KYBERNETIKA — VOLUME *56* (2020), NUMBER 3, PAGES 383–409

A GLOBALLY CONVERGENT NEURODYNAMICS OPTIMIZATION MODEL FOR MATHEMATICAL PROGRAMMING WITH EQUILIBRIUM CONSTRAINTS

Soraya Ezazipour and Ahmad Golbabai

This paper introduces a neurodynamics optimization model to compute the solution of mathematical programming with equilibrium constraints (MPEC). A smoothing method based on NPC-function is used to obtain a relaxed optimization problem. The optimal solution of the global optimization problem is estimated using a new neurodynamic system, which, in finite time, is convergent with its equilibrium point. Compared to existing models, the proposed model has a simple structure, with low complexity. The new dynamical system is investigated theoretically, and it is proved that the steady state of the proposed neural network is asymptotic stable and global convergence to the optimal solution of MPEC. Numerical simulations of several examples of MPEC are presented, all of which confirm the agreement between the theoretical and numerical aspects of the problem and show the effectiveness of the proposed model. Moreover, an application to resource allocation problem shows that the new method is a simple, but efficient, and practical algorithm for the solution of real-world MPEC problems.

Keywords: neural network, mathematical programming with equilibrium constraints, asymptotically stability, globally convergence

Classification: 90C33 , 90C26

1. INTRODUCTION

In this paper, a neurodynamics model for solving the mathematical program with equilibrium constraints (MPEC) is presented:

$$\begin{array}{lll} \text{(MPEC)} & \min & F(z) \\ & \text{s.t.} & G_1(z) \le 0, \quad G_2(z) = 0, \\ & & 0 \le H_1(z) \perp H_2(z) \ge 0. \end{array} \tag{1}$$

where $F : R^n \to R, H_1 : R^n \to R^m, H_2 : R^n \to R^m, G_1 : R^n \to R^p, G_2 : R^n \to R^q$. An equally asymmetric formulation with the formulation (1) is the optimization problem

DOI: 10.14736/kyb-2020-3-0383

with complementarity constraints (OPCC):

$$\begin{aligned} (\text{OPCC}) \quad \min \quad & f(x,y) \\ \text{s.t.} \quad & G_1(x,y) \leq 0, \quad G_2(x,y) = 0, \\ & 0 \leq H(x,y) \perp y \geq 0. \end{aligned} \tag{2}$$

That's a special case of optimization problem with variational inequality constraints (OPVIC):

$$\begin{aligned} (\text{OPVIC}) \quad \min \quad & f(x,y) \\ \text{s.t.} \quad & G_1(x,y) \leq 0, \quad G_2(x,y) = 0, \\ & y \in \Omega, \ \langle H(x,y), y - y^{'} \rangle \leq 0, \ \forall y^{'} \in \Omega, \end{aligned} \tag{3}$$

where $f : R^{n+m} \to R, H : R^{n+m} \to R^m, G_1 : R^{n+m} \to R^p, G_2 : R^{n+m} \to R^q$, and Ω is a closed convex subset of R^m. Sometimes, x, y are called the upper and lower level variables, respectively [17]. Apart of it, y is a solution of complementarity or variational inequality constraints. These problems can be used to model many real-world continuous or discrete events, such as multilevel games, shape optimization or transportation networks [6, 21, 35]. In discrete cases, MPEC can be more efficient than solving mixed-integer formulations of the optimization problems because it avoids the combinatorial difficulties of searching for optimal discrete variables.

The major difficulty in dealing with MPECs is that the typical constraint qualifications such as Mangasarian-Fromovitz constraint qualification (MFCQ) do not hold at any feasible point[26]. Because the feasible region of MPECs determines with the solution of complementarity or variational inequality problem, which is usually non-smooth and non-convex [36]. This encouraged many researchers to conduct in-depth research on MPEC resolution methods. Despite these efforts, few algorithms are capable of solving MPECs successfully [1]. The main methods that have been used to solve MPECs so far are the penalty, the implicit programming, and the piecewise programming approaches [18]. In the application of MPECs, we will encounter high dimension and dense structures [6]. These features have led researchers to seek new ways to address these issues. In recent years, due to the needs of engineers for real-time solutions, artificial neural networks (ANN) have been considered widely. The main purpose of using neural networks for solving various problems is to exploit their parallel processing nature and access to hardware implementations, which make it easier to deal with complex structures of considered problems[12, 40]. Following the interesting work of Hopfield and Tank, the theory, methodology, and application of neural networks have been expanded to solve optimization problems [13, 14, 15, 24, 30, 31, 32].There are also several ANN models to solve bilevel problems and MPECs. Sheng et. al., have suggested an ANN model for solving bilevel programming problem (BLP), using Frank-Wolfe method [38]. Lana et. al., have introduced a hybrid ANN for BLPs [22]. Li et. al. in [23] and He et. al. in [11], have proposed two different models for solving convex quadratic BLP problems based on Morrison method [33]. Using the Lagrange function, a neural network model to solve MPEC proposed by Lv et. al. [25]. A feedback neural network presented in [10] to solve MPECs.

In this paper, with the emphasis on the effectiveness and efficiency of the neural networks, we have tried to solve the MPECs using a novel neurodynamics model. Compared

Abbreviation	Full form
MPEC	Mathematical program with equilibrium constraints
OPCC	Optimization problem with complementarity constraints
OPVIC	Optimization problem with variational inequality constraints
BLP	Bilevel programming problem
ANN	Artificial neural network
KKT	Karush-Kuhn-Tucker conditions

Tab. 1. Summary of abbreviations.

to various iterative algorithms, the proposed network has a straightforward structure to implement and can be used to solve MPEC problems with each sort of stationary point. The smoothing technique in [5], is used to reformulate the non-smooth constraints i.e. complementarity or variational inequality. A neurodynamics optimization model which approximates the optimal solution of MPECs is presented. Unlike the existing networks such as [25], the support of the minimum number of variables is the superior feature of the suggested network. It is therefore particularly proper for large-scale problems and pretty accessible to implement.

The structure of the rest of this article is as follows. In Section 2, MPECs stationary points, and their properties are defined. In Section 3, the smoothing method, energy function, and the corresponding gradient-based network is explained. The theoretical aspects of the suggested model discussed in Section 4. Section 5 allocated to computational experiences on several academic examples and a resource allocation problem. The conclusion of this paper presented in Section 6. In addition Table 7 summarizes the abbreviations used in the remainder of this paper.

2. MPEC STATIONARY

Contrary to the classical nonlinear optimization problem, which has only one kind of KKT conditions, in scientific texts, several different types of KKT-form conditions for the MPEC have been used. The variation in terms of KKT-form conditions has created various kind of stationary points for MPECs. These points are suitable candidates for optimal solution. However, most available numerical algorithms designed to find a specific type of these stationary points. In this section, we first review the definitions of the most common stationary points. Then give some examples to reveal that checking all type of stationary points to gain the optimal point is vital for an algorithm; otherwise, it is highly probable that we lose the main optimal point. In the following the feasible region of the problem (1) is called S, and for a given point $z^* \in S$ the below index-sets is considered:

$$
\begin{aligned}
I(z^*) &:= \{i | G_1^i(z^*) = 0\},\\
J(z^*) &:= \{i | H_1^i(z^*) = 0, H_2^i(z^*) > 0\},\\
K(z^*) &:= \{i | H_1^i(z^*) = 0, H_2^i(z^*) = 0\},\\
F(z^*) &:= \{i | H_1^i(z^*) > 0, H_2^i(z^*) = 0\}.
\end{aligned}
$$

Definition 2.1. (Jane [17]) Let $z^* \in S$. Then z^* is said to be

(1) Weakly stationary (W-stationary) if there exists $\lambda = (\lambda^{G_1}, \lambda^{G_2}, \lambda^{H_1}, \lambda^{H_2}) \in R^m \times R^m \times R^p \times R^q$ such that satisfies

$$\begin{aligned} &0 = \nabla F(z^*) + \lambda^{G_1}\nabla G_1(z^*) + \lambda^{G_2}\nabla G_2(z^*) - \lambda^{H_1}\nabla H_1(z^*) - \lambda^{H_2}\nabla H_2(z^*) \\ &\min\{\lambda^{G_1}, -G_1(z^*)\} = 0, \\ &\lambda_i^{H_1} = 0,\ \forall i \in F \quad \lambda_i^{H_2} = 0,\ \forall i \in J. \end{aligned}$$

(2) Clarke stationary (C-stationary) if it is W-stationary and $\lambda_i^{H_1} \cdot \lambda_i^{H_2} \geq 0,\ \forall i \in K$.

(3) Mordukhovich stationary (M-stationary) if it is W-stationary and $\lambda_i^{H_1} \cdot \lambda_i^{H_2} = 0$ or $\lambda_i^{H_1} > 0\ \lambda_i^{H_2} > 0,\ \forall\, i \in K$.

(4) Strongly stationary (S-stationary) if it is W-stationary and $\lambda_i^{H_1} \geq 0\ \lambda_i^{H_2} \geq 0,\ \forall i \in K$.

(5) Alternatively stationary (A-stationary) if it is W-stationary and $\lambda_i^{H_1} \geq 0$ or $\lambda_i^{H_2} \geq 0,\ \forall\, i \in K$.

The relationship between these stationary concepts is summarized in Fig. 1. Numerous algorithms are proposed to solve these problems, each of which converges to a particular type of stationary point. Roger Fletcher et. al., in 2006 suggested a local convergent SQP method that converges to S-stationary [7]. Lei Guo et. al. (2014) has introduced three various smooth formulations for the C-/M-/S-stationary. Then, a Levengerg-Marquardt method for solving MPECs offered [9]. The main difficulty with this method is that in real problems we do not know at which type of stationeries, the optimal value will happen and therefore we will have to try all these smooth formulations. Christian Kanzow in [19] proposed a regularization method which converges to M-stationary. Here, there are a few examples that confirm the optimum point may occur in each of the stationary points. Accordingly, before-mentioned algorithms may be useless in practice.

Example 2.2. (Guo et al. [9]) $\min\ x_1 - 2x_2 \quad \text{s.t.} \quad x_1 - x_2 \geq 0, \quad 0 \leq x_1 \perp x_2 \geq 0.$
$x^* = (0, 0)$ is a W and M-stationary point for this problem. But it is not S-stationary.

Example 2.3. (Guo et al. [9]) $\min\ x_1 + x_2 - x_3 - \frac{1}{2}x_4$ s.t. $-6x_1 + x_3 + x_4 \leq 0$, $-6x_2 + x_3 \leq 0, \quad 0 \leq x_1 \perp x_2 \geq 0$. The optimal point $x^* = (0, 0, 0, 0)$, is a C-stationary point. However, it is not M-stationary point.

Example 2.4. (Guo et al. [9]) In the example below
$\min\ (x_1-1)^2+(x_2-\frac{1}{2})^2 \quad \text{s.t.} \quad x_1 \leq 0,\ x_2 \geq 0, \quad 0 \leq 2x_1+x_2 \perp 2-(x_1-1)^2-(x_2-1)^2 \geq 0.$
$(1, \sqrt{2}+1), (0, 0)$ and $(-\frac{2}{5}, \frac{4}{5})$ are three W-stationary points. Although the $(1, \sqrt{2}+1)$ is also S-stationary point, the $(0, 0)$ is an M-stationary point, but not S. Its global minimizer point is $(0, 0)$ and its global maximizer point is $(1, \sqrt{2}+1)$.

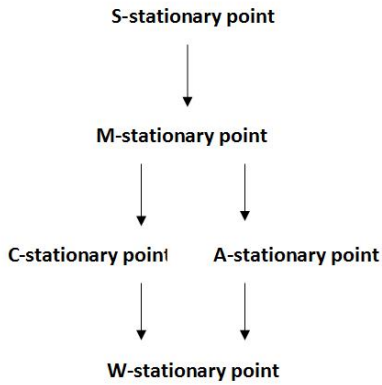


Fig. 1. The relationship between MPEC's stationeries ([17]).

Example 2.5. (Guo et al. [9]) $\min\ (x_1-1)^2+(x_2-\frac{1}{2})^2+\frac{1}{2}x_3(x_1-1)$ s.t. $x_1\leq 1$, $x_2+x_3(x_1-1)\geq 0$, $x_3^2\leq 0$, $0\leq 2x_1+x_2\perp 2-(x_1-1)^2-(x_2-1)^2\geq 0$. This problem has two W-stationary points: $(1,\sqrt{2}+1,0)$ and $(0,0,0)$. $(1,\sqrt{2}+1,0)$ is also S-stationary and its maximizer point. $(0,0,0)$ is its minimizer and also its C-stationary point. It is not M-stationary point.

Example 2.6. (Guo et al. [9]) $\min\ x_1+x_2-x_3$ s.t. $-4x_1+x_3\leq 0$, $-4x_2+x_3\leq 0$, $0\leq x_1\perp x_2\geq 0$. The only weakly stationary point of this problem is: $(0,0,0)$ which is also M-stationary point and its global minimizer. However, it is not S-stationary point.

Several researchers have tested examples as mentioned earlier to illustrate the effectiveness of their approach. In Section 5 and in Tables 3 – 5, we have outlined the results of [9], [16] and [41]. These results indicate that each of these methods failed to solve some of the above-mentioned examples. It motivates us to design an algorithm that converges to the optimal point, regardless of its stationary point type. In the following, we will show that the proposed model can reach the optimal point of all these examples in a finite time.

3. NEURODYNAMICS OPTIMIZATION MODEL

It is well known that the complementarity constraints of problem (1) violate the usual nonlinear constraint qualification especially, Mangasarian-Fromovitz constraint qualification (MFCQ) at each feasible point. This failure has several unpleasant consequences such as the linear dependence of the normal of active constraints and unbounded Lagrange multiplier set [26]. This is the greatest challenge in solving MPECs using conventional numerical methods and even new methods such as artificial neural networks. To get out of this situation, we reformulate the non-convex MPEC (1) as a non-smooth

equivalence problem.

$$\begin{array}{ll} \min & F(z) \\ \text{s.t.} & G_1(z) \leq 0, \\ & G_2(z) = 0, \\ & H_1(z) - v = 0, \\ & H_2(z) - u = 0, \\ & 2\min(v,u) = 0. \end{array} \tag{4}$$

Where $\min(v,u) = (\min(v_1,u_1),\ldots,\min(v_m,u_m))$. In the next lemma, we will clarify the necessity of the multiplicative factor 2 before the minimum function. Because of the presence of the minimum function, the reformulation (4) does not satisfy any regularity assumptions. Consequently, the usual methods of solving nonlinear optimization problems in the solution of this problem are inefficient. Using the smooth perturbation of minimum function, Problem (4) is transferred to a smooth reformulation. See also [5].

Definition 3.1. (Facchinei et al. [5]) The smooth perturbation of the minimum function is the function $\phi_\epsilon : R^2 \to R$,

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

Here, $\epsilon \in R$ is a scalar.

Lemma 3.2. (Facchinei et al. [5]) For any scalar $\epsilon \in R$,the function ϕ_ϵ possesses the properties:

(i) $\phi_\epsilon(a,b)$ is locally Lipschitz continuous and regular.

(ii) $\phi_\epsilon(a,b) = 0 \iff a \geq 0, \quad b \geq 0, \quad a \cdot b = \epsilon^2$.

(iii) For every $\epsilon \neq 0$, the function $\phi_\epsilon(a,b)$ is smooth for every $(a,b) \in R^2$.

(iv) $\lim_{\epsilon\to 0} \phi_\epsilon(a,b) = 2\min(a,b), \qquad \forall\ (a,b) \in \mathbb{R}^2$.

Define the nonlinear functions $G : \mathbb{R}^{n+2m} \to \mathbb{R}^p$ and $H : \mathbb{R}^{n+2m} \to \mathbb{R}^{2m+q}$,

$$\begin{aligned} G(w) &= G(z,u,v) = G_1(z), \\ H(w) &= H(z,u,v) = \begin{pmatrix} G_2(z) \\ H_1(z) - u \\ H_2(z) - v \end{pmatrix}. \end{aligned}$$

Thus, using the gap functions $\phi_\epsilon(u_i,v_i)$ and these functions, the Problem (4) is equivalent to the following problem:

$$\begin{array}{ll} \min & F(w) \\ \text{s.t.} & G(w) \leq 0, \\ & H(w) = 0, \\ & \phi_\epsilon^i(w) = 0,\ i = 1,\ldots,m. \end{array} \tag{5}$$

Definition 3.3. (Lv et al. [25]) Let w be a feasible point of Problem (5) and $K = \{j : G_j(w) = 0, j = 1, \ldots, p\}$. We say that w is a regular point if $\nabla H_1(w), \ldots, \nabla H_{2m+q}(w)$, $\nabla \phi_\epsilon^i(w),\ i = 1, \ldots, m$ and $\nabla G_j(w), j \in K$ are linearly independent.

The next theorem clarifies the relation between problem (1) and (5).

Theorem 3.4. [5] Let $\{\epsilon\}$ be a sequence of nonzero numbers with $\epsilon \to 0$ and w^ϵ be the global solution of Problem (5) corresponds to each ϵ. Then, the sequence $\{w^\epsilon\}$ contained in a compact set, and any limit point of it ,w^*, which is regular, will be a global solution of the MPEC problem (1).

To derive the new neurodynamics optimization model we will use the natural merit function of ϕ_ϵ.

Definition 3.5. (Chen [4]) The natural merit function of ϕ_ϵ is defined by

$$\begin{aligned} &\Phi_\epsilon : R^{2m} \to R, \\ &\Phi_\epsilon(X) = \tfrac{1}{2}\|\phi_\epsilon(X)\|^2 = \tfrac{1}{2}\textstyle\sum_{i=1}^m \phi_\epsilon(a_i, b_i)^2, \end{aligned}$$

where $X = (a_1, b_1, \ldots, a_m, b_m)$.

Lemma 3.6. Let $\psi_\epsilon(a, b) = \frac{1}{2}\phi_\epsilon(a, b)^2$, then

(i) $\psi_\epsilon(a, b) \geq 0\ \forall a, b \in R$ and $\psi_\epsilon(a, b) = 0 \iff a \geq 0,\ b \geq 0,\ a \cdot b = \epsilon^2$;

(ii) $\psi_\epsilon(a, b)$ is continuously differentiable for all $(a, b) \in R^2$, and the partial derivatives are

$$\begin{aligned} \nabla_a \psi_\epsilon(a, b) &= (1 - \tfrac{a-b}{\sqrt{(a-b)^2+4\epsilon^2}})\phi_\epsilon(a, b) \\ \nabla_b \psi_\epsilon(a, b) &= (1 - \tfrac{a-b}{\sqrt{(a-b)^2+4\epsilon^2}})\phi_\epsilon(a, b); \end{aligned}$$

(iii) the gradient of $\psi_\epsilon(a, b)$ is locally Lipschitz continuous i.e. $\exists L_1 > 0\ s.t.$

$$\|\nabla\psi_\epsilon(a_1, b_1) - \nabla\psi_\epsilon(a_2, b_2)\| \leq L_1\|(a_1, b_1) - (a_2, b_2)\|,$$

for all $(a_1, b_1), (a_2, b_2) \in R^2$.

P r o o f . The results (i) and (ii) are directly derived from Lemma 3.2. The proof for the (iii) is analogous to the proof of Proposition 3.5 in [8]. We ignore the details. □

Corollary 3.7. Let $\epsilon \in R$ be a fixed scalar, then $\Phi_\epsilon(X) \geq 0$ and $\Phi_\epsilon(X) = 0 \iff X \geq 0,\ a_i \cdot b_i = 0,\ i = 1, 2, \ldots, m$.

Corollary 3.8. The gradient of $\Phi_\epsilon(X)$ is locally Lipschitz continuous.

Remark 3.9. (Li et al. [23]) for $x \in R^n$, we have

$$x \leq 0 \iff \frac{1}{2}x^T(x + |x|) = 0.$$

Furthermore, $\frac{1}{2}x^T(x+|x|)=0$ is a nonnegative, continuously differentiable and convex function, where $|x|=(|x_1|,|x_2|,\ldots,|x_n|)$.

Remark 3.10. The smooth perturbation of the minimum function can be replaced by any NCP-function which has the same properties, such as Fischer–Burmeister function.

With regard to solving a constrained optimization problem with neural networks, the main method is to design an appropriate energy function so that the lowest energy state corresponds to the optimal solution. With Remark 3.9, we construct following energy function to approximate the problem (5).

$$E(w,\epsilon)=F(w)+\frac{\lambda}{2}\{\|H(w)\|^2+G(w)^T[G(w)+|G(w)|]+\Phi_\epsilon(w)\}, \tag{6}$$

where λ is a large positive scalar.

Lemma 3.11. For every fixed $\epsilon\in R$, $E(w,\epsilon)$ is a non-negative and continuously differentiable function.

We require the minimum value of $E(w,\epsilon)$. Hence, we will have the following unconstrained optimization problem:

$$\min_{w\in R^{n+2m}} E(w,\epsilon) \tag{7}$$

The well-known necessary optimality condition of (7) is $\nabla E(w,\epsilon)=0$, i.e.

$$\nabla_w F(w)+\lambda\{\nabla_w H(w)\cdot H(w)+\nabla_w G(w)^T[G(w)+|G(w)|]+\nabla_w\Phi_\epsilon(w)\}=0. \tag{8}$$

Using the equation (8), Lemma 3.11 and the steepest descent method, we describe an artificial neural network model for solving MPEC problem (1) by the following neurodynamics system:

$$\frac{dw}{dt}=-\nabla E(w,\epsilon), \tag{9}$$

where $w=[w_j]_{j=1,\ldots,n+2m}=[z_1,\ldots,z_n,u_1,\ldots,u_m,v_1,\ldots,v_m]$ and $\Phi_\epsilon(w)=\frac{1}{2}\sum_{i=1}^m\phi_\epsilon(u_i,v_i)^2$.

Let

$$S_i=\begin{cases}1 & \text{if } g_i(w)>0,\\ 0 & \text{if } g_i(w)\le 0.\end{cases}$$

Then, the neurodynamics system (9) can be rewritten as follows:

$$\frac{dw_j}{dt}=-\frac{\partial F(w)}{\partial w_j}-\lambda\left\{\sum_{i=1}^{P}2S_iG_i(w)\frac{\partial G_i(w)}{\partial w_i}+\sum_{i=1}^{2m+q}H_i(w)\frac{\partial H_i(w)}{\partial w_i}+\frac{\partial\Phi_\epsilon(w)}{\partial w_j}\right\}, \tag{10}$$

for $j=1,\ldots,n+2m$.

Figure 2, shows the practical structure for the simulations of the above equations. It can be seen that the recommended model (9) is a one-layer recurrent neural network with $n+2m$ massively connected neurons. Using the ANN model (9), the next algorithm is designed to obtain the optimal solution of the problem (1) with the proper accuracy.

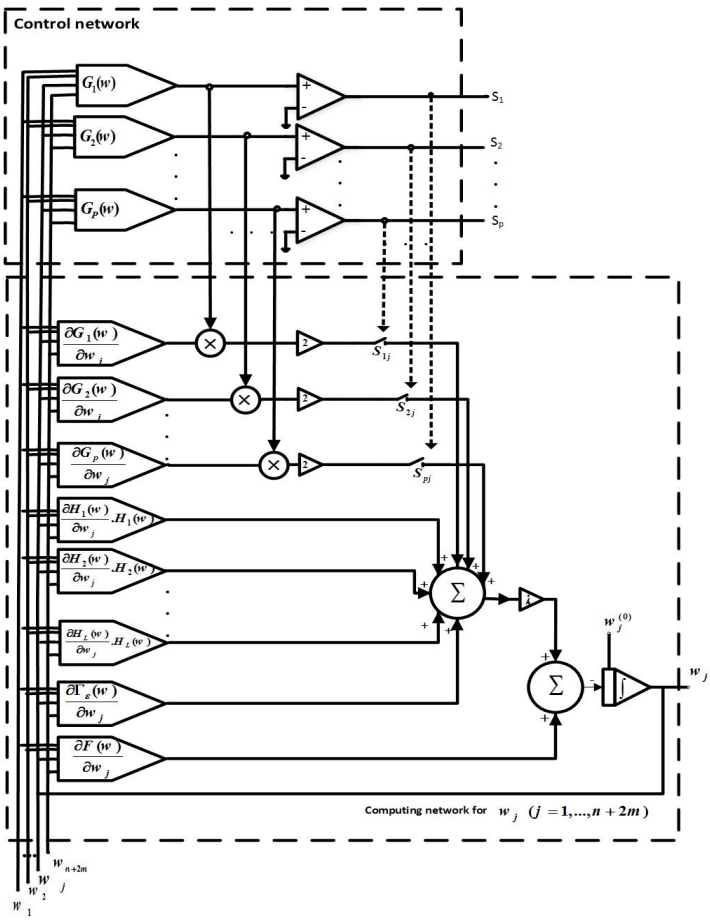


Fig. 2. The architecture of neural network model (9).

Algorithm A.

Step 0 Let $\{\epsilon^k\}$ be any sequence of nonzero numbers with $\lim_{k\to\infty} \epsilon^k = 0$ and $\{\lambda^k\}$ be any sequence of large positive numbers with $\lim_{k\to\infty} \lambda^k = +\infty$ and $\lambda^k \leq \lambda^{k+1}$: Choose $(w^0)^T = (z_0, u_0, q_0)^T \in R^{2m+n}$; and set $k := 1$.

Step 1 Find the stationary point w^k of Neurodynamics model (9)with λ^k and ϵ^k.

Step 2 Terminate, if the solution of (9) is exact enough, otherwise; set $k := k + 1$ and go Step 2.

The next theorem expresses the convergence behavior of the sequence $\{w^k\}$ generated by Algorithm A.

Theorem 3.12. Let $\{w^k\}$ be a sequence generated by Algorithm A. Then, any limit point of the sequence is a solution to (5).

P r o o f . The proof is quite similar to convergence theorem in [20], pp 404. We ignore the details. □

4. THEORETICAL ANALYSIS

In this section, the basic properties of the proposed model, such as its global convergence and stability, are considered.

Definition 4.1. (Liao et al. [27]) A local solution $x(t); t \in [t_0; \tau)$ is called maximal solution, if it cannot be extended to a local solution on a larger interval $[t_0, \tau_1)$; $\tau_1 > \tau$, and the interval $[t_0; \tau)$ is the maximal interval of existence.

Theorem 4.2. (Liao et al. [27]) Let $f : R^n \to R^n$ be a continuous mapping. If $x(t); t \in [t_0, \tau(x_0))$; be a maximal solution and $\tau(x_0) < +\infty$ then

$$\lim_{t \to \tau(x_0)} \|x(t)\| = +\infty.$$

Theorem 4.3. Let $\nabla F(z)$ be a locally Lipschitz continuous function on R^n then

(i) for every initial point $w(t_0) = w_0 \in R^{2m+n}$ there exists a unique maximal solution $w(t)$; $t \in [t_0, \tau(w_0))$ for the neural network model (9).

(ii) If the level set $L(w_0) = \{w \in R^{2m+n} | E(w, \epsilon) \leq E(w_0, \epsilon)\}$ be bounded, then $\tau(w_0) = +\infty$.

P r o o f .

(i) By assumption, $\nabla E(w, \epsilon)$ is locally Lipschitz, since the other terms in the right-hand side of (8) belong to C^1. Given the existence and uniqueness of ordinary equations [34], there is a unique continuous solution $w(t)$ $t \in [t_0, \tau(w_0))$ for the system (9).

(ii) By contradiction method, assume $\tau(w_0) < +\infty$, then Theorem 4.2 concludes

$$\lim_{t \to \tau(w_0)} \|w(t)\| = +\infty.$$

Let $\mu_0 = inf\{t \geq 0 | t < \tau(w_0), w(t) \in L^c(w_0)\} < +\infty$. Where $L^c(w_0)$ denote the complement of the set $L(w_0) \in R^{n+2m}$. Due to the continuity of $E(w, \epsilon)$, $L(w_0)$ is closed. It is also bounded by assumption. Hence, we have $w(\mu_0) \in L(w_0)$ and $\mu_0 < \tau(w_0)$. It concludes:

$$E(w(t), \epsilon) > E(w(\mu_0), \epsilon), \tag{11}$$

for some $t \in (\mu_0, \tau(w_0))$. However,

$$\frac{dE(w(t), \epsilon)}{dt} = \nabla E(w(t), \epsilon)^T \cdot \frac{dw}{dt} = -\nabla E(w(t), \epsilon)^2 \leq 0.$$

i.e., $E(w, \epsilon)$ is nonincreasing on $[t_0, \tau(w_0))$. This is contradictory with (11).Thus $\tau(w_0) = +\infty$. □

Theorem 4.4. If the neurodynamic model (9) possesses a stable isolated equilibrium point,$\bar{w}$, then it is a local minimizer of the function $E(w, \epsilon)$.

P r o o f . The proof is by contradiction. Consider the simple function $\mathfrak{S}(w) = E(\bar{w}, \epsilon) - E(w, \epsilon)$. Note that $\mathfrak{S}(\bar{w}) = 0$, and $\mathfrak{S}(w)$ is a continuously diffrentiable function. Suppose that the equilibrium point $\bar{w}$ is not a local minimizer of $E(w, \epsilon)$. Obviously, in each neighborhood $\mathfrak{N}$ of $\bar{w}$ there exists $\hat{w} \in \mathfrak{N}$ such that $E(\hat{w}, \epsilon) < E(\bar{w}, \epsilon)$. Consequently, $\mathfrak{S}(\hat{w}) > 0$. Due to the fact that $\bar{w}$ is an isolated equilibrium of point (9), we have

$$\frac{d\mathfrak{S}(w)}{dt} = -\nabla E(w, \epsilon)^T \cdot \frac{dw}{dt} = \|\nabla E(w, \epsilon)\|^2 > 0.$$

Thus, there exists a neighborhood $\mathfrak{R}$ of $\bar{w}$ so that if $w \in \mathfrak{R}, w \neq \bar{w}$, then $\nabla E(w, \epsilon) \neq 0$. The Instability Theorem [28] yields the desirable result □

Corollary 4.5. If the isolated equilibrium point of (9) be a local maximizer of $E(w, \epsilon)$, then it is unstable.

Theorem 4.6. Suppose that $w(t, w_0) \in \Omega$ is the trajectory of proposed neurodynamic model (9) in which the initial point is $w_0 \in \Omega$ and $E(w, \epsilon)$ is convex in Ω. Then,

(i) $\mathfrak{D}^+ = \{w(t, w_0) | t > 0\}$ is bounded;

(ii) the neural network (9) has an equilibrium point $\bar{w}$, so that

$$\lim_{t \to +\infty} w(t, w_0) = \bar{w};$$

(iii) the neural network is Lyapunov stable with respect to $\bar{w}$. Moreover, if $E(w, \epsilon)$ has a unique minimum then, this network is asymptotically stable.

P r o o f .

(i) Let $w^* \in \Omega$ be an equilibrium point of (9). Define the function $\mathfrak{H}(w)$ as follows

$$\mathfrak{H}(w) = \frac{1}{2}\|w - w^*\|^2.$$

Using the first order optimization condition, and the convexity of $E(w, \epsilon)$ for $w(t) \in \Omega$, we have

$$E(w^*, \epsilon) - E(w, \epsilon) \geq (w^* - w)^T \nabla E(w, \epsilon).$$

Thus

$$\frac{d\mathfrak{H}(w)}{dt} = (w - w^*)^T \frac{dw}{dt} = (w - w^*)^T(-\nabla E(w, \epsilon)) =$$

$$(w^* - w)^T \nabla E(w, \epsilon) \leq E(w^*, \epsilon) - E(w, \epsilon) \leq 0.$$

Then, $\|w(t) - w^*\|^2 = 2\mathfrak{H}(w(t)) \leq 2\mathfrak{H}(w_0)$.

(ii) $\mathfrak{H}(w)$ is a coercive function, i.e., $\mathfrak{H}(w) \to +\infty$, as $w \to \infty$. So, $L_{\mathfrak{H}}(w_0) = \{w \in \mathbb{R}^{n+m+2l} : \mathfrak{H}(w) \leq \mathfrak{H}(w_0)\}$ is bounded. By Lasalle invariance set theory each trajectory of (9) will converge to the maximum invariant subset $\mathfrak{N}$ of $\{w \in L_{\mathfrak{H}}(w_0) : \frac{d}{dt}\mathfrak{H}(w) = 0\}$.

If $w(t) \in \mathfrak{N}$, i.e., $\frac{d}{dt}\mathfrak{H}(w) = 0$, then

$$\frac{d\mathfrak{H}(w)}{dt} = 0 \leq E(w^*, \epsilon) - E(w, \epsilon) \leq 0,$$

which means w is also an optimal solution of $E(w, \epsilon)$ and $\frac{dw}{dt} = -\nabla E(w, \epsilon) = 0$. On the other hand, if $\frac{dw}{dt} = 0$, then $\frac{d\mathfrak{H}(w)}{dt} = (w - w^*)^T \frac{dw}{dt} = 0$.

Thus,

$$\mathfrak{N} \subseteq \{w \in L_{\mathfrak{H}}(w_0) : \frac{d}{dt}\mathfrak{H}(w) = 0\} = \{w \in L_{\mathfrak{H}}(w_0) : \frac{dw}{dt} = 0\}.$$

Concludes that there exist a convergence subsequence $\{w(t_k, w_0)\}$ such that

$$\lim_{k \to +\infty} w(t_k, w_0) = \bar{w}.$$

It is clear that $\nabla E(\bar{w}, \epsilon) = 0$. Finally, Define the following Lyapunov function

$$\hat{\mathfrak{H}}(w) = \frac{1}{2}\|w - \bar{w}\|^2.$$

Similar $\mathfrak{H}(w)$, it can be seen easily that $\hat{\mathfrak{H}}(w)$ is a decreasing function along the trajectory of (9). Hence, for each $\hat{\epsilon} > 0$, $\exists q > 0$ such that for $t > t_q$

$$\frac{1}{2}\|w(t) - \bar{w}\| = \hat{\mathfrak{H}}(w(t)) < \hat{\mathfrak{H}}(w(t_q)) < \hat{\epsilon}.$$

So, $\lim_{t \to +\infty} w(t) = \bar{w}$.

(iii) $\hat{\mathfrak{H}}(w)$ is a Lyapunov function over $\bar{w}$ for proposed model (9). Thus the network is Lyapunov stable. besides, if $E(w, \epsilon)$ has a unique minimum then, $\frac{d\hat{\mathfrak{H}}(w)}{dt} < 0$; $w \neq \bar{w}$ i.e. the network is asymptotically stable.

□

5. ILLUSTRATIVE EXAMPLES

In this section, several different examples are presented to illustrate the performance of the proposed network. We also compare the performance of the neural network (9) with the methods presented in [9, 16, 41] in Examples 2.2 – 2.6. The computational results indicate that the neurodynamics model (9) produces proper approximate solutions of (1) in a small number of iterations.

Example 5.1. We first tested the method on the Examples 2.2 – 2.6 to compare our method with the methods in [16, 41] and [9]. The neurodynamics model (9) with a different set of initial points, $\epsilon = 10^{-6}$ and $\lambda = 10^5$ is applied to solve problems mentioned

above. Figure 3, shows the convergence behavior of the proposed neural network, starting from random initial points. For all above examples and despite the methods presented in [9, 16, 41], the proposed model successfully converges to optimal points regardless to their stationary type. Table 2, summarizes the results obtained by the proposed neural network and confirms its convergence with the appropriate speed and accuracy in Examples 2.2 – 2.6.

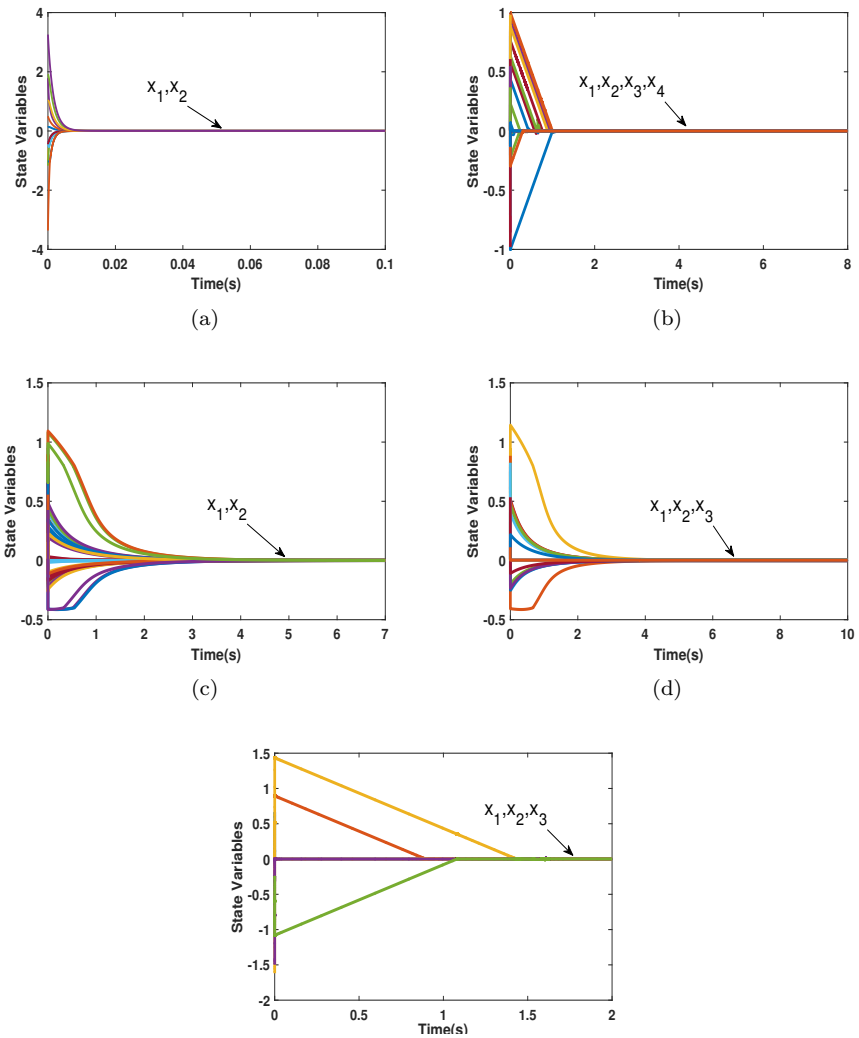


Fig. 3. Convergence behavior of the proposed neural network with random starting points in: 3(a) Example 2.2., 3(b) Example 2.3., 3(c) Example 2.4., 3(d) Example 2.5., 3(e) Example 2.6.

Example	x^k	$f(x^k)$	$G(x^k)^T \cdot H(x^k)$	CPU time(s)
2.2	(2.9515e-04, 3.9515e-04)	-4.9515e-04	1.1663e-07	0.2944
2.3	(3.6538e-04,2.7476e-05,1.6489e-04,0.0020)	-7.8574e-04	1.0039e-08	0.4393
2.4	(2.3751e-04,-4.9999e-05)	1.2496	1.5937e-07	0.2787
2.5	(0.0010,-0.0010, -0.0010)	1.2494	1.2759e-08	0.3592
2.6	(3.1627e-04,3.1627e-04,0.0012)	-6.3258e-04	1.0003e-07	0.4713

Tab. 2. Simulation results for Examples 2.2 – 2.6, using proposed model (9) with one iteration.

In Tables 3 – 5, the results acquired by modified Levenberg–Marquardt method in [9], $l_{\frac{1}{2}}$ penalty method in [41] and partial augmented Lagrangian method in [16], are reported directly. Table 3, shows that the method presented in [9] is incapable of solving Examples 2.3 and 2.5. Using Table 4, the method in [16] failed to solve Examples 2.2, 2.4, 2.6. However, it has the lowest CPU time among these methods. Results in Table 5, show that the method in [41] failed to solve Example 2.3. A comparison of above tested methods in terms of absolute objective function error is presented in Table 7.

M, S and C systems and function $F(w)$ are defined in [9].

Example	systems	Iter	x^k	$\|F(w^k)\|$	CPU time(s)
2.2	M	16	$(0, 0.0000)$	$2.0683e-07$	0.4305
2.3	C	78	(0.0429,0,-0.0000,0)	0.4204	3.5504
2.4	C	11	(0.0000,0)	3.6890e-10	0.3430
	M	15	(-0.0076,0.0157)	0.0013	1.3910
	S	100	(-0.4594,0.7073)	1.1766	3.1200
2.5	C	22	(0.0720,-0.0671,-0.0721)	0.0052	1.0187
	M	18	(-0.0768,0.1526,0.1188)	0.0166	1.1543
	S	100	(0.3736,2.2679,0.0067)	13.4476	11.0329
2.6	M	12	(0.0000,0.0000,0.000)	$8.9677e-08$	0.4540

Tab. 3. Simulation results for Examples 2.2 – 2.6, using the modified Levenberg–Marquardt method in [9].

Example	Iter	x^k	$f(x^k)$	$G(x^k)^T \cdot H(x^k)$	CPU time(s)
2.2	100	$(0.7937, 0.7937)$	-0.7937	0.6300	$3.7640e-04$
2.3	1	(0.0001,0.0015,-0.0016,-0.0000)	0.0032	$1.8072e-07$	$1.2074e-06$
2.4	100	(0.4061,0.0000)	0.4061	0.5257	0.0015
2.5	1	(-0.0000,0.0000,0.0000)	1.2502	$4.0747e-19$	$1.5092e-06$
2.6	100	(1.0000,1.0000,4.000)	-2.0000	1.0000	0.0010

Tab. 4. Simulation results for Examples 2.2 – 2.6, using the partial augmented Lagrangian method in [16].

Ex.	Iter	x^k	$f(x^k)$	$min(G(x^k), H(x^k))$	CPU time(s)
2.2	16	$(-0.0000, -0.0000)$	$1.9158e-07$	-1.8035e-07	0.2974
2.3	100	1.0e+19*(-4.7764,-4.7359,2.9522,-0.1574)	-1.2387e+20	-4.7764e+19	1.1181
2.4	18	(0.0000,0.0000)	1.2500	5.4404e-11	0.3184
2.5	24	(-0.0003,0.0008,0.0003)	1.2502	4.4384e-07	0.4752
2.6	13	(0.0000,0.0000,0.0000)	-2.3305e-05	1.3595e-10	0.1985

Tab. 5. Simulation results for Examples 2.2 – 2.6, using the $l_{\frac{1}{2}}$ penalty method in [41].

Objective function	Numerical optimal value $f(x^*, y^*)$	Numerical optimal solution x_1^*	x_2^*	y^*	CPU time (s)
$f_1(x, y)$	10.4925	2.7101	0.5365	0.0	0.730450
$f_2(x, y)$	2.00	2.5	0.0	0.0	0.536138

Tab. 6. Numerical results of proposed model (9) for Example 5.2. with random initial point.

Example	Proposed ANN	Modified Levenberg–Marquardt [9]	Partial augmented $\frac{1}{2}$ Lagrangian [16]	Penalty[41]
2.2	4.9515e-04	0.0000	0.7937	1.9158e-07
2.3	7.8574e-04	0/04929	0.0032	1.2387e+20
2.4	6.0e-04	0.0000 (C system) 4.9579e-04 (M system) 0.9228 (S system)	0.8439	0.0000
2.5	4.0e-04	0.0338 (C system) 0.0338 (M system) 2.2657 (S system)	0.0002	0.0002
2.6	6.3258e0-4	0.0000 (M system)	2	3.3305e-05

Tab. 7. Comparison of above tested methods in terms of absolute objective function error in Examples 2.2 – 2.6

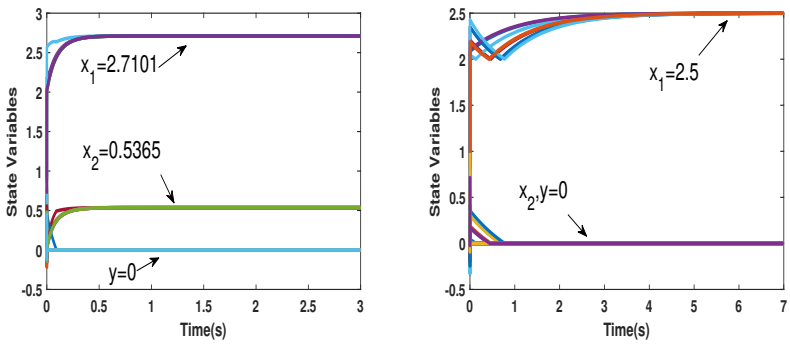
Example 5.2. Next we tested the method on the problem:

$$\begin{aligned} \min \quad & f(x, y) \\ \text{s.t.} \quad & \min\{y; x_1 - e^{x_2} - e^y\} = 0, \\ & x_2 \geq 0. \end{aligned}$$

for various quadratic objective functions $f(x, y)$,

$$f_1(x, y) = x_1^2 + 10(x_2 - 1)^2 + (y + 1)^2,$$
$$f_2(x, y) = (x_1 - 2.5)^2 + (x_2 + 1)^2 + (y + 1)^2.$$

This problem has been given in [39], and was investigated in [29]. The neurodynamics model (9) is applied to solve the above example. Let $\epsilon = 10^{-6}, \lambda = 17 \times 10^3$. Choosing six random initial points, the problem was solved with MATLAB software and the results show in Table 6. The method detected the optimal solutions reported in [39] and the numerical results indicate global convergence in all cases. The solution trajectory with (x, y) versus the run time, is shown in Figure 4, which converge to the optimal point.



(a) (b)

Fig. 4. 4(a) The transient conduct of the neural system model (9) with six various initial points and $f_1(x, y)$ as its objective function in Example 5.2.; 4(b) the output trajectory of (9) with nine various initial points and $f_2(x, y)$ as its objective function in Example 5.2.

Example 5.3. Outrata in [36] and Facchinei in [8], use the following example to illustrate an implicit function-based bundle trust region and a homotopy method for MPECs, respectively. The MPEC is of the form:

$$\begin{aligned}
\min \quad & f(x, y) \\
\text{s.t.} \quad & \min\{x_1, (1 + 0.2y)x_1 - 3 + 1.333y - 0.333x_3 + 2x_1x_4\} = 0, \\
& \min\{x_2, (1 + 0.1y)x_2 - y + x_3 + 2x_2x_4\} = 0, \\
& \min\{x_3, 0.333x_1 - x_2 + 1 - 0.1y\} = 0, \\
& \min\{x_4, 9 + 0.1y - x_1^2 - x_2^2\} = 0, \\
& 0 \le y \le 10,
\end{aligned}$$

with the objective functions

$$
\begin{aligned}
f_1(x,y) &= \frac{1}{2}(x_1-3)^2+\frac{1}{2}(x_2-4)^2\\
f_2(x,y) &= \frac{1}{2}(x_1-3)^2+\frac{1}{2}(x_2-4)^2+\frac{1}{2}(x_3-1)^2\\
f_3(x,y) &= \frac{1}{2}(x_1-3)^2+\frac{1}{2}(x_2-4)^2+5x_4^2\\
f_4(x,y) &= \frac{1}{2}(x_1-3)^2+\frac{1}{2}(x_2-4)^2+\frac{1}{2}(x_3-1)^2+\\
&\quad \frac{1}{2}(x_4-1)^2+\frac{1}{2}y^2.
\end{aligned}
$$

These problems differ in the objective $f(x,y)$. The proposed neural network model is used to obtain optimal solutions with $\lambda = 7\times 10^3$ and $\epsilon = 10^{-6}$. We tested at least five random initial points for each problem. The model reaches to the optimal solutions reported in [36]. The results summarized in Table 8, indicate global convergence in all cases. Moreover, Figure 5 shows the output trajectory of (9) corresponding to every objective function.

Objective function	Optimal objective value	optimal solution					CPU time (s)
		x_1	x_2	x_3	x_4	y	
$f_1(x,y)$	3.2077	2.6822	1.484	0.0	0.662	4.0587	2.3
$f_2(x,y)$	3.4487	2.7487	1.4	0.7253	0.8241	5.1553	3.9
$f_3(x,y)$	4.6032	2.7892	1.2079	0.0	0.3696	2.3891	2.9
$f_4(x,y)$	6.5918	2.8874	0.8944	0.0	0.1991	1.373	2.2

Tab. 8. Numerical results of proposed model (9) Example 5.3. with random initial point.

Example 5.4. The following OPVIC problem is a Stackelberg leader-follower game problem. Facchinei in [5] and LV in [25] used this example to test their methods.

$$
\begin{aligned}
&n=1, \quad m=1,\\
&f(x,y)=0.5x^2+0.5xy-95x,\\
&x\in[0,200],\\
&y\geq 0, \quad \langle 2y+0.5x-100, y-y'\rangle \leq 0, \quad \forall y'\geq 0.
\end{aligned}
$$

Using KKT conditions, the above variational inequality constraint can be written as follows:

$$
\begin{aligned}
&2y+0.5x-100-\lambda_1=0,\\
&y-\mu_1=0,\\
&2\min\{\lambda_1,\mu_1\}=0.
\end{aligned}
$$

So, the neurodynamics model (9) is used to solve this problem. To illustrate the validity of theoretical analysis, ϵ and λ values is tended to zero and infinity, respectively. Results

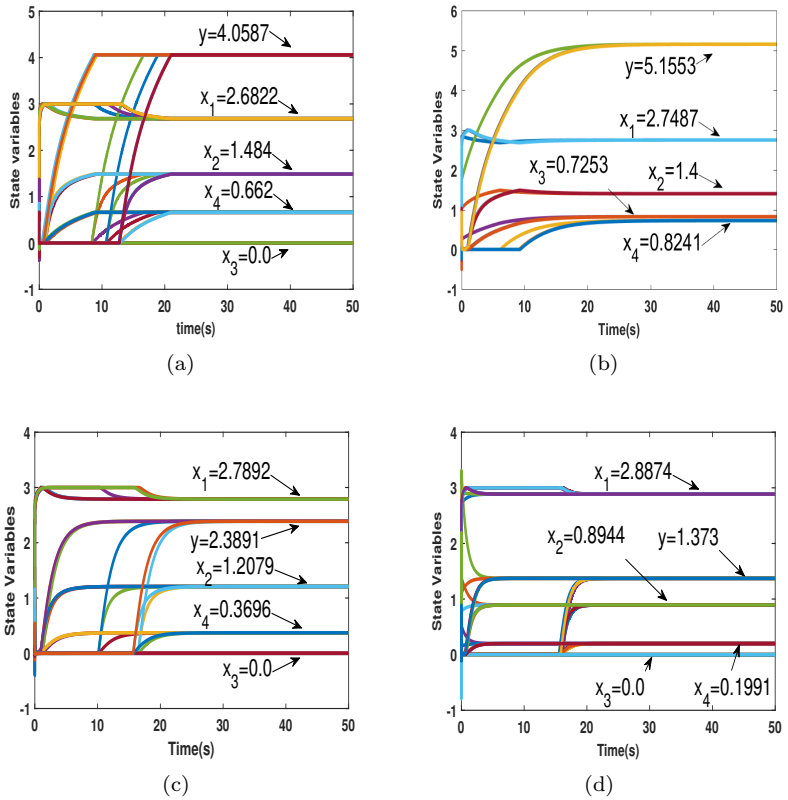


Fig. 5. The transient conduct of the neural system model (9) with arbitrary, initial points and various objective functions in Example 5.3.: 5(a) $f_1(x,y)$, 5(b) $f_2(x,y)$, 5(c) $f_3(x,y)$, 5(d) $f_4(x,y)$.

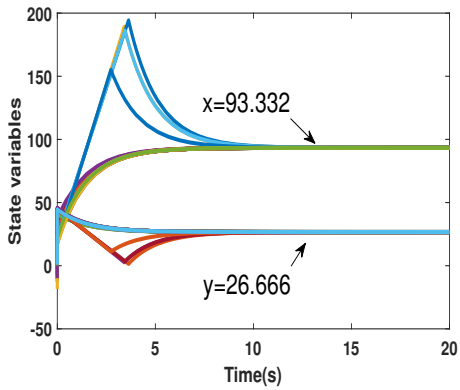
is reported in Table 9. It can be seen that a better estimate for $w^* = (x^*, y^*)$ is obtained by $\epsilon \to 0$ and $\lambda \to \infty$. As shown in Figure 6(a), the trajectories of (9) will converge successfully to $w^* = (x^*, y^*) = (93.33, 26.67)$ i.e. the optimal solution of this problem. The energy function $E(w = (x,y)^T, \lambda_1^*, \mu_1^*)$ is drawn up in Figure 6(b), clearly, $E(w = (x,y)^T, \lambda_1^*, \mu_1^*)$ is convex. In this way, the assumptions of Theorem 10 will be happen.

Example 5.5. The next problem is also an OPVIC problem which has tested by Facchinei in [5] and Lv in [25], for their proposed models.

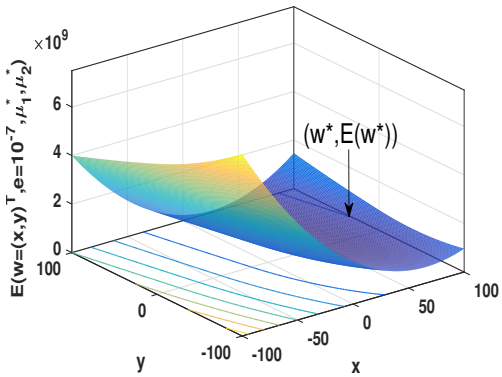
$$
\begin{aligned}
&n = 2, \quad m = 2,\\
&f(x,y) = x_1^2 - 2x_1 + x_2^2 - 2x_2 + y_1^2 + y_2^2,\\
&x_1, x_2 \in [0,2],\\
&H(x,y) = \begin{pmatrix} 2y_1 - 2x_1 \\ 2y_2 - 2x_2 \end{pmatrix},\\
&y \in \mathbf{\Omega}, \quad \langle H(x,y), y - y \rangle \leq 0, \quad \forall y' \in \mathbf{\Omega},
\end{aligned}
$$

Parameters		Optimal solution		Optimal value	CPU time(s)
ϵ	λ	x^*	y^*	$f(x^*, y^*)$	
0.01	100	93.432661975135200	26.496032022488870	-3273.474323938126	0.239
0.001	1000	93.342781461899750	26.649738610115150	-3267.346449808123	0.241
0.0001	10000	93.335142362283520	26.664757914192254	-3266.734636534589	0.281
0.00001	100000	93.333448077624440	26.666492334563888	-3266.673463484818	0.282

Tab. 9. Estimations of optimal solution $w = (x, y)$, when $\epsilon \to 0$ and $\lambda \to \infty$ in Example 5.4.



(a)



(b)

Fig. 6. 6(a) Convergent trajectories of the neural system (9) with ten random starting points in Example 5.4. 6(b) The energy function $E(w = (x, y)^T, \lambda_1^*, \mu_1^*)$ of Example 5.4.

where $\mathbf{\Omega} = \{y|\ \Re_1(x,y) = 0.25 - (y_1-1)^2 \geq 0;\ \Re_2(x,y) = 0.25 - (y_2-1)^2 \geq 0\}$. Similar to pervious example, the above variational inequality constraint can be reformulated as follows:

$$\begin{aligned}
&2y_1 - 2x_1 + 2\mu_1(y_1-1) = 0,\\
&2y_2 - 2x_2 + 2\mu_2(y_2-1) = 0,\\
&0.25 - (y_1-1)^2 - \lambda_1 = 0,\\
&0.25 - (y_2-1)^2 - \lambda_2 = 0,\\
&\min\{\mu_1, \lambda_1\} = 0,\\
&\min\{\mu_2, \lambda_2\} = 0.
\end{aligned}$$

The approach (9) is applied to solve the equivalence problem. Using the MATLAB software, the problem was solved with $\epsilon = 10^{-7}, \lambda = 10^5$, and 20 random initial points. See Figure 7(a), for the convergence to the optimal solution $w = (x_1, x_2, y_1, y_2) = (0.5, 0.5, 0.5, 0.5)$ with optimal value $f(w) = -1$.

Example 5.6. Jonathan F. BARD in [3], considers the following bilevel programming problem which was tested in [5].

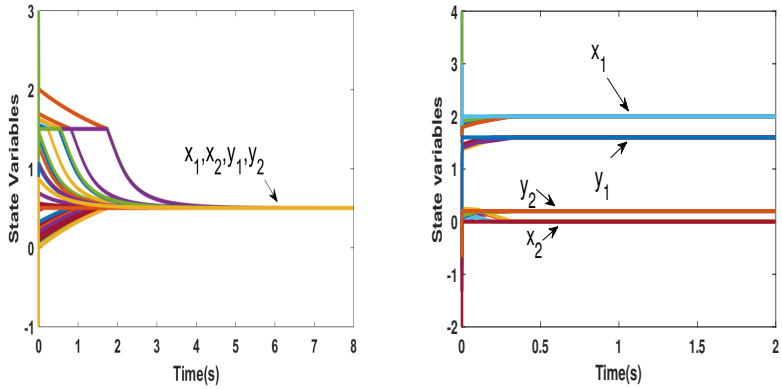
$$\begin{aligned}
\min\quad & F(x,y) = -x_1^2 - 3x_2 - 4y_1 + y_2^2\\
\text{s.t.}\quad & x_1^2 + 2x_2 \leq 4,\\
& x_1 \geq 0;\ \ x_2 \geq 0,\\
& \quad \min\quad f(x,y) = 2x_1^2 + y_1^2 - 5y_2,\\
& \quad \text{s.t.}\quad x_1^2 - 2x_2 + x_2^2 - 2y_1 + y_2 \geq -3,\\
& \qquad\quad x_2 + 3y_1 - 4y_2 \geq 4,\\
& \qquad\quad y_1 \geq 0;\ \ y_2 \geq 0.
\end{aligned}$$

Rewrite the lower level problem as below:

$$\begin{aligned}
&\min\{y_1, 2y_1 + 2\lambda_1 - 3\lambda_2\} = 0,\\
&\min\{y_2, -5 - \lambda_1 + 4\lambda_2\} = 0,\\
&\min\{\lambda_1, x_1^2 - 2x_1 + x_2^2 - 2y_1 + y_2 + 3\} = 0,\\
&\min\{\lambda_2, x_2 + 3y_1 - 4y_2 - 4\} = 0,
\end{aligned}$$

where λ_1 and λ_2 are Lagrange coefficients of the lower level problem. The optimal solution of this problem is $x = (2,0),\ y = (1.6, 0.2)$ [5]. For the neurodynamics optimization model 9, let $\lambda = 10^4, \epsilon = 10^{-6}$. We select four initial points: $(x_0, y_0) = (3,1,0,0), (3,1,-1,-1), (4,-2,0,0), (4,0,-1,0)$. Figure 7(b), displays the output trajectories of proposed model corresponding to selected starting points. All trajectories converge to the optimal solution successfully.

Example 5.7. The following example is a quadratic zero-one programming problem



(a) (b)

Fig. 7. 7(a) The output trajectories of the neural system (9) with twenty random starting points in Example 5.5. 7(b) The convergence behavior of the outgoing paths of the dynamic system to the optimal solution in Example 5.6.

that can be rewritten equally as a MPEC.

$$\begin{aligned}
\min \quad & f(x) = \frac{1}{2} x^T Q x \\
\text{s.t.} \quad & x_1 + x_2 + x_3 = 1, \\
& x_4 + x_5 + x_6 = 1, \\
& x_7 + x_8 + x_9 = 1, \\
& x_1 + x_4 + x_7 = 1, \\
& x_2 + x_5 + x_8 = 1, \\
& x_3 + x_6 + x_9 = 1, \\
& x_i \in \{0, 1\}, \ i = 1, \ldots, 9,
\end{aligned}$$

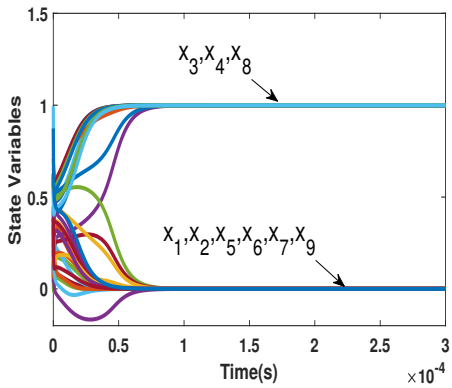
where

$$Q = \begin{pmatrix}
175 & 4 & 11 & 10 & 9 & 27 & 18 & 17 & 49 \\
4 & 175 & 13 & 11 & 20 & 32 & 19 & 36 & 58 \\
11 & 13 & 175 & 28 & 33 & 10 & 50 & 59 & 18 \\
10 & 11 & 28 & 178 & 16 & 44 & 22 & 21 & 60 \\
9 & 20 & 33 & 16 & 185 & 52 & 23 & 44 & 71 \\
27 & 32 & 10 & 44 & 52 & 179 & 61 & 72 & 22 \\
18 & 19 & 50 & 22 & 23 & 61 & 174 & 4 & 11 \\
17 & 36 & 59 & 21 & 44 & 72 & 4 & 177 & 13 \\
49 & 58 & 18 & 60 & 71 & 22 & 11 & 13 & 174
\end{pmatrix}.$$

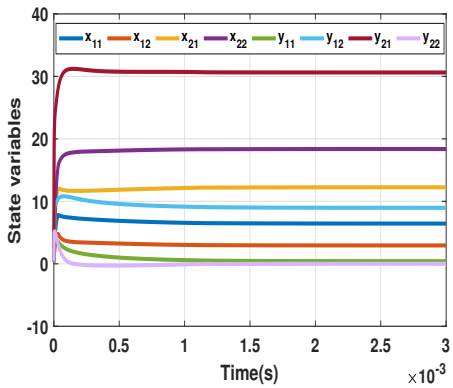
Binary variables are replaced by the following constraints:

$$\begin{aligned} &r_i + s_i = \tfrac{1}{2}, \\ &x_i - r_i + s_i = \tfrac{1}{2}, \\ &r_i.s_i = 0, \\ &r_i, s_i \geq 0,\ i = 1, \ldots, 9. \end{aligned}$$

Fischer–Burmeister NCP-function and the neural network 9, is applied to solve this problem. The optimal solution is $x = (0, 0, 1, 1, 0, 0, 0, 1, 0)$ [37] and according to Figure 8(a), it can be seen that the proposed model is convergent to the optimal solution.



(a)



(b)

Fig. 8. 8(a) The convergence behavior of the outgoing paths of the dynamic system to the optimal solution in Example 5.7.
8(b) The convergence behavior of the outgoing paths of the dynamic system to the optimal solutions in resource allocation problem corresponding with $\mu_1 = \mu_2 = 0.72$ and $\mu_3 = 0.77$.

5.1. An application to resource allocation problem

Consider a product marketing company with a resource allocation center and two factories as a subsystem. Each factory generates two types of products using dedicated resources. The center decides on the number of resources allocated to the factories to maximize its overall profitability in product marketing. Based on three factors: efficiency, quality, and performance, each factory is willing to achieve the goal of its production activity. Let's define the symbols as Table 10. The allocation resource problem is:

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{y}} \quad & F(Y(x)) = -(200 - y_{11} - y_{21})(y_{11} + y_{21}) - (160 - y_{12} - y_{22})(y_{12} + y_{22}) \\
\text{s.t.} \quad & x_{11} + x_{12} + x_{21} + x_{22} \le 40, \\
& 0 \le x_{11} \le 10, \;\; 0 \le x_{21} \le 5, \\
& 0 \le x_{21} \le 15, \;\; 0 \le x_{22} \le 20, \\
& \min \quad f_1(y_1) = (y_{11} - 4)^2 + (y_{12} - 13)^2, \\
& \text{s.t.} \quad 0.4y_{11} + 0.7y_{12} \le x_{11} \\
& \qquad 0.6y_{11} + 0.3y_{12} \le x_{12}, \\
& \qquad 0 \le y_{11}, \; y_{12} \le 20, \\
& \qquad \min \quad f_2(y_2) = (y_{21} - 35)^2 + (y_{22} - 2)^2, \\
& \qquad \text{s.t.} \quad 0.4y_{21} + 0.7y_{22} \le x_{21}, \\
& \qquad\qquad 0.6y_{21} + 0.3y_{22} \le x_{22}, \\
& \qquad\qquad 0 \le y_{21}, \; y_{22} \le 40.
\end{aligned}
$$

This problem, prepared by E. Aiyoshi and K. Shimizu in [2] for the first time . It also, has been tested in [3] and [5] by F. Bard and F. Facchinei. So far, two different optimal solutions have been obtained. First, $\hat{x} = (x_{11}, x_{12}, x_{21}, x_{22}) = (7, 3, 12, 18)$, $\hat{y}_1 = (y_{11}, y_{12}) = (0, 10)$, $\hat{y}_2 = (y_{21}, y_{22}) = (30, 0)$. Second, $\bar{x} = (x_{11}, x_{12}, x_{21}, x_{22}) = (7.91, 4.37, 11.09, 16.63)$, $\bar{y}_1 = (y_{11}, y_{12}) = (2.29, 10)$, $\bar{y}_2 = (y_{21}, y_{22}) = (27.21, 0)$. Due to the fact that the set of solutions to such problems is convex [3], any convex combination of these points will be optimal, i.e. the solutions will be in the form $\mu_1\hat{x} + (1-\mu_1)\bar{x}$, $\mu_2\hat{y}_1 + (1-\mu_2)\bar{y}_1$, $\mu_3\hat{y}_2 + (1-\mu_3)\bar{y}_2$, with $0 \le \mu_1, \mu_2, \mu_3 \le 1$. According to the convexity of the lower level problem, this is equivalent with the following complementarity constraints:

$$
\begin{aligned}
&\min\{y_{11}, 2(y_{11} - 4) + 0.4\lambda_1 + 0.6\lambda_2 + \lambda_3\} = 0, \\
&\min\{y_{12}, 2(y_{12} - 13) + 0.7\lambda_1 + 0.3\lambda_2 + \lambda_4\} = 0, \\
&\min\{y_{21}, 2(y_{21} - 35) + 0.4\lambda_5 + 0.6\lambda_6 + \lambda_7\} = 0, \\
&\min\{y_{22}, 2(y_{22} - 2) + 0.7\lambda_5 + 0.3\lambda_6 + \lambda_8\} = 0, \\
&\min\{\lambda_1, -0.4y_{11} - 0.7y_{12} + x_{11}\} = 0, \\
&\min\{\lambda_2, -0.6y_{11} - 0.3y_{12} + x_{12}\} = 0, \\
&\min\{\lambda_3, -y_{11} + 20\} = 0, \\
&\min\{\lambda_4, -y_2 + 20\} = 0, \\
&\min\{\lambda_5, -0.4y_{21} - 0.7y_{22} + x_{21}\} = 0, \\
&\min\{\lambda_6, x_{22} - 0.6y_{21} - 0.3y_{22}\} = 0, \\
&\min\{\lambda_7, 40 - y_{21}\} = 0, \\
&\min\{\lambda_8, 40 - y_{22}\} = 0.
\end{aligned}
$$

$\lambda_i, \quad i = 1, \ldots, 8$ are Lagrange coefficients of the lower level problem. Utilizing proposed model (9) with $\epsilon = 10^{-5}, \lambda = 10^4$, we will obtain different optimal solutions by changing the starting point. For instance, Figure 8(b), displays optimal solution $x = (x_{11}, x_{12}, x_{21}, x_{22}) = (7.25, 3.38, 11.74, 17.61), \quad \bar{y}_1 = (y_{11}, y_{12}) = (0.64, 10), \quad \bar{y}_2 = (y_{21}, y_{22}) = (29.35, 0)$, corresponding with $\mu_1 = \mu_2 = 0.72$ and $\mu_3 = 0.77$. The optimal objective value is $F = -6600$.

x_{ni},	The amount of i^{th} resource allocated to the n^{th} factory,
y_{nj}	The amount of j^{th} products produced by the n^{th} factory ,
Y_j	The total product marketing j,
$F(Y)$	The profit function of marketing,
$f_n(y_n)$	The goal function of the n^{th} factory.

Tab. 10. Notation used in resource allocation problem.

We now have numerical experiments to explain the effectiveness of the proposed model for solving a variety of MPEC problems. This model was implemented using Matlab R2016b modeling language. Numerical results are obtained only in one iteration, with the proper selection of the parameters λ and ϵ. The results show that the use of the penalty method in designing neural networks to solve MPEC problems can be efficient. Correctly, it can be recognized that the proposed network has made meaningful improvements over the partial augmented Lagrangian method, $\frac{1}{2}$ penalty method, implicit function-based bundle trust region, and other existing methods, both in terms of the repetition number and CPU time. Also, the accuracy of the solutions obtained with this method is quite competitive with other methods.

6. CONCLUSIONS

We prepare a simple model of neurodynamics optimization technique for solving MPECs based on penalty method and NCP- function. The smooth perturbation of the minimum function and its natural merit function are used to achieve a relaxed optimization problem. Using the relaxed problem, a new energy function which its lowest energy state corresponds to the optimal solution introduced. The novel recurrent neural network model is constructed based on the steepest descent method and has a simple one-layer structure. With the theory of stability in differential equations, significant results proved. It demonstrated that its state vector converges to an optimal solution of MPEC under a simple condition. Based on Lyapunov theorem, the proposed model, is asymptotic stable over its equilibrium point. Computational experiences persuaded us to validate theoretical results. As described in the first numerical example, the method can reduce the number of repetitions needed to obtain an optimal response with optimal accuracy than the partial augmented Lagrangian method, $\frac{1}{2}$ penalty method, and modified Levenberg–Marquardt method. It led to reduced computational load and time. In addition, the simulation of this example well illustrates that the new method converges

to the optimal point, regardless of its stationary point type. The proposed model has been used to solve some examples of OPVIC, OPCC, BLPs, and problems that reformulated as an MPEC such as zero-one programming problems. However, if we specialize the new method to bilevel programming problems, it does appear that after applying the KKT approach, the crucial factor influences the transient behavior of the proposed ANN model is its initial point. Thus, to get the optimal solution in this class, we have to choose the appropriate starting point. A case study on the resource allocation problem reveals the potential of the proposed model in engineering and economics applications.

(Received October 6, 2018)

REFERENCES

[1] R. Andreani, and J. M. Martínez: On the solution of mathematical programming problems with equilibrium constraints. Math. Methods Oper. Res. *54* (2001), 345–359. DOI:10.1007/s001860100158

[2] E. Aiyoshi and K. Shimizu: Hierarchical decentralized systems and its new solution by a barrier method. IEEE Trans. Systems Man Cybernet. *11* (1981), 444–449. DOI:10.1109/tsmc.1981.4308712

[3] J. F. Bard: Convex two-level optimization. Math. Program. *40* (1988), 15–27. DOI:10.1109/tsmc.1981.4308712

[4] J. S. Chen: On some NCP-functions based on the generalized Fischer–Burmeister function. Asia-Pacific J. Oper. Res. *24* (2007), 401–420. DOI:10.1142/s0217595907001292

[5] F. Facchinei, H. Jiang, and L. Qi: A smoothing method for mathematical programs with equilibrium constraints. Math. Program. *85* (1999), 107–134. DOI:10.1007/s10107990015a

[6] M. C. Ferris, S. P. Dirkse, and A. Meeraus: Mathematical Programs with Equilibrium Constraints: Automatic Reformulation and Solution via Constrained Optimization. Frontiers in Applied General Equilibrium Modeling, Cambridge University Press 2002, pp. 67–94. DOI:10.1017/cbo9780511614330.005

[7] R. Fletcher, S. Leyffer, D. Ralph, and S. Scholtes: Local convergence of SQP methods for mathematical programs with equilibrium constraints. SIAM J. Optim. *17* (2006), 259–286. DOI:10.1137/s1052623402407382

[8] F. Facchinei and J. Soares: A new merit function for nonlinear complementarity problems and a related algorithm. SIAM J. Optim. *7* (1997), 225–247. DOI:10.1137/s1052623494279110

[9] L. Guo, G. H. Lin, and J. J. Ye: Solving mathematical programs with equilibrium constraints. J. Optim. Theory Appl. *166* (2015), 234–256. DOI:10.1007/s10957-014-0699-z

[10] A. Golbabai and S. Ezazipour: A high-performance nonlinear dynamic scheme for the solution of equilibrium constrained optimization problems. Expert Systems Appl. *82* (2017), 291–300. DOI:10.1016/j.eswa.2017.04.016

[11] X. He, C. Li, T. Huang, and C. H. Li: Neural network for solving convex quadratic bilevel programming problems. Neural Networks *51* (2014), 17–25. DOI:10.1016/j.neunet.2013.11.015

[12] J. J. Hopfiel and D. W. Tank: Neural computation of decisions in optimization problems. Biol. Cybernet. *52* (1985), 141–152. DOI:10.1016/0096-3003(85)90004-9

[13] A. Hosseini and S. M. Hosseini: A new steepest descent differential inclusion-based method for solving general nonsmooth convex optimization problems. J. Optim. Theory Appl. *159* (2013), 698–720. DOI:10.1007/s10957-012-0258-4

[14] N. Hosseinipour-Mahani and A. Malek: A neurodynamic optimization technique based on overestimator and underestimator functions for solving a class of non-convex optimization problems. Math. Comput. Simul. *122* (2016), 20–34. DOI:10.1016/j.matcom.2015.09.013

[15] N. Hosseinipour-Mahani and A. Malek: Solving a class of non-convex quadratic problems based on generalized KKT conditions and neurodynamic optimization technique. Kybernetika *51* (2015), 890–908. DOI:10.14736/kyb-2015-5-0890

[16] X. X. Huang, X. Q. Yang, and K. L. Teo: Partial augmented Lagrangian method and mathematical programs with complementarity constraints. Global Optim. *35* (2006), 235–254. DOI:10.1007/s10898-005-3837-1

[17] J. Y. Jane: Necessary and sufficient optimality conditions for mathematical programs with equilibrium constraints. Math. Anal. Appl. *307* (2005), 350–369. DOI:10.1016/j.jmaa.2004.10.032

[18] M. Kočvara and J. Outrata: Optimization problems with equilibrium constraints and their numerical solution. Math. Program. *101* (2004), 119–149. DOI:10.1007/s10107-004-0539-2

[19] C. Kanzow and A. Schwartz: A new regularization method for mathematical programs with complementarity constraints with strong convergence properties. SIAM J. Optim. *23* (2013), 770–798. DOI:10.1137/100802487

[20] D. G. Luenberger and Y. Ye: Linear and Nonlinear Programming. Springer Science and Business Media, 233 Spring Street, New York 2008. DOI:10.1007/978-0-387-74503-9

[21] Z. Q. Luo, J. S. Pang, and D. Ralph: Mathematical Programs with Equilibrium Constraints. Cambridge University Press, Cambridge 1996. DOI:10.1017/cbo9780511983658

[22] K. M. Lan, U. P. Wen, H. S. Shih, and E. S. Lee: A hybrid neural network approach to bilevel programming problems. Appl. Math. Lett. *20* (2007), 880–884. DOI:10.1016/j.aml.2006.07.013

[23] J. Li, C. Li, Z. Wu, and J. Huang: A feedback neural network for solving convex quadratic bi-level programming problems. Neural Comput. Appl. *25* (2014), 603–611. DOI:10.1007/s00521-013-1530-8

[24] Q. Liu and J. Wang: A one-layer projection neural network for nonsmooth optimization subject to linear equalities and bound constraints. IEEE Trans. Neural Networks Learning Systems *24* (2013), 812–824. DOI:10.1109/tnnls.2013.2244908

[25] Y. Lv, Z. Chen, and Z. Wan: A neural network approach for solving mathematical programs with equilibrium constraints. Expert System Appl. *38* (2011), 231–234. DOI:10.1016/j.eswa.2010.06.050

[26] S. Leyffer: Complementarity constraints as nonlinear equations: Theory and numerical experience. In: Optimization with Multivalued Mappings (S. Dempe and V. Kalashnikov, eds.), Springer Optimization and Its Applications, vol 2. Springer, Boston 2006. DOI:10.1007/0-387-34221-4_9

[27] L. Z. Liao, H. Qi, and L. Qi: Solving nonlinear complementarity problems with neural networks: a reformulation method approach. J. Comput. Appl. Math. *131* (2001), 343–359. DOI:10.1016/s0377-0427(00)00262-4

[28] E. W. Lillo, M. H. Loh, S. Hui, and H. S. Zak: On solving constrained optimization problems with neural networks: a penalty method approach. IEEE Trans. Neural Networks *4* (1993), 931–940. DOI:10.1109/72.286888

[29] G. H. Lin and M. Fukushima: New relaxation method for mathematical programs with complementarity constraints. J. Optim. Theory Appl. *118* (2003), 81–116. DOI:10.1023/a:1024739508603

[30] A. Malek, S. Ezazipour, and N. Hosseinipour-Mahani: Projected dynamical systems and optimization problems. Bull. Iranian Math. Soc. *37* (2011), 81–96.

[31] A. Malek, S. Ezazipour, and N. Hosseinipour-Mahani: Double projection neural network for solving pseudomonotone variational inequalities. Fixed Point Theory *12* (2011), 401–418.

[32] A. Malek, N. Hosseinipour-Mahani, and S. Ezazipour: Efficient recurrent neural network model for the solution of general nonlinear optimization problems. Optim. Methods Software *25* (2010), 489–506. DOI:10.1080/10556780902856743

[33] D. D. Morrison: Optimization by least squares. SIAM J. Numer. Anal. *5* (1968), 83–88. DOI:10.1137/0705006

[34] R. K. Miller and A. N. Michel: Ordinary Differential Equations. Academic Press, New York 1982.

[35] J. V. Outrata, M. Kočvara, and J. Zowe: Nonsmooth Approach to Optimization Problems with Equilibrium Constraints, Theory, Applications and Numerical Results. Kluwer Academic Publishers, Dordrecht 1998. DOI:10.1007/978-1-4757-2825-5

[36] J. V. Outrata: On optimization problems with variational inequality constraints. SIAM J. Optim. *4* (1994), 340–357. DOI:10.1137/0804019

[37] M. Ranjbar, S. Effati, and S. M. Miri: An artificial neural network for solving quadratic zero-one programming problems Neurocomputing *235* (2017), 192–198. DOI:10.1016/j.neucom.2016.12.064

[38] Z. Sheng, Z. Lv, and Z. Xu: A new algorithm based on the Frank–Wolfe method and neural network for a class of bilevel decision making problem. Acta Automat. Sinica *22* (1996), 657–665.

[39] S. Scholtes and M. Stohr: Exact penalization of mathematical programs with complementarity constraints. SIAM J. Control Optim. *37* (1999), 617–652. DOI:10.1137/s0363012996306121

[40] D. W. Tank and J. J. Hopfield: Simple neural optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit. IEEE Trans. Circuits Syst. *33* (1986), 533–541. DOI:10.1109/tcs.1986.1085953

[41] Q. X. Yang and X. X. Huang: Lower-order penalty methods for mathematical programs with complementarity constraints. Optim. Methods Software *19* (2004), 693–720. DOI:10.1080/1055678041001697659

Soraya Ezazipour, Department of Applied Mathematics, Faculty of Mathematical Sciences, Iran University of Science and Technology, P.O.Box 16846-13114, Tehran. Iran.
e-mail: s_ ezazipour@mathdep.iust.ac.ir

Ahmad Golbabai, Department of Applied Mathematics, Faculty of Mathematical Sciences, Iran University of Science and Technology, P.O.Box 16846-13114, Tehran. Iran.
e-mail: golbabai@iust.ac.ir