Linear Programming Problem for Nonlinear Convex Set Constraints Based on Nonlinear Neural Network

Cai Huaxian[1], Tian Tian, Cai Yilin[2]
[1]Kunming University, Kunming
Yunnan, 650214, China
[2]The second occupation Technical Secondary School in Kunming
Kunming, Yunnan, 650214, China
chxkmxy@163.com

Journal of Digital Information Management

ABSTRACT: *Linear programming problem is widely applied in engineering group. And artificial neural network is an effective and practical method and approach for solving linear programming problem of nonlinear convex set constraints in engineering field. Most models of artificial neural network are nonlinear dynamic system. If the objective function of optimization calculation problem is corresponding to some energy function of network, steady state point is the local or global optimal dynamic process solution for optimizing the problem. This paper studied the optimization problem with nonlinear constrained convex set by the application of nonlinear neural network. First, the inverse optimization problem with nonlinear convex set constraints into nonlinear bilevel programming problem. Then the optimization solution was searched by establishing neural network model. In addition, the stability of model was analyzed and sufficient conditions for global asymptotic stability of the balance point were given. Finally, the validity and effectiveness of the conclusion was verified by example.*

Categories and Subject Descriptors:
G.2 [Discrete Mathematics]; G.1.5 [Roots of Nonlinear Equations]: Rate of Convergence

General Terms: Petri net, Mapping

Keywords: Nonlinear Convex Set, Neural Network, Bilevel Programming, Balance Point, Stability

Received: 11 June 2014, Revised 19 July 2014, Accepted 28 July 2014

1. Introduction

In real life, we come cross many optimization problems in fields such as engineering design, resource allocation, business operations, rocket launch trajectory, etc. Most of them are nonlinear optimization problem when abstracting away these problems. Nonlinear optimization problem is approximate description of optimization problem in real life. We can give an accurate answer for practical problem by accurate mathematical linguistics through this approximation. Since 1980 s, method of neural network was adopted into this field. People gradually realized the neural network can provide an almost perfect solution for the optimization problem because it can nearly all demands in many aspects. As the research become deeper, the researchers in this field achieved many important research results and apply them in reality.

In nonlinear problem, nonlinear programming can be understood as optimization problem for objective function with equation or inequation constraint condition in broad sense [1]. So far, convex optimization problem is which people can further study. In real life, many practical engineering problems all require real time solution on optimization problem. Therefore, many scholars studied the nonlinear programming problem based on neural network. Su Peng from Harbin Institute of Technology created a recurrent neural network to solve the pseudo-convex optimization problem with linear equation constraint condition in the perspective of variational inequality in Research on Several Kinds of Nonlinear Optimization Problems Based on Neural Network [2]. Ji Weidong from Northeast Forestry University proposed the application of evolutionary computation in swarm

intelligence to solve main problems of neural network existing in complex system modeling in Research on Learning Method of Evolutionary Computation Optimizing Feed Forward Neural Network [3]. Zhang Haibo from Beijing University of Posts and Telecommunications studied the optimal solution problem in BWCS by chaotic neural network technology in Research on Wireless Resources Optimization Strategy Based on Chaotic Neural Network [4].

2. Overview

Artificial neural network (ANN) expresses knowledge of problem solving by distributed memory of link weight between lots of interconnected neurons. It possesses characteristics such as parallel processing, self learning, and approaching random nonlinear function with arbitrary precision. The above characteristics of neural network achieve its application in complex system modeling. In nature, the learning of neural network is the optimization process of network structure and weight. Purpose of information processing is achieved by adjusting the mutual connection relationship between nodes inside.

Most models of artificial neural network are nonlinear dynamic system. If the objective function of optimization calculation problem is corresponding to some energy function of network, the process of network dynamically moving towards the direction of minimum energy can be regarded as the solution process of optimization problem and steady state point becomes the local or global optimal dynamic process solution for optimizing the problem [5, 6].

3. Linear Programming Problem of Convex Set Constraints

3.1 Statement and Conversion of Problem

Consider the following linear programming problem with nonlinear convex set constraints:

$$\min_{x} \; c^T x \quad s.t. \; h(x) \le 0 \tag{1}$$

Where $c, x \in R^n$ and $h: R^n \to R$ is a continuously differentiable convex function. A set $C \subseteq R^n$ and a real number Z^* are given. According to the literature [7], the inverse optimization problem related linear programming problem (1) is to search for $c \in C$ which can make optimization value of objective function for linear programming problem (1) to approach to z. Therefore, inverse optimization problem can be written as the following form:

$$\begin{array}{ll} \textbf{UP:} \min\limits_{c} f(c) & \textbf{LP:} \min\limits_{x} c^T x \\ s.t. \quad c \in C & s.t. \quad h(x) \le 0 \end{array}$$

Where $f(c) = |Q(c) - z^*|$, $|.|$ is expressed as the Euclidean norm on R. Obviously, problem (3) is a bilevel programming problem. *UP* is termed as upper level problem and LP is termed as lower level problem. We assume that:

1) Cost vector set C is non-void. $C = \{c: g(c) \le 0\}$, among which $g(c)$ is nonlinear convex function;

2) Feasible region $\{x: h(x) \le 0\}$ is non-void and bounded. Without loss of generality, we take place $f(c) = |Q(c) - z^*|$ by $f^2(c) = (Q(c) - z^*)^2$ We assume 2) set up, and then upper level (LP) use KKT most conditional for $c \in C$. The bilevel programming problem can be turned into a single level programming problem.

$$\min_{(c,x,\gamma)} (c^T x - z^*)^2 \quad \begin{array}{l} s.t. \quad g(c) \le 0 \\ \nabla_x L(x, \gamma) = 0 \\ \gamma^T h(x) = 0 \\ h(x) \le 0 \\ \gamma \ge 0 \end{array} \tag{3}$$

$L(x, \gamma) = c^T x + \gamma h(x)$ is Lagrange function. λ is termed as Lagrange multiplier. The inverse optimization problem corresponding to linear programming problem (1) can be turned into a class of convex programming problem similar to programming problem (3) equivalently. Therefore, the solution of inverse optimization problem can be turned into solution of convex programming problem (3) corresponding to it. Programming problem (3) can be turned into the following form by applying Fish-Burmeister function $\Phi'(a, b, \varepsilon)$.

$$\begin{array}{l} \min\limits_{(c,x,\gamma)} (c^T x - z^*)^2 \\ s.t. \quad g(c) \le 0 \\ \qquad \nabla_x L(x, \lambda) = 0 \\ \qquad \sqrt{h_i^2 + \gamma_i^2 + \varepsilon} + h_i - \gamma_i = 0, \; i = 1, 2, ..., n \end{array} \tag{4}$$

Suppose $F(c, x, \gamma) = (c^T x - z^*)^2$, $G(c, x, \gamma) = g(c)$,

$$H(c, x, \gamma) = \begin{bmatrix} \nabla_x L(x, \gamma) = 0 \\ \sqrt{h_i^2 + \gamma_i^2 + \varepsilon} + h_i - \gamma_i = 0, \; i = 1, 2, ..., n \end{bmatrix}$$

and $y^T = (c^T, x^T, \gamma^T)$, (4) can be turned into the following form:

$$\begin{array}{l} \min\limits_{y} F(y) \; s.t. \qquad H_j(y) = 0, j = 1, 2, ..., 2n \\ G_k(y) \le 0, k = 1, 2, ..., m \end{array} \tag{5}$$

Suppose y is a feasible point of linear programming problem (5). If gradient $\nabla H_1(y), ..., \nabla H_{2n}(y)$ and $\nabla G_k(y), k \in \{k: G(y) = 0, k = 1, 2, ..., m\}$ is linearly independent, and then y is a regular point.

3.2 Eestablishment of neural network model

Lagrange function of programming problem (5) is defined as $L(y, \lambda, \mu) = F(y) + \lambda H(y) + \mu(G(y) + B)$. $B \in R^m$ is termed as slack variable; $\lambda \in R^{2n}$; $\mu \in R^m$ is termed as Lagrange multiplier. The following proposition set up based on the famous KKT condition: if there is (y^*, λ^*, μ^*) which can make $\nabla_y L(y^*, \lambda^*, \mu^*) = 0$, $H(y^*) = 0$ and $G(y^*) + B$ set up,

then y^* is an optimization solution for programming problem (5).

According to this proposition, energy function for programming problem (4) can be structured as the following form:

$$E(y,\lambda,\mu)=\frac{1}{2}\|\nabla_y L(y,\lambda,\mu)\|^2+\frac{1}{2}\|\nabla_y L(y,\lambda,\mu)\|^2+\frac{1}{2}\|\nabla_\mu L(y,\lambda,\mu)\|^2,$$

then the corresponding neural network model is established for solving problem (4). This model for neural network is termed as gradient neural network. Its dynamic equation can be described as follows:

$$\begin{cases}\frac{dy}{dt}=-\nabla_y E(y,\lambda,\mu)\\ \frac{d\lambda}{dt}=-\nabla_\lambda E(y,\lambda,\mu)\\ \frac{d\mu}{dt}=-\nabla_\mu E(y,\lambda,\mu)\end{cases} \quad (6)$$

Its component form is as follows:

$$\frac{dy}{dt}=-\nabla_{yy}^2 L(y,\lambda,\mu).\nabla_y L(y,\lambda,\mu)-H^T(y).\nabla_y H(y)-(G(y)+B)^T\nabla_y G(y),$$

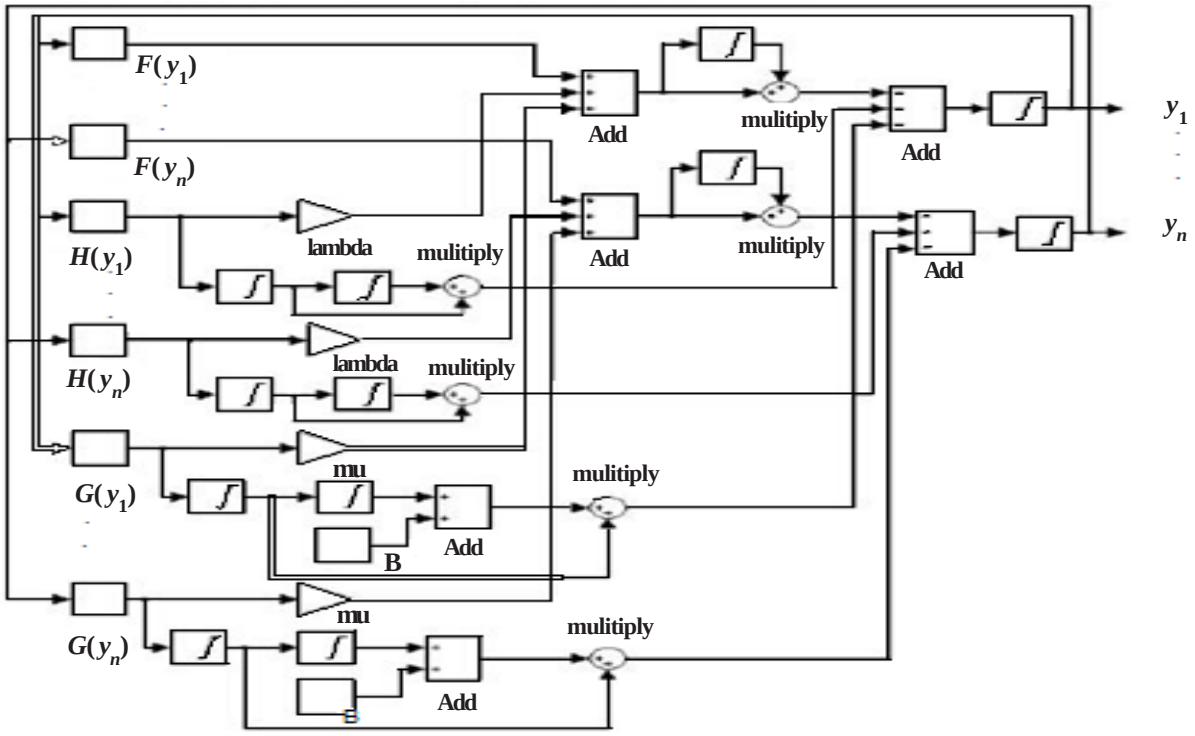

Figure 1. Model structure of neural network (5)

$$\frac{d\lambda}{dt}=-\nabla_{y\lambda}^2 L(y,\lambda,\mu).\nabla_y L(y,\lambda,\mu),$$

$$\frac{d\mu}{dt}=-\nabla_{y\mu}^2 L(y,\lambda,\mu).\nabla_y L(y,\lambda,\mu)$$

Theorem 1 can be derived from the above analysis: if (y^*, λ^*, μ^*) is a balance point, if and only if y^* is an optimization solution for programming problem.

3.3 Stability Analysis of Model

Theorem 2 can be derived from the above analysis: if suppose assumption 1) and 2) set up, then nonlinear neural network (6) exists the only balance point which is globally asymptotically stable.

Proof: in the condition of assumption 1) and 2), programming (5) is a class of convex programming problem. Programming problem exists the only optimization solution. Suppose y^* is the optimization solution for the programming problem (5). According to theorem 1, there are two constant $\lambda \in R^{2n}$ and ;$\mu \in R^m$ which can make (y^*, λ^*, μ^*) as one balance point of neural network (6). Suppose η = (y, λ, μ), and then establish Lyapunov function in following form:

$$V(\eta)=E(\eta)-E(\eta^*)=E(\eta)$$

Obviously, $V(\eta)>0,,\forall\eta\neq\eta^*$ and $V(\eta^*)=0$. We solve the solution $\eta(t)$ of $V(\eta)$ related to neural network (6), and there is:

$$\frac{d}{dt}V(\eta)=\nabla(\eta)^T.\frac{d\eta}{dt}=-\|\nabla E(\eta)\|^2<0,\eta\neq\eta^*$$

According to Lyapunov stability theory, we can know that balance point η^* of neural network (6) is globally asymptotically stable [8]. The proof is fulfilled.

3.4 Numerical Example

An example is applied to explain the validity and effectiveness of neural network method for solving inverse optimization problem.

For example, consider the following inverse optimization problem:

$$\min_{(c,x,\gamma)} (c_1x_1 + c_2x_2 - 14)^2$$

$$\begin{aligned} s.t.\ \ & c_1^2\ c_2^2 - 8c_1 - 12c_2 + 39 \le 0 \\ & -c_1 - 2\gamma_1 x_1 + 2\gamma_2(x_1 - 4) = 0 \\ & -c_2 - 2\gamma_2 x_2 + 2\gamma_1(x_2 - 4) = 0 \\ & \sqrt{\gamma_1^2 + [\,x_1^2 + (x_2-2)^2 - 16\,]^2 + \varepsilon} - \gamma_1 + x_1^2 + (x_2-2)^2 - 16 = 0 \\ & \sqrt{\gamma_2^2 + [(x_1-4)^2 + x_2^2 - 4]^2 + \varepsilon} - \gamma_2 + (x_1-4)^2 + x_2^2 - 4 = 0 \end{aligned} \quad (8)$$

Take the initial value $y_0 = (4.2, 1, 3, 2, 0, 0)$, $\lambda_0 = (0, 0, 0, 0, 0)$, $\mu_0 = 0$. $B = 0.01$ And suppose ε take the value of 0.01, 0.001 and 0.0001. In MATLAB, we applied classical fourth-order Runge-Kutta method to do simulation experiment on neural network (6) corresponding to problem (8). Figure. 2 showed the state trajectory of solution (x_1, x_2, c_1, c_2) for neural network (6) corresponding to problem (8) when $\varepsilon = 0.0001$. When ε take the value of 0.01, 0.001 and 0.0001, the optimization solution of problem (8) take the value of (3.9902, 1.9983, 2.0037, 2. 9975), (3.9997, 1.9998, 2.0004, 2. 9998) and (4.0000, 2.0000, 2.0000, 3.0000) (as shown in table 1). Through table 1, we can know that the optimization solution $(x_1^*, x_2^*, c_1^*, c_2^*)$ for problem (8) is converged to the solution (4, 2, 2, 3) for inverse optimization problem (7) when $\varepsilon \to 0^+$.

In order to compare the performance of the method given in the paper and the penalty function algorithm, we use the same computer (CPU: AMD Athlon II M340 2.20 GHz, RAM: 512 MB) and the same MATLAB (2010a) to solve problem (8) by two methods to compare the calculated performance.

When we solve the solution by the given nonlinear neural network method, we take initial value $y_0 = (4.2, 1, 3, 2, 0, 0)$ $\lambda_0 = (0, 0, 0, 0, 0)$, $\mu_0 = 0$. $B = 0.01$. The optimization solution (4.0000, 2.0000, 2.0000, 3.0000) for problem (8) can be derived when ε take the value of 0.000. At the moment, the calculation time of computer is 1.938 s.

When we use penalty function method in literature [9] to inverse optimization problem (7), the optimization problem (7) is turned into the following form by the method in literature [9]:

$$\begin{aligned} \min\ & [\,(c_1x_1 + c_2x_2 - 14)^2 + k\ ((16 - x_1^2 - (x_2-2)^2)\\ & u_1 + (4 - (x_1-4)^2 - x_2^2)\,u_2 + v_1x_1 + v_2x_2)] \end{aligned}$$

$$\begin{aligned} s.t.\ & c_1^2 + c_2^2 - 8c_1 - 12c_2 + 39 \le 0 \\ & x_1^2 + (x_2-2)^2 \le 16 \\ & (x_1-4)^2 + x_2^2 \le 4 \\ & 2\mu_1x_1 + \mu_2(x_1-4) - v_1 - c_1 = 0 \\ & 2\mu_1(x_2-2) + 2\mu_2x_2 - v_2 - c_2 = 0 \\ & c, u, v, x \ge 0 \end{aligned} \quad (9)$$

According to the algorithm in literature [9], we take the initial value as (4.2, 1, 3, 2, 0, 0, 0, 0). When $k = 100$ and $i = 7$, the optimization solution of problem (9) is (4.0000, 2.0000, 2.0000, 3.0000). CPU operates for 2.1406 s. In conclusion, the method of neural network saves 1.0466 in solving the same problem through comparison of neural network method and penalty function method in literature [9].

Optimization solution $(x_1^*, x_2^*, c_1^*, c_2^*)$ corresponding to different ε values	
$\varepsilon = 0.01$	(3.9902, 1.9983,2.0037, 2.9975)
$\varepsilon = 0.001$	(3.9997, 1.9998, 2.0004, 2.9998)
$\varepsilon = 0.0001$	(4.0000, 2.0000, 2.0000, 3.0000)

Table 1. Comparison of optimization solution when ε take different values in example

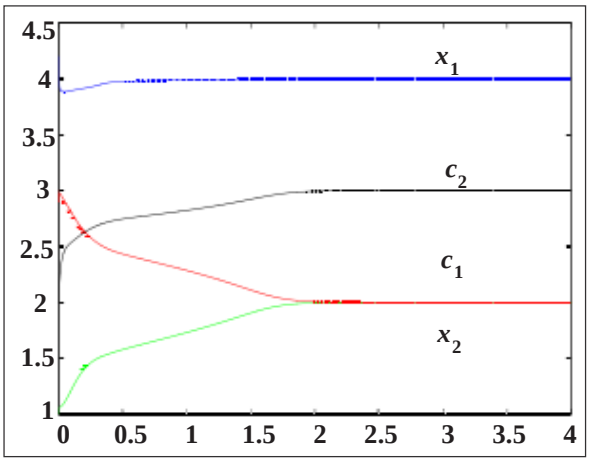

Figure 2.The state curve of neural network when $\varepsilon = 0.0001$, $y_0 = (4.2, 1, 3, 2, 0, 0)$, $\lambda_0 = (0, 0, 0, 0, 0)$, $\mu_0 = 0$, and $B = 0.01$.

4. Conclusion

In nonlinear optimization problem, convex optimization problem occupies a very important position because it has many good performances. Many necessary conditions in nonconvex optimization are also sufficient condition in convex optimization. Undoubtedly, that brings huge convenience for us. Solving optimization problem by convex optimization is just like to solve linear programming problem by least square method [11, 12]. If a problem can be abstracted to a convex optimization problem, then we can use effective method to solve it. Moreover, convex optimization also plays an important function in solving non-convex optimization problem. Most often, when facing

with a non-convex optimization problem that is hard to solve, turning non-convex optimization problem into convex optimization problem is a very effective solution [12]. In addition, many methods for solving nonconvex problem is centered as convex optimization method. In this paper, we constructed a class of nonlinear neural network model to solve the inverse optimization problem with nonlinear convex set constraints. First, we turned the inverse optimization problem with nonlinear convex set constraints into corresponding nonlinear bilevel programming problem. Then we constructed a class of neural network model to solve this bilevel programming problem. Afterwards, we proved that the balance point of corresponding neural network can be converged to the optimal point value of the inverse optimization problem. Through construction of Lyapunov function, we gave the sufficient condition for the global asymptotically stable balance point. This paper also applied numerical example to compare the neural network method and traditional numerical algorithm. Through comparison, we can know that the neural network method saved more timecompared to the penalty function in literature [9] under the same environment. All in all, the neural network method is more superior in the rate of convergence and complex degree of time.

References

[1] Bazaraa, M.S., Sherali, H. D., Shetty, C. M., (2013). Nonlinear Programming: Theory and Algorithms. *John Wiley & Sons*.

[2] Peng, Su. (2013). Research on Several Kinds of Nonlinear Optimization Problems Based on Neural Network. *Harbin: Harbin Institute of Technology.*

[3] Weidong, Ji . (2013). Research on Learning Method of Evolutionary Computation Optimizing Feed Forward Neural Network. *Harbin: Northeast Forestry University.*

[4] Haibo, Zhang. (2013). Research on Wireless Resources Optimization Strategy Based on Chaotic Neural Network. *Beijing:Beijing University of Posts and Telecommunicati-ons*.

[5] Liping, Chen. (2013). Stability and Synchronous Control of Fractional Order Linear System. *Chongqing: Chongqin University.*

[6] Jian, Sun., Chai, Yi., Huafeng, Li ., Zhiqin, Zhu. (2012). A Novel Stable Locally Recurrent Neural Network with Pole Assignment Projection Approach. *Acta Automatica Sinica*, 38 (2) 183-196.

[7] Lv, Y. B., Chen, Z., Wan, Z. P. (2010). A Penalty Function Method based on Bilevel Programming for Solving Inverse Optimal Value Problems. *Applied Mathematics Letters,* 23, p. 170-175.

[8] Shouming, Zhong., Bisen, Liu ., Wang, Xiaomei., Fan Xiaoming. (2008). Neural Network Stability Theory. *Science Press*, p. 39-52.

[9] Y.B. Lv, T.S. Hu, Z.P. Wan. (2008). A Penalty Function Method for Solving Inverse Optimal Value Problem. *Journal of Computational and Applied Mathematics*, 220, p. 175-180.

[10] Fontova, M., Oliveira, A., Lyra. C. (2012). Opfield Neural Networks in Large-scale Linear Optimization Problems. *Applied Mathematics and Computation*, p. 218: 6851-6859.

[11] Yongwei, Liu., Jianfeng, Hu ., Ping, Wang . (2014). A Bisection Method for Convex Programming Problem Solution based on Projection Neural Network. *Hunan Normal University* (JCR-SCI), 27 (1) 15-17.

[12] Liu, Q., Guo, Z., Wang, J. A. (2012). One-layer Recurrent Neural Network for Constrained Pseudoconvex Optimization and Its Application for Dynamic Portfolio Optimization. *Neural Networks*, 26, 99-109.