A Recurrent Neural Network for Solving Nonlinear Projection Equations

Youshen Xia and Jun Wang
Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Shatin, Hong Kong
Email: {ysxia, jwang}@mae.cuhk.edu.hk

I. Introduction

In this paper, we are concerned with the nonlinear projection equations of the following form

\[ P_X(u - F(u)) = u \]  \hspace{1cm} (1)

where \( X = \{u \in \mathbb{R}^l \mid d_i \leq u_i \leq h_i, \forall i \in N \subset L\}, L = \{1, \ldots, l\}, F : \mathbb{R}^m \to \mathbb{R}^m \) is a continuous differentiable mapping, and \( P_X : \mathbb{R}^l \to X \) is a projection operator which is defined by \( P_X(u) = [P_X(u_1), \ldots, P_X(u_l)]^T \) and for \( i \in L - N \), \( P_X(u_i) = u_i \); for \( i \in N \),

\[ P_X(u_i) = \begin{cases} 
    d_i & u_i < d_i \\
    u_i & d_i \leq u_i \leq h_i \\
    h_i & u_i > h_i.
\end{cases} \]

In particular, for \( i \in N \), if \( h_i = +\infty \), then \( P_X(u_i) = \max\{u_i, d_i\} \); if \( d_i = -\infty \), \( P_X(u_i) = \min\{h_i, u_i\} \); and if \( h_i = +\infty \) and \( d_i = 0 \), then \( P_X(u_i) = (u_i)^+ = \max\{0, u_i\} \).

Note that if \( X \) is whole space, then problem (1) reduces to the problem of finding a solution of nonlinear equations \( F(u) = 0 \), whereas, if \( d_i = 0 \) and \( h_i = +\infty \) for all \( i = 1, \ldots, l \), then the problem (1) is just the well-known nonlinear complementarity problem

\[ u \geq 0, \quad F(u) \geq 0, \quad u^T F(u) = 0. \]

Therefore, despite the particular structure of the feasible set \( X \), the problem (1) is still a very general problem in mathematics programming. Moreover, there are a number of important applications which lead to this special class of variational inequalities such as equilibrium models arising in fields of economics and transportation science, etc. [5]. Various numerical solution procedures for the problem (1) have been investigated over decades [1-4]. Because of the nature of digital computers, conventional algorithms are time-consuming for large-scale optimization problems. It is well-known that one promising approach to optimization problems in real time is to employ artificial neural networks implemented in hardware. Recurrent neural networks for solving optimization problems are readily hardware-implementable. Thus, neural networks are a top-choice of real-time solvers for optimization problems. Since the seminal work of Hopfield and Tank [6], the neural network approach to optimization has been investigated and many neural networks for optimization problems have been proposed [7-10].

II. Neural Network Formulation

Recently, we proposed a recurrent neural network model [10] for solving (1):

\[ \frac{du}{dt} = -\lambda\{z + \alpha F(P_X(z)) - P_X(z)\} \]  \hspace{1cm} (2)
where \( z = u - \alpha F(u) \), and \( \alpha \) and \( \lambda \) are positive constants. Under the assumption that \( F(u) \) satisfies

\[
(u - v)^T(F(u) - F(v)) \leq \beta \| u - v \|^2, \quad \forall u, v \in \mathbb{R}^l,
\]

we obtained the following stability result:

- If \( F(u) \) is monotone and \( \alpha < \frac{1}{\beta} \), then the neural network is stable in the Lyapunov sense and is globally convergent to an exact solution to (1).

Since it is difficult to find \( \beta \) sometimes, the condition (3) limits many problems in applications. In paper [12], based on the dynamic model described by Friesz et al. [5], we proposed a recurrent neural network for solving linear projection equations. In this paper, we generalize this model and propose the following neural network model for solving (1)

\[
\frac{du}{dt} = -\lambda\{u - P_X(u - \alpha F(u))\}, \tag{3}
\]

The neural network model in (4) can be realized easily by a circuit shown in Fig. 1 where the projection operator \( P_X(\cdot) \) may be implemented by using a piecewise activation function. In contrast to (2), the present neural network has no in addition term \( F(u) \) and \( F(u - \alpha F(u)) \) and thus decreases the complexity in implementation. Moreover, in the theoretical aspect, we obtain the global stability results of the proposed neural network without condition (3).

Throughout this paper we assume that \( X^* = \{ u \in \mathbb{R}^l \mid u \text{ solves (1)} \} \) is nonempty and bounded, and assume that \( F(u) \) is differentiable in an open convex set \( X_0 \) including \( X \).

---

**Figure 1:** Architecture of the proposed neural network in (4)

---

**III. Stability Results**
First, from the paper [11] we have the following stability results of (4).

**Theorem 1.** Assume that \( F(u) \) is locally Lipschitz continuous in \( \mathbb{R}^l \). Then there exists a unique continuous solution \( u(t) \) for (4). Moreover, when \( x_0 \notin X \), the solution trajectory of (4) will globally exponentially approach to \( X \), that is,

\[
\| u(t) - P_X(u(t)) \| \leq \| u(t_0) - P_X(u(t_0)) \| e^{-\lambda(t-t_0)},
\]

where \( \| \cdot \| \) denotes \( l_2 \) norm, and \( u(t) \subset X \) when \( u_0 \subset X \).

**Theorem 2.** Assume that \( F(u) \) is monotone on \( X \) and \( \nabla F(u) \) is symmetric for any \( u \in X \), then the proposed neural network in (4) with \( u_0 \in X \) is stable in the Lyapunov sense and is globally convergent to the solution set of (1). Specially, (4) is globally asymptotically stable when \( X^* = \{ u^* \} \).

**Remark.** The neural network in (4) is said to converge globally to the solution set \( X^* \) if irrespective of the initial point the trajectory of (4) \( u(t) \) satisfies \( \lim_{t \to \infty} \text{dist}(u(t), X^*) = 0 \), where \( \text{dist}(u, X^*) = \inf_{y \in X^*} \| u - y \| \).

**Theorem 3.** Assume that \( \nabla F(u) \) is symmetric and \( \| \nabla F(u) \| \) has an upper bound on \( X \). If \( F(u) \) is strong monotone on \( X \), then for any initial point \( u_0 \in X \), the proposed neural network in (4) with \( \alpha < \frac{\max_{u \in X} \| \nabla F(u) \|}{2} \) is globally exponentially stable.

Furthermore, we obtain new stability results of (4) below.

**Theorem 4.** Assume that \( \nabla F(u) \) is symmetric for any \( u \in X \) which is bounded. Then the proposed neural network in (4) with \( u_0 \in X \) is stable in the Lyapunov sense and is globally convergent to the solution set of (1). Specially, (4) is globally asymptotically stable when \( X^* = \{ u^* \} \).

**Theorem 5.** Assume that \( F(u) \) is strict monotone, then the proposed neural network in (4) with \( u_0 \in X \) is globally asymptotically stable.

**Remark.** Theorem 4 does not require the monotone condition of \( F \). Theorem 5 does not require the symmetric condition of \( F \).

**V. Illustrate Examples**

**Example 1.** Consider the problem (1), where

\[
F(x) = \begin{bmatrix}
-2(1 - x_1)x_2 + (2 - x_2)^2 \\
(1 - x_1)^2 - 2x_1(2 - x_2)
\end{bmatrix}
\]

where \( X = \{ x \in \mathbb{R}^2 | 0 \leq x \leq 3e \} \), \( e = [1, 1]^T \), and \( F(x) \) is symmetric but not monotone. This problem has two solutions \( x^*_1 = [0, 0]^T \) and \( x^*_2 = [1, 2]^T \). Let \( \alpha = 1 \) and \( \lambda = 5 \). Figure 2 shows the trajectory of (4) with the initial point \( x^*_0 = [3, 3]^T, x^*_0 = [0, 5, 0, 2]^T \), respectively.

**Example 2.** Consider the problem (1), where

\[
F(x) = \begin{bmatrix}
x_1 + x_2x_3x_4x_5/50 \\
x_2 + x_1x_3x_4x_5/50 - 3 \\
x_3 + x_3x_2x_1x_4x_5/50 - 1 \\
x_4 + x_2x_3x_1x_5/50 + 0.5 \\
x_5 + x_2x_3x_4x_1/50
\end{bmatrix}
\]

is strict monotone but not symmetric and \( X = \{ x \in \mathbb{R}^5 | x \geq 0 \} \). This problem has only one solution \( x^* = [0, 3, 1, 0, 0]^T \) and it can be written as \( (x - F(x))^+ = x \). Let \( \alpha = 1 \) and \( \lambda = 5 \). Figure 3 shows the trajectory of (4) with the initial point \( x_0 = [1, -1, 2, -2, 5]^T \).
Figure 2: The transient behavior of the proposed neural network with two initial points in Example 1.

Example 3. Consider the following convex program with box constraints

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad x \in \Omega
\end{align*}
\]

(4)

where \( \Omega = \{ x \in \mathbb{R}^d \mid d \leq x \leq h \} \) and \( f(x) \) is twice continuously differentiable and convex. From the Karush-Kuhn-Tucker conditions we see that \( x^* \) is an optimal solution to (5) if and only if \( x^* \) satisfies

\[
\frac{\partial f}{\partial x_i} \begin{cases} 
\geq 0 & x_i = d_i, \\
= 0 & x_i \in (d_i, h_i), \\
\leq 0 & x_i = h_i.
\end{cases}
\]

Thus (6) can be written a projection equation \( P_{\Omega}(x - \nabla f(x)) = x \), where \( \nabla f(x) \) is the gradient of \( f \). So, the proposed neural network for solving (5) becomes

\[
\frac{dx}{dt} = -\lambda \{ x - P_{\Omega}(x - \alpha \nabla f(x)) \}
\]

(5)

and its asymptotical stability result is below:

**Corollary.** The neural network in (7) is stable in the sense of Lyapunov and is globally convergent to the solution set of (5), and (7) is globally asymptotically stable when the problem (5) has unique optimal solution.

This result appeared also in [11], [13], and [14], respectively.

Now, let \( f(x) = x_1^3 + (x_1 + 2x_2)^3 + e^{x_1+x_2} \) and let \( \Omega = \{ x \in \mathbb{R}^2 \mid 1 \leq x_1 \leq 3, 3 \leq x_2 \leq 7 \} \). Then this problem has only an optimal solution \( x^* = [1, 5, 336]^T \) and

\[
\nabla f(x) = \begin{pmatrix} 3x_1^2 + 3(x_1 - 2x_2)^2 + e^{x_1+x_2} \\ 6(x_1 + 2x_2)^2 + e^{x_1+x_2} \end{pmatrix}.
\]
Let $\alpha = 1$ and $\lambda = 5$. Figure 4 shows the trajectories of (7) with four different initial points, where $P_3(3,6)$ and $P_3(3,3)$ are located in $\Omega$, and $P_1(5,0)$, $P_2(2,10)$ are not in $\Omega$.

REFERENCES


Figure 4: The transient behavior of the proposed neural network with four initial points in Example 3.


