A Neural Network Approach for Nonlinear Bilevel Programming Problem

Yibing Lv[1] Tiesong Hu[1] Zhongping Wan[2]

[1]Institute of Systems Engineering, Wuhan University, Wuhan 430072, P.R.China
[2]School of Mathematics and Statistics, Wuhan University, Wuhan 430072, P.R.China

Abstract

A novel neural network approach is presented for solving nonlinear bilevel programming problem. The proposed neural network is proved to be Lyapunov stable and capable of generating optimal solution to the nonlinear bilevel programming problem. The asymptotic properties of the neural network are analyzed and the condition for asymptotic stability, solution feasibility and solution optimality are derived. The transient behavior of the neural network is simulated and the validity of the network is verified with numerical examples.

Keywords: Nonlinear bilevel programming, Neural network, Asymptotic stability, Optimal solution

1. Introduction

Bilevel programming (BLP) has increasingly been addressed in literature, both from the theoretical and computational points of view. This model has been wildly applied to decentralized planning problems involving a decision progress with a hierarchical structure. It is characterized by the existence of two optimization problems in which the constraint region of the first-level problem is implicitly determined by another optimization problem.

The BLP problem is hard to solve. In fact, the problem has been proved to be NP hard[1]. However the BLP problem is used so extensively in resource allocation, finance budget, price control et al[2]that many researchers have devoted to this field, which leads to a rapid development in theories and algorithms. For the detailed expositions, the reader may consult[3]-[6].

Since McCulloch and Pyne[7]-[8] utilized logical calculus to emulate nervous activities, there have been various types of analogue neural networks proposed for computation, we cite for example[9]-[12]. But generally speaking, there are two methods to solve optimization problem in terms of neural network approach. One method is to construct an appropriate computational energy function(Lyapunov function)so that the lowest energy state will correspond to the desired solutions, and then the derivation of the energy function enables us to transform the optimization problem into a set of ordinary differential equations on the basis of which we can design neural network. Another method is to construct a set of ordinary differential equations and then find an appropriate Lyapunov function such that all trajectory of the system converges to some equilibrium points which correspond to the desired solutions. The approach in [9,12] belongs to the first method, and the approach in [10]-[11] belongs to the second method.

It is noted that in modern science and technology, many optimization problems need to be solved in real time, while the classical methods can not render real-time solutions to these optimization problems, especially large-scale problems. The appearance of neural computing approach satisfies the demand of real-time optimal solutions and using neural network approach to solve the BLP problem is fairly new and there are few reports on the study of neural network approach for the BLP problems but Hsu-Shih Shih[13] and Kan-Ming Lan[14] recently presented their studying results on neural network approach for solving the linear BLP problem.

In this paper, for the nonlinear BLP problem, using the Kuhn-Tucker optimality condition, we reduce the nonlinear BLP problem to a regular nonlinear programming with complementary constraints. Then we propose a novel neural network approach, which belongs to the above second method, for the regular nonlinear programming problem with complementary constraints and get the approximate optimal solution of the nonlinear BLP problem. Towards these ends, the rest of the paper is organized as follows. In section 2, we will firstly introduce the smoothing method for the regular nonlinear programming with complementary constraints. Then in section 3, we propose a neural

network for solving the smoothed problem and derive the condition for asymptotic stability, solution feasibility and solution optimality. Numerical examples are given in section 4. Finally we conclude our paper.

2. Nonlinear BLP problem and smoothing method

Consider the following nonlinear BLP problem.

$$\begin{aligned} (UP) \quad & \min_x F(x,y) \\ & s.t. \;\; h(x) \le 0 \\ (LP) \quad & \min_y f(x,y) \\ & s.t. \;\; g(x,y) \le 0 \end{aligned} \tag{1}$$

where $x \in R^n, y \in R^m$, $F : R^{n\times m} \to R^1$, $f : R^{n\times m} \to R^1$ and $h : R^n \to R^p$, $g : R^{n\times m} \to R^q$ are continuous differentiable functions. The term (UP) is called the upper level problem and (LP) is called the lower level problem and correspondingly the terms x, y are the upper level variable and the lower level variable respectively.

Throughout the rest of the paper, we make the following assumptions:
(H_1) For fixed $x \in \{x : \exists y \in R^m, h(x) \le 0, g(x,y) \le 0\}$, the problem (LP) is a convex optimization problem satisfying $(MFCQ)$ at $y \in \{y : y \in R^m, g(x,y) \le 0\}$.
(H_2) The constraint region of the BLP problem $S = \{(x,y) : h(x) \le 0, g(x,y) \le 0\}$ is nonempty and compact.

If the assumption (H_1) is satisfied, then we can reduce the BLP problem to the one-level programming problem:

$$\begin{aligned} & \min_{x,y,\lambda} F(x,y) \\ & s.t. \;\; h(x) \le 0 \\ & \quad \nabla_y L(x,y,\lambda) = 0 \\ & \quad \lambda^T g(x,y) = 0 \\ & \quad g(x,y) \le 0 \\ & \quad \lambda \ge 0 \end{aligned} \tag{2}$$

where $L(x,y,\lambda)$ is the Lagrange function and $L(x,y,\lambda) = f(x,y) + \lambda^T g(x,y), \lambda \in R^q$.

Remark: If the lower level problem is not a convex parametric optimization problem, then the problem (2) has a larger feasible set including not only global optimal solutions of the lower level problem but also all local optimal solutions and also all stationary points. Then, to simply discussion, we assume that the lower level problem is a convex parametric optimization problem in this paper.

For the problem (2), the regularity assumptions which are needed for successfully handling smooth optimization problems are never satisfied and this situation is not good for using the neural network approach to solve the problem (2). But fortunately, Dempe[5] presents smoothing method for the BLP problem and the similar method is also presented in [16] for programming with complementary constraints. Following the smoothing method we can propose a neural network approach for the nonlinear BLP problem. Before introducing the smoothing method, we give some definitions firstly.

Definition 1 The $Fischer-Burmeister$ function is $\Phi : R^2 \to R$ defined by $\Phi(a,b) = a + b - \sqrt{a^2+b^2}$, and the perturbed $Fischer-Burmeister$ function is $\Phi : R^3 \to R$ defined by $\Phi(a,b,\varepsilon) = a + b - \sqrt{a^2+b^2+\varepsilon}$.

The $Fischer - Burmeister$ function has the property that $\Phi(a,b) = 0$ if and only if $a \ge 0, b \ge 0, ab = 0$, but it is non-differentiable at $a = b = 0$. Its perturbed variant satisfies $\Phi(a,b,\varepsilon) = 0$ if and only if $a > 0, b > 0, ab = \varepsilon/2$ for $\varepsilon > 0$. This function is smooth with respect to a, b for $\varepsilon > 0$.

In order to make the proposed neural network can also be applied to solve the linear BLP problem and satisfies the asymptotic stability conditions well, in this paper we adopt the following changed perturbed $Fischer - Burmeister$ function:

$$\Phi'(a,b,\varepsilon) = \sqrt{a^2+b^2+\varepsilon} - a - b$$

It is obvious that function $\Phi'(a,b,\varepsilon)$ has the same property with the function $\Phi(a,b,\varepsilon)$. Using the changed perturbed $Fischer-Burmeister$ function, the problem (2) can be approximated by

$$\begin{aligned} & \min_{x,y,\lambda} F(x,y) \\ & s.t. \;\; h(x) \le 0 \\ & \quad \nabla_y L(x,y,\lambda) = 0 \\ & \quad \sqrt{\lambda_j^2 + g_j^2(x,y) + \varepsilon} - \lambda_j + g_j(x,y) = 0 \\ & \qquad j = 1, \ldots, q \end{aligned} \tag{3}$$

Using the problem (3), we overcome the difficulty that the problem (2) dose not satisfy any regularity assumptions which are needed for successfully handling smooth optimization problems, and pave the way for using neural network approach to solve the problem (2). To simply our discussion, we introduce the following notations.

$$H(x,y,\lambda) = \begin{pmatrix} G(x,y,\lambda) = h(x), \\ \nabla_y L(x,y,\lambda) \\ \Phi'(\lambda_j, -g_j(x,y), \varepsilon)_{j=1,\ldots,q} \end{pmatrix}$$

Let $x^{'}=(x,y,\lambda)$, then the problem can be written as:

$$\begin{aligned}&\min_{x^{'}} F(x^{'})\\ &s.t.\ \ G_l(x^{'})\le 0,\ l=1,\dots,p\\ &\qquad H_k(x^{'})=0,\ k=1,\dots,m+q\end{aligned}\tag{4}$$

Definition 2 Let $x^{'}$ be a feasible point of the problem (4) and $L=\{l: G_l(x^{'})=0, l=1,\dots,p\}$. We say that $x^{'}$ is a regular point if the gradients $\nabla H_1(x^{'}),\dots,\nabla H_{m+q}(x^{'})$ and $\nabla G_l(x^{'}),\ l\in L$ are linearly independent.

Similar to the main result in [5](Theorem 6.11), we can have the following result.

Theorem 1 Let $\{(x^{'})^{\varepsilon}\}$ be a sequence of solutions of the problem (4). Suppose that the sequence $\{(x^{'})^{\varepsilon}\}$ converges to some $\bar{x}^{'}$ for $\varepsilon\to 0+$. If $\bar{x}^{'}$ is a regular point, then $\bar{x}^{'}$ is a Bouligand stationary solution for the BLP problem (1).

3. Neural network for BLP problem

The Lagrange function of the problem (4) can be defined by $L(x^{'},Y,\gamma,\mu)=F(x^{'})+\sum_{k=1}^{m+q}\gamma_k H_k(x^{'})+\sum_{l=1}^{p}\mu_l[G_l(x^{'})+Y_l^2]$ where $Y\in R^p$ is slack variable and $\gamma\in R^{m+q},\mu\in R^p$ are referred as the Lagrange multiplier.

Then, our aim now is to design a neural network that will settle down to an equilibrium, which is also a stationary point of the Lagrange function $L(x^{'},Y,\gamma,\mu)$. The transient behavior of the neural network can be defined by the following equations.

$$(NLBPNN)\begin{cases}dx^{'}/dt=-\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu)\\ dY/dt=-\nabla_Y L(x^{'},Y,\gamma,\mu)\\ d\gamma/dt=\nabla_\gamma L(x^{'},Y,\gamma,\mu)\\ d\mu/dt=\nabla_\mu L(x^{'},Y,\gamma,\mu)\end{cases}\tag{5}$$

Now we will study the relationship between the equilibrium of $(NLBPNN)$ and the approximate optimal solution of the problem (1) for $\varepsilon\to 0+$. We have the following theorem.

Theorem 2 Let $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ be the equilibrium of the neural network $(NLBPNN)$ for $\varepsilon\to 0+$ and assume that:

(i) $(x^{'})^*$ is a regular point of the problem (4).

(ii) For any $z\in Z=\{z:\nabla H_k((x^{'})^*)z=0,k=1,\dots,m+q;\nabla G_l((x^{'})^*)z=0,\forall l\in L\}$,

$z^T\nabla^2_{x^{'}x^{'}}L((x^{'})^*,Y^*,\gamma^*,\mu^*)z>0$.

Then the equilibrium of the neural network solves the problem (4), and also solves the BLP problem.

Proof. The proof of theorem 2 can be divided into two steps. Firstly, same to the proof of theorem 3 in [10], we can get that $\mu_l>0, l=1,\dots,p$ and the equilibrium of the neural network is the KT point of the problem (4); Secondly, following the sufficiency optimality conditions of second order for the problem (4), we can get that $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ solves the problem (4). Following theorem 1, we can finish this proof. □

While for the network to be of practical sense, $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ should furthermore to be of asymptotically stable, so that the network will always converge to $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ from an arbitrary initial point within the attraction domain of $((x^{'})^*,Y^*,\gamma^*,\mu^*)$. With this in mind we state and prove the following theorem, which in other words represents the local stability of the network.

Theorem 3 Let $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ be the equilibrium of the neural network $(NLBPNN)$ for $\varepsilon\to 0+$. Assume that the Hessian $\nabla^2_{x^{'}x^{'}}L((x^{'})^*,Y^*,\gamma^*,\mu^*)$ is positive definite and $(x^{'})^*$ is a regular point of the problem (4). Then $((x^{'})^*,Y^*,\gamma^*,\mu^*)$ is a asymptotically stable point of the neural network.

Proof. Let $E(x^{'},Y,\gamma,\mu)=\frac{1}{2}|\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu)|^2+\frac{1}{2}|\nabla_Y L(x^{'},Y,\gamma,\mu)|^2+\frac{1}{2}|\nabla_\gamma L(x^{'},Y,\gamma,\mu)|^2+\frac{1}{2}|\nabla_\mu L(x^{'},Y,\gamma,\mu)|^2$ denote the Lyapunov function of the network $(NLBPNN)$. Differentiating $E(x^{'},Y,\gamma,\mu)$ with respect to time t along the trajectory of the neural network gives:

$$\begin{aligned}&\tfrac{dE(x^{'},Y,\gamma,\mu)}{dt}=\tfrac{\partial E}{\partial x^{'}}.\tfrac{dx^{'}}{dt}+\tfrac{\partial E}{\partial Y}.\tfrac{dY}{dt}+\tfrac{\partial E}{\partial \gamma}.\tfrac{d\gamma}{dt}+\tfrac{\partial E}{\partial \mu}.\tfrac{d\mu}{dt}\\ &=[\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu).\nabla^2_{x^{'}x^{'}}L(x^{'},Y,\gamma,\mu)+\\ &\nabla_\gamma L(x^{'},Y,\gamma,\mu).\nabla^2_{\gamma x^{'}}L(x^{'},Y,\gamma,\mu)+\\ &\nabla_\mu L(x^{'},Y,\gamma,\mu).\nabla^2_{\mu x^{'}}L(x^{'},Y,\gamma,\mu)]\tfrac{dx^{'}}{dt}+\\ &[\nabla_Y L(x^{'},Y,\gamma,\mu).\nabla^2_{YY}L(x^{'},Y,\gamma,\mu)+\\ &\nabla_\mu L(x^{'},Y,\gamma,\mu).\nabla^2_{\mu Y}L(x^{'},Y,\gamma,\mu)]\tfrac{dY}{dt}+\\ &[\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu).\nabla^2_{x^{'}\gamma}L(x^{'},Y,\gamma,\mu)]\tfrac{d\gamma}{dt}+\\ &[\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu).\nabla^2_{x^{'}\mu}L(x^{'},Y,\gamma,\mu)+\\ &\nabla_Y L(x^{'},Y,\gamma,\mu).\nabla^2_{Y\mu}L(x^{'},Y,\gamma,\mu)]\tfrac{d\mu}{dt}=\\ &-\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu).\nabla^2_{x^{'}x^{'}}L(x^{'},Y,\gamma,\mu).\nabla_{x^{'}}L(x^{'},Y,\gamma,\mu)-\\ &\nabla_Y L(x^{'},Y,\gamma,\mu).\nabla^2_{YY}L(x^{'},Y,\gamma,\mu).\nabla_Y L(x^{'},Y,\gamma,\mu)\end{aligned}$$

As $\nabla^2_{YY}L((x^{'})^*,Y^*,\gamma^*,\mu^*)=Diag(2\mu_1^*,\dots,2\mu_p^*)$, following the proof of theorem 2, we have that $\mu_l>0, l=1,\dots,p$, and $\nabla^2_{YY}L((x^{'})^*,Y^*,\gamma^*,\mu^*)$ is positive definite. Then, we can deduce that $\frac{dE((x^{'})^*,Y^*,\gamma^*,\mu^*)}{dt}\le 0$. It means

that $((x^{'})^*, Y^*, \gamma^*, \mu^*)$ is a asymptotically stable point of the neural network. □

Remark: The above proposed neural network can also solve the linear BLP problem, moreover we adopt the perturbed $Fischer - Burmeister$ function $\Phi^{'}(a, b, \varepsilon) = \sqrt{a^2 + b^2 + \varepsilon} - a - b$, then the condition $\nabla^2_{x^{'}x^{'}} L((x^{'})^*, Y^*, \gamma^*, \mu^*)$ *is positive definite*, which is an assumption in theorem 2 and theorem 3, will be obvious for the linear BLP problem.

4. Computer simulations

In this section we will present two examples to illustrate the validity of the neural network approach for the nonlinear bilevel programming.

Example 1[15] Consider the following nonlinear BLP problem, where $x \in R^1, y \in R^1$.

$$\begin{aligned}
&\min_{x \geq 0} F(x, y) = x^2 + (y - 10)^2 \\
&s.t. \;\; 0 \leq x \leq 15 \\
&\min_{y \geq 0} f(x, y) = (x + 2y - 30)^2 \\
&s.t. \;\; x - y^2 \geq 0 \\
&\qquad 20 - x - y^2 \geq 0
\end{aligned}$$

After applying the Kuhn-Tucker transformation and the smoothing method, the above problem reduces to a problem similar to the problem (3). Then similar to the problem (5), we can get a set of ordinary differential equations, which describes the transient behavior of the neural network, and adopt the classical fourth-order Runge-Kutta method to solve these equations. We make program with Microsoft Visual C++ 6.0 and use a personal computer(CPU: Intel Pentium 1.7GHz, RAM: 256MB)to execute the program. Following theorem 1, we let ε have different small values and table 1 presents the different optimal solutions of Example 1 over the different ε. The initial condition is $(x, y, \lambda) = (1, 1, 0, 0, 0)$,$(Y, \gamma, \mu) = (1, 3, 0, 0, 0, 0, 0, 0)$, and figure 1 shows a typical transient process of x, y corresponding to the above initial point and $\varepsilon = 0.001$.

Example 2[4] Consider the following nonlinear BLP problem, $x \in R^1, y \in R^2$.

$$\begin{aligned}
&\min_{x \geq 0} (x - 1)^2 + 2y_1^2 - 2x \\
&s.t. \;\; \min_{y \geq 0} (2y_1 - 4)^2 + (2y_2 - 1)^2 + xy_1 \\
&\qquad s.t. \;\; 4x + 5y_1 + 4y_2 \leq 12 \\
&\qquad\qquad -4x - 5y_1 + 4y_2 \leq -4 \\
&\qquad\qquad 4x - 4y_1 + 5y_2 \leq 4 \\
&\qquad\qquad -4x + 4y_1 + 5y_2 \leq 4
\end{aligned}$$

For Example 2, following the same procedure of Example 1, we also study the different optimal solutions over the different ε, the result is also presented in table 1. Moreover figure 2 shows the transient behavior of x, y_1, y_2 corresponding to an initial condition $(x, y_1, y_2, \lambda_1, \ldots, \lambda_5) = (1, 1, 0, 0, 0, 0, 0, 0)$, $(Y, \gamma, \mu) = (0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ and $\varepsilon = 0.001$.

From table 1 and the two figures, we can find that the computed results converge to the optimal solution with the decreasing of ε. It shows that the neural network approach is feasible to the nonlinear BLP problem.

5. Conclusion

In this paper we present a novel neural network for the nonlinear BLP problem, and the numerical results show that the computed results converge to the optimal solution with the decreasing of ε, which corresponds to the result in theorem 1. In fact, a mass of additional numerical experiments show that we can get a satisfying approximate solution of the nonlinear BLP problems when $\varepsilon = 0.01$.

It deserves pointing out that the initial point of the neural network is the key factor of influencing the transient behavior of the proposed neural network. An appropriate initial point can get perfect transient behavior of the variables. The reason why such thing happens is that the neural network proposed only has asymptotic stability, then in order to get the optimal solution rapidly, we should choose the point, which satisfies the constraints (3) possibly, as the initial point. How to design neural network with global stability for BLP problems is still a challenge topic.

Acknowledgement

This work is supported by Natural Nature Science Foundation of China(Grant No. 50479039).

References

[1] O.Ben-Ayed, O.Blair. Computational difficulity of bilevel linear programming, *Operations Research.* 38:556–560,1990.

[2] W.F.Bialas, M.H.Karwan. On two-level optimization, *IEEE Transaction on Automatic Control.* 27(1):211–214,1982.

[3] E.S.Lee, H.S.Shih. Fuzzy and multi-level decision making: An interactive computational approach, Springer-Verlag, London, 2001.

examples in this paper	different optimal solution (x^*, y^*) corresponding to different ε			optimal solution (x^*, y^*) from corresponding references
	$\varepsilon = 0.01$	$\varepsilon = 0.001$	$\varepsilon = 0.0001$	
Exam.1	$(2.600, 1.613)$	$(2.600, 1.613)$	$(2.600, 1.612)$	$(2.600, 1.612)$
Exam.2	$(1.883, 0.891, 0.003)$	$(1.888, 0.889, 0)$	$(1.889, 0.889, 0)$	$(\frac{17}{9}, \frac{8}{9}, 0)$

Table 1: The comparison of the optimal solution.

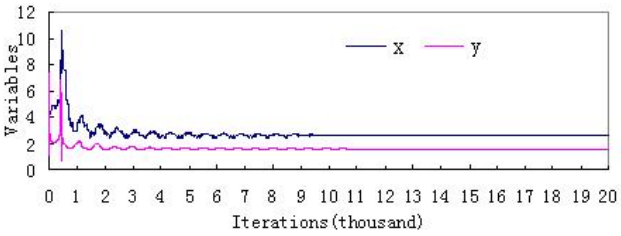

Figure 1: The behavior of the variables in Example 1.

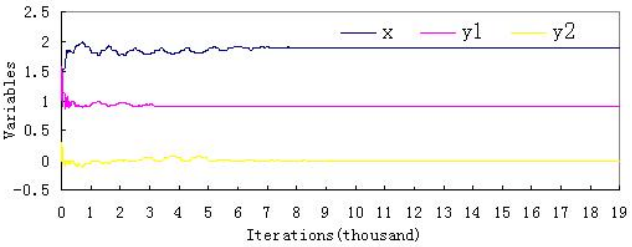

Figure 2: The behavior of the variables in Example 2.

[4] J.Bard. Practical Bilevel Optimization: Algorithm and Applications, Kluwer Academic Publishers, Dordrecht, 1998.

[5] S.Dempe. Foundation of bilevel programming, Kluwer Academic Publishers, London, 2002.

[6] Y.Lv, T.Hu, Z.Wan. A penalty functin method for solving weak price control problem[J]. *Applied Mathematics and Computation*, 186(2):1520–1525,2007.

[7] W.S.McCulloch and W.A.Pitts. A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematics and Biophysics*, 5:115–133,1943.

[8] I.B.Pyne. Linear programming on an electronic analog computer, *Transactions of the American Institute Electrical Engineers*, 75:139–143,1956.

[9] D.W.Tank, J.J.Hopfield. Simple ¡®neural¡¯ optimization networks: An A/D convert, siginal decision circuit, and a linear programming circuit, *IEEE Transactions on Circuits and Systems*, CAS-33:533–541,1986.

[10] S.Zhang, A.G.Constantinides. Lagrange programming neural networks, *IEEE Transactions on Circuits and Systems*, 39(7):441–452,1992.

[11] T.Hu, Y.Guo. Neural network for multiobjective dynamic programming, *ACTA Electronica Sinica*, 27(10):70–72,1999.

[12] M.P.Kennedy and L.O.Chua. Neural Network for nonlinear programming, *IEEE Transaction on Circuits and Systems*, 35:554–562,1998.

[13] H.S.Shih, U.P.Wen, E.S.Lee et al. A neural network approach to multi-objective and multilevel programming problems, *Computers and Mathematics with Applications*, 48:95–108,2004.

[14] K.M.Lan, U.P.Wen et al. A hybrid neural network approach to bilevel programming problems, *Applied Mathematics Letters*, 20(8):880–884,2007.

[15] W.Zhong, N.Xu. Boltzmann machine method for two-level decision making, *Journal of Systems Engineering*, 10(1):7–13,1995.

[16] F.Facchinei, H.Jiang, L.Qi. A smoothing method for mathematical programmings with equilibrium constraints, *Mathematical Programming*, 85:107–134,1999.