ORIGINAL RESEARCH

Solving linear bilevel programs via a new neural network

Yibing Lv*[1], Zhongping Wan[2]

[1]School of Information and Mathematics, Yangtze University, Jingzhou, P.R.China
[2]School of Mathematics and Statistics, Wuhan University, Wuhan, P.R.China

Received: July 22, 2015 **Accepted:** October 7, 2015 **Online Published:** October 22, 2015
DOI: 10.5430/air.v5n1p49 **URL:** http://dx.doi.org/10.5430/air.v5n1p49

Abstract

A new neural network model containing fewer neurons is proposed for the linear bilevel programs. The Lyapunov stability and convergence of the neural network are proved. To illustrate the feasibility of the neural network approach, we give the corresponding numerical experiments. The comparison results show that the new neural network approach is feasible and has lesser iterations to obtain the optimal solution.

Key Words: Linear bilevel programs, Neural network, Lyapunov stability, Equilibrium point, Optimal solution

1. Introduction

As a powerful implement to describe the hierarchical relationships in the real life, bilevel programming has attracted more and more attentions, and been used extensively in traffic network designing,[1] parameter calibrated,[2] supply chain,[3] and resource allocation,[4] *etc*. And on the other hand, the extensive applications of bilevel programming in the real life have been driving people to propose some effective algorithms to solve the bilevel programming problem. For more detail theories and algorithms, the reader may consult Refs. 5 and 6 and the references therein.

Linear bilevel programming problem (LBLP), where both the objective functions and the constraint functions are linear functions, has been studied extensively both from theories and solving approaches. In fact, more and more researchers have been devoting to the algorithm aspects. Roughly speaking, we can category these algorithms into four types. The first one is the so-called extreme point searching approach. The basic idea can be described as follows. Based on the fact that the optimal solution to the LBLP can be obtained in the vertex of the constraint region of the LBLP, through searching the vertexes and judging the feasibility, one can obtain the optimal solution of the LBLP. "Kth-best" algorithm[7] and grid-search algorithm[8] are two typical extreme point searching approaches. The second one is the transforming method. That's, based on the optimality conditions of the lower level programs, one can transform the LBLP into the usual one level programming problem, and some traditional optimization approaches can be used to obtain the optimal solutions. For more details on the transforming approach, the reader may consult Refs.9-11. The third one is the intelligent algorithms. In the last years, there have been various intelligent algorithms for the LBLP problem, such as Tabu search,[12] neural network approach.[13] The last category is interior point approach, such as primal-dual algorithm by Weng and Wen.[14]

*Correspondence: Yibing Lv; Email: lvyibing2001@gmail.com; Address: School of Information and Mathematics, Yangtze University, Jingzhou, 434023, P.R.China.

It has been verified that neural network has the capacity of fast converging to the equilibrium point of the status space, and obtaining the online solutions for some practical problems. Then, it has been a promising implement for some problems. In fact, neural network has been used extensively in solving equations, bilevel programs, variational inequalities, *et al.*[15–17]

Nowadays, there are some researchers attempt to solve the LBLP by neural network approach. For example, based on the method of transforming the LBLP into the mixed integer programming problem, Shih[17] proposes a neural network model for the LBLP. As the neural network model proposed contains a large enough penalty constant, it can cause the instability in simulation calculations. In order to overcome this drawback, Lan[12] combines the tabu search strategy to the above neural network. Based on the smoothing method for the complementary constraints, Hu[13] also proposes a neural network approach. However, in Ref.13, the authors introduce the slackness variables to transform the inequality constraints into the equality constraints, then it can increase the number of the neurons inevitably when proposing neural network model.

In this paper, inspired by the method presented in Ref.13, we will propose a new neural network model for the LBLP. Compared with the existing approach presented in Ref.13, the neural network approach proposed in this paper is different in the two aspects. The first one is that for the inequality constraints, we needn't introduce the slackness variables. Then, the neural network proposed here contains fewer neurons. The second one is that for the proof of the asymptotic stability, we adopt the concept of trajectory in ordinary differential equations. It can make the readers more directly understanding that with time changing the trajectory of the neural network can converge to the equilibrium point, which corresponds to the optimal solution of the LBLP.

In the next section, the LBLP will be transformed into the single level programs, meanwhile the smoothing function for the complementary constraints in the single level programs will be introduced. In Sect.3, based on the optimality conditions of the smoothed single level programs, the neural network model will be proposed. Then, the Lyapunov stability and convergence of the neural network model will be analyzed. In Sect.4, to illustrate the feasibility and effectiveness of the neural network model, we will solve some linear bilevel programming problems using our neural network model. Finally, we give some remarks to conclude the paper.

2. LBLP AND SMOOTHING APPROACH

The LBLP, which is considered in this paper, can be written as:

$$\begin{aligned} &\min_x \; c_1x + d_1y \\ &s.t. \quad A_1x + B_1y \le b_1, \\ &\qquad \min_y \; c_2x + d_2y \\ &\qquad s.t. \quad A_2x + B_2y \le b_2, \end{aligned} \tag{1}$$

Where $x \in R^n, y \in R^m; c_1, c_2 \in R^n; d_1, d_2 \in R^m; b_1 \in R^p; b_2 \in R^q; A_1 \in R^{p\times n}, B_1 \in R^{p\times m}; A_2 \in R^{q\times n}, B_2 \in R^{q\times m}$.

To well define the LBLP, in the rest of the paper, the following assumption is satisfied.

$(A1)$ For any $(x, y) \in R = \{(x, y)|A_1x + B_1y \le b_1, A_2x + B_2y \le b_2, x \ge 0, y \ge 0\}$, the vectors $B_2, i \in I = \{i|A_{2i}x + B_{2i}y = b_{2i}, i = 1, \cdots, q\}$ are linear independent, where the vector B_{2i} denotes the i-th row of B_2.

In problem (1), for the given upper level variable x, the lower level problem

$$\begin{aligned} &\min_y \; c_2x + d_2y \\ &s.t. \; A_2x + B_2y \le b_2, \end{aligned}$$

is a linear programming problem. Then, based on the above assumption , we can transform problem (1) into the following single level programs by the corresponding optimality conditions of the lower level programs.

$$\begin{aligned} &\min_{x,y,u,v} \; c_1x + d_2y \\ &\quad s.t. \; A_1x + B_1y \le b_1, \\ &\qquad A_2x + B_2y \le b_2, \\ &\qquad uB_2 - v = -d_2, \\ &\qquad u(b_2 - A_2x - B_2y) + vy = 0, \\ &\qquad x \ge 0, y \ge 0, u \ge 0, v \ge 0 \end{aligned} \tag{2}$$

Where $u \in R^q, v \in R^m$ are complementary variables.

Problem (2) is the non-smooth optimization problem as the existence of the complementary constraints, and proposing the neural network model for the non-smooth optimization problem is a quite tough task. To overcome this difficulty, we adopt the following smoothing method to deal with problem (2). Through transforming the complementary constraints equivalently, we can obtain the following programs, which is equivalent to the above programs (2).

$$\begin{aligned}
&\min_{x,y,u,v} c_1x+d_1y\\
&s.t. \quad A_1x+B_1y\le b_1,\\
&\qquad A_2x+B_2y\le b_2,\\
&\qquad uB_2-v=-d_2,\\
&\qquad -2\min(u_i,(b_2-A_2x-B_2y)_i)=0,\ i=1,\dots,q\\
&\qquad -2\min(v_j,y_j)=0,\ \ j=1,\dots,m\\
&\qquad x\ge 0, y\ge 0, u\ge 0, v\ge 0.
\end{aligned}$$

Let $\varepsilon \in R$ be a parameter. Define the function $\phi_\varepsilon : R^2 \to R$ by

$$\phi_\varepsilon(a,b)=\sqrt{(a-b)^2+4\varepsilon^2}-(a+b).$$

The following Proposition 2.1 gives the characters of the above function $\phi_\varepsilon(a,b)$.

Proposition 2.1.[6] For every $\varepsilon \in R$, we have

$$\phi_\varepsilon(a,b)=0 \Leftrightarrow a\ge 0, b\ge 0, ab=\varepsilon^2$$

It is obvious that for $\varepsilon = 0, \phi_\varepsilon(a,b) = -2\min(a,b)$. And for every $\varepsilon \neq 0, \phi_\varepsilon(a,b)$ is differential for every (a,b). Furthermore, for every $(a,b), \lim_{\varepsilon\to 0}\phi_\varepsilon(a,b) = -2\min(a,b)$.

Based on the above analysis, we can take the function $\phi_\varepsilon(a,b)$ as the approximation of the complementary conditions. That's, problem (2) can be approximated by

$$\begin{aligned}
&\min_{x,y,u,v} c_1x+d_1y\\
&s.t.\ A_1x+B_1y\le b_1,\\
&\quad uB_2-v=-d_2,\\
&\sqrt{[u_i-(b_2-A_2x-B_2y)_i]^2+4\varepsilon^2}-u_i-(b_2-A_2x-B_2y)_i=0,\ i=1,\dots,q\\
&\sqrt{(v_j-y_j)^2+4\varepsilon^2}-v_j-y_j=0,\ \ j=1,\dots,m\\
&\ x\ge 0.
\end{aligned} \tag{3}$$

Problem (3) is the usual smooth optimization problem. And based on the above transformation, it is convenient to propose the corresponding neural network model for problem (2). Before this, we give the following notations to simply the following presentations.

$$G(x,y,u,v)=\begin{pmatrix} A_1x+B_1y-b_1\\ -x\end{pmatrix}$$

$$H(x,y,u,v)=\begin{pmatrix} uB_2-v+d_2\\ \phi_\varepsilon(u_i,(b_2-A_2x-B_2y)_i), i=1,\dots,q\\ \phi_\varepsilon(v_j,y_j), j=1,\dots,m\end{pmatrix}$$

Let $x' = (x^T, y^T, u^T, v^T)^T$, based on the above notations, problem (3) can be rewritten as,

$$\begin{aligned}
&\min f(x')=c_1x+d_1y\\
&s.t.\ G_l(x')\le 0,\ l=1,\dots,n+p\\
&\qquad H_k(x')=0,\ k=1,\dots,2m+q.
\end{aligned} \tag{4}$$

Definition 2.1. Suppose that is feasible to the above programs (4), and denote

$$L=\{l: G_l(x')=0, l=1,\dots,n+p\}$$

If the gradient vectors $\nabla H_k(x'), \nabla G_l(x'), l \in L$ are linearly independent, then x' is called a regular point to the above programs (4).

Following some results (Theorem 6.11) on the smooth method for the bilevel programming problem in Ref.6, on the relationships between the optimal solutions to problem (2) and that to problem (4), we can also give the following theorem without proof.

Theorem 2.1. Suppose that $\{(x')^\varepsilon\}$ is a solution sequence to the above programs (4). If the sequence $\{(x')^\varepsilon\}$ converges to some $\tilde{x}$ for $\varepsilon \to 0$, meanwhile $\tilde{x}$ is also regular to the above programs (4). Then, $\tilde{x}$ is an optimal solution to the programs (2).

Following some results in optimization theory,[18] we have the following results on the optimal solutions of problem (4).

Suppose that there exist a point (x'^*, λ^*, μ^*) satisfying that for all $x' \in R^{2m+n+q}$, and all $(\lambda,\mu) \in R^{2m+n+p+q}$ with $\mu \ge 0$,

$$L(x'^*,\lambda,\mu)\le L(x'^*,\lambda^*,\mu^*)\le L(x',\lambda^*,\mu^*). \tag{5}$$

Then the point $(x')^*$ is an optimal solution to the above programs (4), where $L(x',\lambda,\mu)=f(x')+\mu^T G(x')+\lambda^T H(x')$ denotes the Lagrange function to the programs (4).

Proof. See Ref.18.

In fact, the above result is the so called saddle point theorem. Let x' be a regular point, and also feasible to problem (4). Following (5), there exist $(\lambda,\mu) \in R^{2m+n+p+q}$ with $\mu \ge 0$, such that the following K-T optimality conditions are satisfied at (x',λ,μ).

$$\begin{aligned}
&\nabla(x')+\nabla G(x')\mu+\nabla H(x')\lambda\mu=0,\\
&G(x')\le 0, H(x')=0,\\
&\mu^T G(x')=0, x'\ge 0, \mu\ge 0
\end{aligned} \tag{6}$$

3. Neural network model for problem (4)

Following the equality systems (6), the following so-called energy function can be constructed

$$\Phi(w)=\frac{1}{3}\sum_{l=1}^{n+p}[G_l^+(x')]^3+\frac{1}{2}\|\nabla f(x')+\nabla G(x')\mu+\nabla H(x')\lambda\|_2^2 +\frac{1}{2}(\mu^T G(x'))^2+\frac{1}{2}\|H(x')\|_2^2 \tag{7}$$

where $G_l^+ = \max\{0, G_l(x')\}$. It is obvious that the point $W^* = (x'^*, \lambda^*, \mu^*)$ is a minimization optimal solution of the function $\Phi(w)$ if and only if w^* satisfies the above system (6). That's, w^* is a K-T point to problem (4).

Then, we can construct the following neural network, whose transient behavior is depicted by the equations (8).

$$\text{(LBLPNN)}\quad \frac{dw}{dt}=-\nabla\Phi(w) \tag{8}$$

where $w \in \Omega = \{(x', \lambda, \mu) \in R^{4m+2n+p+2q} | x' \geq 0, \mu \geq 0\}$. In fact, the above equations depict the changing procession of the variable w with time t.

Remark 3.1. It is noted that for the inequality constraints, we don't introduce the slack variables, then compared with the existing neural network in Ref.13, our neural network model (8) contains fewer neurons.

Now we are the position of exploring that whether the equilibrium point of the LBLPNN is also an approximately optimal solution to problem (4) as $\varepsilon \to 0+$. On the above problem, we have the following result.

Theorem 3.1. Suppose that the point w^* is an equilibrium point of the network model LBLPNN. Meanwhile the part $(x')^*$ in w^* is regular to problem (4). Then, the point $(x')^*$ solves problem (4) locally.

Proof. Suppose that the point w^* is an equilibrium point of the network model (8), *i.e.*, $\bigtriangledown\Phi(w^*) = 0$. Since $\Phi(w)$ is twice continuously differentiable on Ω, there exists a ball $B(w^*;\delta)=\left\{w\in R^{4m+2n+p+2q} \mid \|w-w^*\|\le\delta\right\}$ with center w^* and radius δ such that $\Phi(w)$ is convex in the ball $B(w^*; \delta)$. Following the characters of the convex function, we have

$$\nabla\Phi(w^*)^T(w^*-w)\ge\Phi(w^*)-\Phi(w),\ \forall w\in B(w^*;\delta) \tag{9}$$

In (9), let $\bigtriangledown\Phi(w^*) = 0$, we can obtain that

$$\Phi(w^*)-\Phi(w)\le 0,\ \forall w\in B(w^*;\delta) \tag{10}$$

It means that the point w^* solves problem (4) locally.

Theorem 3.2. Suppose that the network model LBLPNN has only one equilibrium point, which is denoted by w^*, then w^* is uniformly and asymptotically stable.

Proof. Let $E(w) = \Phi(w) - \Phi(w^*) = Phi(w) \geq 0$, then $\forall w \neq w^*, E(w) > 0$. That's, $E(w)$ is a positive definitive function. Moreover, given an arbitrary initial point w^0, there exists a unique trajectory $w = w(t, w^0)$ of the system (8), along it there is

$$\begin{aligned}\frac{d}{dt}E(w)&=\frac{d}{dt}E(w(t,w^0))\\ &=\nabla E(w)^T\cdot\frac{dw}{dt}\\ &=-\|\nabla\Phi(w)\|^2<0.\end{aligned}$$

Following the Lyapunov Theorem, Theorem 3.2 is proved.

The following theorem shows that under some conditions, the trajectory of the LBLPNN (8) does converge to the equilibrium point.

Theorem 3.3. Suppose that the level set $L(w^0) = \{w : \Phi(w) \leq \Phi(w^0)\}$ is bound, then there exists an equilibrium point $\bar{w}$ and a strictly increasing sequence $\{t_n\}(t_n > 0)$, such that

$$\lim_{n\to+\infty} w(t_n,w^0)=\bar{w},\quad \nabla\Phi(\bar{w})=0.$$

Proof. Firstly, we will prove the following two results.

(a) The function $\Phi(w(t, w^0))$ is monotone non-increasing along the trajectory $w = w(t, w^0)$. In fact, following Theorem 3.2, the above result (a) is obvious.

(b) $W = \{w(t, w^0) : t \geq 0\}$ is a bound positive semi-trajectory.

Following the definition of the energy function $\Phi(w)$, it is obvious that is bound form below. In addition, $\Phi(w)$ is continuous, following the above result (a), the set $L(w^0)$ is bound and close, and

$$W=\left\{w(t,w^0):t\ge 0\right\}\subseteq L(w^0)$$

Hence, the result (b) is proved.

Now, we will prove $\lim_{n\to+\infty} w(t_n, w^0) = \bar{w}$.

Firstly, W is a bound points set. Take strictly increasing sequence

$$\{\bar{t}_n\}, 0 \le \bar{t}_1 < \bar{t}_2 < \cdots < \bar{t}_n \to +\infty,$$

$w(t_n, w^0)$ is a bound and infinite sequence. Then, the sequence $w(t_n, w^0)$ exist some limiting point $\bar{w}$. In other words, there exists a trajectory $w(t_n, w^0)$ with the rigidly increasing time sequence

$$\{t_n\} \subseteq \{\bar{t}_n\}, t_n \to +\infty$$

Satisfying

$$\lim_{n \to +\infty} w(t_n, w^0) = \bar{w}$$

Secondly, we prove that $\bigtriangledown\Phi(\bar{w}) = 0$.

By (a), $\Phi(w)$ is a Lyapunov function. $\Phi(\bar{w}) = 0 \Leftrightarrow \bigtriangledown\Phi(\bar{w}) = 0$. By LaSalle invariance principle,[19] $w(\bar{t_n}, w^0) \to W'(t \to +\infty)$, where the term W' denotes the largest invariant set in the set of equilibrium point. That means, there exists a time sequence $\{t_n\}(t_n \ge 0)$ such that

$$\lim_{n \to +\infty} w(t_n, w^0) = \bar{w}, \quad \nabla\Phi(\bar{w}) = 0.$$

The following theorem shows that the neural network (8) is locally and asymptotically stable.

Theorem 3.4. Suppose that the level set $L(W^0) = \{w : \Phi(w) \le \Phi(w^0)\}$ is bound, then the neural network model (8) is Lyapunov stable, and the limiting point of the trajectory of the neural network model (8) solves problem (4) locally.

Proof. Following Theorem 3.1, 3 .2 and 3.3, this theorem is obvious.

4. Computation simulations

In order to illustrate the feasibility and effectiveness of the new network model (8), and compare with the existing approach in Ref.13, we consider the following three LBLP, which are also considered in Ref.13.

Example 1[13] In this linear bilevel programming problem, the upper level variable $x \in R^1$ and the lower level variable $y \in R^1$.

$$\begin{aligned}
&\max_x \ -x + 4y \\
&s.t. \ \max_y -y \\
&\qquad s.t. \ x + y \ge 3, \\
&\qquad\qquad 2x - y \ge 0, \\
&\qquad\qquad 2x + y \le 12, \\
&\qquad\qquad 3x - 2y \le 4, \\
&\qquad\qquad x \ge 0, y \ge 0.
\end{aligned}$$

Example 2[13] In this linear bilevel programming problem, the upper level variable $x \in R^1$ and the lower level variable $y \in R^2$.

$$\begin{aligned}
&\max_x \ 4x + y_1 + y_2 \\
&s.t. \ \max_y \ x + 3y_1 \\
&\qquad s.t. \ x + y_1 + y_2 \le \frac{25}{9}, \\
&\qquad\qquad x + y_2 \le 2, \\
&\qquad\qquad y_1 + y_2 \le \frac{8}{9}, \\
&\qquad\qquad x \ge 0, y \ge 0.
\end{aligned}$$

Example 3[9] In this linear bilevel programming problem, the upper level variable $x \in R^2$ and the lower level variable $y \in R^3$.

$$\begin{aligned}
&\max_x \ 8x_1 + 4x_2 - 4y_1 + 40y_2 + 4y_3 \\
&s.t. \ \ x_1 + 2x_2 - y_3 \le 1.3, \\
&\qquad \max_y \ -2y_1 - y_2 - 2y_3 \\
&\qquad s.t. \ -y_1 + y_2 + y_3 \le 1, \\
&\qquad\qquad 4x_1 - 2y_1 + 4y_2 - y_3 \le 2, \\
&\qquad\qquad 4x_2 + 4y_1 - 2y_2 - y_3 \le 2, \\
&\qquad\qquad x \ge 0, y \ge 0.
\end{aligned}$$

For the above three linear bilevel programming problems, firstly we transform them into the usual single level programming problem, then adopt the smoothing function given in Proposition 2.1 to smooth the complementary conditions, and obtain the smoothed problem similar to problem (3). Following the above (4), (6) and (8), we can obtain the corresponding neural network model for the linear bilevel programming problem, that's, LBLPNN (8).

To solve the corresponding ordinary differential equations, *i.e.*, LBLPNN (8), we adopt the fourth order Runge-Kutta approach. We make C++ programs and employ a PC(Central Processing Unit: Intel Pentium 2.26 GHz, Random Access Memory: 1G) to execute the programs.

Following the result in Theorem 2.1, we can get the optimal solution to the LBLP (1) as $\varepsilon \to 0$. Then we consider taking the different parameters ε, and Table 1 presents the different optimal solutions of the three examples over different parameter ε. The initial point is that the variables x, y are identity and the rest is zero. What's more, Figures 1 and 2 show the changing procession of the variables in the first two examples with time corresponding to the parameter $\varepsilon = 0.001$.

Table 1. The equilibrium point to different ε

Example in this paper	Equilibrium point (x^*, y^*) corresponding over different ε			Optimal solution in the references
	ε= 0.01	ε= 0.001	ε= 0.0001	
Example 1	(3.99,4.00)	(4.00,4.00)	(4.00,4.00)	(4.00,4.00)
Example 2	(1.834,0.892,0.004)	(1.833,0.891,0)	(1.833,0.891,0)	(17/9,8/9,0)
Example 3	(0.499,0.798,0,0.197,0.798)	(0.499,0.8,0,0.198,0.8)	(0.499,0.8,0,0.198,0.8)	(0.5,0.8,0,0.2,0.8)

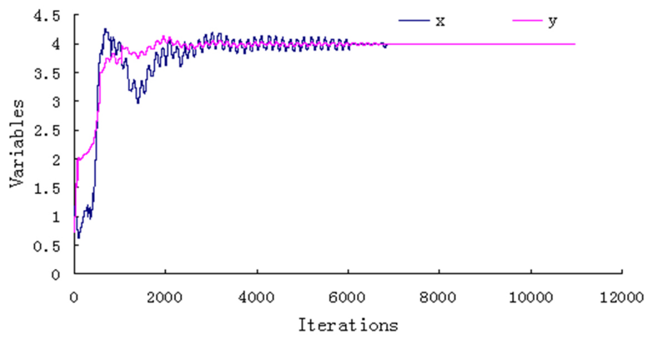

Figure 1. The changing procession of the variables in Example 1

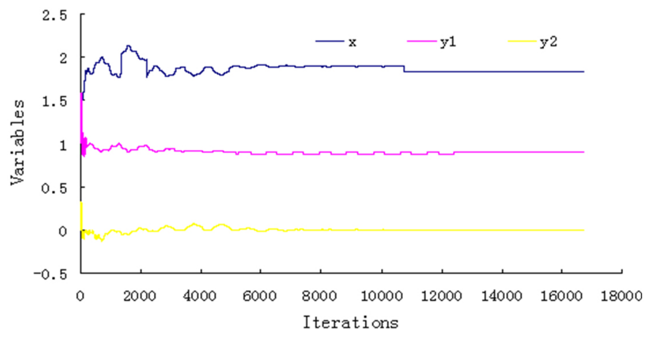

Figure 2. The changing procession of the variables in Example 2

To illustrate that the new neural network model is more efficient in the computation aspect than the neural network presented in Ref.13, we compare the iterations of the two kinds of neural network models in solving the same threes examples, the detailed comparisons are presented in Table 2.

Table 2. Simulation results in this paper comparing with that in Ref.[13] over $\varepsilon = 0.001$

Examples in this paper	Network model in this paper		Network model in Ref. [13]	
	Optimal solution	Iterations	Optimal solution	Iterations
Example 1	(4.00,4.00)	10895	(4.00,4.00)	11986
Example 2	(1.833,0.891,0)	16789	(1.833,0.891,0)	18878
Example 3	(0.499,0.8,0,0.198,0.8)	26467	(0.499,0.8,0,0.198,0.8)	30678

From the above two tables and two figures, we can draw the following conclusions. (1) The trajectory of the network model does converge to the equilibrium point, which is an optimal solution to the LBLP considered with the decreasing of the parameter ε. (2) Compared with the neural network presented in Ref.13, the new network model proposed here needs lesser iterations to get the optimal solutions because of containing fewer neurons.

5. Remarks

In this paper, we propose a new neural network model for the LBLP based on the smoothing method. The main contribution of the paper is that through adopting the max operator approach, we needn't introduce the slackness variables for the inequality constraints. That means that the new neural network model contains fewer neurons. The numerical result illustrate that the new network model has fewer iterations over the existing approach in Ref.13.

References

[1] Yang H, Bell MGH. Transport bilevel programming problems: recent methodological advances. Transportation Research. 2001; 35: 1-4. `http://dx.doi.org/10.1016/S0191-2615(00)00025-4`

[2] Lv Y, Wan Z. A penalty method based on BLP for solving inverse optimal value problem. Applied Mathematics Letters. 2010; 23: 170-5. `http://dx.doi.org/10.1016/j.aml.2009.09.007`

[3] Ma S, Yang W, Li L. Response-time optimization of supply chain with multi-stage based on bilevel programming. Journal of Management Sciences in China. 2004; 7(5): 18-22.

[4] Lv Y, Wan Z, Hu T. Bilevel model of water resources optimal allocation. Systems Engineering-Theory and Practice. 2009; 29(6): 115-20.

[5] Bard JF. Practical Bilevel Optimization: Algorithms and Application. Kluwer Academic Publisher: The Netherlands; 1998.

[6] Dempe S. Foundation of Bilevel Programming. Kluwer Academic Publishers: London; 2002.

[7] Bialas WF, Karwan MH. On two-level optimization. IEEE Transactions on Automatic Control. 1982; 27(1): 211-4. `http://dx.doi.org/10.1109/TAC.1982.1102880`

[8] Bard JF. A grid search algorithm for the linear bilevel programming problem. Proceeding of the 14th Annual Meeting of the American Institute for Decision Sciences. 1982; 2: 256-8.

[9] Hansen P, Jaumard B, Savard G. New branch-and-bound rules for linear bilevel programming. SIAM Journal on Science and Statistical Computing. 1992; 13: 1194-217. http://dx.doi.org/10.1137/0913069

[10] Anandalingam G, White JD. A solution for the linear static Stackelberg problem using penalty function. IEEE Transactions on Automatic Control. 1990; 35: 1170-3. http://dx.doi.org/10.1109/9.58565

[11] Lv Y, Hu T, Wang G. A penalty function method based on Kuhn-Tucker condition for solving linear bilevel programming. Applied Mathematics and Computation. 2007; 188: 808-13. http://dx.doi.org/10.1016/j.amc.2006.10.045

[12] Lan KM, Wen UP, Lee ES. A hybrid neural network approach to bilevel programming problems. Applied Mathematics Letters. 2007; 20(8): 880-4. http://dx.doi.org/10.1016/j.aml.2006.07.013

[13] Hu T, Guo X, Fu X, *et al*. A neural network approach for solving linear bilevel programming problem. Knowledge-Based Systems. 2010; 23: 239-42. http://dx.doi.org/10.1016/j.knosys.2010.01.001

[14] Wend WT, Wen UP. A primal-dual interior point algorithm for solving bilevel programming problems. Asia-Pacific Journal of Operational Research. 2000; 2: 213-31.

[15] Zhang S, Constantinides AG. Lagrange programming neural networks. IEEE Transaction on Circuits and Systems. 1992; 39(7): 441-52. http://dx.doi.org/10.1109/82.160169

[16] Sheng Z, Lv Q. A new algorithm based on the Frank-Wolfe method and neural network for a class of bilevel decision making problems. ACTA Automatica Sinica. 1996; 22(6): 657-65.

[17] Shih HS, Wen UP. A neural network approach to multi-objective and multilevel programming problems. Computers & Mathematics with Applications. 2004; 48: 95-108. http://dx.doi.org/10.1016/j.camwa.2003.12.003

[18] Luenberger DG. Linear and Nonlinear Programming, 2nded. Addison-Wesley: Massachusetts; 1984.

[19] LaSalle JP. The Stability of Dynamical Systems. Springer: New York; 1976.