Iranian Journal of Fuzzy Systems

Volume 21, Number 3, (2024), pp. 37-63

Original paper

University of Sistan and Baluchestan

A projection neural network model for solving fuzzy convex nonlinear programming problems

M. Jahangiri [1] and A. Nazemi [2]

[1,2] *Faculty of Mathematical Sciences, Shahrood University of Technology, Shahrood, Iran*

jahangiri.sut@gmail.com, nazemi20042003@gmail.com

Abstract

In the proposed manuscript, the solution of the fuzzy nonlinear optimization problems (FNLOPs) is gained using a projection recurrent neural network (RNN) scheme. Since there is a few research on resolving FNLOP by RNNs, we establish a new scheme to solve the problem. By reducing the original program to an interval problem and then a weighting problem, the Karush–Kuhn–Tucker (KKT) conditions are presented. Moreover, we apply the KKT conditions to a RNN as an efficient tool to solve the problem. Besides, the convergence properties and stability analysis of the system model are provided. In the final step, several simulation examples are verified to support the obtained results. Reported results are compared with some other previous neural networks.

Keywords: Neural networks, fuzzy nonlinear programming problem, fuzzy parameters, stability, convergence.

1 Introduction

Various problems in engineering and science, such as optimal control, adaptive signal processing, kinematic control of redundant robot manipulators, nonlinear model predictive control, pattern recognition, regression and classification can be planned as neuro-dynamic optimization problems [1, 3, 4, 5, 6, 19, 35, 41, 59, 61]. With knowledge of advanced computing systems, FNLOP becomes one of the most hopeful approaches to solving practical application problems. FNLOP has been recently applied to finance and engineering problems more frequently [27, 40, 50, 75, 80, 87, 88]. For example, Zhu et al. [87] considered an interval FNLOP scheme for seasonal planning, in [80] Wu and Liao discussed a FNLOP approach to evaluate processes by imprecise data, in [88] Zhu et al. discussed an interval-parameter FNLOP for stream water quality management under uncertainty, in [75] Tang and Wang used a type of FNLOP for modeling a manufacturing system, in [40] Lu and Liu expanded a fuzzy optimization model to calculate the fuzzy S/N ratio for the assessment of the manufacturing processes with fuzzy observations, in [80] Wu and Liao founded fuzzy numbers to perform the quality characteristic measurements, and a nonlinear programming scheme was enforced to solve the fuzzy estimator, in [50] Mousavi et al. studied a distance-based decision framework for multi-attributes analysis by examining intuitionistic fuzzy sets and grey relations with an application to inspection planning in manufacturing firms, and in [27] Hsu et al. examined an agent-based fuzzy constraint-directed negotiation mechanism to solve distributed job shop scheduling problems, and the experimental results disclosed that their model can sort cost-effective job shop scheduling.

Over the years, solving FNLOPs was one of the most crucial problems in the field of fuzzy sets and systems. Obtaining real-time optimal solutions for FNLOPs is the difficulty of a dynamic system model. There are a variety of numerical plans to solve optimization problems, especially in the presence of uncertainty (see [4, 17, 23, 24, 39, 58, 62, 64, 65, 70, 77]). The general numerical algorithms have usually less capability in real-time applications. This is why, to solve nonlinear optimization, the computing time depends on the dimension and the structure of the problem. In usual, the RNN frameworks for optimization are ability practiced more than conventional optimization schemes, because of

Corresponding Author: A. Nazemi
Received: October 2023; Revised: May 2024; Accepted: June 2024.
https://doi.org/10.22111/IJFS.2024.46972.8276

their weighty views such as biological plausibility, inherent nature of parallel and distributed information processing, and hardware parallelizability.

Neural-dynamic optimization has been obtained great development in the past three decades. The pioneering works on a RNN for constrained programming problems are achieved by Hopfield and Tank [26, 76]. There are several popular RNNs for constrained optimization which can be divided into the following two classes: First, the gradient–based RNNs in [2, 11, 12, 37, 43, 52, 53, 54, 57, 63, 72, 73]. Second, the projection type RNNs in [2, 13, 14, 16, 18, 19, 22, 25, 28, 29, 30, 36, 38, 45, 46, 47, 55, 56, 57, 60, 84]. Among gradient models, a globally convergent plan based on Fischer–Burmeister operators for solving second-order cone constrained variational inequality problems by Nazemi and Sabeghi in [54], by Nazemi and Mortezaee in [53] for min-max problems, for semidefinite programming problems by Nikseresht and Nazemi in [57], by Miao et al. [47] for efficiently solving nonlinear convex programs with second-order cone constraints. For projection models, Hu and Wang [28] for solving pseudo-monotone variational inequalities and pseudo-convex optimization problems examined a projection neural network, Arjmandzadeh et al. [2] also studied a new neural network model for solving random interval linear programming problems, Effati et al. [14] established a projection type neural network model to solve bilinear programming problems, He et al. [25] introduced a projection type neural network model to solve variational inequalities, Mansoori et al. [46] discussed on a model to solve the absolute value equations, Nikseresht and Nazemi in [55, 56] addressed two projection networks for linear and nonlinear semi-definite programming problems, Karbasi et al. [29, 30] for fuzzy regression and Feizi and Nazemi [18] for stochastic support vector regression.

As some applications of NNs in real-life engineering, industrial and financial situations, several other NNs were reported in the recent years. For example, a RNN was proposed to solve quadratic programming problems and was applied to solving the k-winner take-all problem [34]. A cooperative kinematic control of multiple manipulators using distributed RNNs was considered [32, 33]. To deal with time-varying data in portfolio analysis, a RNN with rigorous theoretical analysis on the convergence and the optimality of portfolio optimization was established [9]. A novel dynamic model with superior performance in both solution accuracy and time efficiency is developed based on softmax equalization for the effective selection of portfolios [10]. An advanced continuous-time RNN model was addressed for solving continuous time-varying matrix inversion, which has incomparable disturbance-suppression property [67]. A discrete-time RNN, which does not depend on the theory and method of continuous-time RNN model, was introduced for solving a serial robot manipulator problem [69]. A discrete-time RNN model was extended to handle the cable-driven planar robot for discrete real-time tracking control, and the corresponding discrete time NN model was established [68]. A different discrete-time recurrent neurodynamics method was studied and developed to solve equality-constrained discrete time-variant nonlinear optimization problem [66].

Motivated by the former discussion, the aim of this paper is to introduce a RNN to solve the FNLOP. In fact, we study the FNLOP, and reformulate it into an interval program. It is also converted into a bi-objective program, and then the weighting problem is raised. Employing the weighting problem, the KKT optimality conditions are given. A neural network model is proposed by utilizing the KKT conditions. Additionally, the stability and global convergence of the model are debated. It should be noted that, this is the first neural network model for solving the FNLOP in general form. The most important advantage of this model is that it can solve nonconvex programs. Also, the presented model is applied to solving real world problems such as the manufacturing system and NUM problem (it is a nonconvex program applied to analyze distributed rate allocation in a communication network). The main contributions put forward in this paper are the following:

• To the best of our knowledge, this is the first attempt to solve the general form of FNLOP via a RNN. For the first time, Mansoori et al. [43] studied a Lagrangian gradient based neural network model for solving FNLOP with only inequality constraints. For the second time, Mansoori and effati [42] solved the FNLOP with only inequality constraints via a parametric NCP–based RNN. An important limitation of the first and second models is that they solve only strictly convex FNLOP. The main shortcoming of the second model is computing a gradient of complex merit functions, which includes the high computational complexity of the method and costs more computation time.

• The distinguishing features of the recommended network in this paper to solve FNLOPs are that the primal and dual problems can be solved simultaneously, all necessary and sufficient optimality conditions are incorporated, and no penalty parameter is involved.

• The neural solution is insured to have global convergence to the optimal solution of the problem.

• The model in this paper can be used for solving non-convex programs. For example, the model solves real world problems such as manufacturing systems and NUM problems (it is a non-convex optimization, tested to analyze distributed rate allocation in a communication network).

• The convergence of the suggested neural network is guaranteed in the case where it is a defect in the models stated in [51, 52, 81].

• The vast experimental results guarantee the convergence and honest achievement of the established neural solution.

This manuscript is organized as follows. Section 2 presents some preliminaries. In Section 3, we study FNLOP and related optimality conditions. We establish the RNN model in Section 4. In Section 5, the stability and global convergence for the given network are stated. Comparing with some other models is provided in Section 6. Illustrative examples are given in Section 7. Finally, Section 8 concludes this manuscript.

2 Preliminaries

In this section, we give some preliminaries of fuzzy set theory and optimization. Throughout this paper, $\mathbb{R}^n$ denotes the space of n-dimensional real column vectors and $^\top$ denotes the transpose. In what follows, $\|.\|$ denotes l^2-norm of $\mathbb{R}^n$ and $x = (x_1, x_2, ..., x_n)^\top$. For any differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f(x) \in \mathbb{R}^n$ means the gradient of f at x and $\nabla^2 f(x) \in \mathbb{R}^{n\times n}$ means the Hessian matrix of $f(x)$. For any differentiable mapping $F = (F_1, ..., F_m)^T : \mathbb{R}^n \to \mathbb{R}^m$, $\nabla F = [\nabla F_1(x), ..., \nabla F_m(x)] \in \mathbb{R}^{n\times m}$, denotes the transposed Jacobian of $\tilde{F}$ at x. If $A \in \mathbb{R}^{m\times n}$, then the ith row of A is denoted by $A_{i.}$ and the jth column of A is denoted by $A_{.j}$.

2.1 Fuzzy set theory

Here, we state some requirements from fuzzy set theory.

Definition 2.1. [74] *A fuzzy number is a fuzzy set $\tilde{u} : \mathbb{R} \to [0, 1]$ with the property that $\tilde{u}$ is fuzzy convex, that is, $\tilde{u}((1-\lambda)(x) + \lambda(y)) \geq min\{\tilde{u}(x), \tilde{u}(y)\}$, where $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$. E^1 denotes the set of all fuzzy numbers. Also, for fuzzy number $\tilde{v}(x) \in E^1$, the α-cut set, shown by $[\underline{u}(\alpha), \overline{u}(\alpha)]$, is given by*

$$\tilde{u}(\alpha) = \begin{cases} \{x \in \mathbb{R}, u(x) \geq \alpha\} & if \quad 0 < \alpha \leq 1, \\ cl(supp\ \tilde{u}), & if \quad \alpha = 0. \end{cases} \tag{1}$$

Definition 2.2. [71]

- *A triangular fuzzy number is denoted by $\tilde{u} = (a, b, c)$, and its α-cuts is denoted by $\tilde{u}[\alpha] = [a+\alpha(b-a), c-\alpha(c-b)]$.*
- *If $\tilde{u}[\alpha] = [\underline{u}(\alpha), \overline{u}(\alpha)]$, then $-\tilde{u}[\alpha] = [-\overline{u}(\alpha), -\underline{u}(\alpha)]$.*
- *$(\tilde{u} \bigoplus \tilde{v})[\alpha] = [\underline{u}(\alpha) + \underline{v}(\alpha), \overline{u}(\alpha) + \overline{v}(\alpha)]$.*

For any $u \in \mathbb{R}$ we have $\tilde{u}(r) = \begin{cases} 1 & \text{if} \quad r = m, \\ 0 & \text{if} \quad r \neq m, \end{cases}$ and also we can write $\tilde{u} = (u, u)$. Also, we denote the $m \times n$ fuzzy matrix $\tilde{A}$ by $(\underline{A}(\alpha), \overline{A}(\alpha))$ where

$$\underline{A}(\alpha) = (\underline{a_{ij}}(\alpha))_{m\times n}, \overline{A}(\alpha) = (\overline{a_{ij}}(\alpha))_{m\times n}.$$

Definition 2.3. [44] *Let $A = [\underline{a}, \overline{a}]$ and $B = [\underline{b}, \overline{b}]$ be two closed interval in $\mathbb{R}$. We note $A \preceq B$, if and only if, $\underline{a} \leq \underline{b}$ and $\overline{a} \leq \overline{b}$. Also, we write $A \prec B$, if and only if, $A \preceq B$ and $A \neq B$. Equivalently, we observe that $A \prec B$, if and only if,*

$$\begin{cases} \underline{a} < \underline{b}, \\ \overline{a} < \overline{b}, \end{cases} \quad or \quad \begin{cases} \underline{a} \leq \underline{b}, \\ \overline{a} < \overline{b}, \end{cases} \quad or \quad \begin{cases} \underline{a} < \underline{b}, \\ \overline{a} \leq \overline{b}. \end{cases}$$

2.2 Fuzzy optimization

Here, our requirements from the fuzzy optimization are established.

Definition 2.4. [74] *A fuzzy mapping $\tilde{h} : C \to E^1$ defined on a convex subset C in Ω is convex, if and only if,*

$$\tilde{h}((1-\lambda)x + \lambda y) \leq (1-\lambda)\tilde{h}(x) + \lambda\tilde{h}(y),$$

for every $\lambda \in [0, 1]$, $x, y \in C$.

A fuzzy mapping $\tilde{h} : C \to E^1$ defined on a convex subset C in Ω is called strictly convex if,

$$\tilde{h}((1-\lambda)x + \lambda y) < (1-\lambda)\tilde{h}(x) + \lambda\tilde{h}(y), \forall x, y \in C, x \neq y, \lambda \in (0,1).$$

According to the parametric representation of fuzzy number, a fuzzy mapping $\tilde{h}(x)$ can be written as follows:

$$\tilde{h}(x) = \{(\underline{h(x)}(\alpha), \overline{h(x)}(\alpha), \alpha) : \alpha \in [0,1]\}.$$

Theorem 2.5. [78] *Let C be a convex subset in Ω and $\tilde{h} : C \to E^1$ be a fuzzy mapping, then $\tilde{h}$ is convex, if and only if, both of $\underline{h(x)}(\alpha)$ and $\overline{h(x)}(\alpha)$ are convex functions at x for any fixed $\alpha \in [0,1]$.*

Theorem 2.6. [78] *If $\tilde{h}_1, \tilde{h}_2, \ldots, \tilde{h}_m$ are convex fuzzy mappings defined on $C \subseteq \Omega, \lambda_i \geq 0$ $(i = 1, \ldots, m)$, then $\lambda_1\tilde{h}_1 + \lambda_2\tilde{h}_2 + \cdots + \lambda_m\tilde{h}_m$ is a convex fuzzy mappings on C.*

Definition 2.7. [48] *Let $h(x) = [\underline{h}(x), \overline{h}(x)]$ be an interval-valued function on $\Omega \subseteq \mathbb{R}^n$. We consider the following problem*

$$\begin{aligned} \text{minimize} \quad & h(x) = [\underline{h}(x), \overline{h}(x)], \\ \text{subject to} \quad & x \in \Omega, \end{aligned} \tag{2}$$

where the feasible set Ω is a convex subset on $\mathbb{R}^n$. From [79]*, problem* (2) *can be written as the following bi-objective optimization problem*

$$\begin{aligned} \text{minimize} \quad & h(x) = \quad (\underline{h}(x), \overline{h}(x)), \\ \text{subject to} \quad & x \in \Omega. \end{aligned} \tag{3}$$

The image of Ω under h is denoted by $Y := h(\Omega) \subseteq \mathbb{R}^2$ and referred to as the image space. Owing to the lack of a total order on the space $\mathbb{R}^2$, the following partial orders will be used to order it. Given two points $y, \bar{y} \in \mathbb{R}^2$:

$y < \bar{y} \iff y_k < \bar{y_k}$ for all $k = 1, 2$,
$y \leqq \bar{y} \iff y_k \leqslant \bar{y_k}$ for all $k = 1, 2$,
$y \leq \bar{y} \iff y_k \leqslant \bar{y_k}$ for all $k = 1, 2$ and $y \neq \bar{y}$.

Definition 2.8. [48] *A feasible solution $x^* \in \Omega$ is called*

i) *an efficient (a Pareto optimal) solution for problem* (3)*, if there is no other $x \in \Omega$ such that $h(x) \leq h(x^*)$.*

ii) *a weakly efficient (a weakly Pareto optimal) solution for problem* (3)*, if there is no other $x \in \Omega$ such that $h(x) < h(x^*)$.*

In order to obtain Pareto optimal solutions, one widely used approach is the weighted sum method [48], which transforms the bi-objectives into a single one by multiplying each objective with a weight. Given a weighting vector $w = (w_1, w_2)^\top$, problem (3) is converted into

$$\begin{aligned} \text{minimize} \quad & w_1\underline{h}(x) + w_2\overline{h}(x), \\ \text{subject to} \quad & x \in \Omega, \end{aligned} \tag{4}$$

where w_1 and w_2 are positive real numbers and $w_1 + w_2 = 1$.

Theorem 2.9. [15] *If $x^* \in \Omega$ is an optimal solution of the weighting problem* (4) *where either $w_1, w_2 > 0$, or x^* is a unique optimal solution, then x^* is a Pareto optimal solution of the bi-objective program* (3)*.*

Theorem 2.10. [15] *Let the bi-objecive optimization problem* (3) *be convex. If $x^* \in \Omega$ is Pareto optimal, then there exists a weighting vector $w \in W$ which is a solution to the weighting problem* (4)*.*

3 Problem setting of FNLOP

In this paper,we study the following nonlinear programming problem with fuzzy coefficients:

$$\text{minimize} \quad \tilde{f}(x), \tag{5}$$

subject to

$$\begin{cases} \tilde{g}(x) \preceq \tilde{d}, \\ \tilde{A}x \cong \tilde{b}, \end{cases} \tag{6}$$

where $x \in \mathbb{R}^n$, $\tilde{f} : E^n(\mathbb{R}) \to E(\mathbb{R})$, $\tilde{g} = (\tilde{g}_1(x), \tilde{g}_2(x), \ldots, \tilde{g}_m(x))^\top$ is an m-dimensional vector-valued continuous mapping (E^n shows the collection of all fuzzy numbers described in Definition 2.1), $\tilde{A}$ is fuzzy matrix of order $l \times n$, $\tilde{d} = \left(\tilde{d}_1, \tilde{d}_2, \ldots, \tilde{d}_m\right)^\top$ and $\tilde{b} = \left(\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_l\right)^\top$ are m-dimensional and l-dimensional fuzzy vectors. The twice differentiable fuzzy functions $\tilde{f}, \tilde{g}_1, \tilde{g}_2, \ldots, \tilde{g}_m$ are convex in the sense of fuzzy functions (see Definition 2.4). Utilizing Definition 2.7, we can write the problem (5)-(6) as the following interval program

$$\text{minimize} \quad \left[\underline{f}(x)(\alpha), \overline{f}(x)(\alpha)\right], \tag{7}$$

subject to

$$\begin{cases} \left[\underline{g}(x)(\alpha), \overline{g}(x)(\alpha)\right] \leq \left[\underline{d}(\alpha), \overline{d}(\alpha)\right], \\ \left[\underline{A}(\alpha), \overline{A}(\alpha)\right] x = \left[\underline{b}(\alpha), \overline{b}(\alpha)\right], \\ 0 \leq \alpha \leq 1, \end{cases} \tag{8}$$

or it can be reduced as the following bi-objective program

$$\text{minimize} \quad (\underline{f}(x)(\alpha), \overline{f}(x)(\alpha)), \tag{9}$$

subject to

$$\begin{cases} \underline{g}(x)(\alpha) \leq \underline{d}(\alpha), \\ \overline{g}(x)(\alpha) \leq \overline{d}(\alpha), \\ \underline{A}(\alpha)x = \underline{b}(\alpha), \\ \overline{A}(\alpha)x = \overline{b}(\alpha), \\ 0 \leq \alpha \leq 1. \end{cases} \tag{10}$$

The related weighting problem of (9)-(10) can be written as

$$\text{minimize} \quad w_1 \underline{f}(x)(\alpha) + w_2 \overline{f}(x)(\alpha), \tag{11}$$

subject to

$$\begin{cases} \underline{g}(x)(\alpha) \leq \underline{d}(\alpha), \\ \overline{g}(x)(\alpha) \leq \overline{d}(\alpha), \\ \underline{A}(\alpha)x = \underline{b}(\alpha), \\ \overline{A}(\alpha)x = \overline{b}(\alpha), \\ 0 \leq \alpha \leq 1, \end{cases} \tag{12}$$

where $w_1 + w_2 = 1$ and $w_1, w_2 \geq 0$.

$$\begin{cases} f(x)(\alpha) = \begin{bmatrix} \underline{f}(x)(\alpha) \\ \overline{f}(x)(\alpha) \end{bmatrix}, \nabla f(x)(\alpha) = \begin{bmatrix} \nabla \underline{f}(x)(\alpha) \\ \nabla \overline{f}(x)(\alpha) \end{bmatrix}, g(x)(\alpha) = \begin{bmatrix} \underline{g}(x)(\alpha) \\ \overline{g}(x)(\alpha) \end{bmatrix}, \\ \nabla g(x)(\alpha) = \begin{bmatrix} \nabla \underline{g}(x)(\alpha) \\ \nabla \overline{g}(x)(\alpha) \end{bmatrix}, d(\alpha) = \begin{bmatrix} \underline{d}(\alpha) \\ \overline{d}(\alpha) \end{bmatrix}, A(\alpha) = \begin{bmatrix} \underline{A}(\alpha) \\ \overline{A}(\alpha) \end{bmatrix}, b(\alpha) = \begin{bmatrix} \underline{b}(\alpha) \\ \overline{b}(\alpha) \end{bmatrix}, \\ \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}, \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}, W = \{w \in \mathbb{R}^2 | w_1 + w_2 = 1, w \geq 0\}. \end{cases} \tag{13}$$

Then, problem (11)-(12) can be written as

$$\text{minimize} \quad w^\top f(x)(\alpha), \tag{14}$$

subject to

$$\begin{cases} g(x)(\alpha) \leq d(\alpha), \\ A(\alpha)x = b(\alpha), \\ 0 \leq \alpha \leq 1, \ w \in W. \end{cases} \tag{15}$$

The KKT optimality conditions for the problem (14)-(15) is

$$\begin{cases} \lambda^* \geq 0, \quad g(x^*)(\alpha) \leq d(\alpha), \quad {\lambda^*}^\top (g(x^*)(\alpha) - d(\alpha)) = 0, \\ w^\top \nabla f(x^*)(\alpha) + \nabla g(x^*)(\alpha)^\top \lambda^* + A(\alpha)^\top \mu^* = 0, \\ A(\alpha)x^* = b(\alpha), \quad 0 \leq \alpha \leq 1, \ w \in W. \end{cases} \tag{16}$$

x^* is called a KKT point of (14)-(15) and a pair $({\lambda^*}^\top, {\mu^*}^\top)^\top$ is called the Lagrangian multiplier vector corresponding to x^*. Moreover, if $\underline{f}$, $\overline{f}$, $\underline{g}$ and $\overline{g}$ are all convex, then x^* is an optimal solution of (14)-(15), if and only if x^* is a KKT point of (14)-(15).

4 The neural network scheme

In this section, we will construct a neural network for solving (14)-(15). Throughout this paper, we assume $0 \leq \alpha \leq 1$, $w \in W$.

Theorem 4.1. [22] *$x^* \in S$ is an optimal solution of* (14)-(15)*, if and only if there exists $u^* = (u_1^*, u_2^*, \ldots, u_{2m}^*)^\top \in \mathbb{R}^{2m}$ and $v^* = (v_1^*, v_2^*, \ldots, v_{2l}^*)^\top \in \mathbb{R}^{2l}$ such that $({x^*}^\top, {u^*}^\top, {v^*}^\top)^\top$ satisfies the following Karush–Kuhn–Tucker (*K.K.T*) conditions:*

$$\begin{cases} (x - x^*)^\top (\sum_{k=1}^K w^\top \nabla f(x^*)(\alpha) + \nabla g(x^*)(\alpha)^\top u^* - A(\alpha)^\top v^*) \geq 0, \quad \forall x \in \Omega, \\ u_j^*(g_j(x^*)(\alpha) - d_j(\alpha)) = 0, \quad u_j^* \geq 0, \quad g_j(x^*)(\alpha) - d_j(\alpha) \leq 0, \quad j = 1, ..., 2m, \ A(\alpha)x^* - b(\alpha) = 0. \end{cases} \tag{17}$$

Lemma 4.2. [83] *x^* is an optimal solution of* (14)-(15) *if and only if there exists $u^* = (u_1^*, u_2^*, \ldots, u_{2m}^*)^\top \in \mathbb{R}^{2m}$ and $v^* = (v_1^*, v_2^*, \ldots, v_{2l}^*) \in \mathbb{R}^{2l}$ such that:*

$$\begin{cases} x^* - P_\Omega[x^* - w^\top \nabla f(x^*)(\alpha) - \nabla g(x^*)(\alpha)^\top u^* + A(\alpha)^\top v^*] = 0, \\ u^* - [u^* + g(x^*)(\alpha) - d(\alpha)]^+ = 0, \quad A(\alpha)x^* - b(\alpha) = 0, \end{cases} \tag{18}$$

where $P_\Omega(x) = [(x_1)_{l_1}^{h_1}, (x_2)_{l_2}^{h_2}, \ldots, (x_n)_{l_n}^{h_n}] \in \mathbb{R}^n$, $(x_i)_{l_i}^{h_i} = \min\{h_i, \max\{x_i, h_i\}\}$ and $(x_i)^+ = \max\{0, x_i\}$.

Lemma 4.3. [31] *Assume that the set $\Omega \subset \mathbb{R}^n$ is a closed convex set. Then*

$$(V - P_\Omega(V))^\top (P_\Omega(V) - U) \geq 0, \quad V \in \mathbb{R}^n, U \in \Omega, \tag{19}$$

$$\|P_\Omega(U) - P_\Omega(V)\| \leq \|U - V\|, \quad U, V \in \mathbb{R}^n. \tag{20}$$

For simplicity, we denote

$$\begin{aligned} &z(t) = (x^\top, u^\top, v^\top)^\top \in \mathbb{R}^{n+2m+2l}, \\ &D = \Omega \times \mathbb{R}^{2m+2l}, \\ &\breve{u}(\alpha) = [u + g(x)(\alpha) - d(\alpha)]^+, \\ &\breve{x}(\alpha) = P_\Omega[x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top u + A(\alpha)^\top (v - A(\alpha)x + b(\alpha))], \\ &D^* = \{z \in \mathbb{R}^{n+2m+2l} | z \text{ solves } (18)\}. \end{aligned}$$

Based on the above results and by an extension of the neural network in [22] we introduce the following neural network to solve (14)-(15):

$$\frac{dz}{dt} = \frac{d}{dt}\begin{pmatrix} x \\ u \\ v \end{pmatrix} = -\tau F(z) = -\tau \begin{pmatrix} 2(x - \breve{x}(\alpha)) \\ u - \breve{u}(\alpha) \\ A(\alpha)x - b(\alpha) \end{pmatrix}, \tag{21}$$

here τ is a scale parameter and indicates the convergence rate of the neural network (21). For simplicity of our analysis, we let $\tau = 1$.

Let D^e be the equilibrium point set of the neural network (21). From Lemma 4.2 we have that $x \in D^e$ is an equilibrium point of the neural network (21) if and only if $x \in D^*$ is an optimal solution of (14)-(15). That is, $D^* = D^e$.

In the following, we give a more technical description of the proposed neural network algorithm whose framework is shown in Algorithm 1. The procedures start with choosing an arbitrary initial vector. Then iterative procedures are controlled by the principal while-loop of Algorithm 1. In each iteration, calculate the right side of system (21), update initial vector, calculation of error and stopping criteria, must be scheduled. The pseudo code that explains the implementation of the proposed algorithm is stated as follows:

Algorithm 1: The working of the proposed neural network algorithm.

The working of the proposed neural network algorithm.
Step 1: Initialization
Arbitrary choose initial vector $x \in \mathbb{R}^n$ $u \in \mathbb{R}^{2m}$, and $v \in \mathbb{R}^{2l}$, $\Delta t > 0$, $t = 0$ and error $\epsilon = 10^{-9}$, while $t \leq T$ (maximum iteration).
Step 2: With an arbitrary value of τ, calculate

$$\begin{aligned}\nu(t) &= 2\tau(-x + \breve{x}(\alpha)),\\ \vartheta(t) &= \tau\left(-u + \breve{u}(\alpha)\right),\\ \iota(t) &= \tau(-A(\alpha)x + b(\alpha)).\end{aligned}$$

Step 3: Update the initial vector

$$\begin{aligned}x(t + \Delta t) &= x(t) + \Delta t\nu(t),\\ u(t + \Delta t) &= u(t) + \Delta t\vartheta(t),\\ v(t + \Delta t) &= v(t) + \Delta t\iota(t).\end{aligned}$$

Step 4: Calculation

$$r(t) = (||\nu(t)||^2 + ||\vartheta(t)||^2 + ||\iota(t)||^2).$$

Step 5: Stopping criteria: if $r(t) < \epsilon$; else $t = t + 1$.
end.
Output:

$$x(t + \Delta t),\ u(t + \Delta t), v(t + \Delta t).$$

5 Stability analysis

In this section, we will establish some stability and convergence properties for (21). We now explain our main results as follows.

Theorem 5.1. *For any initial point* $z(t_0) = (x(t_0)^\top, u(t_0)^\top, v(t_0)^\top)^\top \in \mathbb{R}^{n+2m+2l}$, *there exists a unique continuous solution* $z(t) = (x(t)^\top, u(t)^\top, v(t)^\top)^\top$ *for system* (21). *Moreover,* $x(t) \in \Omega$ *and* $u(t) \geq 0$, *provided that* $x(t_0) \in \Omega$ *and* $u(t_0) \geq 0$.

Proof. The projection mappings $P_\Omega(.)$ and $(.)^+$ are non-expansive. Since $\nabla f(x)(\alpha)$ and $\nabla g_j(x)(\alpha)$, $j = 1, 2, \ldots, 2m$ are continuously differentiable on an open convex set $\mathcal{E} \subseteq \mathbb{R}^n$ including Ω, then $x - P_\Omega[x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top u + A(\alpha)^\top(v - A(\alpha)x + b(\alpha))]$, $u - [u + g(x)(\alpha) - d(\alpha)]^+$ and $A(\alpha)x - b(\alpha)$ are locally Lipschitz continuous. According to the local existence of ordinary differential equations, the initial value problem of the system (21) has a unique solution of (21) on $[t_0, T)$ as is its maximal interval of existence. Moreover, it can be seen in the proof of Theorem 5.4 that the solution $z(t)$ is bounded and thus $T \to +\infty$.

Let the initial point $x_0 = x(t_0) \in \Omega$ and $u_0 = u(t_0) \geq 0$. Since

$$\begin{cases}\frac{dx}{dt} + x = P_\Omega[x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - A(\alpha)^\top(v - A(\alpha)x + b(\alpha))],\\ \frac{du}{dt} + u = [u + g(x)(\alpha)]^+.\end{cases} \tag{22}$$

We have

$$\begin{cases} \int_{t_0}^{t} (\frac{dx}{dt} + x)e^t dt = \int_{t_0}^{t} P_\Omega[x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - A(\alpha)^\top (v - A(\alpha)x + b(\alpha))]e^t dt, \\ \int_{t_0}^{t} (\frac{du}{dt} + u)e^t dt = \int_{t_0}^{t} [u + g(x)(\alpha) - d(\alpha)]^+ e^t dt. \end{cases} \tag{23}$$

Then

$$\begin{cases} x(t) = e^{-(t-t_0)}x_0 + e^{-t}\int_{t_0}^{t} e^t P_\Omega[x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - A(\alpha)^\top (v - A(\alpha)x + b(\alpha))]dt, \\ u(t) = e^{-(t-t_0)}u_0 + e^{-t}\int_{t_0}^{t} e^t [u + g(x)(\alpha) - d(\alpha)]^+ dt. \end{cases} \tag{24}$$

By the integration mean value theorem, we have:

$$\begin{cases} x(t) = e^{-(t-t_0)}x_0 + (1 - e^{-(t-t_0)})P_\Omega[\hat{x} - w^\top \nabla f(\hat{x})(\alpha) - \nabla g(\hat{x})(\alpha)^\top \hat{u}(\alpha) - A(\alpha)^\top (\hat{v} - A(\alpha)\hat{x} + b(\alpha))], \\ u(t) = e^{-(t-t_0)}u_0 + (1 - e^{-(t-t_0)})[\hat{u} + g(\hat{x})(\alpha) - d(\alpha)]^+, \end{cases} \tag{25}$$

It follows that $x(t) \in \Omega$ and $u(t) \geq 0$ from $x(t_0) \in \Omega$ and $u(t_0) \geq 0$. This completes the proof. □

Remark 5.2. *The results of Theorem 5.1 indicate that neural network* (21) *is well defined. In the proof of Theorem 5.1, we assume* $(x_0^\top, u_0^\top, v_0^\top)^\top \in \Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$*, that is to say, the initial point should be feasible. But if* $(x_0^\top, u_0^\top, v_0^\top)^\top)^\top \notin \Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$*, as* $t \to +\infty$*,* $(x(t, x_0)^\top, u(t, x_0)^\top, v(t, v_0)^\top)^\top$ *will eventually approach* $\Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$ *by* (25)*. So there exists a moment* t_0*,* $t > t_0$ *such that* $(x(t, x_0)^\top, u(t, x_0)^\top, v(t, v_0)^\top)^\top$ *can be regarded as a feasible point (or approximate) and* $(x(t, x_0)^\top, u(t, x_0)^\top, v(t, v_0)^\top)^\top$ *will stay in* $\Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$ *from then on.*

Lemma 5.3. *Let* $(x^*, u^*, v^*)^\top \in \Omega^e$ *be an equilibrium point of* (21) *and*

$$V(z, z^*) = \Xi(z) - \Xi(z^*) - (z - z^*)\nabla\Xi(z^*) + \frac{1}{2}\|z - z^*\|^2, \tag{26}$$

where

$$\Xi(z) = w^\top f(x)(\alpha) + \frac{1}{2}(\|\breve{u}(\alpha)\|^2 + \|v - A(\alpha)x + b(\alpha)\|^2). \tag{27}$$

Then for $0 \leq \alpha \leq 1,\ w \in W$*,*

i) $V(z, z^*)$ *is convex on* D *and continuously differentiable on* $E \times \mathbb{R}^{2m+2l}$*.*

ii)

$$V(z, z^*) \geq \frac{1}{2}\|z - z^*\|^2,\ \forall z \in D. \tag{28}$$

Proof. i) It is easy to clarify that $\|v - A(\alpha)x + b(\alpha)\|^2$ is differentially convex on $\mathbb{R}^{n+2l}$. From [4], $\|\breve{u}(\alpha)\|^2$ is convex differentiable on $\Omega \times \mathbb{R}^{2m}$ and continuously differentiable on $\mathcal{E} \times \mathbb{R}^{2m}$. By the differentiable convexity of $f(x)(\alpha)$ we see that $\Xi(z)$ defined in (27) is also convex on D and continuously differentiable on $\mathcal{E} \times \mathbb{R}^{\in \Updownarrow} \times \mathbb{R}^{\in \updownarrow}$. Thus i) is obtained.

ii) Using the analysis of i), it follows that

$$\Xi(z) \geq \Xi(z^*) + (z - z^*)^\top \nabla\Xi(z^*),\ \forall z \in D,$$

By (26) we get

$$V(z, z^*) \geq \frac{1}{2}\|z - z^*\|^2, \forall z \in D. \tag{29}$$

□

Theorem 5.4. *The neural network model* (21) *is stable in the sense of Lyapunov. Moreover, the solution trajectory of neural network* (21) *is extendable to the global existence.*

Proof. By Theorem 5.1, $\forall (x_0^\top, u_0^\top, v_0^\top)^\top \in \Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$, there exists a unique continuous solution $(x(t)^\top, u(t)^\top, v(t)^\top)^\top \subseteq \Omega \times \mathbb{R}_+^{2m} \times \mathbb{R}^{2l}$ for system (21).

Define a Lyapunov function as (26). Noting that $\|\breve{u}(\alpha)\|^2 = \Sigma_{j=1}^{2m}[(u_j + g_j(x)(\alpha) - d_j(\alpha))^+]^2$ and

$$[(u_j + g_j(x)(\alpha) - d_j(\alpha))^+]^2 = \begin{cases} [(u_j + g_j(x)(\alpha) - d_j(\alpha))]^2, & u_j + g_j(x)(\alpha) - d_j(\alpha) \geq 0, \\ 0, & o.w. \end{cases} \tag{30}$$

We have

$$\nabla\|\breve{u}(\alpha)\|^2 = \nabla(\sum_{j=1}^{2m}[(u_j + g_j(x)(\alpha) - d_j(\alpha))^+]^2) = \begin{pmatrix} 2\nabla g(x)(\alpha)^\top \breve{u}(\alpha) \\ 2\breve{u}(\alpha) \end{pmatrix}. \tag{31}$$

Using i) in Lemma 5.3 and (26), we have

$$\nabla\Xi(z) = \begin{pmatrix} w^\top \nabla f(x)(\alpha) + \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - A(\alpha)^\top (v - A(\alpha)x + b(\alpha)) \\ \breve{u}(\alpha) \\ v - A(\alpha)x + b(\alpha) \end{pmatrix},$$

and

$$\nabla V(z, z^*) = z - z^* + \nabla\Xi(z) - \nabla\Xi(z^*).$$

Thus, by (18),(21) and $\breve{u}(\alpha) - u = g(x)(\alpha) - d(\alpha) + (-u - g(x)(\alpha) + d(\alpha))^+$, $\forall z \in D$ one has

$$\begin{aligned}
&\frac{dV(z,z^*)}{dt} = \frac{\partial V(z,z^*)}{\partial x}\frac{dx}{dt} + \frac{\partial V(z,z^*)}{\partial u}\frac{du}{dt} + \frac{\partial V(z,z^*)}{\partial v}\frac{dv}{dt} \\
&= 2(x - \breve{x}(\alpha))^\top [x - x^* - w^\top \nabla f(x^*)(\alpha) - \nabla g(x^*)(\alpha)^\top u^* + A(\alpha)^\top v^* \\
&\quad + w^\top \nabla f(x)(\alpha) + \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - A(\alpha)^\top (v - A(\alpha)x + b(\alpha))] - \|A(\alpha)x - b(\alpha)\|^2 \\
&\quad + 2(v - v^*)^\top (A(\alpha)x - b(\alpha)) + (u - \breve{u}(\alpha))^\top (u - 2u^* + \breve{u}(\alpha)) \\
&= 2(\breve{x}(\alpha) - x^*)^\top [x - \breve{x}(\alpha) + w^\top \nabla f(x^*)(\alpha) + \nabla g(x^*)(\alpha)^\top u^* \\
&\quad - A(\alpha)^\top v^* - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) + A(\alpha)^\top (v - A(\alpha)x + b(\alpha))] + 2\|x - \breve{x}(\alpha)\|^2 \\
&\quad + 2(x - x^*)^\top [w^\top \nabla f(x)(\alpha) + \nabla g(x)(\alpha)^\top \breve{u}(\alpha) - w^\top \nabla f(x^*)(\alpha) - \nabla g(x^*)(\alpha)^\top u^*] \\
&\quad + \|A(\alpha)x - b(\alpha)\|^2 + \|u - \breve{u}(\alpha)\|^2 + 2(\breve{u}(\alpha) - u^*)^\top [-g(x)(\alpha) + d(\alpha) - (-g(x)(\alpha) + d(\alpha) - u)^+] \\
&= 2(\breve{x}(\alpha) - x^*)^\top [x - \breve{x}(\alpha) - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) + A(\alpha)^\top (v - A(\alpha)x + b(\alpha))] \\
&\quad + 2(\breve{x}(\alpha) - x^*)^\top [w^\top \nabla f(x^*)(\alpha) + \nabla g(x^*)(\alpha)^\top u^* - A(\alpha)^\top v^*] + 2\|x - \breve{x}(\alpha)\|^2 \\
&\quad + \|u - \breve{u}(\alpha)\|^2 + \|A(\alpha)x - b(\alpha)\|^2 + 2(x - x^*)^\top [w^\top \nabla f(x)(\alpha) - w^\top \nabla f(x^*)(\alpha)] \\
&\quad + 2\breve{u}(\alpha)^\top [-g(x)(\alpha) + d(\alpha) - (-g(x)(\alpha) + d(\alpha) - u)^+ + \nabla g(x)(\alpha)^\top (x - x^*)] \\
&\quad - 2{u^*}^\top [-g(x)(\alpha) + d(\alpha) - (-g(x)(\alpha) + d(\alpha) - u)^+ + \nabla g(x^*)(\alpha)^\top (x - x^*)] \\
&\geq 2\|x - \breve{x}(\alpha)\|^2 + 2(x - x^*)^\top [w^\top \nabla f(x)(\alpha) - w^\top \nabla f(x^*)(\alpha)] - 2\breve{u}(\alpha)^\top (g(x^*)(\alpha) - d(\alpha)) \\
&\quad + \|u - \breve{u}(\alpha)\|^2 + \|A(\alpha)x - b(\alpha)\|^2 + 2\breve{u}(\alpha)^\top [g(x^*)(\alpha) - g(x)(\alpha) + \nabla g(x)(\alpha)^\top (x - x^*)] \\
&\quad - 2{u^*}^\top [g(x^*)(\alpha) - g(x)(\alpha) + \nabla g(x^*)(\alpha)^\top (x - x^*)],
\end{aligned} \tag{32}$$

In the above calculations we used

$$\begin{aligned}
&[(-u - g(x)(\alpha) + d(\alpha))^+]^\top \breve{u}(\alpha) = 0, \\
&(g(x^*)(\alpha) - d(\alpha))^\top u^* = 0, \\
&[(-u - g(x)(\alpha) + d(\alpha))^+]^\top u^* \geq 0, \\
&(g(x^*)(\alpha) - d(\alpha))^\top \breve{u}(\alpha) \leq 0.
\end{aligned}$$

Moreover, in the inequality of Lemma 4.3, let $V = x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) + A(\alpha)^\top (v - A(\alpha)x + b(\alpha))$ and $U = x^*$, we obtain:

$$(x - w^\top \nabla f(x)(\alpha) - \nabla g(x)(\alpha)^\top \breve{u}(\alpha) + A(\alpha)^\top (v - A(\alpha)x + b(\alpha)) - \breve{x}(\alpha))^\top (\breve{x}(\alpha) - x^*) \geq 0. \tag{33}$$

From the differentiable convexities of $\underline{f}(x)(\alpha), \overline{f}(x)(\alpha), \underline{g}_j(x)(\alpha), \overline{g}_j(x)(\alpha),\ j = 1, 2, ..., m$, $\forall x \in \Omega$, we have:

$$\begin{cases}
\left[w^\top \nabla f(x)(\alpha) - w^\top \nabla f(x^*)(\alpha)\right]^\top (x - x^*) \geq 0, \\
g(x^*)(\alpha) - g(x)(\alpha) - \nabla g(x)(\alpha)^\top (x^* - x) \geq 0, \\
g(x)(\alpha) - g(x^*)(\alpha) - \nabla g(x^*)(\alpha)^\top (x - x^*) \geq 0.
\end{cases} \tag{34}$$

Substituting (17), (33) and (34) into (32), $\forall t \geq t_0$, one has

$$\frac{dV(z,z^*)}{dt} \leq -2\|x(t)-\breve{x}(\alpha)\|^2 - \|A(\alpha)x(t)-b(\alpha)\|^2 - \|u(t)-\breve{u}(\alpha)\|^2 \leq 0. \tag{35}$$

This means neural network (21) is globally stable in the sense of Lyapunov.

From (35) we know that V is a non-increasing function with respect to t, so

$$\frac{1}{2}\|z(t)-z^*\|^2 \leq V(z,z^*) \leq V(z(t_0),z^*), \quad \forall t_0 \leq t \leq T, \tag{36}$$

This shows the state trajectory of neural network (21) is bounded. Thus $T=+\infty$. □

Theorem 5.5. *Assume that $\nabla f(x)(\alpha)$ and $\nabla g_j(x)(\alpha)$, $j=1,2,\ldots,2m$ are continuously differentiable on an open convex set $\mathcal{E} \subseteq \mathbb{R}^n$ including Ω. Then for $0 \leq \alpha \leq 1$, $w \in W$, the solution trajectory of the neural network* (21) *is globally convergent to an optimal point of the problem* (14)-(15). *In particular, neural network* (21) *is asymptotically stable when D^* has only a point.*

Proof. From (28), the level set $L(z_0)=\{z \in D | V(z,z^*) \leq V(z(t_0),z^*)\}$ is bounded. Thus, there exists a convergent subsequence

$$\{z(t_k)|t_0 < t_1 < ... < t_k < t_{k+1}\}, \text{ and } t_k \to \infty \text{ as } k \to \infty,$$

such that

$$\lim_{k\to\infty} z(t_k) = \breve{z}, \tag{37}$$

where $\breve{z}$ satisfies

$$\frac{dV(z,z^*)}{dt} = 0,$$

which indicates that $\breve{z}$ is an ω-limit point of $\{z(t)|t \geq t_0\}$. Using the LaSalle Invariant Set Theorem [49], one has that $\{z(t) \to M\}$ as $t \to \infty$, where M is the largest invariant set in $\Lambda = \{z(t)|\frac{dV(z,z^*)}{dt}=0\}$. From (21) and (35), it follows that $\frac{dx}{dt}=\frac{du}{dt}=\frac{dv}{dt}=0 \Leftrightarrow \frac{dV(z,z^*)}{dt}=0$. Thus $\breve{z} \in D^*$ by $M \subseteq \Lambda \subseteq D^*$.

Substituting $z^*=\hat{z}$ in (26), similar to the analysis above, we can prove that the function $V[z(t),\hat{z}]$ is monotonically non-increasing in $[t_0,+\infty)$, and

$$V(z,\hat{z}) \geq \|z-\hat{z}\|^2/2, \ \forall z \in D$$

.

From the continuity of $V(z,\hat{z})$ and (37), $\forall \epsilon > 0$, there exist a t_q such that

$$V[z(t),\hat{z}] \leq \frac{1}{2}\epsilon^2, \quad when\ t \geq t_q. \tag{38}$$

Thus

$$\|z(t)-\hat{z}\| \leq \sqrt{2V[z(t),\hat{z}]} \leq \sqrt{2V[z(t_q),\hat{z}]} \leq \epsilon, \quad when\ t \geq t_q.$$

That is,$\lim_{k\to\infty} z(t) = \hat{z}$. So, the solution trajectory of the neural network (21) is globally convergent to an equilibrium point $\hat{z}$ is also an optimal solution of (14)-(15). In particular, if $D^*=\{z^*\}$, then for each $x_0 \in \Omega$, v free in sign, and $u_0 \geq 0$, the solution z with the initial point $z(t_0)=z^0$ will approach z^* by the analysis above. That is, the neural network (21) is globally asymptotically stable.

□

Theorem 5.6. *The convergence rate of the neural network in* (21) *increases as τ increases.*

Proof. From (35) and (36), one can see that for any $\tau > 0$ in (21) we get

$$\|z(t)-z^*\|^2 \leq 2V(z,z^*) \leq 2V(z(t_0),z^*),$$
$$+2\tau \int_0^t \{-2\|x(s)-\breve{x}(\alpha)\|^2 - \|A(\alpha)x(s)-b(\alpha)\|^2 - \|u(s)-\breve{u}(\alpha)\|^2\}ds.$$

Therefore, the convergence rate of the trajectory $z(t)$ increases as τ increases.

□

6 Comparison with some neural networks

In order to see how well the presented neural network (21) can be applied to solve (14)-(15), we compare it with some existing neural network models. First, let us consider the following problem

$$\text{minimize} \quad w^\top f(x)(\alpha), \tag{39}$$

subject to

$$\begin{cases} g(x)(\alpha) \leq d(\alpha), \\ A(\alpha)x = b(\alpha), \quad x \in \Theta, \end{cases} \tag{40}$$

where $\Theta = \{x \in \mathbb{R}^n : \ l_i \leq x_i \leq h_i, \ i = 1, 2, \ldots, n\}$, and some h_i (or $-l_j$) could be $+\infty$, $f(x)(\alpha), g(x)(\alpha), d(\alpha), A(\alpha), b(\alpha)$, are defined in (13) and the functions $\underline{f}(x)(\alpha)$, $\overline{f}(x)(\alpha)$ and $g_j(x)(\alpha)$ $(j = 1, \ldots, 2m)$ are assumed to be convex on Θ and twice continuous differentiable on an open convex set $\mathcal{E} \subseteq \mathbb{R}^n$ including Θ. By extending the globally projected dynamic systems [21] to constrained optimization problems by Xia in [81], a neural network can be utilized for solving (39)–(40) as

$$\frac{dx}{dt} = \left(P_\Theta(x - (w^\top \nabla f(x)(\alpha) + \nabla g(x)(\alpha)^\top \lambda - A(\alpha)^\top \mu)) - x\right), \tag{41}$$

$$\frac{d\lambda}{dt} = \left(-\lambda + (\lambda + g(x)(\alpha) - d(\alpha))^+\right), \tag{42}$$

$$\frac{d\mu}{dt} = -A(\alpha)x + b(\alpha), \tag{43}$$

where $\Theta \subset \mathbb{R}^n$ is a closed convex set and $P_\Theta : \mathbb{R}^n \to \Theta$ is a projection operator [22] defined by

$$P_\Theta(x_i) = \begin{cases} l_i, & x_i < l_i, \\ x_i, & l_i \leq x_i \leq h_i, \\ h_i, & x_i > h_i. \end{cases}$$

It is well known that the asymptotical stability of the dynamic model (41) – (43) for a monotone and a symmetric mapping could not be guaranteed [82]. Thus, the system described by (41) – (43) cannot solve problem (39) – (40) in some cases. For instance, one can see Examples 7.1-7.2 in this manuscript.

The proposed model in [52] can be used for constructing the following neural network for solving (14)-(15) as

$$\frac{d(y(t))}{dt} = -\nabla E_2(y(t)), \tag{44}$$

$$y(0) = y_0, \quad y(t) = (x(t)^\top, \lambda(t)^\top, \mu(t)^\top)^\top, \tag{45}$$

where

$$E_2(y) = \frac{1}{2}\|\rho(y)\|^2, \tag{46}$$

$$\rho(y) = \begin{bmatrix} w^\top \nabla f(x)(\alpha) + A(\alpha)^\top \mu + \nabla g(x)(\alpha)^\top \lambda \\ \phi^\varepsilon_{\mathrm{FB}}\left(\lambda, -g(x)(\alpha) + d(\alpha)\right) \\ b(\alpha) - A(\alpha)x \end{bmatrix}, \tag{47}$$

and

$$\phi^\varepsilon_{\mathrm{FB}}(a, b) = (a + b) - \sqrt{a^2 + b^2 + \varepsilon}, \quad \varepsilon \to 0_+.$$

Using [51], under the condition that $\underline{f}(x)(\alpha)$ and $\overline{f}(x)(\alpha)$ are strictly convex and $g_j(x)(\alpha)(j = 1, ..., 2m)$ is convex or $\underline{f}(x)(\alpha)$ and $\overline{f}(x)(\alpha)$ are convex and $g_j(x)(\alpha)(j = 1, ..., 2m)$ is strictly convex, a lagrangian network model can be constructed for solving (14)-(15) as

$$\frac{dy}{dt} = \Phi(y), \tag{48}$$

$$y(t_0) = y_0, \quad y(t) = (x(t)^\top, \lambda(t)^\top, \mu(t)^\top)^\top, \quad \lambda(t_0) > 0, \tag{49}$$

where

$$\Phi(y) = \begin{bmatrix} -\left(w^\top \nabla f(x)(\alpha) + \frac{1}{2}\nabla g(x)(\alpha)^\top \lambda^2 + A(\alpha)^\top \mu\right) \\ \operatorname{diag}(\lambda_1, \ldots, \lambda_{2m})(g(x)(\alpha) - d(\alpha)) \\ A(\alpha)x - b(\alpha) \end{bmatrix}.$$

The convergence of the models (44)-(45) and (48)-(49) is guaranteed under the condition that $w^\top \nabla^2 f(x)(\alpha) + \sum_{j=1}^{2m} \nabla^2 g_j(x)(\alpha)\lambda_j$ is positive definite matrix for all $x \in \mathbb{R}^n$ and $\lambda_j \geq 0$. Thus, these models can not solve fuzzy linear programming problems which limits their application. Moreover, our neural network can converge to the optimal solution under the condition of positive semi-definite matrix $w^\top \nabla^2 f(x)(\alpha) + \sum_{j=1}^{2m} \nabla^2 g_j(x)(\alpha)\lambda_j$ for all $x \in \mathbb{R}^n$ and $\lambda_j \geq 0$. For instance, one can see Examples 7.1-7.2 in this manuscript.

7 Numerical simulations

In order to demonstrate the feasibility of (21), we consider several examples. The simulation is conducted on Matlab R2022.

Example 7.1. [7] *A very simple example illustrated in the following*

$$\begin{aligned} \text{maximize} \quad & \tilde{4}x \\ \text{subject to} \quad & \tilde{1}x \leq \tilde{2}, \\ & x \geq 0, \end{aligned} \tag{50}$$

where,

$$\tilde{1} = (0, 1, 2),\ \tilde{2} = (1, 2, 3),\ \tilde{4} = (3, 4, 5).$$

Simulation results show that the state trajectory $x(t)$ of the model (21) *with $\alpha = 0.35$, $w_1 = 0.35$, $w_2 = 0.65$ converges to the unique optimal solution $x^* = 1.606$ as seen in Figure 1. To make a comparison, we solve the above problem via the neural networks* (41)-(43) *and* (48)-(49)*. Figure 2 illustrates the transient behavior of $x(t)$ based on* (41)-*(43). Figure 3 also depicts the transient behavior of $x(t)$ based on* (48)-(49)*. It is clearly shown that these two models are not stable, and naturally they are not suitable for solving the optimization problem* (50) *with triangular coefficients.*

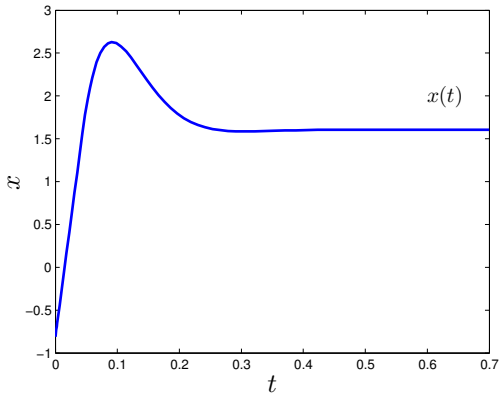


Figure 1: Transient behaviors of x with $w_1 = 0.35, w_2 = 0.65, \alpha = 0.35$ of the neural network (21) in Example 7.1.

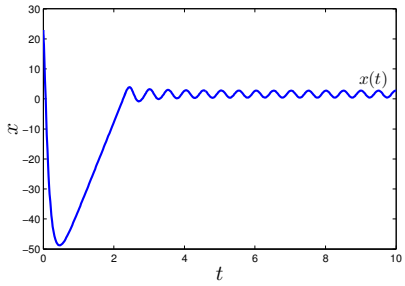


Figure 2: Divergent behavior of the system (41)–(43) in Example 7.1.

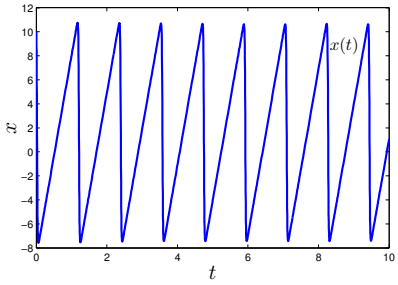


Figure 3: Divergent behavior of the system (48) – (49) in Example 7.1.

Example 7.2. [8] *Consider the following fuzzy linear programming problem*

$$\begin{aligned} \text{maximize} \quad & 15x_1 + 28x_2, \\ \text{subject to} \quad & \tilde{2}x_1 + \tilde{3}x_2 \leq \tilde{100}, \\ & \tilde{4}x_1 - \tilde{2}x_2 \leq \tilde{70}, \\ & x_1, x_2 \geq 0, \end{aligned} \tag{51}$$

where,

$$\tilde{3} = (2,3,5), \tilde{2} = (0,2,4), \tilde{4} = (1,4,5), -\tilde{2} = (-3,-2,-1), \tilde{70} = (60,70,100), \tilde{100} = (80,100,130).$$

Simulation results show that the trajectories of the model (21) *with* $w_1 = 0.75$, $w_2 = 0.25$, $\alpha = 0.45$ *converge to the unique optimal solution* $x^* = (0.00, 30.08)^\top$ *as shown in Figure 4. To make a comparison, we solve the above problem via the neural networks via the neural networks* (41)-(43) *and* (48)-(49). *Figure 5 illustrates the transient behavior of* $(x_1(t), x_2(t))^\top$ *based on* (41) - (43). *Figure 6 also depicts the transient behavior of* $(x_1(t), x_2(t))^\top$ *based on* (48)-(49). *It is clear that these two models are not stable, and naturally they can not solve the optimization problem* (51) *with triangular coefficients.*

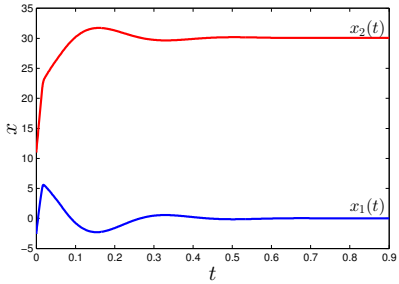


Figure 4: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.75, w_2 = 0.25, \alpha = 0.45$ of the neural network (21) in Example 7.2.

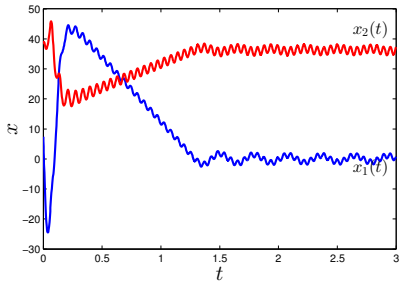


Figure 5: Divergent behavior of the system (41)–(43) in Example 7.2.

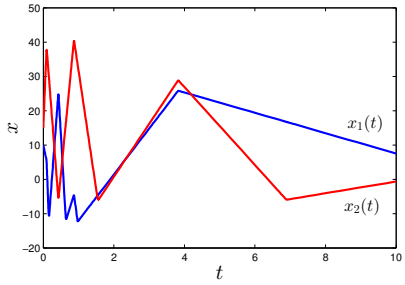


Figure 6: Divergent behavior of the system (48) – (49) in Example 7.2.

Example 7.3. [43] *(Portfolio Selection) An investor has 5000 dollar and two potential investments. Let x_j for $j = 1$ and $j = 2$ denote his allocation to investment j in thousands of dollars. From historical data, investments 1 and 2 have an expected annual return of 20 and 16 percent, respectively. Also, the total risk involved with investments 1 and 2, as measured by the variance of total return, is given by $2x_1^2 + x_2^2 + (x_1 + x_2)^2$, so that risk increases with total investment and with the amount of each individual investment. The investor would like to maximize his expected return and at the same time minimize his risk. Clearly, both of these objectives cannot, in general, be satisfied simultaneously. There are several possible approaches. For example, he can minimize risk subject to a constraint imposing a lower bound on expected return. Alternatively, expected return and risk can be combined in an objective function, to give the model:*

$$\begin{aligned} \text{minimize} \quad & -\tilde{20}x_1 - \tilde{16}x_2 + \theta[\tilde{2}x_1^2 + \tilde{1}x_2^2 + (\tilde{1}x_1 + \tilde{1}x_2)^2], \\ \text{subject to} \quad & \tilde{1}x_1 + \tilde{1}x_2 \leq \tilde{5}, \\ & \tilde{1}x_1, \tilde{1}x_2 \geq \tilde{0}. \end{aligned} \tag{52}$$

The nonnegative constant θ reflects his tradeoff between risk and return. If $\theta = 0$, the model is a linear programming problem, and he will invest completely in the investment with greatest expected return. For very large θ , the objective contribution due to expected return becomes negligible and he is essentially minimizing his risk. Here, we assume $\theta = \tilde{1}$ and it means the tradeoff between risk and return is a fuzzy triangular number $\tilde{1} = (0, 1, 2)$. The other coefficients are studied as triangular fuzzy numbers as follow:

$$\tilde{1} = (0, 1, 2),\ \tilde{2} = (1, 2, 3),\ \tilde{5} = (4, 5, 6),\ \tilde{16} = (15, 16, 17),\ \tilde{20} = (19, 20, 21).$$

The optimal solution of the crisp problem is $x^* = (\frac{7}{3}, \frac{8}{3})^\top$. Table 1 shows results of utilizing suggested model for fixed values $w_1 = w_2 = 0.5$ and some different values of α-cuts. We use the model in (21) to solve (52). Figure 7 depicts the transient behavior of $x(t)$ of the model (21) with 6 random initial points and $\alpha = 1$, $w_1 = w_2 = \frac{1}{2}$. In fact, the Figure 7 shows the optimal solution of the problem (52) when the fuzziness goes to zero and choosing the parameter are $w_1 = w_2 = \frac{1}{2}$. Also, Figure 8 displays the transient behavior of $x(t)$ by choosing $\alpha = 0.7$, $w_1 = 0.1$, $w_2 = 0.9$ with 15 random initial points.

Table 1: Results for some values of α and fixed values $w_1 = w_2 = 0.5$ in Example 7.3.

Cuts	Objective Function	Optimal Solution		
α	f	x_1^*	x_2^*	$CPU - Time(s)$
0	-23.7541	1.0765	0.615	0.080605
0.2	-31.3422	1.3217	0.9081	0.067497
0.4	-41.2544	1.6223	1.3446	0.067429
0.5	-46.9933	1.7901	1.6223	0.066574
0.6	-52.8665	1.9589	1.9232	0.09526
0.8	-62.3922	2.2244	2.4401	0.060178
1	-66.3266	2.3333	2.666	0.080748

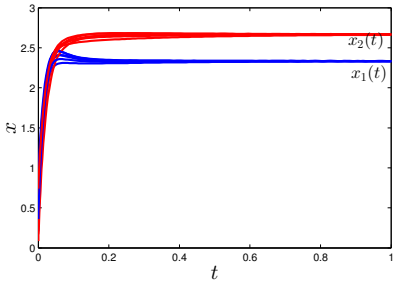


Figure 7: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = w_2 = 0.5, \alpha = 1$ for 6 different initial points of the neural network (21) in Example 7.3.

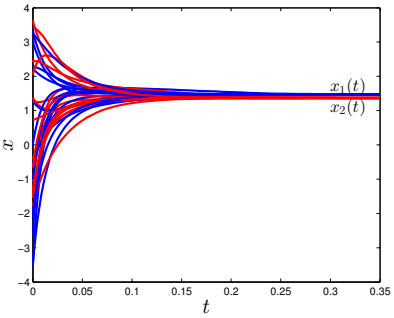


Figure 8: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.1, w_2 = 0.9, \alpha = 0.7$ for 15 different initial points of the neural network (21) in Example 7.3.

Example 7.4. [13] *Consider the following FNLOP (the crisp properties are from* [13]*):*

$$\begin{aligned} \text{minimize} \quad & \tilde{1}x_1^2 + \tilde{2}x_2^2 + \tilde{2}x_1x_2 - \tilde{10}x_1 - \tilde{12}x_2 \\ \text{subject to} \quad & \tilde{1}x_1 + \tilde{3}x_2 \leq \tilde{8} \\ & \tilde{1}x_1^2 + \tilde{1}x_2^2 + \tilde{2}x_1 - \tilde{2}x_2 \leq \tilde{3}, \end{aligned} \tag{53}$$

where,

$$\tilde{1} = (0,1,2),\ \tilde{2} = (1,2,3),\ \tilde{3} = (2,3,4),\ \tilde{8} = (7,8,9),\ \tilde{10} = (9,10,11),\ \tilde{12} = (11,12,13).$$

The optimal solution to the crisp problem is $x^* = (1,2)^\top$. *Table 2 depicts results of using suggested model for fixed values* $w_1 = w_2 = 0.5$ *and some different* α*-cuts. We use the model in* (21) *to solve the mathematical programming* (53). *Also, Figure 9 shows the transient behavior of* $x(t)$ *based on the model* (21) *with 8 random initial points,* $\alpha = 1,\ w_1 = w_2 = \frac{1}{2}$. *The optimal solution of the problem for* $\alpha = 1$ *via* $w_1 = w_2 = \frac{1}{2}$ *is shown in Figure 9. Moreover, Figure 10 establishes*

the transient behavior of $x(t)$ of the model (21) with 9 various initial points, $w_1 = 0.6,\ w_2 = 0.3,\ \alpha = 0.7$. According to the Table 2, the α-cut of f represents the possibility that the objective value will appear in the associated range. For instance, at $\alpha = 0.5$, the value of $f = -19.0588$ occurs at $x_1 = 0.7752$ and $x_2 = 1.6793$ and at $\alpha = 1$ (i.e., when the fuzziness goes to zero) the value of $f = -20.9991$ occurs at $x_1 = 0.9998$ and $x_2 = 2$.

Table 2: Results for some values of α and fixed values $w_1 = w_2 = 0.5$ in Example 7.4.

Cuts	Objective Function	Optimal Solution		
α	f	x_1^*	x_2^*	$CPU - Time(s)$
0	-16.9081	0.6	1.4132	0.101796
0.2	-17.7052	0.656	1.5094	0.091607
0.4	-18.5763	0.7278	1.6187	0.111554
0.5	-19.0588	0.7752	1.6793	0.09564
0.6	-19.5515	0.8292	1.7423	0.07426
0.8	-20.626	0.979	1.8738	0.079761
1	-20.9991	0.9998	2	0.095675

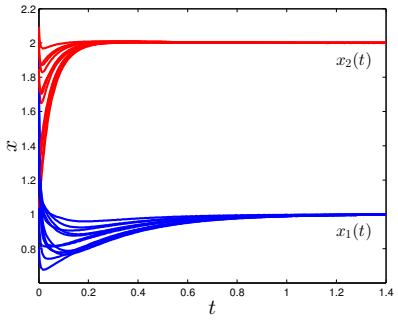


Figure 9: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = w_2 = 0.5, \alpha = 1$ for 8 different initial points of the neural network (21) in Example 7.4.

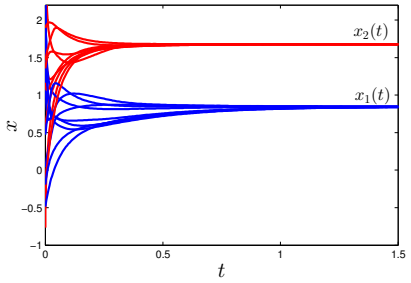


Figure 10: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.3, w_2 = 0.7, \alpha = 0.6$ for 9 different initial points of the neural network (21) in Example 7.4.

Example 7.5. [85] *Consider the following FNLOP (the crisp properties are from* [85]*):*

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\Big((\tilde{1}x_1 - \tilde{1}x_2)^4 + (\tilde{1}x_2 + \tilde{1}x_3)^2 + (\tilde{1}x_1 + \tilde{1}x_3)^2\Big) \\ \text{subject to} \quad & \tilde{1}x_1^2 + \tilde{3}x_2^4 - \tilde{1}x_3 \leq \tilde{0} \\ & (\tilde{2} - \tilde{1}x_1)^2 + (\tilde{2} - \tilde{1}x_2)^2 - \tilde{1}x_3 \leq \tilde{0} \\ & \tilde{2}\exp(-\tilde{1}x_1 + \tilde{1}x_2) - \tilde{1}x_3 \leq \tilde{0} \\ & \tilde{1}x_1^2 + \tilde{1}x_2^2 - \tilde{2}x_1 + \tilde{1}x_2 - \tilde{4} \leq \tilde{0} \\ & |\tilde{1}x_1| \leq \tilde{2}, \quad |\tilde{1}x_2| \leq \tilde{2}, \quad \tilde{1}x_3 \geq \tilde{0}, \end{aligned} \tag{54}$$

where,

$$\tilde{1} = (0,1,2),\ \tilde{4} = (3,4,5),\ \tilde{2} = (1,2,3), -\tilde{1} = (-2,-1,0), -\tilde{2} = (-3,-2,-1),\ \tilde{0} = (-1,0,1).$$

The optimal solution to the crisp problem is $x^* = (1.109, 0.922, 1.954)^\top$ *. Table 3 shows results of using the suggested model via* $w_1 = w_2 = 0.5$ *and some different values of* α*-cuts. We utilize the model in* (21) *to solve the mathematical programming* (54)*. Also, Figure 11 shows the transient behavior of* $x(t)$ *of the recurrent model* (21) *with 12 random initial points and* $\alpha = 1$ *(when the fuzziness moves to zero),* $w_1 = w_2 = \frac{1}{2}$*. In fact, Figure 11 displays the solution of the crisp original problem (i. e.,* $\alpha = 1$*) via* $w_1 = w_2 = \frac{1}{2}$*. One can see all simulation results illustrate that the recurrent model* (21) *is globally asymptotically stable. Moreover, Figure 12 depicts the convergence behavior of the model* (21) *with 6 random initial points and* $w_1 = 0.4,\ w_2 = 0.6,\ \alpha = 0.2$*.*

Table 3: Results for some values of α and fixed values $w_1 = w_2 = 0.5$ in Example 7.5.

Cuts	Objective Function	Optimal Solution			
α	f	x_1^*	x_2^*	x_3^*	$CPU - Time(s)$
0	3.517	0.3294	-0.051	1.1642	0.256053
0.2	6.3426	0.9817	0.2492	0.9362	0.259256
0.4	8.0763	1.2599	0.5347	1.0689	0.29027
0.5	8.1488	1.3184	0.6541	1.2324	0.243795
0.6	7.203	1.3306	0.7346	1.2562	0.228097
0.8	5.3845	1.1614	0.7581	1.2862	0.168201
1	8.8278	1.109	0.9224	1.9539	0.215884

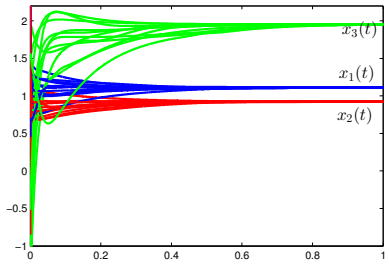


Figure 11: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = w_2 = 0.5, \alpha = 1$ for 12 different initial points of the neural network (21) in Example 7.5.

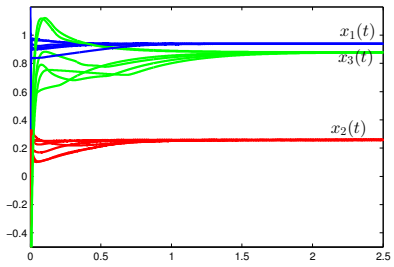


Figure 12: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.4, w_2 = 0.6, \alpha = 0.2$ for 6 different initial points of the neural network (21) in Example 7.5.

Example 7.6. [85] *(Manufacturing Systems) A manufacturing factory is going to produce 2 kinds of products A and B in a period (such as one month). The production of A and B requires 3 kinds of resources R_1, R_2, and R_3 . The requirements of each kind of resource to produce each product A are 2, 4 and 3 units, respectively. To produce each product, B, 3, 2 and 2 units are to be required, respectively. The planned available resource of R_1 and R_2 are 50 and 44 units, respectively. But there are additional 30 and 20 units safety store of material which are administrated by the general manager. The estimate value of available quantity of resource R_3 is 36 units, estimate error are 5 units. Assuming that the planned production quantity of A and B is x_1 and x_2 respectively. Further, assuming that unit cost and sale's price of product A and B are $UC_1 = c_1$, $UC_2 = c_2$, $US_1 = \frac{k_1}{x_1^{\frac{1}{\alpha_1}}}$, and $US_1 = \frac{k_2}{x_2^{\frac{1}{\alpha_2}}}$, respectively. The Decision Maker hopes that the total profits reach an aspiration level z_0 and not less than a lower level $z_0 - p_0$. This is a typical FNLOP problem and it can be described as follow:*

$$\begin{aligned} \text{maximize} \quad & k_1 x_1^{1-\frac{1}{a_1}} - c_1 x_1 + k_2 x_2^{1-\frac{1}{a_2}} - c_2 x_2, \\ \text{subject to} \quad & 2x_1 + 3x_2 \leq \tilde{50}, \\ & 4x_1 + 2x_2 \leq \tilde{44}, \\ & 3x_1 + 2x_2 \leq \tilde{36}, \\ & x_1, x_2 \geq 0. \end{aligned} \tag{55}$$

$$\tilde{36} = (31, 36, 41),\ \tilde{44} = (44, 44, 64),\ \tilde{50} = (50, 50, 80), k_1 = 50,\ k_2 = 45,\ c_1 = 8,\ c_2 = 10,\ a_1 = a_2 = 2.$$

Table 4 displays results of using the proposed model via $w_1 = w_2 = 0.5$ and some different values of α-cuts. We use the model in (21) to solve the mathematical programming (55). Figure 13 depicts the transient behavior of $x(t)$ of the model (21) with 9 random initial points and $w_1 = w_2 = 0.5$, $\alpha = 1$. Therefore, the solution can be obtained as $x_1 = 8.5967$, $x_2 = 4.8065$, and $f = 128.4188$. One can see all simulation results illustrate that model (21) is globally asymptotically stable. Furthermore, we solve this problem by letting 11 random initial points with $w_1 = 0.85$, $w_2 = 0.15$, $\alpha = 0.5$. Figure 14 depicts the convergence behavior of (21).

Table 4: Results for some values of α and fixed values $w_1 = w_2 = 0.5$ in Example 7.6.

Cuts	Objective Function	Optimal Solution		
α	f	x_1^*	x_2^*	$CPU - Time(s)$
0	127.2091	7.4296	4.3557	0.067986
0.2	127.5721	7.7031	4.4455	0.072493
0.4	127.8814	7.9773	4.5338	0.083693
0.5	128.0172	8.1152	4.5773	0.059678
0.6	128.1407	8.2536	4.6205	0.079472
0.8	128.3506	8.5299	4.7057	0.093484
1	128.4188	8.5967	4.8065	0.065527

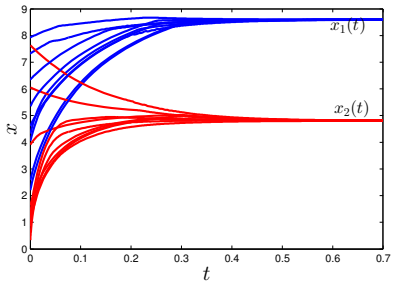


Figure 13: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = w_2 = 0.5, \alpha = 1$ for 9 different initial points of the neural network (21) in Example 7.6.

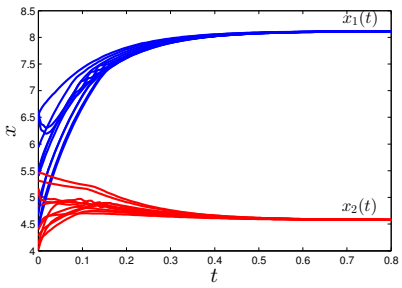


Figure 14: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.85, w_2 = 0.15, \alpha = 0.5$ for 11 different initial points of the neural network (21) in Example 7.6.

Example 7.7. [43] *Consider the following fuzzy quadratic programming problem:*

$$\begin{aligned} \text{minimize} \quad & \tilde{c}^\top x + \frac{1}{2} x^\top \tilde{H} x \\ \text{subject to} \quad & \tilde{A}x \leq \tilde{b} \\ & x \geq 0, \end{aligned} \tag{56}$$

where,

$$\tilde{b} = \begin{bmatrix} \tilde{3} & \tilde{1} & \tilde{2} \end{bmatrix}^\top, \ \tilde{c} = \begin{bmatrix} -\tilde{1} & \tilde{4} & -\tilde{1} & \tilde{1} & \tilde{0} & \tilde{0} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{0} \end{bmatrix}^\top$$

$$\tilde{H} = \begin{bmatrix} \tilde{2} & \tilde{1} & \tilde{0} & \tilde{0} & \dots & \tilde{0} & \tilde{0} \\ \tilde{1} & \tilde{2} & \tilde{1} & \tilde{0} & \dots & \tilde{0} & \tilde{0} \\ \tilde{0} & \tilde{1} & \tilde{2} & \tilde{1} & \dots & \tilde{0} & \tilde{0} \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \tilde{0} & \tilde{0} & \dots & \tilde{1} & \tilde{2} & \tilde{1} & \tilde{0} \\ \tilde{0} & \tilde{0} & \dots & \tilde{0} & \tilde{1} & \tilde{2} & \tilde{1} \\ \tilde{0} & \tilde{0} & \dots & \tilde{0} & \tilde{0} & \tilde{1} & \tilde{2} \end{bmatrix}_{10\times 10}, \tilde{A} = \begin{bmatrix} \tilde{1} & -\tilde{1} & -\tilde{1} & \tilde{1} & \tilde{1} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{1} & -\tilde{1} \\ \tilde{0} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{1} & \tilde{1} & \tilde{1} & \tilde{0} \\ -\tilde{1} & \tilde{1} & -\tilde{1} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{1} & \tilde{0} & \tilde{1} & \tilde{0} \end{bmatrix}$$

and

$$\tilde{1} = (0,1,2),\ \tilde{2} = (1,2,3),\ \tilde{4} = (3,4,5),\ \tilde{3} = (2,3,4), -\tilde{1} = (-2,-1,0),\ \tilde{0} = (-1,0,1).$$

The optimal solution of the problem (56) *for* $\alpha = 1$ *is* $x^* = [0.5, 0, 0.5, 0, 0, 0, 0, 0, 0, 0]\top$. *Table 5 depicts results of utilizing the offered model for* $w_1 = w_2 = 0.5$ *and some different values of* α*-cuts. We use the proposed model* (21) *to solve the mathematical programming* (56). *Figure 15 shows the transient behavior of* $x(t)$ *of the recurrent model* (21) *with 2 random initial points and* $w_1 = w_2 = 0.5,\ \alpha = 1$. *One can see all simulation results illustrate that the model* (21) *is globally asymptotically stable. Moreover, we solve this problem by getting 5 random initial points and* $\alpha = 0.25,\ w_1 = 0.95,\ w_2 = 0.05$. *Figure 16 displays the convergence behavior of* (21).

Table 5: Results for some values of α and fixed values $w_1 = w_2 = 0.5$ in Example 7.7.

Cuts	Obj Fun	Optimal Solution										
α	f	x_1^*	x_2^*	x_3^*	x_4^*	x_5^*	x_6^*	x_7^*	x_8^*	x_9^*	x_{10}^*	$CPU-Time(s)$
0	-0.4984	0.5	0	0.4999	0	0	0	0	0	0	0	0.337054
0.2	-0.4989	0.5	0	0.4999	0	0	0	0	0	0	0	0.298779
0.4	-0.4987	0.5	0	0.4999	0	0	0	0	0	0	0	0.290264
0.5	-0.4994	0.5	0	0.5	0	0	0	0	0	0	0	0.296896
0.6	-0.4997	0.5	0	0.5	0	0	0	0	0	0	0	0.264569
0.8	-0.4992	0.5	0	0.4999	0	0	0	0	0	0	0	0.274842
1	-0.499	0.5	0	0.4999	0	0	0	0	0	0	0	0.226127

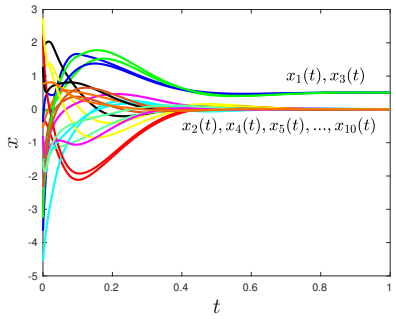


Figure 15: Transient behaviors of $(x_1, x_2, \ldots, x_{10})^\top$ with $w_1 = w_2 = 0.5, \alpha = 1$ for 2 different initial points of the neural network (21) in Example 7.7.

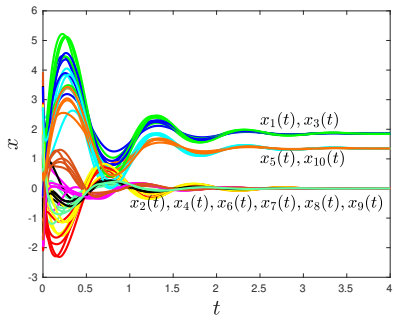


Figure 16: Transient behaviors of $(x_1, x_2, \ldots, x_{10})^\top$ with $w_1 = 0.95, w_2 = 0.05, \alpha = 0.25$ for 5 different initial points of the neural network (21) in Example 7.7.

Example 7.8. [16] *The NUM problem is a non-convex program applied to analyze distributed rate allocation in a communication network. Assume that we have a network with L links having a finite fixed capacity c_l bps and S sources having rate of source transmission x_s bps. Each source s emits one flow, using a fixed set $L(s)$ of links in its path, and it has a utility function $U_s(x_s)$. Each link l is shared by a set $S(l)$ of sources. Consider the NUM problem as the following problem:*

$$\begin{aligned} \text{maximize} \quad & \sum_{s\in S} U_s(x_s) \\ \text{subject to} \quad & \sum_{s\in S(l)} x_s \le c_l, \quad \forall l, \\ & \tilde{1}x_s \ge 0, \quad \forall s \in S. \end{aligned}$$

Here, we consider that the finite capacity is a fuzzy value $\tilde{c}_l$ and also all of the other coefficients are fuzzy parameters. Now, consider two links and three network users. Considering sigmoidal utility functions, the resource allocation problem is obtained by the following fuzzy non-convex optimization problem (the crisp problem is from [16]*):*

$$\begin{aligned} \text{maximize} \quad & \sum_{s=1}^{3} \frac{\tilde{1}}{\tilde{1}+e^{-(\tilde{1}x_s-\tilde{5})}} \\ \text{subject to} \quad & \tilde{1}x_1+\tilde{1}x_2 \leq \tilde{4} \\ & \tilde{1}x_1+\tilde{1}x_3 \leq \tilde{8} \\ & \tilde{1}x_s \geq \tilde{0},\ s=1,2,3, \end{aligned} \tag{57}$$

where,

$$\tilde{1}=(0,1,2),\ \tilde{5}=(4,5,6),\ \tilde{4}=(3,4,5),\ \tilde{8}=(7,8,9), -\tilde{1}=(-2,-1,0),\ \tilde{0}=(-1,0,1).$$

Table 6: Results for some values of α and fixed values $w_1=w_2=0.5$ in Example 7.8.

Cuts	Objective Function	Optimal Solution			
α	f	x_1^*	x_2^*	x_3^*	$CPU-Time(s)$
0.1	1.5189	8.998	8.999	8.999	0.626326
0.2	1.2192	4.0008	4.0004	4.0004	0.604005
0.4	0.4397	1.4995	1.4997	3.9367	0.404121
0.5	0.5411	1.0004	2.0315	4.7056	0.271788
0.6	0.6385	0.6666	2.5273	5.3608	0.312834
0.8	0.9417	0.2498	3.3349	6.607	0.282628
1	1.2282	0	4	8	0.312856

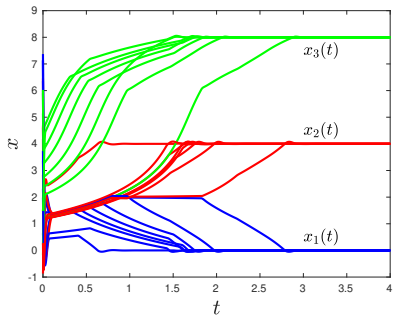


Figure 17: Transient behaviors of $(x_1,x_2,\ldots,x_{10})^\top$ with $w_1=w_2=0.5, \alpha=1$ for 8 different initial points of the neural network (21) in Example 7.8.

We use the suggested model for solving the non-convex program (57). The unique optimal solution of the problem is $x^*=(0,4,8)^\top$. Table 6 displays results of the model (21) for $w_1=w_2=0.5$ and some different values of α-cuts. The problem is solved, by considering $w_1=w_2=0.5, \alpha=1$. Figure 17 shows the convergence behavior of the proposed model for 8 random initial points. Also, Figure 18 illustrates the trajectories of the model with $w_1=0.9,\ w_2=0.1,\ \alpha=0.45$ and 4 random initial points.

8 Conclusion

In this paper we proposed a neural network framework for FNLOPs. The provided neural network has been shown to be Lyapunov stable and converge globally to the FNLOPs solution. The convergence of the neural network has been

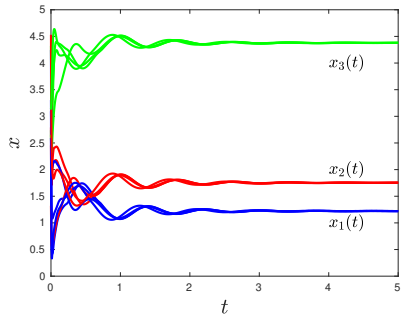


Figure 18: Transient behaviors of $(x_1, x_2)^\top$ with $w_1 = 0.9, w_2 = 0.1, \alpha = 0.45$ for 4 different initial points of the neural network (21) in Example 7.8.

validated with the simulation results of the FNLOP examples. The exhibited neural network does not also require any adjustable parameters and its structure is simple. Thus, the model is more appropriate to be implemented in hardware. Moreover, since FNLOP has wide applications, the novel model and the obtained results in this paper are critical in both theory and application. The FNLOP simulation results illustrate the capability of the suggested framework. We summarize the highlights of investigation results that convey the key findings to the readers as follows:

• We compare our neural network model with some existing models in Section 6. At first glance, we observe that the difference of the numerical performance is very marginal by testing some FNLOP.

• From Theorem 5.5, we see that the solution converges with any initial point. Thus, changing initial points may not have much effect for our neural network model. In fact, our model is globally convergent to the optimal solution of the problem.

• The neural network (21) can be implemented without a penalty parameter and can be convergent to an exact solution of the FNLOPs.

• We compare the recurrent model (21) with a existing model (when the fuzziness goes to zero, i.e., $\alpha = 1$). The RNN (21) look having lower CPU-Time than some other neural networks.

• It should be noted that we use the proposed recurrent neural network model to solve applicable examples in manufacturing and Portfolio Selection. One can see Examples 7.3 and 7.6.

As more future works, one can extend the proposed neural network model for solving fuzzy convex optimization problems under Hukuhara differentiability, generalized Hukuhara differentiability and granular differentiability [86].

Acknowledgments

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest Both of the authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with animals performed by any of the authors.

References

[1] S. K. Agrawal, B. C. Fabien, *Optimization of dynamic systems*, Kluwer Academic Publishers, Netherlands, 1999. https://doi.org/10.1007/978-94-015-9149-2

[2] Z. Arjmandzadeh, M. Safi, A. Nazemi, *A new neural network model for solving random interval linear programming problems*, Neural Networks, **89** (2017), 11-18. https://doi.org/10.1016/j.neunet.2016.12.007

[3] M. Avriel, *Nonlinear programming: Analysis and methods*, Prentice-Hall, EnglewoodCliffs, NJ, 1976.

[4] M. S. Bazaraa, C. Shetty, H. D. Sherali, *Nonlinear programming, theory and algorithms*, John Wiley and Sons, NewYork, 1979. https://doi.org/10.1002/0471787779

[5] D. P. Bertsekas, J. N. Tsitsiklis, *Parallel and distributed computation: Numerical methods*, Prentice Hall. New Jersey: Englewood Cliffs, 1989.

[6] S. Boyd, L. Vandenberghe, *Convex optimization*, Cambridge University Press, Cambridge, 2004. https://doi.org/10.1109/TAC.2006.884922

[7] J. J. Buckley, *Possibilistic linear programming with triangular fuzzy numbers*, Fuzzy Sets and Systems, **26** (1998), 135-138. https://doi.org/10.1016/0165-0114(88)90013-9

[8] L. Campos, J. L. Verdegay, *Linear programming problems and ranking of fuzzy numbers*, Fuzzy Sets and Systems, **32** (1989), 1-11. https://doi.org/10.1016/0165-0114(89)90084-5

[9] X. Cao, A. Francis, X. Pu, Z. Zhang, V. Katsikis, P. Stanimirovic, I. Brajevic, S. Li, *A novel recurrent neural network based online portfolio analysis for high frequency trading*, Expert Systems with Applications, **233** (2023), 120934. https://doi.org/10.1016/j.eswa.2023.120934

[10] X. Cao, C. Peng, Y. Zheng, S. Li, T. T. Ha, V. Shutyaev, V. Katsikis, P. Stanimirovic, *Neural networks for portfolio analysis in high-frequency trading*, IEEE Transactions on Neural Networks and Learning Systems, 2023. https://doi.org/10.1109/TNNLS.2023.3311169

[11] Y. H. Chen, S. C. Fang, *Neurocomputing with time delay analysis for solving convex quadratic programming problems*, IEEE Transactions on Neural Networks, **11** (2000), 230-240. https://doi.org/10.1109/72.822526

[12] J. S. Chen, C. H. Ko, S. Pan, *A neural network based on the generalized Fischer-Burmeister function for nonlinear complementarity problems*, Information Sciences, **180**(5) (2010), 697-711. https://doi.org/10.1016/j.ins.2009.11.014

[13] S. Effati, A. Ghomashi, A. Nazemi, *Application of projection neural network in solving convex programming problems*, Applied Mathematics and Computation, **188** (2007), 1103-1114. https://doi.org/10.1016/j.amc.2006.10.088

[14] S. Effati, A. Mansoori, M. Eshaghnezhad, *An efficient projection neural network for solving bilinear programming problems*, Neurocomputing, **168** (2015), 1188-1197. https://doi.org/10.1016/j.neucom.2015.05.003

[15] M. Ehrgott, M. Wiecek, *Multiobjective programming*, In: Figueira, J., Greco, S., Ehrgott, M. (eds.) Multicriteria Decision Analysis: State of the Art Surveys., Springer Science + Business Media, New York, (2005), 667-722. https://doi.org/10.1007/b100605

[16] M. Eshaghnezhad, S. Effati, A. Mansoori, *A neurodynamic model to solve nonlinear pseudo-monotone projection equationand its applications*, IEEE Transactions on Cybernetics, **47**(10) (2017), 3050-3062. https://doi.org/10.1109/TCYB.2016.2611529

[17] M. A. Fatma, *A differential equation approach to fuzzy non-linear programming problems*, Fuzzy Sets and Systems, **93** (1998), 57-61. https://doi.org/10.1016/S0165-0114(96)00217-5

[18] A. Feizi, A. Nazemi. A. Feizi, A. Nazemi, M. Rabiei, *Solving the stochastic support vector regression with probabilistic constraints by a high-performance neural network model*, Engineering with Computers, **38** (2021), 1-16. https://doi.org/10.1007/s00366-020-01214-5

[19] R. Fletcher, *Practical methods of optimization*, Wiley, NewYork, 1981. https://doi.org/10.1002/9781118723203

[20] M. Friedman, M. Ma, A. Kandel, *Numerical solution of fuzzy differential and integral equations*, Fuzzy Sets and Systems, **106** (1999), 35-48. https://doi.org/10.1016/S0165-0114(98)00355-8

[21] T. L. Friesz, D. H. Bernstein, N. J. Mehta, R. L. Tobin, S. Ganjlizadeh, *Day-to-day dynamic network disequilibria and idealized traveler information systems*, Operation Research, **42** (1994), 1120-1136. https://doi.org/10.1287/opre.42.6.1120

[22] X. Gao, *A novel neural network for nonlinear convex programming*, IEEE Transactions on Neural Networks, **15** (2004), 613-621. https://doi.org/10.1109/TNN.2004.824425

[23] R. Ghanbari, K. Ghorbani-Moghadam, N. Mahdavi-Amiri, *A variables neighborhood search algorithm for solving fuzzy quadratic programming problems using modified Kerre's method*, Soft Computing, **23** (2019), 12305-12315. https://doi.org/10.1007/s00500-019-03771-4

[24] R. Ghanbari, K. Ghorbani-Moghadam, N. Mahdavi-Amiri, *A time variant multi-objective particle swarm optimization algorithm for solving fuzzy number linear programming problems using modified Kerre's method*, OPSEARCH, **58** (2021), 403-424. https://doi.org/10.1007/s12597-020-00482-5

[25] X. He, T. Huang, J. Yu, C. Li, C. Li, *An inertial projection neural network for solving variational inequalities*, IEEE Transactions on Cybernetics, **47** (2016), 809-814. https://doi.org/10.1109/TCYB.2016.2523541

[26] J. J. Hopfield, D. W. Tank, *Neural computation of decisions in optimization problems*, Biological Cybernetics, **52** (1985), 141-152. https://doi.org/10.1007/BF00339943

[27] C. Y. Hsu, B. R. Kao, V. L. Hob, K. R. Lai, *Agent-based fuzzy constraint-directed negotiation mechanism for distributed job shop scheduling*, Engineering Applications of Artificial Intelligence, **53** (2016), 140-154. https://doi.org/10.1016/j.engappai.2016.04.005

[28] X. Hu, J. Wang, *Solving pseudo-monotone variational inequalities and pseudo-convex optimization problems using the projection neural network*, IEEE Transactions on Neural Networks, **17** (2006), 1487-1499. https://doi.org/10.1109/TNN.2006.879774

[29] D. Karbasi, A. Nazemi, M. Rabiei, *A parametric recurrent neural network scheme for solving a class of fuzzy regression models with some real-world applications*, Soft Computing, **24** (2020), 11159-11187. https://doi.org/10.1007/s00500-020-05008-1

[30] D. Karbasi, A. Nazemi, M. Rabiei, *An optimization technique for solving a class of ridge fuzzy regression problems*, Neural Processing Letters, **53** (2021), 3307-3338. https://doi.org/10.1007/s11063-021-10538-2

[31] D. Kinderlehrer, G. Stampacchia, *An introduction to variational inequalities and their applications*, Society for Industrial and Applied Mathematics, 2000. https://doi.org/10.1137/1.9780898719451

[32] S. Li, J. He, Y. Li, M. U. Rafique, *Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective*, IEEE Transactions on Neural Networks and Learning Systems, **28**(2) (2016), 415-426. https://doi.org/10.1109/TNNLS.2016.2516565

[33] S. Li, L. Jin, M. A. Mirza, *Kinematic control of redundant robot arms using neural networks*, John Wiley and Sons, 2019. http://dx.doi.org/10.1002/9781119557005

[34] S. Li, Y. Li, Z. Wang, *A class of finite-time dual neural networks for solving quadratic programming problems and its k-winners-take-all application*, Neural Networks, **39** (2013), 27-39. https://doi.org/10.1016/j.neunet.2012.12.009

[35] Z. Li, H. Xiao, C. Yang, Y. Zhao, *Model predictive control of nonholonomic chained systems using general projection neural networks optimization*, IEEE Transactions on Systems, Man, and Cybernetics, **45** (2015), 1313-1321. https://doi.org/10.1109/TSMC.2015.2398833

[36] G. Li, Z. Yan, J. Wang, *A one-layer recurrent neural network for constrained nonconvex optimization*, Neural Networks, **61** (2015), 10-21. https://doi.org/10.1016/j.neunet.2014.09.009

[37] L. Z. Liao, H. Qi, L. Qi, *Solving nonlinear complementarity problems with neural networks: A reformulation method approach*, Journal of Computational and Applied Mathematics, **131**(1-2) (2001), 343-359. https://doi.org/10.1016/S0377-0427(00)00262-4

[38] Q. Liu, J. Wang, *L1-minimization algorithms for sparse signal reconstruction based on a projection neural network*, IEEE Transactions on Neural Networks and Learning Systems, **27**(3) (2016), 698-707. https://doi.org/10.1109/TNNLS.2015.2481006

[39] J. Lu, S. C. Fang, *Solving nonlinear optimization problems with fuzzy relation equation constraints*, Fuzzy Sets and Systems, **119** (2001), 1-20. https://doi.org/10.1016/S0165-0114(98)00471-0

[40] T. Lu, S. T. Liu, *Fuzzy nonlinear programming approach to the evaluation of manufacturing processes*, Engineering Applications of Artificial Intelligence, **72** (2018), 183-189. https://doi.org/10.1016/j.engappai.2018.04.003

[41] K. Majeed, Z. Masood, R. Samar, M. A. Z. Raja, *A genetic algorithm optimized Morlet wavelet artificial neural network to study the dynamics of nonlinear Troesch's system*, Applied Soft Computing, **56** (2017), 420-435. https://doi.org/10.1016/j.asoc.2017.03.028

[42] A. Mansoori, S. Effati, *An efficient neurodynamic model to solve nonlinear programming problems with fuzzy parameters*, Neurocomput, **334** (2019), 125-133. https://doi.org/10.1016/j.neucom.2019.01.012

[43] A. Mansoori, S. Effati, M. Eshaghnezhad, *An efficient recurrent neural network model for solving fuzzy non-linear programming problems*, Applied Intelligence, **46** (2017), 308-327. https://doi.org/10.1007/s10489-016-0837-4

[44] A. Mansoori, S. Effati, M. Eshaghnezhad, *A neural network to solve quadratic programming problems with fuzzy parameters*, Fuzzy Optimization and Decision Making, **17**(1) (2018), 75-101. https://doi.org/10.1007/s10700-016-9261-9

[45] A. Mansoori, M. Erfanian, *A dynamic model to solve the absolute value equations*, Journal of Computational and Applied Mathematics, **333** (2018), 28-35. https://doi.org/10.1016/j.cam.2017.09.032

[46] A. Mansoori, M. Eshaghnezhad, S. Effati, *An efficient neural network model for solving the absolute value equations*, IEEE Transactions on Circuits and Systems II: Express Briefs, **65**(3) (2018), 391-395. https://doi.org/10.1109/TCSII.2017.2750065

[47] X. Miao, J. S. Chen, C. H. Ko, *A smoothed NR neural network for solving nonlinear convex programs with second-order cone constraints*, Information Sciences, **268**(1) (2014), 255-270. https://doi.org/10.1016/j.ins.2013.10.017

[48] K. M. Miettinen, *Non-linear multi objective optimization*, Kluwer Academic Publishers, 1999. https://doi.org/10.1007/978-1-4615-5563-6

[49] R. K. Miller, A. N. Michel, *Ordinary differential equations*, Academic Press, NewYork, 1982. http://dx.doi.org/10.1109/MCS.2008.930834

[50] S. M. Mousavi, S. Mirdamadi, A. Siadat, J. Dantan, R. Tavakkoli-Moghaddam, *An intuitionistic fuzzy grey model for selection problems with an application to the inspection planningin manufacturing firms*, Engineering Applications of Artificial Intelligence, **39** (2015), 157-167. https://doi.org/10.1016/j.engappai.2014.12.004

[51] A. R. Nazemi, *A dynamic system model for solving convex nonlinear optimization problems*, Communications in Nonlinear Science and Numerical Simulation, **17** (2012), 1696-1705. https://doi.org/10.1016/j.cnsns.2011.08.035

[52] A. R. Nazemi, S. Effati, *Anapplication of a merit function for solving convex programming problems*, Computers and Industrial Engineering, **66** (2013), 212-221. https://doi.org/10.1016/j.cie.2013.07.017

[53] A. R. Nazemi, M. Mortezaee, *A new gradient-based neural dynamic framework for solving constrained min-max optimization problems with an application in portfolio selection models*, Applied Intelligence, **49** (2019), 396-419. https://doi.org/10.1007/s10489-018-1268-1

[54] A. R. Nazemi, A. Sabeghi, *A new neural network framework for solving convex secondorder cone constrained variational inequality problems with an application in multi-finger robot hands*, Journal of Experimental and Theoretical Artificial Intelligence, **32**(2) (2019), 181-203. https://doi.org/10.1080/0952813X.2019.1647559

[55] A. Nikseresht, A. R. Nazemi, *A novel neural network for solving semidefinite programming problems with some applications*, Journal of Computational and Applied Mathematics, **350** (2019), 309-323. https://doi.org/10.1016/j.cam.2018.10.025

[56] A. Nikseresht, A. R. Nazemi, *A novel neural network model for solving a class of nonlinear semidefinite programming problems*, Journal of Computational and Applied Mathematics, **338** (2018), 69-79. https://doi.org/10.1016/j.cam.2018.01.023

[57] A. Nikseresht, A. R. Nazemi, *A smoothing gradient-based neural network strategy for solving semidefinite programming problems*, NET-Network: Computation in Neural Systems, **33**(3-4) (2022), 187-213. https://doi.org/10.1080/0954898X.2022.2104463

[58] J. Nocedal, S. Wright, *Numerical optimization*, Second ed., Springer-Verlag, Berlin, NewYork, 2006. https://doi.org/10.1007/0-387-22742-3_18

[59] Z. Peng, J. Wang, *Output-feedback path-following control of autonomous under water vehicles based on an extended state observer and projection neural networks*, IEEE Transactions on Systems, Man, and Cybernetics, **48**(4) (2018), 535-544. https://doi.org/10.1109/TSMC.2017.2697447

[60] S. Qin, X. Le, J. Wang, *A neurodynamic optimization approach to bilevel quadratic programming*, IEEE Transactions on Neural Networks and Learning Systems, **28** (2017), 2580-2591. https://doi.org/10.1109/TNNLS.2016.2595489

[61] M. A. Z. Raja, M. A. Manzar, F. H. Shah, *Intelligent computing for Mathieu's systems for parameter excitation, vertically driven pendulum and dusty plasma models*, Applied Soft Computing, **62** (2018), 359-372. https://doi.org/10.1016/j.asoc.2017.10.049

[62] D. Rani, A. Ebrahimnejad, G. Gupta, *Generalized techniques for solving intuitionistic fuzzy multi-objective nonlinear optimization problems*, Expert Systems with Applications, **202** (2022), 117264. https://doi.org/10.1016/j.eswa.2022.117264

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

[64] S. Rao, *Engineering optimization: Theory and practice*, Fourth ed., John Wiley and Sons, Hoboken, NewJersey, 2009. https://doi.org/10.1002/9780470549124

[65] J. D. L. C. Sama, K. Some, *Solving fuzzy nonlinear optimization problems using null set concept*, International Journal of Fuzzy System, **26** (2023), 674-685. https://doi.org/10.1007/s40815-023-01626-7

[66] Y. Shi, W. Sheng, S. Li, B. Li, X. Sun, *Neurodynamics for equality-constrained time-variant nonlinear optimization using discretization*, IEEE Transactions on Industrial Informatics, **20** (2024), 2354-2364. https://doi.org/10.1109/TII.2023.3290187

[67] Y. Shi, Q. Shi, X. Cao, B. Li, X. Sun, *An advanced discrete-time RNN for handling discrete time-varying matrix inversion: Form model design to disturbance-suppression analysis*, CAAI Transactions on Intelligence Technology, (2023), 607-621. https://doi.org/10.1049/cit2.12229

[68] Y. Shi, J. Wang, S. Li, B. Li, X. Sun, *Tracking control of cable-driven planar robot based on discrete-time recurrent neural network with immediate discretization method*, IEEE Transactions on Industrial Informatics, **19** (2023), 7414-7423. https://doi.org/10.1109/TII.2022.3210255

[69] Y. Shi, W. Zhao, S. Li, B. Li, X. Sun, *Novel discrete-time recurrent neural network for robot manipulator: A direct discretization technical route*, IEEE Transactions on Neural Networks and Learning Systems, **34** (2023), 2781-2790. https://doi.org/10.1109/TNNLS.2021.3108050

[70] G. Singh, A. Singh, *Extension of particle swarm optimization algorithm for solving transportation problem in fuzzy environment*, Applied Soft Computing, **110** (2021), 107619. https://https://doi.org/10.1016/j.asoc.2021.107619

[71] L. Stefanini, L. Sorini, M. L. Guerra, *Parametric representation of fuzzy numbers and application to fuzzy calculus*, Fuzzy Sets and Systems, **157** (2006), 2423-2455. https://doi.org/10.1016/j.fss.2006.02.002

[72] J. Sun, J. S. Chen, C. H. Ko, *Neural networks for solving second-order cone constrained variational inequality problem*, Computational Optimization and Applications, **51** (2012), 623-648. https://doi.org/10.1007/s10589-010-9359-x

[73] J. Sun, L. Zhang, *A globally convergent method based on Fischer-Burmeister operators for solving second-order coneconstrained variational inequality problems*, Computational Optimization and Applications, **58** (2009), 1936-1946. https://doi.org/10.1016/j.camwa.2009.07.084

[74] Y. R. Syau, *On convex and concave fuzzy mappings*, Fuzzy Sets and Systems, **103** (1999), 163-168. https://doi.org/10.1016/S0165-0114(97)00210-8

[75] J. Tang, D. Wang, *Modelling and optimization for a type of fuzzy nonlinear programming problems in manufacturing systems*, in: Proceedings of the 35th IEEE Conference on Decision and Control, (1996), 4401-4405. https://doi.org/10.1109/CDC.1996.577485.

[76] D. W. Tank, J. J. Hopfield, *Simple neural optimization networks: An a/d converter, signal decisioncircuit, and a linear programming circuit*, IEEE Transactions on Circuits and Systems, **33** (1986), 533-541. https://doi.org/10.1109/TCS.1986.1085953

[77] P. Vasant, *Hybrid LS-SA-PS methods for solving fuzzy non-linear programming problems*, Mathematical and Computer Modelling, **57** (2013), 180-188. https://doi.org/10.1016/j.mcm.2011.08.002

[78] G. Wang, C. Wu, *Directional derivatives and sub-differential of convex fuzzy mappings and application in convex fuzzy programming*, Fuzzy Sets and Systems, **138** (2003), 559-591. https://doi.org/10.1016/S0165-0114(02)00440-2

[79] H. C. Wu, *Evaluate fuzzy optimization problems based on biobjective programming problems*, Computers and Mathematics with Applications, **47** (2004), 893-902. https://doi.org/10.1016/S0898-1221(04)90073-9

[80] C. W. Wu, M. Y. Liao, *Fuzzy nonlinear programming approach for evaluating and ranking process yields with imprecise data*, Fuzzy Sets and Systems, **246** (2014), 142-155. https://doi.org/10.1016/j.fss.2013.10.014

[81] Y. Xia, *An extended neural network for constrained optimization*, Neural Computation, **16** (2004), 863-883. https://doi.org/10.1162/089976604322860730

[82] Y. Xia, J. Wang, *A recurrent neural network for solving linear projection equations*, Neural Networks, **13** (2000), 337-350. https://doi.org/10.1016/S0893-6080(00)00019-8

[83] Y. Xia, J. Wang, *A recurrent neural network for nonlinear convex optimization subject to nonlinear inequality constraints*, IEEE Transactions on Circuits and Systems I: Regular Papers, **51**(7) (2004), 1385-1394. https://doi.org/10.1109/TCSI.2004.830694

[84] Y. Xia, J. Wang, *A bi-projection neural network for solving constrained quadratic optimization problems*, IEEE Transactions on Neural Networks and Learning Systems, **27**(2) (2015), 214-224. https://doi.org/10.1109/TNNLS.2015.2500618

[85] Y. Yang, X. Xu, *The projection neural network for solving convex nonlinear programming*, In: Huang D-S, Heutte L, Loog M (eds) ICIC 2007, LNAI, vol **4682**. Springer-Verlag, Heidelberg, (2007), 174-181. https://doi.org/10.1007/978-3-540-74205-0_20

[86] J. Zhang, X. Chen, L. Li, X. Ma, *Optimality conditions for fuzzy optimization problems under granular convexity concept*, Fuzzy Sets and Systems, **447** (2022), 54-75. https://doi.org/10.1016/j.fss.2022.01.004

[87] H. Zhu, G. H. Huang, P. Guo, *SIFNP: Simulation–based interval–fuzzy nonlinear programming for seasonal planning of stream water quality management*, Water, Air, and Soil Pollution, **223** (2012), 2051-2072. https://doi.org/10.1007/s11270-011-1004-5

[88] H. Zhu, G. H. Huang, P. Guo, X. S. Qin, *A fuzzy robust nonlinear programming model for stream water quality management*, Water Resources Management, **23** (2009), 2913-2940. https://doi.org/10.1007/s11269-009-9416-3