be a feasible and regular point of problem* (13), *and $u^* = ((x^*)^T; (y^*)^T)^T$ be the corresponding KKT point of the problem satisfying the second-order sufficiency conditions in Lemma 3. Then there exists $p > 0$ such that the neural network* (15) *is asymptotically stable at u^*, and x^* is a strict local minimum of the problem.*

Example 3 *Consider the following nonconvex programming problem in (Li & Sun, 2000)*

$$\begin{aligned} min \quad & f(x) = 1 - x_1x_2, \\ s.t. \quad & g(x) = x_1 + 4x_2 \le 1, \quad x \in \Re^2_+. \end{aligned}$$

This problem has only one local solution $x^* = (0.5, 0.125)^T$, thus also the global solution. The solution is located on the boundary of the feasible region (see Fig. 6). It can be verified that

$$y^* = 0.125 \quad and \quad M(x^*) = \{d \in \Re^2 | d_1 + 4d_2 = 0\}.$$

The Hessian of the Lagrangian function is

$$\nabla_{xx} L(x^*, y^*) = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix},$$

which is indefinite.

Now consider the partial p-power formulation (14) *of the problem. When p* = 3, *a direct calculation yields the new optimal Lagrangian multiplier y** = 0.0417 *and the Hessian of the new Lagrangian*

$$\nabla_{xx} L_p(x^*, y^*) = \begin{pmatrix} 0.25 & 0 \\ 0 & 4 \end{pmatrix},$$

which is a positive definite matrix. We simulate the neural network (15) *to solve the problem. Fig. 7 shows the transient behavior of the neural network* (15) *with the initial point* $u(t_0) = (0.5, 0.2, 0.125)^T$ *that is very close to u*. When p* = 1, *the neural network is identical to* (12) *and it does not converge to u*. But when p*≥1.5, *the neural network converges. When p* = 3, *Fig. 8 displays the transient behavior of x*(*t*) *with several initial points* $u(t_0) = (x(t_0), y(t_0))$ *chosen as follows:* $y(t_0)$ *is random chosen and* $x(t_0)$ *is chosen as* $P_1(0.8, 0.1)$, $P_2(0.3, 0.5)$, $P_3(0, 0.2)$, $P_4(0.4,-0.3)$. *From Fig. 8, it is observed that all four trajectories converges to x* eventually, although the trajectory started from* P_2 *exhibits obvious instability at its earlier evolving stage.*

Moreover, all simulations show that the neural network does not converge to the other KKT points corresponding to the maximum solution $\overline{x}^* = (0, 0)^T$, *even after the partial p-power transformation. This is because*

$$\nabla^2 f(\bar{x}^*) = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$$

is not positive semidefinite.

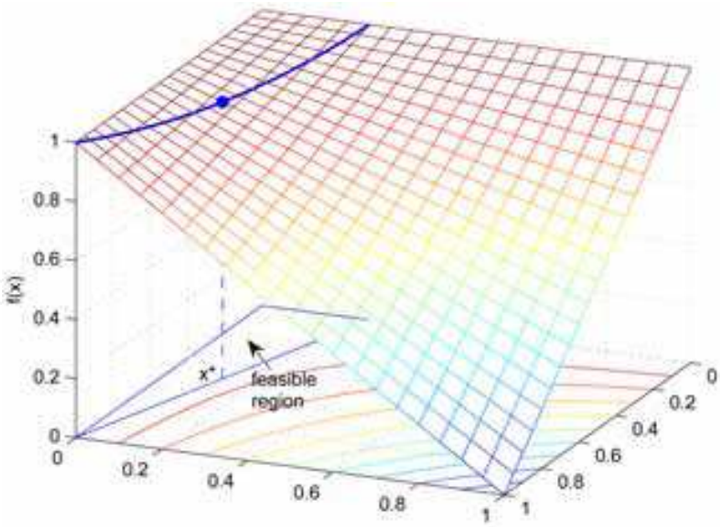
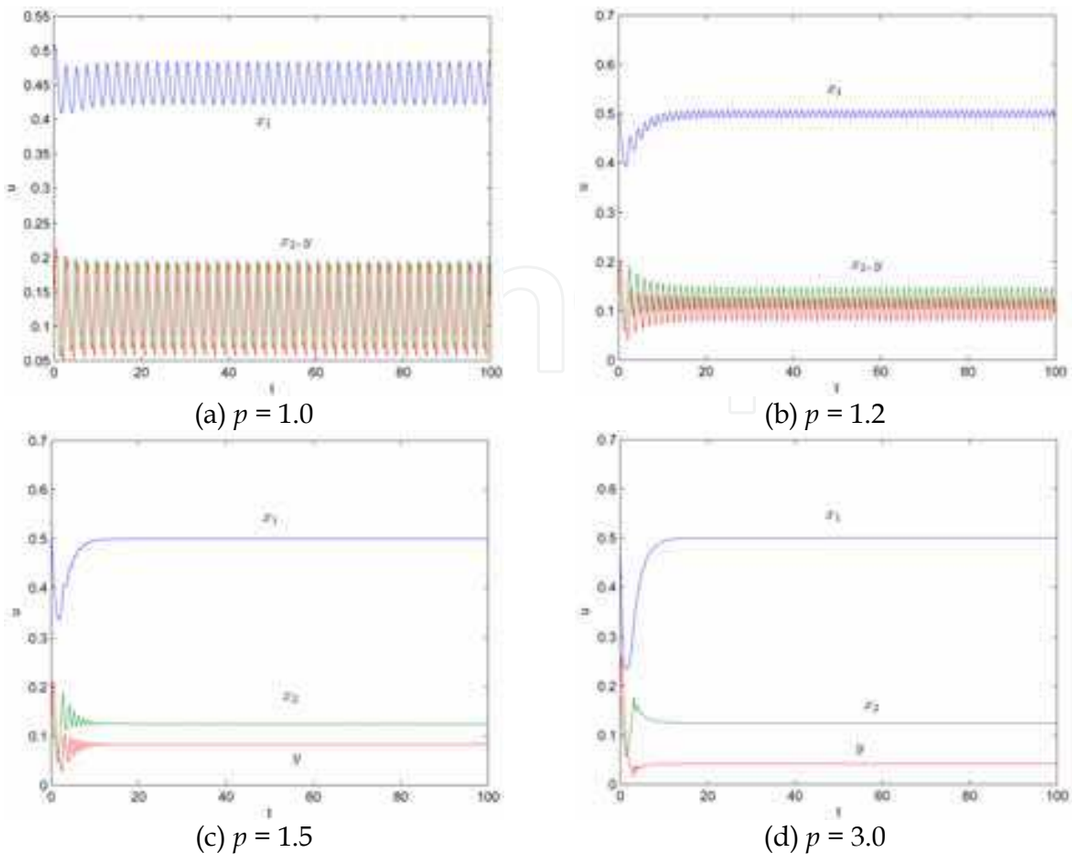


Figure 6. Isometric view of the objective function and constraints in Example 3. Reprint of Fig. 9.1 in (Hu, 2007).

4.2.2 A new augmented Lagrangian

Consider the following augmented Lagrangian function associated with problem (7) slightly differing from that in (Huang, 2005):



(a) p = 1.0 (b) p = 1.2

(c) p = 1.5 (d) p = 3.0

Figure 7. Transient behavior of the neural network (15) with $u(t_0) = (0.5, 0.2, 0.125)^T$ and different values of p in Example 3. Reprint of Fig. 9.2 in (Hu, 2007).

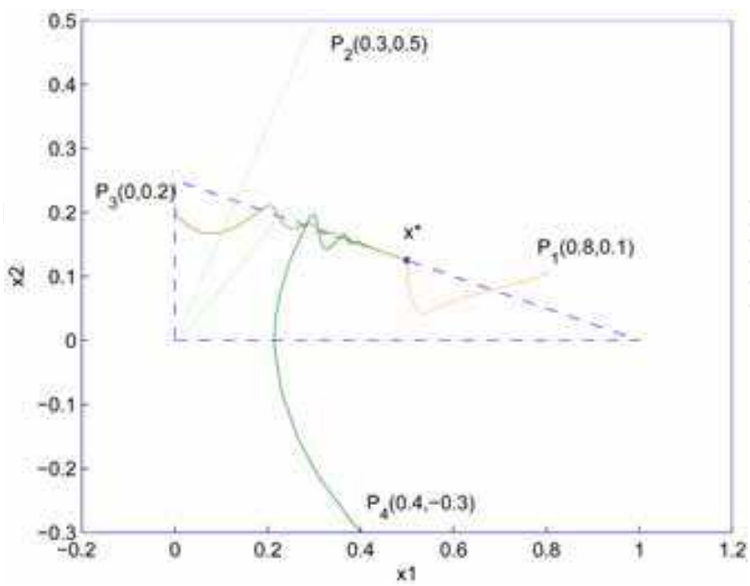


Figure 8: State trajectories $(x_1(t), x_2(t))$ of the neural network (15) with p = 3 and four initial points in Example 3. Reprint of Fig. 9.3 in (Hu, 2007).

$$L_c(x,y) = L(x,y) + \frac{c}{2}\sum_{j=1}^{m}(y_j g_j(x))^2,$$

where $L(x, y)$ is the regular Lagrangian function defined in (9) and $c > 0$ is a scalar. Let Ω^e denote the solution set of the following equations

$$\begin{cases} P_{\mathcal{X}}(x - \nabla_x L_c(x,y)) = x, \\ (y + \alpha g(x))^+ = y, \end{cases}$$

where $\alpha > 0$. We have the following theorem.

Theorem 5 $\Omega^* = \Omega^e$.

Consider a recurrent neural network with its dynamic behavior governed by

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} P_{\mathcal{X}}(x - \alpha(\nabla f(x) + \nabla g(x)y + c\nabla g(x)\Gamma(y^2)g(x))) - x \\ -y + (y + \alpha g(x))^+ \end{pmatrix}, \tag{16}$$

where $\alpha > 0$, $c > 0$ are two contents. Note that the term $\nabla f(x) + \nabla g(x)y + c\nabla g(x)\Gamma(y^2)g(x)$ on the right-hand-side is the expansion of $\nabla_x L_c(x, y)$. Therefore the equilibrium set of the neural network is actually Ω^e, which is equal to Ω^* as claimed in Theorem 5.

Theorem 6 *Let x^* be a feasible and regular point of problem* (7)*, and $u^* = ((x^*)^T, (y^*)^T)^T$ be the corresponding KKT point of the problem satisfying the second-order sufficiency conditions in Lemma 3. Then there exists $c > 0$ such that the neural network* (16) *is asymptotically stable at u^*, and x^* is a strict local minimum of the problem.*

The proofs of Theorems 5 and 6 can be found in (Hu & Wang, 2007a) and (Hu, 2007).

Example 4 *Consider the problem in Example 3 again. This time we use the new augmented Lagrange network* (16) *to solve it. Fig. 9 shows the transient behavior of the neural network* (16) *with the initial point $u(t_0) = (0.5, 0.2, 0.125)^T$ (same as in Example 3). When $c \le 1.5$, the neural network does not converge, and when $c \ge 2$ the neural network converges to u^*. When $c = 5$, Fig. 10 displays the transient behavior of $x(t)$ with four initial points chosen in a similar way as in Example* 3*. It is observed that all four trajectories converges to x^* eventually.*

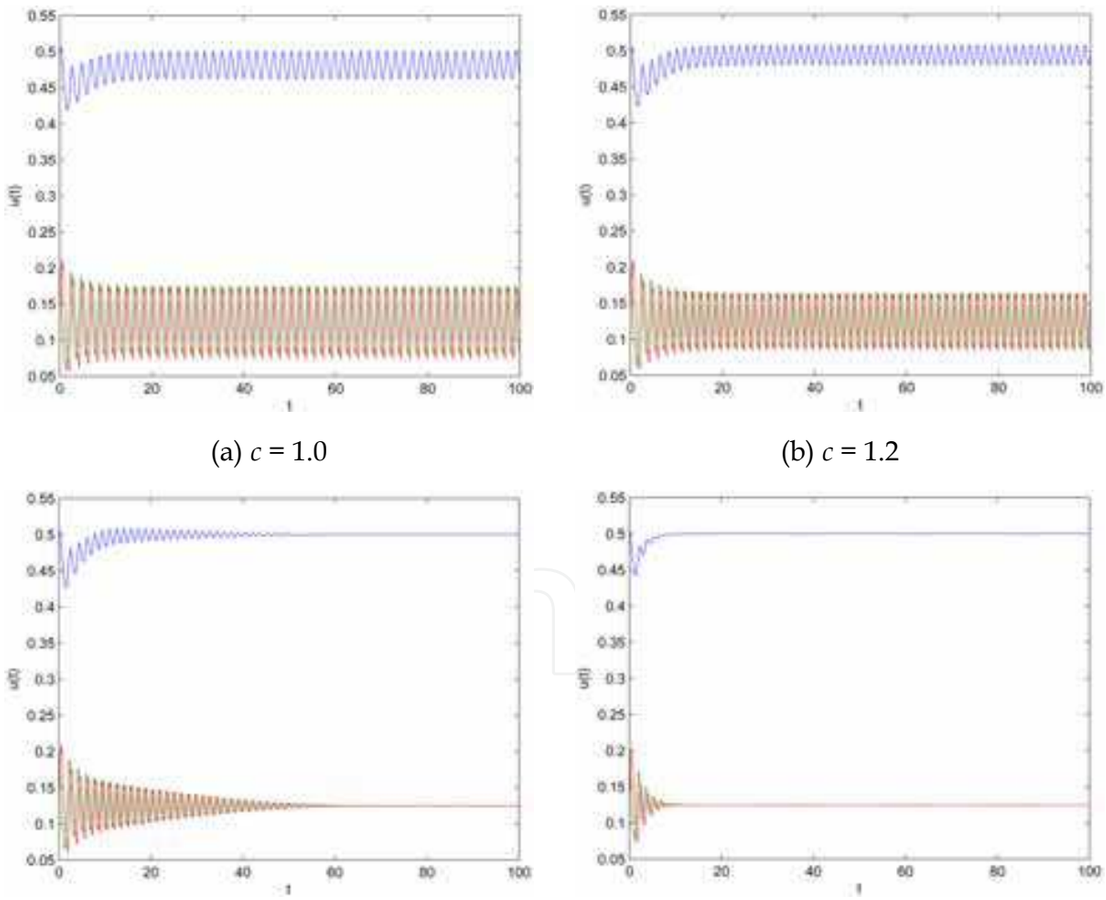
Example 5 *Consider the following problem*

$$\begin{aligned} min \quad & f(x) = 5 - (x_1^2 + x_2^2)/2, \\ s.t. \quad & g_1(x) = -x_1^2 + x_2 - 1 \le 0, \\ & g_2(x) = x_1^4 - x_2 \le 0. \end{aligned}$$

As both $f(x)$ and $g_1(x)$ are concave, the problem is a nonconvex optimization problem. Fig. 11 *shows the contour of the objective function and the solutions to $g_1(x) = 0$* and *$g_2(x) = 0$ on the x_1-x_2 plane. The feasible region is the nonconvex area enclosed by the bold curves. Simple calculations yield*

$$\nabla_{xx} L(x,y) = \begin{pmatrix} -2y_1 + 12x_1^2 y_2 - 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Evidently, $\nabla_{xx}L(x, y)$ *is not positive definite over the entire real space, and the neural network* (12) *can not be applied to solve the problem. Now we check if the neural network (16) can be used to search for the KKT points. There are four KKT points associated with the problem:* u_1^* = (-1.272, 2.618, 4.013, 1.395)^T, u_2^* = (1.272, 2.618, 4.013, 1.395)^T, u_3^* = (0, 0, 0, 0)^T, u_4^* = (0, 1, 1, 0)^T, *but only the first two correspond to local minima. Moreover, it is verified that at either* u_1^* or u_2^*, $J(x^*)$ *defined in Lemma 3 is equal to* {1, 2}, *and* $\nabla g_1(x^*)$, $\nabla g_2(x^*)$ *are linearly independent, which indicates* $M(x^*)$ = 0. *So the second-order sufficiency conditions holds trivially at either point. According to Theorem* 6, *the neural network* (16) *can be made asymptotically stable at* u_1^* *and* u_2^* *by choosing appropriate* $c > 0$. *Fig. 12 displays the state trajectories of the neural network with different values of c started from the same initial point* (-2, 3, 0, 0)^T. *When* c = 0, *the neural network reduces to the neural network* (12). *It is seen from Fig.* 12(a) *that some trajectories diverge to infinity. When* c = 0.1, *the neural network is not convergent, either, as shown in Fig.* 12(b). *However, when* $c \geq 0.2$, *in Figs.* 12(c) *and* 12(d) *we observe that the neural network converges to* u_1^* *asymptotically.*



(a) c = 1.0 (b) c = 1.2

(c) p = 1.5 (d) p = 3.0

Figure 10. State trajectories ($x_1(t)$, $x_2(t)$) of the neural network (16) with c = 5 and four initial points in Example 4.

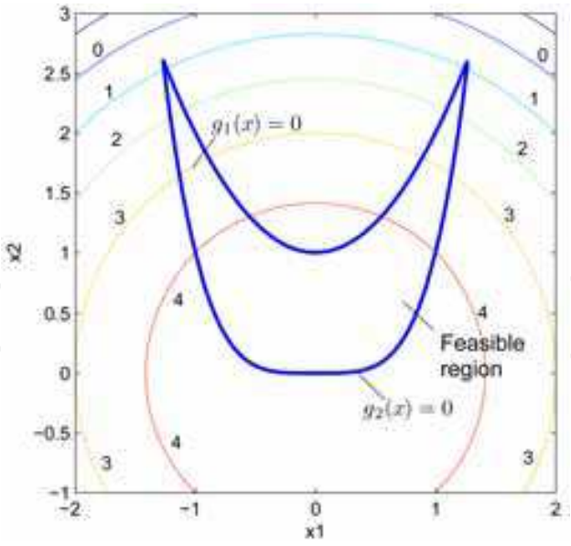


Figure 11. Contour of the objective function and the feasible region in Example 5. Reprint of Fig. 1 in (Hu & Wang, 2007a).

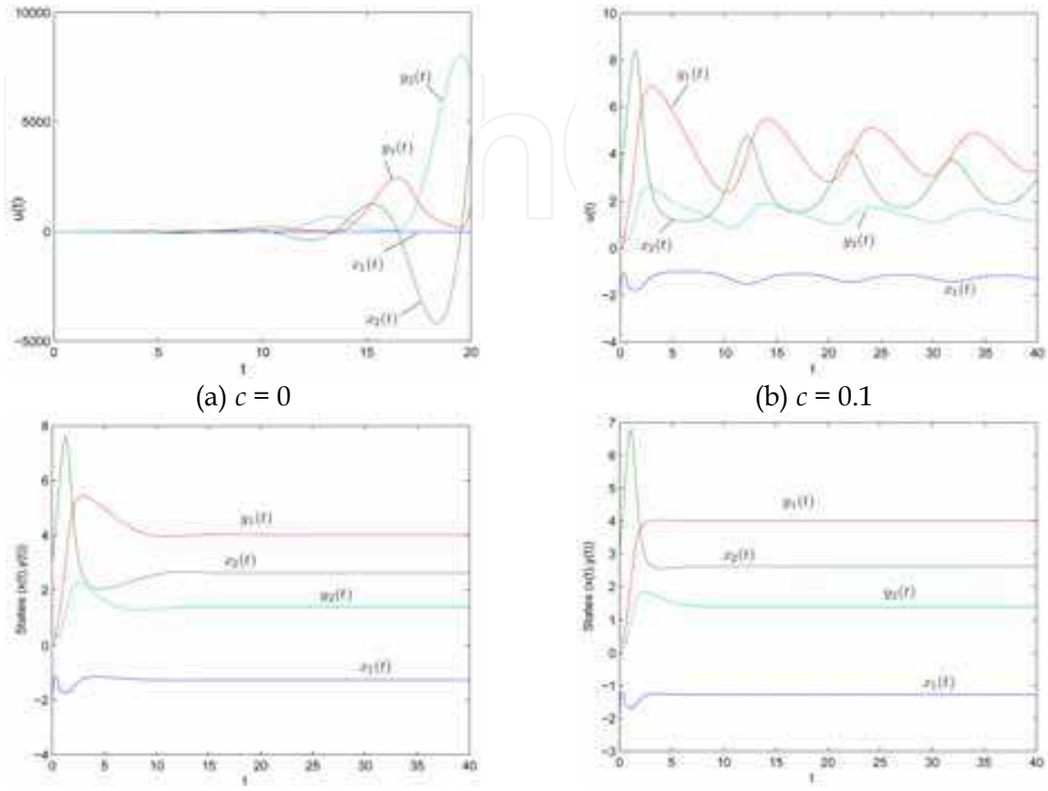
5. Concluding remarks

5.1 Summary of contents

This chapter summarizes our recent work in designing recurrent neural networks for solving nonconvex optimization problems. It is required that the designed neural networks should converge, either locally or globally, to exact local or global solutions of the problems, which is different from the principle of simple penalty-based methods. (Here, the words "locally" and "globally" characterize the convergence behavior of recurrent neural networks while the words "local" and "global" characterize the inherent property of a solution to the problem; they are in general uncorrelated with each other.) First, a special class of nonconvex optimization problems, pseudoconvex optimization problems, were considered. Because any local solution of such a problem is global as well, it is possible to design neural networks which can globally converge to the global solutions. We have revealed that two existing neural networks, called TLPNN and PNN, are capable of accomplishing this task with appropriate conditions.

Second, general nonconvex optimization problems were discussed from the viewpoint of designing neural networks to search for their Karush-Kuhn-Tucker (KKT) points especially the corresponding local solutions. The extended projection neural network (EPNN), originated from solving convex optimization problems in the literature, was studied in this context. The local convergence of the EPNN to KKT points was studied and a set of sufficient conditions was given. Since in many cases these conditions fail to exist, an effective method, augmented Lagrangian techniques were proposed to conquer this difficulty. Two augmented Lagrangian function methods were investigated: one is the partial p-power Lagrangian function existing in the literature and the other is new. Two prominent augmented Lagrange networks were then obtained. For both neural networks, a nice property is that their equilibria are in exact correspondence with the KKT points. Another nice property lies in that by choosing an appropriate control parameter each neural network can be made asymptotically stable at those KKT points associated with local optima under some standard assumptions in the optimization context, although locally. This can be

regarded as a meaningful progress for designing neural networks for completely solving nonconvex optimization problems.
During discussion, numerical examples were provided to illustrate as well as validate the theoretical results.



(a) $c = 0$ (b) $c = 0.1$

(c) $c = 0.2$ (d) $c = 0.5$

Figure 12. Transient behavior the neural network (16) with different values of c in Example 5. Reprint of Fig. 2 in (Hu & Wang, 2007a).

5.2 Future directions

If we classify the nonconvex optimization problems into two categories *Type-I* and *Type-II*, referring to those whose local optima are also global optima and those otherwise, respectively, our primary goal at current stage is to devise some neural networks that can converge globally to the solutions of Type-I problems and can converge globally to local optima sets of Type-II problems. Towards this goal, there is still a long way to walk. Related to the contents of this chapter, some meaningful future directions are as follows. Notice that in Section 4.1 it was shown that the EPNN is locally asymptotically stable at a KKT point (x^*, y^*) (corresponding to a local solution) of the Type-II problem provided that the Hessian of the Lagrange function $\nabla L_{xx}L(x, y) > 0$ at this point. The main idea of the proof of this result (see Hu, 2007, Chapter 9.2) is to construct a neighborhood $\Omega_c(x^*, y^*)$ around this KKT point in which $\nabla L_{xx}L(x, y) > 0$. Then the trajectory originated in it will converge to the KKT point. Hence, for the size of the neighborhood, the larger the better. This condition is actually somewhat too strong. For ensuring the local convergence, it is required $\nabla L_{xx}L(x, y) > 0$ on the trajectory of the network in $\Omega_c(x^*, y^*)$ only, not necessarily on the entire $\Omega_c(x^*, y^*)$. This

new condition can be utilized to state global convergence of the EPNN to a KKT point, while the original one cannot. The reason is that it is impossible for a nonconvex optimization problem that $\nabla L_{xx}L(x, y) > 0$ over the entire space, but it is possible that this inequality holds over a particular trajectory. This is one of the main ideas of a most recent article (Xia et al., 2007). Obviously, this idea can be also applied to the two augmented Lagrange networks discussed in the chapter.

For solving optimization problems with general constraints, the EPNN and its variants play the dominant roles in the community. Recently, a notable progress has been made in (Xia & Feng, 2007) where a much different model was proposed for solving convex optimization problems. It deserves further investigation from the viewpoint of nonconvex optimization.

6. Acknowledgement

The work was supported by the National Natural Science Foundation of China under Grant 60621062 and by the National Key Foundation R&D Project under Grants 2003CB317007, 2004CB318108 and 2007CB311003. It has been partially presented in (Hu, 2007; Hu & Wang, 2006b; Hu & Wang, 2006a; Hu & Wang, 2007a).

7. References

Avriel, M., Diewert, W. E., Schaible, S. and Zang, I. (1988). *Generalized concav ity*, Mathematical Concepts and Methods in Science and Engineering, Plenum Publishing Corporation, New York.

Bazaraa, M. S., Sherali, H. D. and Shetty, C. M. (1993). *Nonlinear Programming: Theory and Algorithms*, 2nd edn, Wiley, New York.

Cambini, A., Crouzeix, J. P. and Martein, L. (2002). On the pseudoconvexity of a quadratic fractional function, *Optimization* 51(4): 677-687.

Gao, X. (2004). A novel neural network for nonlinear convex programming, *IEEE Trans. Neural Netw.* 15(3): 613-621.

Gao, X., Liao, L. and Qi, L. (2005). A novel neural network for variational inequalities with linear and nonlinear constraints, *IEEE Trans. Neural Netw.* 16(6): 1305- 1317.

Hadjisavvas, N. and Schaible, S. (1993). On strong pseudomonotonicity and (semi)strict quasimonotonicity, *Journal of Optimization Theory and Applications* 79(1): 139-155.

Hopfield, J. J. and Tank, D. W. (1986). Computing with neural circuits: a model, *Scienc* 233(4764): 625-633.

Hu, X. (2007). *Solving Variational Inequalities and Related Problems Using Recurrent Neural Networks*, PhD thesis, The Chinese University of Hong Kong, Hong Kong, China.

Hu, X. andWang, J. (2006a). Global stability of a recurrent neural network for solving pseudomonotone variational inequalities, *Proc. IEEE International Symposium on Circuits and Systems*, Island of Kos, Greece, pp. 755-758.

Hu, X. and Wang, J. (2006b). Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network, *IEEE Trans. Neural Netw.* 17(6): 1487-1499.

Hu, X. and Wang, J. (2007a). Convergence of a recurrent neural network for nonconvex optimization based on an augmented lagrangian function, *Proc. 4th International Symposium on Neural Networks*, Vol. 4493 of *Lecture Notes in Computer Science*, Nanjing, China, pp. 194-203.

Hu, X. and Wang, J. (2007b). Design of general projection neural networks for solving monotone linear variational inequalities and linear and quadratic optimization problems, *IEEE Trans. Syst., Man, Cybern. B* 37(5): 1414-1421.

Hu, X. and Wang, J. (2007c). Solving generally constrained generalized linear variational inequalities using the general projection neural networks, *IEEE Trans. Neural Netw.* 18(6): 1697-1708.

Hu, X. and Wang, J. (2008). An improved dual neural network for solving a class of quadratic programming problems and its k-winners-take-all application, IEEE Trans. Neural Netw. accepted.

Huang, Y. (2005). Lagrange-type neural networks for nonlinear programming problems with inequality constraints, *Proc. 44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, pp. 4129-4133.

Karamardian, S. and Schaible, S. (1990). Seven kinds of monotone maps, *Journal of Optimization Theory and Applications* 66(1): 37-46.

Kennedy, M. P. and Chua, L. O. (1988). Neural networks for nonlinear programming, *IEEE Trans. Circuits Syst.* 35: 554-562.

Kinderlehrer, D. and Stampcchia, G. (1980). *An Introduction to Variational Inequalities and Their Applications*, Academic, New York.

Li, D. and Sun, X. L. (2000). Local convexification of the lagrangian function in nonconvex optimization, *Journal of Optimization Theory and Applications* 104(1): 109-120.

Rodríguez-Vázquez, A., Domínguez-Castro, R., Rueda, A., Huertas, J. L. and Sánchez-Sinencio, E. (1990). Nonlinear switched-capacitor neural networks for optimization problems, *IEEE Trans. Circuits Syst.* 37: 384-397.

Sun, X. L., Li, D. and Mckinnon, K. I. M. (2005). On saddle points of augmented lagrangians for constrained nonconvex optimization, *SIAM J. Optim.* 15(4): 1128- 1146.

Tank, D. W. and Hopfield, J. J. (1986). Simple neural optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit, *IEEE Trans. Circuits Syst.* 33(5): 533-541.

Wang, J. (1994). A deterministic annealing neural network for convex programming, *Neural Networks* 7(4): 629-641.

Xia, Y. (2004). An extended projection neural network for constrained optimization, *Neural Computation* 16(4): 863-883.

Xia, Y. and Feng, G. (2005). On convergence conditions of an extended projection neural network, *Neural Computation* 17(3): 515-525.

Xia, Y. and Feng, G. (2007). A new neural network for solving nonlinear projection equations, *Neural Networks* 20: 577-589.

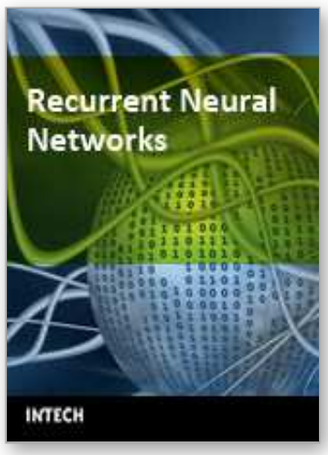
Xia, Y., Feng, G. and Kamel, M. (2007). Development and analysis of a neural dynamical approach to nonlinear programming problems, *IEEE Trans. Automatic Control* 52(11): 2154-2159.

Xia, Y. and Wang, J. (1998). A general methodology for designing globally convergent optimization neural networks, *IEEE Trans. Neural Netw.* 9(6): 1331-1343.

Xia, Y. and Wang, J. (2000). Global exponential stability of recurrent neural networks for solving optimization and related problems, *IEEE Trans. Neural Netw.* 11(4): 1017-1022.

Xia, Y. and Wang, J. (2004). A recurrent neural network for nonlinear convex optimization subject to nonlinear inequality constraints, *IEEE Trans. Circuits Syst. I* 51(7): 1385-1394.

Zhang, S. and Constantinides, A. G. (1992). Lagrange programming neural networks, *IEEE Trans. Circuits Syst. II* 39: 441-452.



Recurrent Neural Networks

Edited by Xiaolin Hu and P. Balasubramaniam

ISBN 978-953-7619-08-4
Hard cover, 400 pages
Publisher InTech
Published online 01, September, 2008
Published in print edition September, 2008

The concept of neural network originated from neuroscience, and one of its primitive aims is to help us understand the principle of the central nerve system and related behaviors through mathematical modeling. The first part of the book is a collection of three contributions dedicated to this aim. The second part of the book consists of seven chapters, all of which are about system identification and control. The third part of the book is composed of Chapter 11 and Chapter 12, where two interesting RNNs are discussed, respectively.The fourth part of the book comprises four chapters focusing on optimization problems. Doing optimization in a way like the central nerve systems of advanced animals including humans is promising from some viewpoints.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Xiaolin Hu (2008). Neurodynamic Optimization: towards Nonconvexity, Recurrent Neural Networks, Xiaolin Hu and P. Balasubramaniam (Ed.), ISBN: 978-953-7619-08-4, InTech, Available from: http://www.intechopen.com/books/recurrent_neural_networks/neurodynamic_optimization__towards_nonconvexity

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2008 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen