mathematics

IMPACT FACTOR 2.3

CITESCORE 4.0

Article

Simulations and Bisimulations between Weighted Finite Automata Based on Time-Varying Models over Real Numbers

Predrag S. Stanimirović, Miroslav Ćirić, Spyridon D. Mourtas, Pavle Brzaković and Darjan Karabašević

Special Issue
Advances in Fuzzy Logic and Artificial Neural Networks

Edited by
Prof. Dr. Francisco Rodrigues Lima Junior

MDPI

https://doi.org/10.3390/math12132110

mathematics

Article

Simulations and Bisimulations between Weighted Finite Automata Based on Time-Varying Models over Real Numbers

Predrag S. Stanimirović [1,2], Miroslav Ćirić [1], Spyridon D. Mourtas [2,3], Pavle Brzaković [4] and Darjan Karabašević [4,5,*]

1 Faculty of Sciences and Mathematics, University of Niš, Višegradska 33, 18000 Niš, Serbia; pecko@pmf.ni.ac.rs (P.S.S.); miroslav.ciric@pmf.edu.rs (M.Ć.)
2 Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, Prosp. Svobodny 79, Krasnoyarsk 660041, Russia; spirmour@econ.uoa.gr
3 Department of Economics, Division of Mathematics-Informatics and Statistics-Econometrics, National and Kapodistrian University of Athens, Sofokleous 1 Street, 10559 Athens, Greece
4 Faculty of Applied Management, Economics and Finance, University Business Academy in Novi Sad, Jevrejska 24, 11000 Belgrade, Serbia; pavle.brzakovic@mef.edu.rs
5 College of Global Business, Korea University, Sejong 30019, Republic of Korea
* Correspondence: darjan.karabasevic@mef.edu.rs

Abstract: The zeroing neural network (ZNN) is an important kind of continuous-time recurrent neural network (RNN). Meanwhile, the existence of forward and backward simulations and bisimulations for weighted finite automata (WFA) over the field of real numbers has been widely investigated. Two types of quantitative simulations and two types of bisimulations between WFA are determined as solutions to particular systems of matrix and vector inequations over the field of real numbers $\mathbb{R}$. The approach used in this research is unique and based on the application of a ZNN dynamical evolution in solving underlying matrix and vector inequations. This research is aimed at the development and analysis of four novel ZNN dynamical systems for addressing the systems of matrix and/or vector inequalities involved in simulations and bisimulations between WFA. The problem considered in this paper requires solving a system of two vector inequations and a couple of matrix inequations. Using positive slack matrices, required matrix and vector inequations are transformed into corresponding equations and then the derived system of matrix and vector equations is transformed into a system of linear equations utilizing vectorization and the Kronecker product. The solution to the ZNN dynamics is defined using the pseudoinverse solution of the generated linear system. A detailed convergence analysis of the proposed ZNN dynamics is presented. Numerical examples are performed under different initial state matrices. A comparison between the ZNN and linear programming (LP) approach is presented.

Keywords: weighted finite automata; Zhang neural network; forward simulation; backward simulation; pseudoinverse

MSC: 65F20; 68T05; 68Q70

check for updates

Citation: Stanimirović, P.S.; Ćirić, M.; Mourtas, S.D.; Brzaković, P.; Karabašević, D. Simulations and Bisimulations between Weighted Finite Automata Based on Time-Varying Models over Real Numbers. *Mathematics* **2024**, *12*, 2110. https://doi.org/10.3390/math12132110

Academic Editor: Francisco Rodrigues Lima Junior

Received: 5 June 2024
Revised: 27 June 2024
Accepted: 3 July 2024
Published: 5 July 2024

1. Preliminaries on Weighted Finite Automata and Zeroing Neural Networks

Simulations between WFA ensure its containment, while bisimulations ensure the equivalence of WFA. As a result of the transition from various boolean to quantitative systems, both simulations and bisimulations become quantitative. Corresponding models are based on the use of matrices whose entries supply a quantitative measurement of the relationship between states of underlying systems.

Hereafter, $\mathbb{R}$ denotes the field of real numbers, and $\mathbb{N}$ denotes the set of natural numbers without zero, while the set of all positive real numbers is denoted by $\mathbb{R}_+$. Additionally, $X = \{x_1, \ldots, x_r\}$ is a non-empty finite set with k elements, where $k \in \mathbb{N}$, called an *alphabet*, while $X^+ = \{x_1x_2\dots x_s \mid s \in \mathbb{N}, x_1, x_2, \dots, x_s \in X\}$ is the set of all finite sequences of elements of X, which are called *words* over the alphabet X, and $X^* = X^+ \cup \{\varepsilon\}$, where $\varepsilon \notin X^+$ is a symbol that denotes the *empty word* of length 0. With respect to the conventional concatenation operation on words (sequences), X^+ forms a semigroup, while X^* is a structure representing a monoid with the identity element ε.

A *weighted finite automaton* over the field of real numbers $\mathbb{R}$ and the alphabet X is defined as a quadruple $\mathcal{A} = \big(m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A\big)$, where $m \in \mathbb{N}$ denotes the *dimension* of $\mathcal{A}$; $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ are the *initial vector* and *terminal vector*, respectively, and $\{M_x^A\}_{x\in X} \subset \mathbb{R}^{m\times m}$ is a collection of *transition matrices*. The initial vector σ^A is treated as a row vector, while the terminal vector τ^A is treated as a column vector. The behavior of a weighted finite automaton is expressed as the product σ^A, representing the initial weights, matrices $\{M_x^A\}_{x\in X}$ representing the weights of the transitions induced by input letters, and the column vector τ^A representing the terminal weights.

The collection $\{M_x^A\}_{x\in X}$ is extended up to a collection $\{M_u^A\}_{u\in X^*} \subset \mathbb{R}^{m\times m}$ of *compound transition matrices* expressed as

$$M_u^A = \begin{cases} I_m, & u = \varepsilon, \\ M_{x_1}^A\, M_{x_2}^A \cdots M_{x_s}^A, & u = x_1x_2\cdots x_s \in X^+, \end{cases} \tag{1}$$

where I_m denotes the $m \times m$ identity matrix. The matrices M_u^A, $u \in X^*$, defined in (1), are known as the *compound transition matrices* of $\mathcal{A}$. The multiplication of transition matrices carry numerical values over $\mathbb{R}$, known as weights. A function $f : X^* \to \mathbb{R}$ is called a *word function*. In particular, each weighted finite automaton $\mathcal{A} = \big(m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A\big)$ gives rise to a word function $[\![\mathcal{A}]\!] : X^* \to \mathbb{R}$ defined as follows:

$$[\![\mathcal{A}]\!](u) = \begin{cases} \sigma^A\, M_u^A\, \tau^A = \sigma^A\, M_{x_1}^A\, M_{x_2}^A \cdots M_{x_s}^A\, \tau^A, & u = x_1x_2\dots x_s \in X^+, \\ \sigma^A\, M_\varepsilon^A\, \tau^A = \sigma^A\, \tau^A, & u = \varepsilon. \end{cases} \tag{2}$$

The word function $[\![\mathcal{A}]\!]$ defined in (2) is called the *behavior* of $\mathcal{A}$, or a *word function computed by* $\mathcal{A}$. The behavior of an automaton is a mapping that relates a weight to words over a semiring.

Consider the weighted finite automata (WFA) $\mathcal{A} = \big(m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A\big)$ and $\mathcal{B} = \big(n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B\big)$ over the field of real numbers $\mathbb{R}$ and X. The following notations are used:

$[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!] \iff [\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u)$, for every $u \in X^*$;

$[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!] \iff [\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$, for every $u \in X^*$.

WFA $\mathcal{A}$ and $\mathcal{B}$ over $\mathbb{R}$ and the alphabet X are said to be *equivalent* if $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$. On the other hand, if $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$, then $\mathcal{A}$ is said to be *contained in* $\mathcal{B}$. The problem of determining whether WFA are equivalent is called the *equivalence problem*, and the problem of determining whether one of two WFA is contained in another is called the *containment problem*. A solution to the equivalence problem decides whether two WFA compute the same word function. On the other hand, a solution to the containment problem determines whether the word function computed by one WFA is less than or equal to the word function corresponding to the other WFA

A matrix (resp. vector) is said to be a *positive matrix* (resp. *positive vector*) if all its entries are positive real numbers, and a weighted finite automaton $\mathcal{A}$ is said to be a *positive automaton* if its initial and terminal vectors, as well as all its transition matrices, are positive.

Weighted automata have been applied to describe quantitative properties in various systems, as well as to represent probabilistic models, image compression, speech recognition, and finite representations of formal languages. Context–free grammars are used in the development of programming languages as well as in artificial intelligence.

The theoretical foundations of current investigations involve two types of simulations and two types of bisimulations defined in [1], in the general context of WFA over a semiring. The approach we use consists of defining quantitative simulations and bisimulations as matrices that are solutions to certain systems of matrix inequations. Such an approach was introduced in [2], where quantitative simulations and bisimulations between fuzzy finite automata were introduced and their basic properties were examined. Algorithms for testing their existence were developed in [3]. The same algorithms compute the greatest simulations and bisimulations in cases when they exist. Then, the same approach was applied to the study of bisimulations and simulations for non-deterministic automata [4], WFA over an additively idempotent semiring [5], and max-plus automata [6], as well as for WFA over an arbitrary semiring [1,7], which encompass all the previous ones. It turns out that an almost identical methodology can also be applied to social networks [8]. In [9], it was proven that two probabilistic finite automata are equivalent if and only if there is a bisimulation between them, where the bisimulation is defined as a classical binary relation between the vector spaces corresponding to those automata.

In the present paper, we investigate forward and backward simulations and bisimulations for WFA over the field of real numbers. It is worth noting that there are some very important specifics in this case. For most WFA types, the *problem of equivalence* (determining whether two automata compute the same word function) and the *minimization problem* (determining an automaton with the minimal number of states equivalent to a given automaton) are computationally hard. In these cases, bisimulations have two very important roles. The first role is to provide an efficient procedure for witnessing the existence of the equivalence of two automata, and the second one is to provide an efficient way to construct an automaton equivalent to a given one, with a not necessarily minimal but reasonably smaller number of states. However, it is not the case with WFA over the field of real numbers, for which there are efficient algorithms for testing the equivalence and performing minimization. Despite this observation, the importance of bisimulations for these automata is not diminished. Bisimulations are still needed as a means of determining the measure of similarity between the states of different automata, which algorithms for testing the equivalence are unable to do. In the context of weighted automata over the field of real numbers, such measures have already been studied in [10] by means of bisimulation seminorms and pseudometrics, and in [11] by means of linear bisimulations; in our upcoming research, we will deal with the relationships between bisimulation seminorms, linear bisimulations, and our concepts of bisimulations.

Following the definitions of simulations and bisimulations over various algebraic structures, an analogous approach has been used in defining simulations and bisimulations for WFA over the field of real numbers. The problem of simulations and bisimulations for WFA over the field of real numbers reduces to the system of two vector inequations and a number of matrix inequations. There is a notable lack of numerical methods for solving simulation and bisimulationproblems. Urabe and Hasuo proposed the idea of reducing the problem of testing the existence of simulations to the problem of linear programming (LP) and implemented it in [7] (Section 5). Seen more generally, the research described in this paper shows that the ZNN design is usable in solving systems of matrix and vector inequations in linear algebra. Our goal is to show that the zeroing neural network (ZNN) dynamics are an effective tool to decide on the containment or equivalence between WFA. A comparison between the ZNN and LP approach is presented.

On the other hand, the application of dynamical systems is a robust tool for solving various matrix algebra problems, primarily owing to the global exponential convergence, parallel distributed essence, convenience of hardware implementation, suitability for online computations involving TV objects, and possibility of providing convergence in a finite time frame [12,13]. First, ZNN models have been used to solve the TV matrix inversion problem [14]. Standard and finite-time convergent ZNN dynamical systems aimed at solving time-varying (TV) linear matrix equations have been widely investigated [12,15–18]. The applications of ZNN design, mainly focusing on robot manipulator path tracking, motion planning, and chaotic systems, were surveyed in [19]. ZNN dynamical systems for solving TV linear matrix–vector inequalities (TVLMVI) and TV linear matrix inequalities (TVLMI) have been broadly investigated [12,15,20–27]. Moreover, various ZNN models for solving TVLMI have been applied, mainly in obstacle avoidance for redundant robots and robot manipulator control [12,28,29]. Typically, TVLMVI and TVLMI of type "$\leq$" are solved by utilizing an additional matrix or vector of appropriate dimensions with non-negative entries. A TV matrix inequality of the Stein form $A(t)X(t)B(t) + X(t) \leq C(t)$ was considered in [21]. A TVLMVI problem of the general form $A(t)x(t) \leq b(t)$ was considered in [24,26,27]. Two ZNN models for solving systems of two TVLMVI were developed in [15]. In [22], the authors proposed ZNNs for solving TV nonlinear inequalities. Finite-time dynamics for solving general TVLMVI $A(t)X(t)B(t) \leq C(t)$ were proposed in [25]. A comparison between ZNN and gradient-based networks for solving $A(t)x(t) \leq b(t)$ was investigated in [23]. The computational time for solving TV equations increases due to the large number of calculations of TV requirements [30].

The problem under consideration is more complex because it requires us to solve systems of linear matrix and vector inequations. The structure of ZNN models developed in the current research is based on composite models with a prescribed number of error functions in matrix form and two in vector form. The ZNN dynamics aim to force the convergence of the involved error functions to zero over the considered time interval [13]. But the ZNN model in this paper aims to solve several matrix–vector equations that are inconsistent in the general case. Our strategy is to utilize ZNN neurodynamics to generate simulations between two WFA with weights over real numbers. In this way, our objective involves the topic of numerical linear algebra.

This research is aimed at the development and analysis of four novel ZNN models for addressing the systems of matrix and vector inequalities involved in simulations between WFA. The problem considered in this paper is specific and complex, and it requires solving a system of two vector inequations and a couple of matrix inequations. Using positive slack matrices, matrix and vector inequalities are transformed into corresponding equalities. In this case, it is useful to utilize the development of ZNN dynamics based on several inequalities and Zhang error functions. ZNN algorithms established upon a few error functions have been investigated in several studies, such as [31–34]. Our motivation for the application of ZNN arises from a verified fact that it is a powerful tool for solving various matrix algebra models, possessing global exponential convergence and a parallel distributed structure [12,13]. Therefore, it is interesting to construct the ZNN evolution for such a problem and study its behavior. A detailed convergence analysis is considered. Numerical examples are performed with different initial state matrices.

The main results are emphasized as follows.

(1) Two types of quantitative simulations and two types of bisimulations between WFA are determined as solutions to particular systems of several matrix and two vector inequations over $\mathbb{R}$.
(2) The approach used to solve the problem of simulations and bisimulations in this research is unique and based on the application of the ZNN dynamical evolution in solving underlying matrix and vector inequations.
(3) A detailed convergence analysis of the proposed ZNN dynamics is presented.
(4) Numerical examples are performed under different initial state matrices, and a comparison between the ZNN and LP approach is presented.

The overall organization of the sections is as follows. Preliminaries on WFA and ZNN are presented in Section 1. Global results are highlighted in the same section. Two types of simulations and four types of bisimulations proposed in [1] in the general context of WFA over a semiring are generalized in the context of WFA over the field of real numbers in Section 2. ZNN designs for simulations and bisimulations of WFA over real numbers are presented in Section 3. Section 4 is aimed at testing the developed ZNN dynamical systems and making comparisons with the LP solver. Concluding remarks are given in Section 5.

2. Simulations and Bisimulations of WFA over Real Numbers

As a continuation of the research presented in [1], here we correspondingly introduce definitions of two types of simulations and two types of bisimulations in the context of WFA over $\mathbb{R}$. For this purpose, consider two WFA $\mathcal{A} = \big(m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A\big)$ and $\mathcal{B} = \big(n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B\big)$ over the field of real numbers $\mathbb{R}$ and the alphabet X. A matrix $U \in \mathbb{R}^{m\times n}$ is called a *forward simulation* between $\mathcal{A}$ and $\mathcal{B}$ if it satisfies the following conditions with respect to U:

$$\begin{aligned} &\text{(fs-1)} \quad \sigma^A \leqslant \sigma^B U^\top \\ &\text{(fs-2)} \quad U^\top M_x^A \leqslant M_x^B U^\top \quad (\forall x \in X) \\ &\text{(fs-3)} \quad U^\top \tau^A \leqslant \tau^B, \end{aligned} \tag{3}$$

and it is termed as *backward simulation* between $\mathcal{A}$ and $\mathcal{B}$ if it fulfills

$$\begin{aligned} &\text{(bs-1)} \quad \tau^A \leqslant U\tau^B \\ &\text{(bs-2)} \quad M_x^A U \leqslant U M_x^B \quad (\forall x \in X) \\ &\text{(bs-3)} \quad \sigma^A U \leqslant \sigma^B. \end{aligned} \tag{4}$$

Our intention is to apply the notion of transposed automaton from [35] to reverse the transitions' flow direction. If both U and $U^\top$ are forward simulations between $\mathcal{A}$ and $\mathcal{B}$ and vice versa, i.e., if they fulfil

$$\begin{aligned} &\text{(fb-1)} \ \sigma^A \leqslant \sigma^B U^\top, \quad \sigma^B \leqslant \sigma^A U \\ &\text{(fb-2)} \ U^\top M_x^A \leqslant M_x^B U^\top, \quad U M_x^B \leqslant M_x^A U \quad (\forall x \in X) \\ &\text{(fb-3)} \ U^\top \tau^A \leqslant \tau^B, \quad U\tau^B \leqslant \tau^A \end{aligned} \tag{5}$$

then U is termed as a *forward bisimulation* between $\mathcal{A}$ and $\mathcal{B}$, and if both U and $U^\top$ are backward simulations between $\mathcal{A}$ and $\mathcal{B}$ and vice versa, i.e., if they satisfy

$$\begin{aligned} &\text{(bb-1)} \ \tau^A \leqslant U\tau^B, \quad \tau^B \leqslant U^\top \tau^A \\ &\text{(bb-2)} \ M_x^A U \leqslant U M_x^B, \quad M_x^B U^\top \leqslant U^\top M_x^A \quad (\forall x \in X) \\ &\text{(bb-3)} \ \sigma^A U \leqslant \sigma^B, \quad \sigma^B U^\top \leqslant \sigma^A \end{aligned} \tag{6}$$

then U is known as a *backward bisimulation* between $\mathcal{A}$ and $\mathcal{B}$.

It is important to note that, for any $\omega \in \{\text{fs}, \text{bs}, \text{fb}, \text{bb}\}$, the conditions ($\omega$-1), ($\omega$-2), and ($\omega$-3) can be treated a system of matrix inequations with the unknown matrix U, and simulations or bisimulations of type ω are precisely solutions to this system. This is extremely important because simulations between weighted automata over the field of real numbers are searched for by solving the corresponding systems of matrix inequalities.

Another important note is that the main role of simulations is to witness containment between automata $\mathcal{A}$ and $\mathcal{B}$, while the main role of bisimulations is to witness equivalence between $\mathcal{A}$ and $\mathcal{B}$. However, forward and backward simulations and bisimulations are defined by matrix inequations. On that note, in order to prove that simulations achieve containment and bisimulations achieve equivalence, we need the inequations to be preserved by multiplying, on either side, by the transition matrices, as well as by the initial and terminal vectors. Multiplication by matrices and vectors containing negative entries can violate inequalities, and, therefore, in order for simulations and bisimulations defined by systems of inequations to make full sense, we consider these types of bisimulations and simulations only between positive automata.

Theorem 1 is a modified version of [1] (Theorem 1).

Theorem 1. *The following statements are valid for positive WFA $\mathcal{A}$ and $\mathcal{B}$ over $\mathbb{R}$:*

(a) *For $\omega \in \{\text{fs}, \text{bs}\}$, if there is a simulation of type ω between $\mathcal{A}$ and $\mathcal{B}$, then $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$.*

(b) *For $\omega \in \{\text{fb}, \text{bb}\}$, if there is a bisimulation of type ω between $\mathcal{A}$ and $\mathcal{B}$, then $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$.*

The modification is reflected in the following. A slightly different version of Theorem 1 was proved in [1] [Theorem 1] for WFA over a positive semiring. Theorem 1 could also be formulated for $\mathcal{A}$ and $\mathcal{B}$ as WFA over the positive semiring $\mathbb{R}_+$ of nonnegative real numbers, but such a formulation would mean that the simulations and bisimulations between $\mathcal{A}$ and $\mathcal{B}$ should also be over the semiring $\mathbb{R}_+$, that is, they should be positive matrices, which is not necessary. Namely, for positive WFA over an arbitrary ordered semiring (not necessarily positive), the proof of [1] (Theorem 1) also holds for simulations and bisimulations that contain negative entries, and Theorem 1 is formulated to allow for such simulations and bisimulations as well.

As this article is primarily concerned with solving systems of matrix inequations, nothing important will change if we consider the more general case and allow the transition matrices, as well as the initial and terminal vectors, to have negative entries, which is performed below. On the other hand, in some applications of simulations and bisimulations, for example in the dimensionality reduction for WFA, there is a need to find positive solutions of the considered systems of matrix inequations. For this reason, we consider systems with an additional condition requiring the positivity of the solution. It should be noted that the proposed procedures for solving the systems remain valid even in the case when this condition is omitted, and in the same way, in that case we obtain solutions that do not have to be positive.

3. ZNN Designs for Simulations and Bisimulations of WFA over Real Numbers

This section defines and analyzes four novel ZNN models for addressing the systems of inequations (3)–(6). For the remainder of this section, let $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ be two WFA over $\mathbb{R}$, where $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$ with $i = 1, \dots r$.

Also, it is crucial to mention that the process of building a ZNN model usually involves two primary steps. The error matrix equation's (EME) function, $E(t)$, must be initially declared. Secondly, the dynamic system represented by the continuous differential equation of the general form

$$\dot{E}(t) = -\lambda E(t), \tag{7}$$

needs to be employed. The dynamical evolution (7) relates the time derivative $\dot{E}(t)$ to $E(t)$ in proportion to the positive real coefficient λ. The convergence rate of the dynamical system (7) is altered by manipulating the parameter $\lambda \in \mathbb{R}^+$. More precisely, with increasing values of λ, any ZNN model converges even faster [13,36,37]. The primary goal of the dynamics (7) is to force $E(t)$ to approach 0 as $t \to \infty$. The continuous learning principle that emerges from the EME's construction in Equation (7) is used to manage this goal. EME is, therefore, considered as a tracking indication in the context of the ZNN model's learning.

Special attention should be paid to a few notations that are used in the remainder of this work. The $p \times 1$ matrices with all ones and all zeros as entries are indicated by $\mathbf{1}_p$ and $\mathbf{0}_p$, whereas the $p \times r$ matrices with all ones and all zeros as entries are indicated by $\mathbf{1}_{p,r}$ and $\mathbf{0}_{p,r}$. Furthermore, the $p \times p$ identity matrix is indicated by I_p, whereas $\mathrm{vec}()$, $\otimes$, $\odot$, $(\,)^{\odot}$, $(\,)^{\dagger}$, and $\|\,\|_{\mathrm{F}}$ stand for the vectorization process, the Kronecker product, the Hadamard (or elementwise) product, the Hadamard exponential, pseudoinversion, and the matrix Frobenius norm, respectively. Finally, $\mathrm{rand}(m, n)$ denotes an $m \times n$ matrix whose entries consist of random numbers.

3.1. The ZNN-fs Model

In line with (3), the following group of inequations must be satisfied:

$$\begin{cases} U^{\mathrm{T}}(t)\tau^A - \tau^B \leqslant \mathbf{0}_n, \\ \sigma^A - \sigma^B U^{\mathrm{T}}(t) \leqslant \mathbf{0}_m^{\mathrm{T}}, \\ U^{\mathrm{T}}(t) M_{x_i}^A - M_{x_i}^B U^{\mathrm{T}}(t) \leqslant \mathbf{0}_{n,m},\ i = 1, \dots, r, \\ U(t) \geqslant \mathbf{0}_{m,n}, \end{cases} \tag{8}$$

with respect to an unknown matrix $U(t) \in \mathbb{R}^{m\times n}$. Utilizing the vectorization in conjunction with the Kronecker product, the system (8) is reformulated into the vector inequations form

$$\begin{cases} ((\tau^A)^{\mathrm{T}} \otimes I_n)\mathrm{vec}(U^{\mathrm{T}}(t)) - \tau^B \leqslant \mathbf{0}_n, \\ -(I_m \otimes \sigma^B)\mathrm{vec}(U^{\mathrm{T}}(t)) + (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ \left((M_{x_i}^A)^{\mathrm{T}} \otimes I_n - I_m \otimes M_{x_i}^B\right)\mathrm{vec}(U^{\mathrm{T}}(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}. \end{cases} \tag{9}$$

To calculate $U(t)$ more efficiently, (9) must be simplified. Thus, the vectorization-related Lemma 1 derived from [38] is given.

Lemma 1. *The vectorization* $\mathrm{vec}(W^{\mathrm{T}}) \in \mathbb{R}^{mn}$ *of the transpose* W^{T} *of* $W \in \mathbb{R}^{m\times n}$ *is defined by*

$$\mathrm{vec}(W^{\mathrm{T}}) = P\,\mathrm{vec}(W), \tag{10}$$

where $P \in \mathbb{R}^{mn\times mn}$ *is a constant permutation matrix that depends on the number of columns* n *and number of rows* m *in* W.

The algorithmic procedure for generating the permutation matrix P in (10) is presented in the following Algorithm 1.

Algorithm 1 The permutation matrix P formation.

```
Input: The number of rows $m$ and columns $n$ of a matrix $W \in \mathbb{R}^{m\times n}$.
1: procedure PERM_MAT($m, n$)
2:     Put $g$ =eye($mn$) and $W$ =reshape($1 : mn, n, m$)
3:     return $P = g$(:,reshape($W^{\mathrm{T}}, 1, mn$))
4: end procedure
Output: $P$
```

Using the permutation matrix P for generating $\mathrm{vec}(U^{\mathrm{T}}(t))$, inequations (9) can be rewritten in the form

$$\begin{cases} ((\tau^A)^{\mathrm{T}} \otimes I_n)P\,\mathrm{vec}(U(t)) - \tau^B \leqslant \mathbf{0}_n, \\ -(I_m \otimes \sigma^B)P\,\mathrm{vec}(U(t)) + (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ \left((M_{x_i}^A)^{\mathrm{T}} \otimes I_n - I_m \otimes M_{x_i}^B\right)P\,\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}, \end{cases} \tag{11}$$

wherein the last constraint imposes non-negativity on the solution. The corresponding block matrix form of (11) is given by

$$L_{fs}\,\mathrm{vec}(U(t)) - \mathbf{b}_{fs} \leqslant \mathbf{0}_z, \tag{12}$$

such that $z = (r+1)mn + m + n$ and

$$L_{fs}=\begin{bmatrix}((\tau^A)^{\mathrm{T}}\otimes I_n)P\\ -(I_m\otimes\sigma^B)P\\ W_{fs}\\ -I_{mn}\end{bmatrix}\in\mathbb{R}^{z\times mn},\ \mathbf{b}_{fs}=\begin{bmatrix}\tau^B\\ -(\sigma^A)^{\mathrm{T}}\\ \mathbf{0}_{(r+1)mn}\end{bmatrix}\in\mathbb{R}^{z},\ W_{fs}=\begin{bmatrix}((M^A_{x_1})^{\mathrm{T}}\otimes I_n-I_m\otimes M^B_{x_1})P\\ ((M^A_{x_2})^{\mathrm{T}}\otimes I_n-I_m\otimes M^B_{x_2})P\\ \dots\\ ((M^A_{x_r})^{\mathrm{T}}\otimes I_n-I_m\otimes M^B_{x_r})P\end{bmatrix}\in\mathbb{R}^{rmn\times mn}. \tag{13}$$

Then, considering the vector of slack variables $K(t)=\begin{bmatrix}k_1(t)\\ \dots\\ k_z(t)\end{bmatrix}\in\mathbb{R}^z$, the inequation (12) can be converted into the corresponding equation

$$L_{fs}\,\mathrm{vec}(U(t))-\mathbf{b}_{fs}+K^{\odot 2}(t)=\mathbf{0}_z, \tag{14}$$

in which $K^{\odot 2}(t)=\begin{bmatrix}k_1^2(t)\\ \dots\\ k_z^2(t)\end{bmatrix}$ is the time-varying term with secured non-negative entries.

Thereafter, the ZNN approach considers the following EME, which is based on (12), to simultaneously satisfy all the inequations in (8):

$$E_{fs}(t)=L_{fs}\,\mathrm{vec}(U(t))-\mathbf{b}_{fs}+K^{\odot 2}(t), \tag{15}$$

where $U(t)$ and $K(t)$ are the unknown matrices that need to be found. The ZNN design (7) exploits the first time derivative of (15)

$$\dot{E}_{fs}(t)=L_{fs}\mathrm{vec}(\dot{U}(t))+2(I_z\odot K(t))\dot{K}(t). \tag{16}$$

Combining Equations (15) and (16) with the generic ZNN design (7), we obtain

$$L_{fs}\mathrm{vec}(\dot{U}(t))+2(I_z\odot K(t))\dot{K}(t)=-\lambda E_{fs}(t). \tag{17}$$

As a result, setting

$$H_{fs}=\begin{bmatrix}L_{fs} & 2(I_z\odot K(t))\end{bmatrix}\in\mathbb{R}^{z\times(mn+z)},\quad \dot{\mathbf{x}}(t)=\begin{bmatrix}\mathrm{vec}(\dot{U}(t))\\ \dot{K}(t)\end{bmatrix}\in\mathbb{R}^{mn+z},\quad \mathbf{x}(t)=\begin{bmatrix}\mathrm{vec}(U(t))\\ K(t)\end{bmatrix}\in\mathbb{R}^{mn+z},$$

the next system of linear equations with respect to $\dot{\mathbf{x}}$ is obtained:

$$H_{fs}\,\dot{\mathbf{x}}=-\lambda E_{fs}(t). \tag{18}$$

The ZNN dynamics are applicable in solving (18) if the mass matrix H_{fs} is invertible. To avoid this restriction, it is appropriate to use the pseudoinverse (best approximate) solution

$$\dot{\mathbf{x}}=H_{fs}^{\dagger}\Big(-\lambda E_{fs}(t)\Big). \tag{19}$$

An appropriate `ode` MATLAB R2022a solver can be used to handle the ZNN dynamics (19), additionally referred to as the ZNN-fs model. The ZNN-fs model's convergence and stability investigation is shown in Theorem 2.

Theorem 2. *Let* $\mathcal{A}=\left(m,\sigma^A,\left\{M^A_{x_i}\right\}_{x_i\in X},\tau^A\right)$ *and* $\mathcal{B}=\left(n,\sigma^B,\left\{M^B_{x_i}\right\}_{x_i\in X},\tau^B\right)$ *be the WFA over* $\mathbb{R}$ *and the alphabet* $X=\{x_1,\dots,x_r\}$*, where* $M^A_{x_i}\in\mathbb{R}^{m\times m}$, $\sigma^A\in\mathbb{R}^{1\times m}$, $\tau^A\in\mathbb{R}^{m\times 1}$ *and* $M^B_{x_i}\in\mathbb{R}^{n\times n}$, $\sigma^B\in\mathbb{R}^{1\times n}$, $\tau^B\in\mathbb{R}^{n\times 1}$ *with* $i=1,\dots,r$. *The dynamics (17) in linewith the ZNN method (7) lead to the theoretical solution (TSOL), determined by* $\mathbf{x}_{\mathcal{S}}(t)=\begin{bmatrix}\mathrm{vec}(U_{\mathcal{S}}(t))^{\mathrm{T}} & K^{\mathrm{T}}_{\mathcal{S}}(t)\end{bmatrix}^{\mathrm{T}}$*, which is stable according to Lyapunov.*

Proof. Let

$$\begin{cases} U_{\mathcal{S}}^{\mathrm{T}}(t)\tau^A - \tau^B \leqslant \mathbf{0}_n, \\ \sigma^A - \sigma^B U_{\mathcal{S}}^{\mathrm{T}}(t) \leqslant \mathbf{0}_m^{\mathrm{T}}, \\ U_{\mathcal{S}}^{\mathrm{T}}(t) M_{x_i}^A - M_{x_i}^B U_{\mathcal{S}}^{\mathrm{T}}(t) \leqslant \mathbf{0}_{n,m},\ i = 1, \dots, r, \\ U_{\mathcal{S}}(t) \geqslant \mathbf{0}_{m,n}. \end{cases} \tag{20}$$

Using vectorization, Kronecker product, and the permutation matrix P for constructing $\mathrm{vec}(U^{\mathrm{T}}(t))$, defined by Algorithm 1, the system (20) is reformulated as

$$\begin{cases} \left((\tau^A)^{\mathrm{T}} \otimes I_n\right) P\, \mathrm{vec}(U_{\mathcal{S}}(t)) - \tau^B \leqslant \mathbf{0}_n, \\ -\left(I_m \otimes \sigma^B\right) P\, \mathrm{vec}(U_{\mathcal{S}}(t)) + (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ \left((M_{x_i}^A)^{\mathrm{T}} \otimes I_n - I_m \otimes M_{x_i}^B\right) P\, \mathrm{vec}(U_{\mathcal{S}}(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ -\mathrm{vec}(U_{\mathcal{S}}(t)) \leqslant \mathbf{0}_{mn}. \end{cases} \tag{21}$$

The equivalent form of (21) is

$$L_{fs}\mathrm{vec}(U_{\mathcal{S}}(t)) - \mathbf{b}_{fs} \leqslant \mathbf{0}_z \tag{22}$$

where L_{fs} and $\mathbf{b}_{fs}$ are declared in Equation (13). Then, considering the slack variable $K_{\mathcal{S}}(t) \in \mathbb{R}^z$, the inequation (22) can be converted into the equation

$$L_{fs}\, \mathrm{vec}(U_{\mathcal{S}}(t)) - \mathbf{b}_{fs} + K_{\mathcal{S}}^{\odot 2}(t)(t) = \mathbf{0}_z,$$

in which $K_{\mathcal{S}}^{\odot 2}(t)$ is always a non-negative time-varying term.

The substitution

$$\mathbf{x}_{\mathcal{O}}(t) := -\mathbf{x}(t) + \mathbf{x}_{\mathcal{S}}(t) = \begin{bmatrix} -\mathrm{vec}(U(t)) + \mathrm{vec}(U_{\mathcal{S}}(t)) \\ -K(t) + K_{\mathcal{S}}(t) \end{bmatrix} := \begin{bmatrix} \mathrm{vec}(U_{\mathcal{O}}(t)) \\ K_{\mathcal{O}}(t) \end{bmatrix}$$

gives

$$\mathbf{x}(t) = \mathbf{x}_{\mathcal{S}}(t) - \mathbf{x}_{\mathcal{O}}(t) = \begin{bmatrix} \mathrm{vec}(U_{\mathcal{S}}(t)) - \mathrm{vec}(U_{\mathcal{O}}(t)) \\ K_{\mathcal{S}}(t) - K_{\mathcal{O}}(t) \end{bmatrix}.$$

The 1st derivative of $\mathbf{x}(t)$ is equal to

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_{\mathcal{S}}(t) - \dot{\mathbf{x}}_{\mathcal{O}}(t) = \begin{bmatrix} \mathrm{vec}(\dot{U}_{\mathcal{S}}(t)) - \mathrm{vec}(\dot{U}_{\mathcal{O}}(t)) \\ \dot{K}_{\mathcal{S}}(t) - \dot{K}_{\mathcal{O}}(t) \end{bmatrix}.$$

As a result, after substituting (14) for $\mathbf{x}(t) = \mathbf{x}_{\mathcal{S}}(t) - \mathbf{x}_{\mathcal{O}}(t)$, the following holds

$$E_{\mathcal{S}}(t) = L_{fs}(\mathrm{vec}(U_{\mathcal{S}}(t)) - \mathrm{vec}(U_{\mathcal{O}}(t))) - \mathbf{b}_{fs} + (K_{\mathcal{S}}(t) - K_{\mathcal{O}}(t))^{\odot 2},$$

or

$$E_{\mathcal{S}}(t) = \begin{bmatrix} L_{fs} & (I_z \odot (K_{\mathcal{S}}(t) - K_{\mathcal{O}}(t))) \end{bmatrix} (\mathbf{x}_{\mathcal{S}}(t) - \mathbf{x}_{\mathcal{O}}(t)) - \mathbf{b}_{fs},$$

where L_{fs} and $\mathbf{b}_{fs}$ are declared in (13). Then, the following results follow from (7):

$$\dot{E}_{\mathcal{S}}(t) = L_{fs}\mathrm{vec}\left(\dot{U}(t) + \dot{U}(t)\right) + 2(I_z \odot (K_{\mathcal{S}}(t) - K_{\mathcal{O}}(t)))\left(\dot{K}_{\mathcal{S}}(t) - \dot{K}_{\mathcal{O}}(t)\right) = -\lambda E_{\mathcal{S}}(t),$$

or equivalently

$$\dot{E}_{\mathcal{S}}(t) = \begin{bmatrix} L_{fs} & 2(I_z \odot (K_{\mathcal{S}}(t) - K_{\mathcal{O}}(t))) \end{bmatrix} (\dot{\mathbf{x}}_{\mathcal{S}}(t) - \dot{\mathbf{x}}_{\mathcal{O}}(t)) = -\lambda E_{\mathcal{S}}(t). \tag{23}$$

Next, for confirming the convergence, we choose the plausible Lyapunov function

$$\mathcal{Z}(t) = \frac{1}{2}\|E_{\mathcal{S}}(t)\|_{\mathrm{F}}^2 = \frac{1}{2}\mathrm{tr}\left(E_{\mathcal{S}}(t)(E_{\mathcal{S}}(t))^{\mathrm{T}}\right).$$

The following is confirmed for $\mathcal{Z}(t)$:

$$\dot{\mathcal{Z}}(t)=\frac{2\text{tr}\Big((E_{\mathcal{S}}(t))^{\mathrm{T}}\dot{E}_{\mathcal{S}}(t)\Big)}{2}=\text{tr}\Big((E_{\mathcal{S}}(t))^{\mathrm{T}}\dot{E}_{\mathcal{S}}(t)\Big)=-\lambda\text{tr}\Big((E_{\mathcal{S}}(t))^{\mathrm{T}}E_{\mathcal{S}}(t)\Big). \tag{24}$$

Because of (24), the following is valid:

$$\begin{aligned}
&\dot{\mathcal{Z}}(t)\begin{cases}<0, E_{\mathcal{S}}(t)\neq 0,\\ =0, E_{\mathcal{S}}(t)=0,\end{cases}\\
\Leftrightarrow&\dot{\mathcal{Z}}(t)\begin{cases}<0, \Big[L_{fs}\quad (I_z\odot(K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)))\Big](\mathbf{x}_{\mathcal{S}}(t)-\mathbf{x}_{\mathcal{O}}(t))-\mathbf{b}_{fs}\neq 0,\\ =0, \Big[L_{fs}\quad (I_z\odot(K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)))\Big](\mathbf{x}_{\mathcal{S}}(t)-\mathbf{x}_{\mathcal{O}}(t))-\mathbf{b}_{fs}=0,\end{cases}\\
\Leftrightarrow&\dot{\mathcal{Z}}(t)\begin{cases}<0, \Big[L_{fs}\quad (I_z\odot(K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)))\Big]\begin{bmatrix}\text{vec}(U_{\mathcal{S}}(t))-\text{vec}(U_{\mathcal{O}}(t))\\ K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)\end{bmatrix}-\mathbf{b}_{fs}\neq 0,\\ =0, \Big[L_{fs}\quad (I_z\odot(K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)))\Big]\begin{bmatrix}\text{vec}(U_{\mathcal{S}}(t))-\text{vec}(U_{\mathcal{O}}(t))\\ K_{\mathcal{S}}(t)-K_{\mathcal{O}}(t)\end{bmatrix}-\mathbf{b}_{fs}=0,\end{cases}\\
\Leftrightarrow&\dot{\mathcal{Z}}(t)\begin{cases}<0, \quad \begin{bmatrix}\text{vec}(U_{\mathcal{O}}(t))\\ K_{\mathcal{O}}(t)\end{bmatrix}\neq 0,\\ =0, \quad \begin{bmatrix}\text{vec}(U_{\mathcal{O}}(t))\\ K_{\mathcal{O}}(t)\end{bmatrix}=0.\end{cases}\\
\Leftrightarrow&\dot{\mathcal{Z}}(t)\begin{cases}<0, \quad \mathbf{x}_{\mathcal{O}}(t)\neq 0,\\ =0, \quad \mathbf{x}_{\mathcal{O}}(t)=0.\end{cases}
\end{aligned}$$

With $\mathbf{x}_{\mathcal{O}}(t)$ being the equilibrium point of the system (23), we have

$$\forall\, \mathbf{x}_{\mathcal{O}}(t)\neq 0, \quad \dot{\mathcal{Z}}(t)\leq 0.$$

It appears that the equilibrium state

$$\mathbf{x}_{\mathcal{O}}(t)=-\mathbf{x}(t)+\mathbf{x}_{\mathcal{S}}(t)=\begin{bmatrix}-\text{vec}(U(t))+\text{vec}(U_{\mathcal{S}}(t))\\ -K(t)+K_{\mathcal{S}}(t)\end{bmatrix}=0$$

is stable in accordance with Lyapunov theory. Afterwards, when $t\to\infty$, the following holds:

$$\mathbf{x}(t)=\begin{bmatrix}\text{vec}(U(t))\\ K(t)\end{bmatrix}\to\mathbf{x}_{\mathcal{S}}(t)=\begin{bmatrix}\text{vec}(U_{\mathcal{S}}(t))\\ K_{\mathcal{S}}(t)\end{bmatrix},$$

which finalizes the proof. $\square$

Theorem 3. *Let* $\mathcal{A}=\left(m,\sigma^A,\left\{M^A_{x_i}\right\}_{x_i\in X},\tau^A\right)$ *and* $\mathcal{B}=\left(n,\sigma^B,\left\{M^B_{x_i}\right\}_{x_i\in X},\tau^B\right)$ *be the WFA over* $\mathbb{R}$ *and* $X=\{x_1,\dots,x_r\}$*, where* $M^A_{x_i}\in\mathbb{R}^{m\times m}$*,* $\sigma^A\in\mathbb{R}^{1\times m}$*,* $\tau^A\in\mathbb{R}^{m\times 1}$ *and* $M^B_{x_i}\in\mathbb{R}^{n\times n}$*,* $\sigma^B\in\mathbb{R}^{1\times n}$*,* $\tau^B\in\mathbb{R}^{n\times 1}$ *with* $i=1,\dots,r$*. Beginning from any initial point* $\mathbf{x}(0)$*, the ZNN-fs model of* (19) *converges exponentially to* $\mathbf{x}^*(t)$*, which refers to the TSOL of* (3)*.*

Proof. Firstly, the system of (8) is considered to find the solution $\mathbf{x}(t)=[\text{vec}(U(t))^{\mathrm{T}},\ K^{\mathrm{T}}(t)]^{\mathrm{T}}$ that is affiliated to the time-varying backward-forward bisimulation between $\mathcal{A}$ and $\mathcal{B}$ of (3). Secondly, the system of (8) is reformulated into the system of (9) utilizing vectorization and the Kronecker product and, then, into the system of (12) utilizing the operational permutation matrix P for $\text{vec}(U^{\mathrm{T}}(t))$. Thirdly, considering the slack variable $K(t)$, the inequality constraint of the system of (12) is converted into an equality constraint in the system of (14). Fourthly, the EME of (15) is constructed, in keeping with the ZNN technique and the system of (14), to generate the solution $\mathbf{x}(t)$ that is affiliated with the system of (3).

Fifthly, the model of (17) is yielded in accordance to the ZNN technique of (7) for zeroing (15). According to Theorem 2, the EME of (15) converges to zero as $t \to \infty$. Consequently, the solution of (19) converges to $\mathbf{x}^*(t) = \left[\mathrm{vec}(U^*(t))^{\mathrm{T}},\ (K^*(t))^{\mathrm{T}}\right]^{\mathrm{T}}$ as $t \to \infty$. Furthermore, it is obvious that (19) is (17) in a different form because of the derivation process. After that, the proof is accomplished. $\square$

3.2. The ZNN-bs Model

In line with (4), the following group of inequations must be satisfied:

$$\begin{cases} \tau^A - U(t)\tau^B \leqslant \mathbf{0}_m, \\ \sigma^A U(t) - \sigma^B \leqslant \mathbf{0}_n^{\mathrm{T}}, \\ M_{x_i}^A U(t) - U(t) M_{x_i}^B \leqslant \mathbf{0}_{m,n},\ i = 1, \dots, r, \\ U(t) \geqslant \mathbf{0}_{m,n}, \end{cases} \tag{25}$$

where $U(t) \in \mathbb{R}^{m \times n}$ denotes the unknown matrix to be found. Utilizing vectorization and the Kronecker product, the system of inequations (25) is rewritten in the equivalent form

$$\begin{cases} -\left((\tau^B)^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) + \tau^A \leqslant \mathbf{0}_m, \\ (I_n \otimes \sigma^A)\mathrm{vec}(U(t)) - (\sigma^B)^{\mathrm{T}} \leqslant \mathbf{0}_n, \\ \left(I_n \otimes M_{x_i}^A - (M_{x_i}^B)^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}, \end{cases}$$

and its corresponding matrix form is

$$L_{\mathrm{bs}}\mathrm{vec}(U(t)) - \mathbf{b}_{\mathrm{bs}} \leqslant \mathbf{0}_z, \tag{26}$$

where

$$L_{\mathrm{bs}} = \begin{bmatrix} -(\tau^B)^{\mathrm{T}} \otimes I_m \\ I_n \otimes \sigma^A \\ W_{\mathrm{bs}} \\ -I_{mn} \end{bmatrix} \in \mathbb{R}^{z \times mn},\ \ \mathbf{b}_{\mathrm{bs}} = \begin{bmatrix} -\tau^A \\ (\sigma^B)^{\mathrm{T}} \\ \mathbf{0}_{(r+1)mn} \end{bmatrix} \in \mathbb{R}^z,\ \ W_{\mathrm{bs}} = \begin{bmatrix} I_n \otimes M_{x_1}^A - (M_{x_1}^B)^{\mathrm{T}} \otimes I_m \\ I_n \otimes M_{x_2}^A - (M_{x_2}^B)^{\mathrm{T}} \otimes I_m \\ \cdots \\ I_n \otimes M_{x_r}^A - (M_{x_r}^B)^{\mathrm{T}} \otimes I_m \end{bmatrix} \in \mathbb{R}^{rmn \times mn}.$$

Then, considering the slack variable $K(t) \in \mathbb{R}^z$, the inequation (26) can be converted into the equation

$$L_{\mathrm{bs}}\mathrm{vec}(U(t)) - \mathbf{b}_{\mathrm{bs}} + K^{\odot 2}(t) = \mathbf{0}_z, \tag{27}$$

where $K^{\odot 2}(t)$ is always a non-negative time-varying term.

Thereafter, the ZNN approach considers the following EME, which is based on (27), for simultaneously satisfying all the inequations in (25):

$$E_{\mathrm{bs}}(t) = L_{\mathrm{bs}}\,\mathrm{vec}(U(t)) - \mathbf{b}_{\mathrm{bs}} + K^{\odot 2}(t), \tag{28}$$

where $U(t)$ and $K(t)$ are the unknown matrices to be found. The first time derivative of (28) is

$$\dot{E}_{\mathrm{bs}}(t) = L_{\mathrm{bs}}\,\mathrm{vec}(\dot{U}(t)) + 2(I_z \odot K(t))\dot{K}(t). \tag{29}$$

Then, combining Equations (28) and (29) with the ZNN design (7), we obtain

$$L_{\mathrm{bs}}\mathrm{vec}(\dot{U}(t)) + 2(I_z \odot K(t))\dot{K}(t) = -\lambda E_{\mathrm{bs}}(t). \tag{30}$$

As a result, setting

$$H_{\mathrm{bs}} = \begin{bmatrix} L_{\mathrm{bs}} & 2(I_z \odot K(t)) \end{bmatrix} \in \mathbb{R}^{z \times (mn+z)},\ \ \dot{\mathbf{x}}(t) = \begin{bmatrix} \mathrm{vec}(\dot{U}(t)) \\ \dot{K}(t) \end{bmatrix} \in \mathbb{R}^{mn+z},\ \ \mathbf{x}(t) = \begin{bmatrix} \mathrm{vec}(U(t)) \\ K(t) \end{bmatrix} \in \mathbb{R}^{mn+z},$$

the next model is obtained:

$$H_{\mathrm{bs}}\,\dot{\mathbf{x}} = -\lambda E_{\mathrm{bs}}(t). \tag{31}$$

Since the ZNN dynamics in solving (31) requires invertibility of the mass matrix H_{bs}, it is practical to use the best approximate solution to (31), which leads to

$$\dot{\mathbf{x}} = H_{\mathrm{bs}}^{\dagger}\left(-\lambda E_{\mathrm{bs}}(t)\right). \tag{32}$$

An appropriate `ode` MATLAB solver can be used to handle the ZNN model of (32), additionally referred to as the ZNN-bs flow. The ZNN-bs model's convergence and stability investigation is shown in Theorem 4.

Theorem 4. *Let $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ be WFA over $\mathbb{R}$, where $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$ with $i = 1,\ldots,r$. The dynamics (30) in line with the ZNN method of (7) lead to the TSOL, shown by $\mathbf{x}_{\mathcal{S}}(t) = \left[\mathrm{vec}(U_{\mathcal{S}}(t))^{\mathrm{T}} \quad K^{\mathrm{T}}_{\mathcal{S}}(t)\right]^{\mathrm{T}}$, which is stable according to Lyapunov.*

Proof. The proof is omitted since it is similar to the proof of Theorem 2. □

Theorem 5. *Let $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ be WFA over $\mathbb{R}$, where $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$ with $i = 1,\ldots,r$. Beginning from any initial point $\mathbf{x}(0)$, the ZNN-bs design (32) converges exponentially to $\mathbf{x}^*(t)$, which refers to the TSOL of (4).*

Proof. The proof is omitted since it is similar to the proof of Theorem 3. □

3.3. The ZNN-fb Model

In line with (5), the following group of inequations must be satisfied:

$$\begin{cases} U^{\mathrm{T}}(t)\tau^A - \tau^B \leqslant \mathbf{0}_n, \\ U(t)\tau^B - \tau^A \leqslant \mathbf{0}_m, \\ \sigma^A - \sigma^B U^{\mathrm{T}}(t) \leqslant \mathbf{0}^{\mathrm{T}}_m, \\ \sigma^B - \sigma^A U(t) \leqslant \mathbf{0}^{\mathrm{T}}_n, \\ U^{\mathrm{T}}(t) M^A_{x_i} - M^B_{x_i} U^{\mathrm{T}}(t) \leqslant \mathbf{0}_{n,m},\ i = 1,\ldots,r, \\ U(t) M^B_{x_i} - M^A_{x_i} U(t) \leqslant \mathbf{0}_{m,n},\ i = 1,\ldots,r, \\ U(t) \geqslant \mathbf{0}_{m,n}, \end{cases} \tag{33}$$

where $U(t) \in \mathbb{R}^{m\times n}$ implies the unknown matrix to be generated. Utilizing vectorization in combination with the Kronecker product, the system of (33) is reformulated as

$$\begin{cases} \left((\tau^A)^{\mathrm{T}} \otimes I_n\right)\mathrm{vec}(U^{\mathrm{T}}(t)) - \tau^B \leqslant \mathbf{0}_n, \\ \left((\tau^B)^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) - \tau^A \leqslant \mathbf{0}_m, \\ -\left(I_m \otimes \sigma^B\right)\mathrm{vec}(U^{\mathrm{T}}(t)) + (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ -\left(I_n \otimes \sigma^A\right)\mathrm{vec}(U(t)) + (\sigma^B)^{\mathrm{T}} \leqslant \mathbf{0}_n, \\ \left((M^A_{x_i})^{\mathrm{T}} \otimes I_n - I_m \otimes M^B_{x_i}\right)\mathrm{vec}(U^{\mathrm{T}}(t)) \leqslant \mathbf{0}_{mn},\ i = 1,\ldots,r, \\ \left((M^B_{x_i})^{\mathrm{T}} \otimes I_m - I_n \otimes M^A_{x_i}\right)\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1,\ldots,r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}. \end{cases} \tag{34}$$

Using the permutation matrix P for $\mathrm{vec}(U^{\mathrm{T}}(t))$, (34) is rewritten as

$$\begin{cases} ((\tau^A)^{\mathrm{T}} \otimes I_n) P \,\mathrm{vec}(U(t)) - \tau^B \leqslant \mathbf{0}_n, \\ ((\tau^B)^{\mathrm{T}} \otimes I_m)\mathrm{vec}(U(t)) - \tau^A \leqslant \mathbf{0}_m, \\ -(I_m \otimes \sigma^B) P \,\mathrm{vec}(U(t)) + (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ -(I_n \otimes \sigma^A)\mathrm{vec}(U(t)) + (\sigma^B)^{\mathrm{T}} \leqslant \mathbf{0}_n, \\ \left((M^A_{x_i})^{\mathrm{T}} \otimes I_n - I_m \otimes M^B_{x_i}\right) P\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ \left((M^B_{x_i})^{\mathrm{T}} \otimes I_m - I_n \otimes M^A_{x_i}\right)\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \dots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}, \end{cases}$$

and its corresponding matrix form is

$$L_{fb}\,\mathrm{vec}(U(t)) - \mathbf{b}_{fb} \leqslant \mathbf{0}_y, \tag{35}$$

where $y = (2r+1)mn + 2m + 2n$ and

$$L_{fb} = \begin{bmatrix} ((\tau^A)^{\mathrm{T}} \otimes I_n)P \\ (\tau^B)^{\mathrm{T}} \otimes I_m \\ -(I_m \otimes \sigma^B)P \\ -I_n \otimes \sigma^A \\ W_{fb} \\ -I_{mn} \end{bmatrix} \in \mathbb{R}^{y \times mn}, \quad \mathbf{b}_{fb} = \begin{bmatrix} \tau^B \\ \tau^A \\ -(\sigma^A)^{\mathrm{T}} \\ -(\sigma^B)^{\mathrm{T}} \\ \mathbf{0}_{(2r+1)mn} \end{bmatrix} \in \mathbb{R}^y, \quad W_{fb} = \begin{bmatrix} ((M^A_{x_1})^{\mathrm{T}} \otimes I_n - I_m \otimes M^B_{x_1})P \\ (M^B_{x_1})^{\mathrm{T}} \otimes I_m - I_n \otimes M^A_{x_1} \\ ((M^A_{x_2})^{\mathrm{T}} \otimes I_n - I_m \otimes M^B_{x_2})P \\ (M^B_{x_2})^{\mathrm{T}} \otimes I_m - I_n \otimes M^A_{x_2} \\ \dots \\ ((M^A_{x_r})^{\mathrm{T}} \otimes I_n - I_m \otimes M^B_{x_r})P \\ (M^B_{x_r})^{\mathrm{T}} \otimes I_m - I_n \otimes M^A_{x_r} \end{bmatrix} \in \mathbb{R}^{2rmn \times mn}.$$

Then, considering the slack variables vector $K(t) \in \mathbb{R}^y$, the inequation (35) is converted into the equation

$$L_{fb}\,\mathrm{vec}(U(t)) - \mathbf{b}_{fb} + K^{\odot 2}(t) = \mathbf{0}_y.$$

Thereafter, the ZNN approach considers the following EME, which is based on (35), for simultaneously satisfying all the inequations in (33):

$$E_{fb}(t) = L_{fb}\,\mathrm{vec}(U(t)) - \mathbf{b}_{fb} + K^{\odot 2}(t), \tag{36}$$

where $U(t)$ and $K(t)$ are the unknown matrices to be found. The first time derivative of (36) is equal to

$$\dot{E}_{fb}(t) = L_{fb}\,\mathrm{vec}(\dot{U}(t)) + 2(I_y \odot K(t))\dot{K}(t). \tag{37}$$

Then, combining Equations (36) and (37) with the ZNN design (7), we obtain the following:

$$L_{fb}\,\mathrm{vec}(\dot{U}(t)) + 2(I_y \odot K(t))\dot{K}(t) = -\lambda E_{fb}(t). \tag{38}$$

As a result, setting

$$H_{fb} = \begin{bmatrix} L_{fb} & 2(I_y \odot K(t)) \end{bmatrix} \in \mathbb{R}^{y \times (mn+y)}, \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} \mathrm{vec}(\dot{U}(t)) \\ \dot{K}(t) \end{bmatrix} \in \mathbb{R}^{mn+y}, \quad \mathbf{x}(t) = \begin{bmatrix} \mathrm{vec}(U(t)) \\ K(t) \end{bmatrix} \in \mathbb{R}^{mn+y},$$

(38) is transformed into the model

$$H_{fb}\,\dot{\mathbf{x}} = -\lambda E_{fb}(t)$$

whose pseudoinverse solution is equal to

$$\dot{\mathbf{x}} = H^{\dagger}_{fb}(-\lambda E_{fb}(t)). \tag{39}$$

An appropriate `ode` MATLAB solver can be used to handle the ZNN model (39), additionally referred to as the ZNN-fb model. The ZNN-fb model's convergence and stability investigation is shown in the next theorem.

Theorem 6. *Let* $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ *and* $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ *be the WFA over* $\mathbb{R}$*, where* $M^A_{x_i} \in \mathbb{R}^{m\times m}$*,* $\sigma^A \in \mathbb{R}^{1\times m}$*,* $\tau^A \in \mathbb{R}^{m\times 1}$ *and* $M^B_{x_i} \in \mathbb{R}^{n\times n}$*,* $\sigma^B \in \mathbb{R}^{1\times n}$*,* $\tau^B \in \mathbb{R}^{n\times 1}$ *with* $i = 1, \ldots, r$*. The dynamics of (38) in line with the ZNN method of (7) lead to the TSOL, shown by* $\mathbf{x}_{\mathcal{S}}(t) = \left[\mathrm{vec}(U_{\mathcal{S}}(t))^{\mathrm{T}} \quad K^{\mathrm{T}}_{\mathcal{S}}(t)\right]^{\mathrm{T}}$*, which is stable according to Lyapunov.*

Proof. The proof is omitted since it is similar to the proof of Theorem 2. □

Theorem 7. *Let* $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ *and* $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ *be the WFA over* $\mathbb{R}$*, where* $M^A_{x_i} \in \mathbb{R}^{m\times m}$*,* $\sigma^A \in \mathbb{R}^{1\times m}$*,* $\tau^A \in \mathbb{R}^{m\times 1}$ *and* $M^B_{x_i} \in \mathbb{R}^{n\times n}$*,* $\sigma^B \in \mathbb{R}^{1\times n}$*,* $\tau^B \in \mathbb{R}^{n\times 1}$ *with* $i = 1, \ldots, r$*. Beginning from any initial point* $\mathbf{x}(0)$*, the ZNN-bs model of (39) converges exponentially to* $\mathbf{x}^*(t)$*, which refers to the TSOL of (5).*

Proof. The proof is similar to the proof of Theorem 3. □

3.4. The ZNN-bb Model

In line with (6), the following group of inequations must be satisfied:

$$\begin{cases} \tau^A - U(t)\tau^B \leqslant \mathbf{0}_m, \\ \tau^B - U^{\mathrm{T}}(t)\tau^A \leqslant \mathbf{0}_n, \\ \sigma^A U(t) - \sigma^B \leqslant \mathbf{0}_n^{\mathrm{T}}, \\ \sigma^B U^{\mathrm{T}}(t) - \sigma^A \leqslant \mathbf{0}_m^{\mathrm{T}}, \\ M^A_{x_i} U(t) - U(t) M^B_{x_i} \leqslant \mathbf{0}_{m,n},\ i = 1, \ldots, r, \\ M^B_{x_i} U^{\mathrm{T}}(t) - U^{\mathrm{T}}(t) M^A_{x_i} \leqslant \mathbf{0}_{n,m},\ i = 1, \ldots, r, \\ U(t) \geqslant \mathbf{0}_{m,n}, \end{cases} \tag{40}$$

where $U(t) \in \mathbb{R}^{m\times n}$ stands for the unknown matrix. The system of (40) is reformulated as follows:

$$\begin{cases} -\left((\tau^B)^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) + \tau^A \leqslant \mathbf{0}_m, \\ -\left((\tau^A)^{\mathrm{T}} \otimes I_n\right)\mathrm{vec}(U^{\mathrm{T}}(t)) + \tau^B \leqslant \mathbf{0}_n, \\ \left(I_n \otimes \sigma^A\right)\mathrm{vec}(U(t)) - (\sigma^B)^{\mathrm{T}} \leqslant \mathbf{0}_n, \\ \left(I_m \otimes \sigma^B\right)\mathrm{vec}(U^{\mathrm{T}}(t)) - (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ \left(I_n \otimes M^A_{x_i} - (M^B_{x_i})^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \ldots, r, \\ \left(I_m \otimes M^B_{x_i} - (M^A_{x_i})^{\mathrm{T}} \otimes I_n\right)\mathrm{vec}(U^{\mathrm{T}}(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \ldots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}. \end{cases} \tag{41}$$

Using the permutation matrix P for generating $\mathrm{vec}(U^{\mathrm{T}}(t))$, (41) is rewritten as

$$\begin{cases} -\left((\tau^B)^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) + \tau^A \leqslant \mathbf{0}_m, \\ -\left((\tau^A)^{\mathrm{T}} \otimes I_n\right)P\,\mathrm{vec}(U(t)) + \tau^B \leqslant \mathbf{0}_n, \\ \left(I_n \otimes \sigma^A\right)\mathrm{vec}(U(t)) - (\sigma^B)^{\mathrm{T}} \leqslant \mathbf{0}_n, \\ \left(I_m \otimes \sigma^B\right)P\,\mathrm{vec}(U(t)) - (\sigma^A)^{\mathrm{T}} \leqslant \mathbf{0}_m, \\ \left(I_n \otimes M^A_{x_i} - (M^B_{x_i})^{\mathrm{T}} \otimes I_m\right)\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \ldots, r, \\ \left(I_m \otimes M^B_{x_i} - (M^A_{x_i})^{\mathrm{T}} \otimes I_n\right)P\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn},\ i = 1, \ldots, r, \\ -\mathrm{vec}(U(t)) \leqslant \mathbf{0}_{mn}, \end{cases}$$

and its corresponding matrix form is the following:

$$L_{\mathscr{bb}}\mathrm{vec}(U(t)) - \mathbf{b}_{\mathscr{bb}} \leqslant \mathbf{0}_y, \tag{42}$$

where

$$L_{\mathscr{bb}} = \begin{bmatrix} -(\tau^B)^{\mathrm{T}} \otimes I_m \\ -((\tau^A)^{\mathrm{T}} \otimes I_n)P \\ I_n \otimes \sigma^A \\ (I_m \otimes \sigma^B)P \\ W_{\mathscr{bb}} \\ -I_{mn} \end{bmatrix} \in \mathbb{R}^{y\times mn}, \ \mathbf{b}_{\mathscr{bb}} = \begin{bmatrix} -\tau^A \\ -\tau^B \\ (\sigma^B)^{\mathrm{T}} \\ (\sigma^A)^{\mathrm{T}} \\ \mathbf{0}_{(2r+1)mn} \end{bmatrix} \in \mathbb{R}^y,$$

$$W_{\mathscr{bb}} = \begin{bmatrix} I_n \otimes M^A_{x_1} - (M^B_{x_1})^{\mathrm{T}} \otimes I_m \\ \left(I_m \otimes M^B_{x_1} - (M^A_{x_1})^{\mathrm{T}} \otimes I_n\right)P \\ I_n \otimes M^A_{x_2} - (M^B_{x_2})^{\mathrm{T}} \otimes I_m \\ \left(I_m \otimes M^B_{x_2} - (M^A_{x_2})^{\mathrm{T}} \otimes I_n\right)P \\ \cdots \\ I_n \otimes M^A_{x_r} - (M^B_{x_r})^{\mathrm{T}} \otimes I_m \\ \left(I_m \otimes M^B_{x_r} - (M^A_{x_r})^{\mathrm{T}} \otimes I_n\right)P \end{bmatrix} \in \mathbb{R}^{2rmn\times mn}.$$

Then, considering the slack variable $K(t) \in \mathbb{R}^y$, the inequation (42) can be converted into the equation

$$L_{\mathscr{bb}}\mathrm{vec}(U(t)) - \mathbf{b}_{\mathscr{bb}} + K^{\odot 2}(t) = \mathbf{0}_y,$$

in which $K^{\odot 2}(t)$ is always a non-negative time-varying term.

Thereafter, the ZNN approach considers the following EME, which is based on (42), for simultaneously satisfying all the equations in (40):

$$E_{\mathscr{bb}}(t) = L_{\mathscr{bb}}\mathrm{vec}(U(t)) - \mathbf{b}_{\mathscr{bb}} + K^{\odot 2}(t), \tag{43}$$

where $U(t)$ and $K(t)$ are the unknown matrices to be found. The first time derivative of (43) is given as

$$\dot{E}_{\mathscr{bb}}(t) = L_{\mathscr{bb}}\mathrm{vec}(\dot{U}(t)) + 2(I_y \odot K(t))\dot{K}(t). \tag{44}$$

Then, combining Equations (43) and (44) with the ZNN design of (7), we can obtain

$$L_{\mathscr{bb}}\,\mathrm{vec}(\dot{U}(t)) + 2(I_y \odot K(t))\dot{K}(t) = -\lambda E_{\mathscr{bb}}(t). \tag{45}$$

As a result, setting

$$H_{\mathscr{bb}} = \begin{bmatrix} L_{\mathscr{bb}} & 2(I_y \odot K(t)) \end{bmatrix} \in \mathbb{R}^{y\times(mn+y)}, \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} \mathrm{vec}(\dot{U}(t)) \\ \dot{K}(t) \end{bmatrix} \in \mathbb{R}^{mn+y}, \quad \mathbf{x}(t) = \begin{bmatrix} \mathrm{vec}(U(t)) \\ K(t) \end{bmatrix} \in \mathbb{R}^{mn+y},$$

the next model is obtained:

$$H_{\mathscr{bb}}\,\dot{\mathbf{x}} = -\lambda E_{\mathscr{bb}}(t),$$

or an equivalent:

$$\dot{\mathbf{x}} = H^{\dagger}_{\mathscr{bb}}\,(-\lambda E_{\mathscr{bb}}(t)). \tag{46}$$

An appropriate `ode` MATLAB solver can be used to handle the ZNN model of (46), additionally referred to as the ZNN-bb model. The ZNN-bb model's convergence and stability investigation is shown in the next theorem.

Theorem 8. *Let* $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i\in X}, \tau^A\right)$ *and* $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i\in X}, \tau^B\right)$ *be the WFA over* $\mathbb{R}$*, where* $M^A_{x_i} \in \mathbb{R}^{m\times m}$*,* $\sigma^A \in \mathbb{R}^{1\times m}$*,* $\tau^A \in \mathbb{R}^{m\times 1}$ *and* $M^B_{x_i} \in \mathbb{R}^{n\times n}$*,* $\sigma^B \in \mathbb{R}^{1\times n}$*,* $\tau^B \in \mathbb{R}^{n\times 1}$ *with* $i = 1,\ldots,r$*. The dynamics of (45) in line with the ZNN method of (7) lead to the TSOL, shown by* $\mathbf{x}_{\mathcal{S}}(t) = \left[\mathrm{vec}(U_{\mathcal{S}}(t))^{\mathrm{T}} \quad K^{\mathrm{T}}_{\mathcal{S}}(t)\right]^{\mathrm{T}}$*, which is stable according to Lyapunov.*

Proof. The proof is omitted since it is similar to the proof of Theorem 2. □

Theorem 9. *Let $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$ be the WFA over $\mathbb{R}$, where $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$ with $i = 1, \ldots, r$. Beginning from any initial point $\mathbf{x}(0)$, the ZNN-bb model of (46) converges exponentially to $\mathbf{x}^*(t)$, which refers to the TSOL of (6).*

Proof. The proof is similar to the proof of Theorem 3. □

4. ZNN Experiments

The performances of the ZNN-fs model of (19), the ZNN-bs model of (32), the ZNN-fb model of (39), and the ZNN-bb model of (46) are examined in each of the five numerical experiments presented in this section. Keep in mind that during the computation in all experiments, the MATLAB `ode45` solver was applied with time span of $[0, 10]$ under a relative and absolute tolerance of 10^{-12} and 10^{-8}, respectively. Additionally, we contrast the output of the ZNN models with the results of the MATLAB function `linprog` (with the default settings). Following the model proposed in [7], the zero initial point is used.

Example 1. *Let us choose $m = 2$, $n = 3$, $r = 2$, and $X = \{x_1, x_2\}$, and consider WFA over $\mathbb{R}$ defined by $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i}\}_{x_i \in X}, \tau^B\right)$. Clearly, $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$. Consider $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}, = 1, 2\}, \tau^A\right)$ defined by*

$$\sigma^A = \begin{bmatrix} -7 & -8 \end{bmatrix}, \quad \tau^A = \begin{bmatrix} -13 & -13 \end{bmatrix}^{\mathrm{T}},$$
$$M^A_{x_1} = \begin{bmatrix} -6 & 9 \\ -13 & -14 \end{bmatrix}, \quad M^A_{x_2} = \begin{bmatrix} 13 & -1 \\ -15 & -9 \end{bmatrix}$$

and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}, \ i = 1, 2\right\}, \tau^B\right)$ defined by

$$\sigma^B = \begin{bmatrix} 5 & -14 & 9 \end{bmatrix}, \quad \tau^B = \begin{bmatrix} -3 & 1 & 1 \end{bmatrix}^{\mathrm{T}},$$
$$M^B_{x_1} = \begin{bmatrix} -4 & -2 & 2 \\ -13 & 17 & 9 \\ -15 & -14 & 17 \end{bmatrix}, \quad M^B_{x_2} = \begin{bmatrix} -2 & 12 & 7 \\ -1 & 16 & -3 \\ -2 & -5 & 7 \end{bmatrix}.$$

Furthermore, the design parameter of ZNN is set to $\lambda = 10$, and the following initial conditions (ICs) are used:

- *IC1: $\mathbf{x}(0) = \mathbf{1}_{23}$,*
- *IC2: $\mathbf{x}(0) = -\mathbf{1}_{23}$,*
- *IC3: $\mathbf{x}(0) = \mathrm{rand}(23, 1)$.*

The results of the ZNN-fs model are presented in Figure 1.

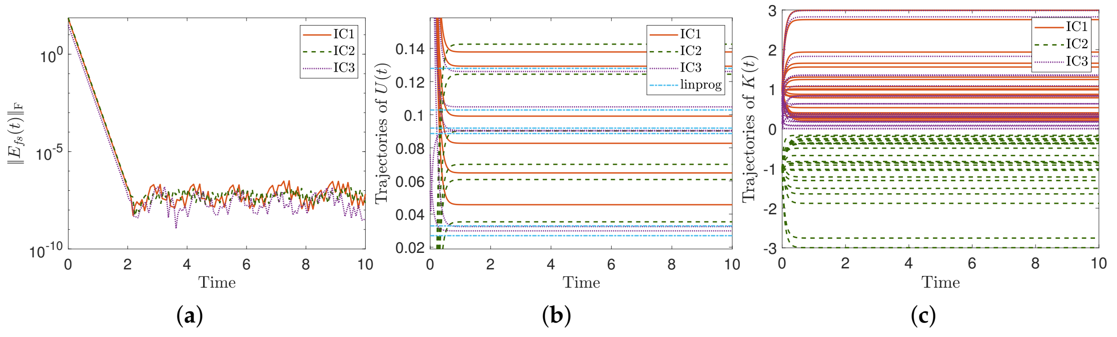


Figure 1. *Cont.*

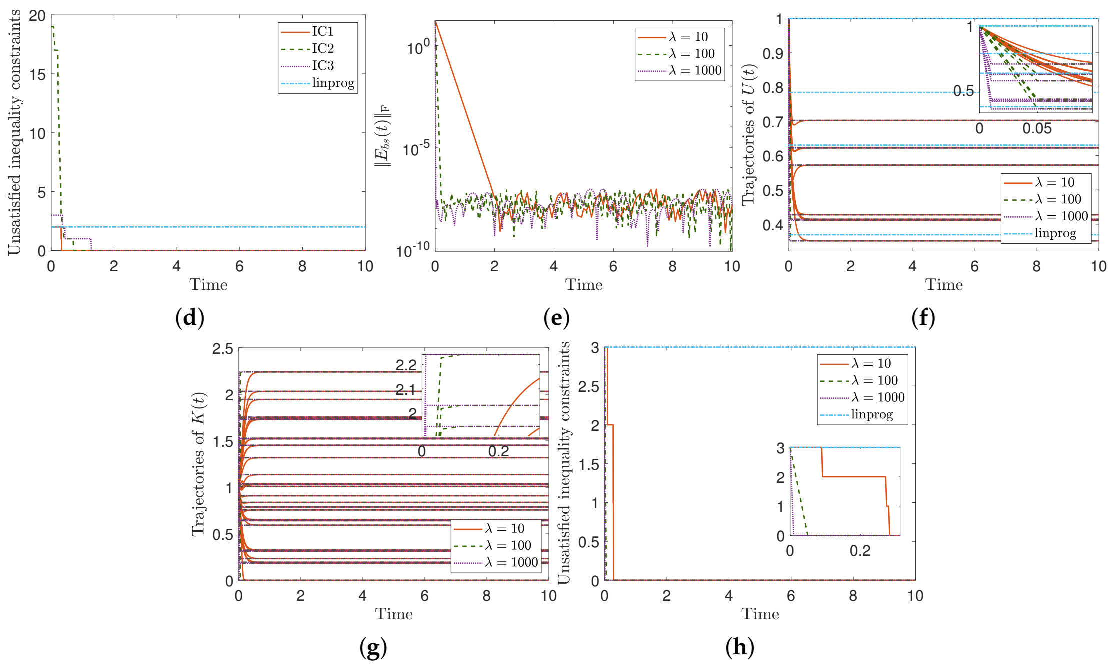


Figure 1. Errors and trajectories in Examples 1 and 2. (**a**) Example 1: EME errors. (**b**) Example 1: Trajectories of $U(t)$. (**c**) Example 1: Trajectories of $K(t)$. (**d**) Example 1: Number of unsatisfied constraints. (**e**) Example 2: EME errors. (**f**) Example 2: Trajectories of $U(t)$. (**g**) Example 2: Trajectories of $K(t)$. (**h**) Example 2: Number of unsatisfied constraints.

Example 2. *Let $m = 4$, $n = 2$, $r = 2$, and $X = \{x_1, x_2\}$, and consider WFA $\mathcal{A} = \left(m, \sigma^A, \left\{M^A_{x_i}\right\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}\right\}_{x_i \in X}, \tau^B\right)$. Clearly, $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$. Consider $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}, = 1,2\}, \tau^A\right)$ defined by*

$$\sigma^A = \begin{bmatrix} -1 & 1 & -2 & 1 \end{bmatrix}, \quad \tau^A = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}},$$
$$M^A_{x_1} = \begin{bmatrix} 2 & -1 & 3 & -1 \\ 1 & -2 & 1 & -2 \\ 3 & -1 & 2 & -1 \\ 3 & -1 & 2 & -1 \end{bmatrix}, \quad M^A_{x_2} = \begin{bmatrix} 1 & 4 & -2 & 4 \\ 2 & -1 & 2 & -1 \\ -2 & 4 & 1 & 4 \\ -2 & 4 & 1 & 4 \end{bmatrix}$$

and $\mathcal{B} = \left(n, \sigma^B, \left\{M^B_{x_i}, i = 1,2\right\}, \tau^B\right)$ defined by

$$\sigma^B = \begin{bmatrix} 1 & 2 \end{bmatrix}, \quad \tau^B = \begin{bmatrix} 1 & 1 \end{bmatrix}^{\mathrm{T}},$$
$$M^B_{x_1} = \begin{bmatrix} 6 & 4 \\ -4 & 4 \end{bmatrix}, \quad M^B_{x_2} = \begin{bmatrix} 4 & 6 \\ 6 & 4 \end{bmatrix}.$$

Also, the design parameters of ZNN are $\lambda = 10$, $\lambda = 100$ and $\lambda = 100$, whereas the IC is set to $\mathbf{x}(0) = \mathbf{1}_{30}$. The results of the ZNN-bs model are presented in Figure 1.

Example 3. *Let $m = n = k = 10$, $r = 2$, and $X = \{x_1, x_2\}$, and consider WFA $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}\}_{x_i \in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i}\}_{x_i \in X}, \tau^B\right)$ over $\mathbb{R}$. Clearly, $M^A_{x_i} \in \mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$ and $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$. Consider $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}, = 1,2\}, \tau^A\right)$ defined by*

$$\sigma^A = \mathbf{1}_k^{\mathrm{T}}, \quad \tau^A = \mathbf{1}_k, \quad M^A_{x_1} = I_k, \quad M^A_{x_2} = I_k$$

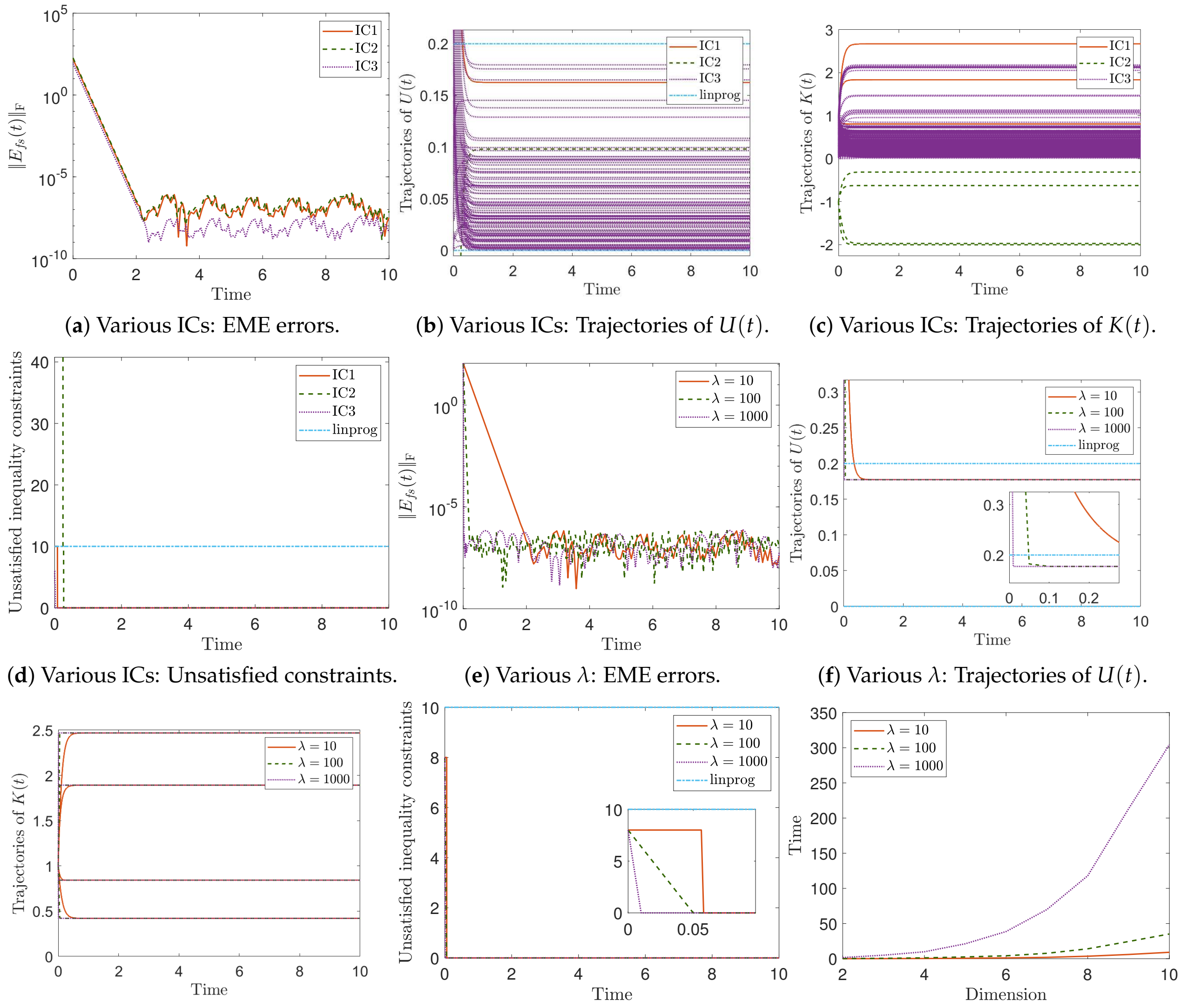
and $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i},\ i = 1, 2\}, \tau^B\right)$ *defined by*

$$\sigma^B = 5 \cdot \mathbf{1}_k^{\mathrm{T}}, \quad \tau^B = 5 \cdot \mathbf{1}_k, \quad M^B_{x_1} = 5 \cdot I_k, \quad M^B_{x_2} = 5 \cdot I_k.$$

Furthermore, the design parameter of ZNN is set to $\lambda = 10, 100, 1000$*, and the following ICs are used:*

- *IC1:* $\mathbf{x}(0) = \mathbf{1}_{420}$,
- *IC2:* $\mathbf{x}(0) = -\mathbf{1}_{420}$,
- *IC3:* $\mathbf{x}(0) = \mathrm{rand}(420, 1)$.

The results of the ZNN-fs model are presented in Figure 2.



(**a**) Various ICs: EME errors. (**b**) Various ICs: Trajectories of $U(t)$. (**c**) Various ICs: Trajectories of $K(t)$.

(**d**) Various ICs: Unsatisfied constraints. (**e**) Various λ: EME errors. (**f**) Various λ: Trajectories of $U(t)$.

(**g**) Various λ: Trajectories of $K(t)$. (**h**) Various λ: Unsatisfied constraints. (**i**) Various λ: Time consumption.

Figure 2. Errors and trajectories in Example 3.

Example 4. *Let* $m = n = k = 10$, $r = 2$, *and* $X = \{x_1, x_2\}$, *and consider WFA* $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}\}_{x_i \in X}, \tau^A\right)$ *and* $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i}\}_{x_i \in X}, \tau^B\right)$ *over* $\mathbb{R}$*. Clearly,* $M^A_{x_i} \in \mathbb{R}^{m \times m}$, $\sigma^A \in \mathbb{R}^{1 \times m}$, $\tau^A \in \mathbb{R}^{m \times 1}$ *and* $M^B_{x_i} \in \mathbb{R}^{n \times n}$, $\sigma^B \in \mathbb{R}^{1 \times n}$, $\tau^B \in \mathbb{R}^{n \times 1}$*. Consider* $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}, = 1, 2\}, \tau^A\right)$ *defined by*

$$\sigma^A = \mathbf{1}_k^{\mathrm{T}}, \quad \tau^A = \mathbf{1}_k, \quad M^A_{x_1} = I_k, \quad M^A_{x_2} = I_k$$

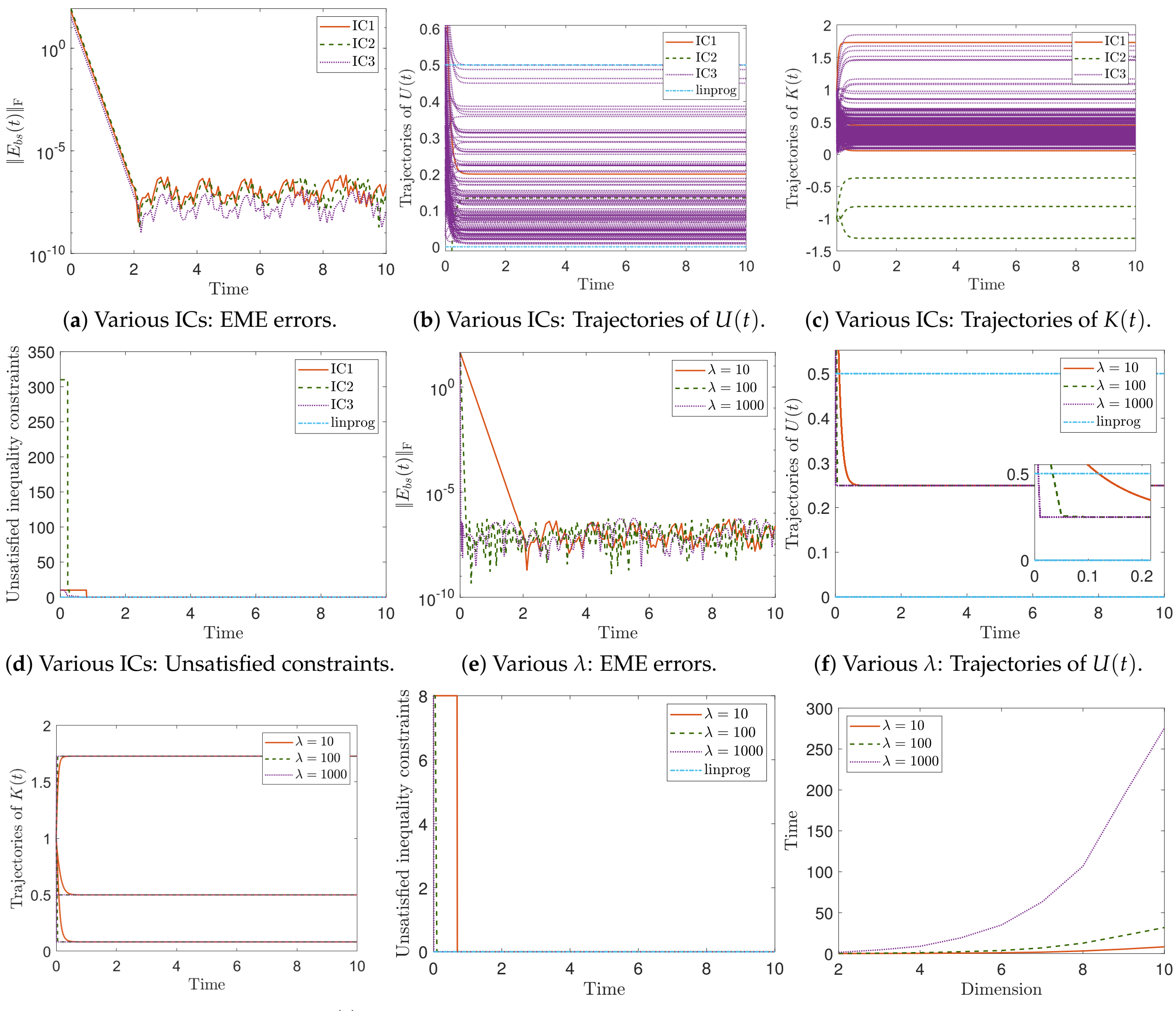
and $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i},\ i = 1,2\}, \tau^B\right)$ *defined by*

$$\sigma^B = 2 \cdot \mathbf{1}_k^{\mathrm{T}}, \quad \tau^B = 2 \cdot \mathbf{1}_k, \quad M^B_{x_1} \;= 2 \cdot I_k, \quad M^B_{x_2} = 2 \cdot I_k.$$

Furthermore, the design parameter of ZNN is set to $\lambda = 10, 100, 1000$, *and the following ICs are used:*

- *IC1:* $\mathbf{x}(0) = \mathbf{1}_{420}$,
- *IC2:* $\mathbf{x}(0) = -\mathbf{1}_{420}$,
- *IC3:* $\mathbf{x}(0) = \mathrm{rand}(420, 1)$.

The results of the ZNN-bs model are presented in Figure 3.



(**a**) Various ICs: EME errors. (**b**) Various ICs: Trajectories of $U(t)$. (**c**) Various ICs: Trajectories of $K(t)$.

(**d**) Various ICs: Unsatisfied constraints. (**e**) Various λ: EME errors. (**f**) Various λ: Trajectories of $U(t)$.

(**g**) Various λ: Trajectories of $K(t)$. (**h**) Various λ: Unsatisfied constraints. (**i**) Various λ: Time consumption.

Figure 3. Errors and trajectories in Example 4.

Example 5. *Let* $m = k + 1$, $n = k$ *with* $k = 10$, $r = 2$, *and* $X = \{x_1, x_2\}$, *and consider WFA* $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}\}_{x_i \in X}, \tau^A\right)$ *and* $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i}\}_{x_i \in X}, \tau^B\right)$ *over* $\mathbb{R}$. *Clearly,* $M^A_{x_i} \in$

$\mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$, and $M_{x_i}^B \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$. Consider $\mathcal{A} = \left(m, \sigma^A, \{M_{x_i}^A, = 1,2\}, \tau^A\right)$ defined by

$$\sigma^A = -\mathbf{1}_{k+1}^{\mathrm{T}}, \quad \tau^A = \begin{bmatrix} -\mathbf{1}_k \\ 1 \end{bmatrix}, \quad M_{x_1}^A = \begin{bmatrix} -\mathbf{1}_{k,k+1} \\ \mathbf{1}_{k+1}^{\mathrm{T}} \end{bmatrix}, \quad M_{x_2}^A = 2\cdot\begin{bmatrix} -\mathbf{1}_{k,k+1} \\ \mathbf{1}_{k+1}^{\mathrm{T}} \end{bmatrix}$$

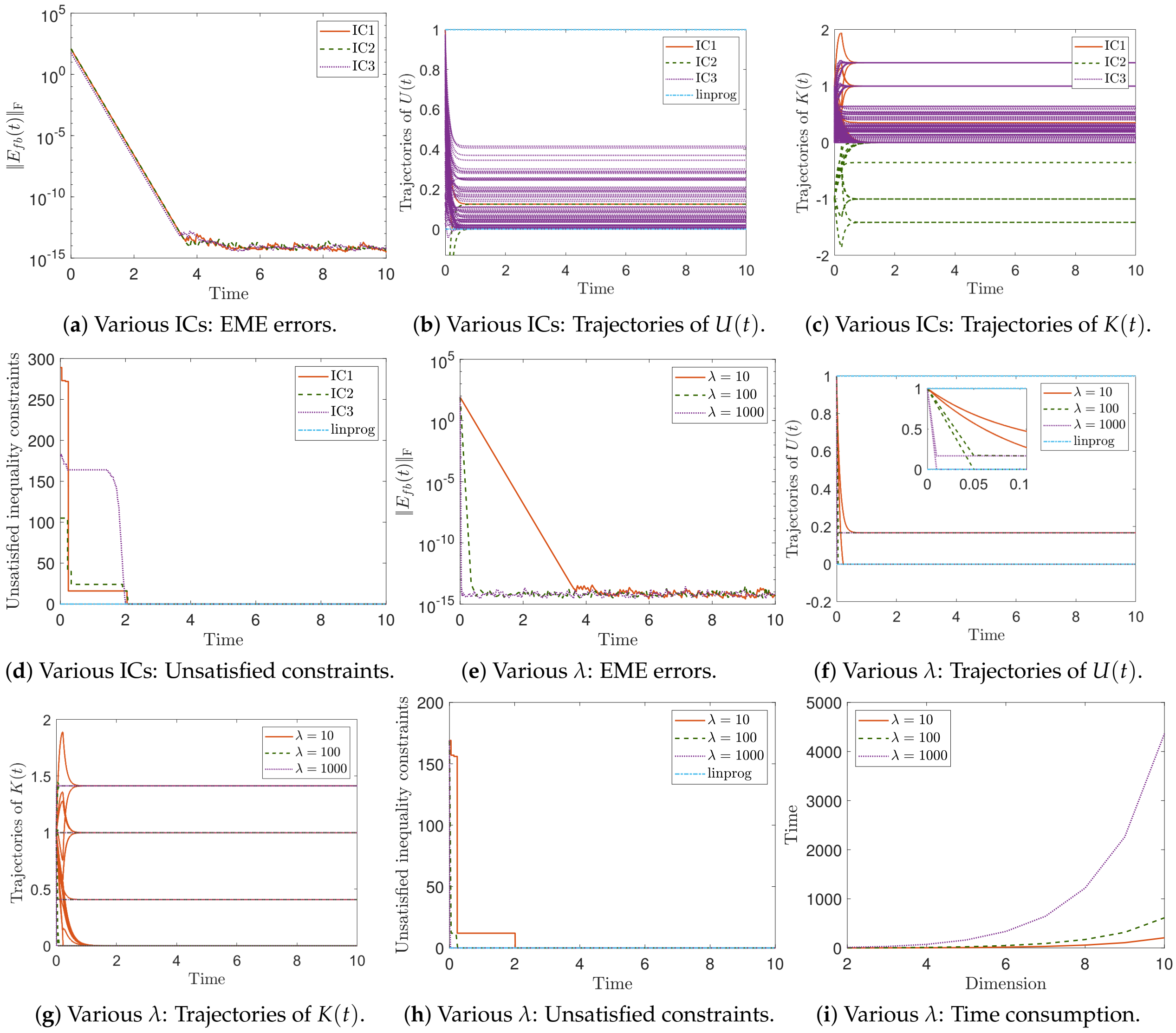
and $\mathcal{B} = \left(n, \sigma^B, \{M_{x_i}^B, i = 1,2\}, \tau^B\right)$ defined by

$$\sigma^B = -\mathbf{1}_k^{\mathrm{T}}, \quad \tau^B = -\mathbf{1}_k, \quad M_{x_1}^B = -\mathbf{1}_{k,k}, \quad M_{x_2}^B = -2\cdot\mathbf{1}_{k,k}.$$

Furthermore, the design parameter of ZNN is set to $\lambda = 10, 100, 1000$, and the following ICs are used:

- *IC1: $\mathbf{x}(0) = \mathbf{1}_{702}$,*
- *IC2: $\mathbf{x}(0) = -\mathbf{1}_{702}$,*
- *IC3: $\mathbf{x}(0) = \mathrm{rand}(702, 1)$.*

The results of the ZNN-fb model are presented in Figure 4.



(**a**) Various ICs: EME errors. (**b**) Various ICs: Trajectories of $U(t)$. (**c**) Various ICs: Trajectories of $K(t)$. (**d**) Various ICs: Unsatisfied constraints. (**e**) Various λ: EME errors. (**f**) Various λ: Trajectories of $U(t)$. (**g**) Various λ: Trajectories of $K(t)$. (**h**) Various λ: Unsatisfied constraints. (**i**) Various λ: Time consumption.

Figure 4. Errors and trajectories in Example 5.

Example 6. *Let $m = k+1$, $n = k$ with $k = 10$, $r = 2$, and $X = \{x_1, x_2\}$, and consider WFA $\mathcal{A} = \left(m, \sigma^A, \{M_{x_i}^A\}_{x_i\in X}, \tau^A\right)$ and $\mathcal{B} = \left(n, \sigma^B, \{M_{x_i}^B\}_{x_i\in X}, \tau^B\right)$ over $\mathbb{R}$. Clearly, $M_{x_i}^A \in$*

$\mathbb{R}^{m\times m}$, $\sigma^A \in \mathbb{R}^{1\times m}$, $\tau^A \in \mathbb{R}^{m\times 1}$, *and* $M^B_{x_i} \in \mathbb{R}^{n\times n}$, $\sigma^B \in \mathbb{R}^{1\times n}$, $\tau^B \in \mathbb{R}^{n\times 1}$. *Consider* $\mathcal{A} = \left(m, \sigma^A, \{M^A_{x_i}, = 1,2\}, \tau^A\right)$ *defined by*

$$\sigma^A = \begin{bmatrix} 2\cdot\mathbf{1}_k^{\mathrm{T}} & 1\end{bmatrix}, \tau^A = \begin{bmatrix} -2\cdot\mathbf{1}_k \\ -1 \end{bmatrix}, M^A_{x_1} \ = \begin{bmatrix} -\mathbf{1}_{k+1,k} & \mathbf{1}_{k+1}\end{bmatrix}, M^A_{x_2} = 2\cdot\begin{bmatrix} -\mathbf{1}_{k+1,k} & \mathbf{1}_{k+1}\end{bmatrix}$$

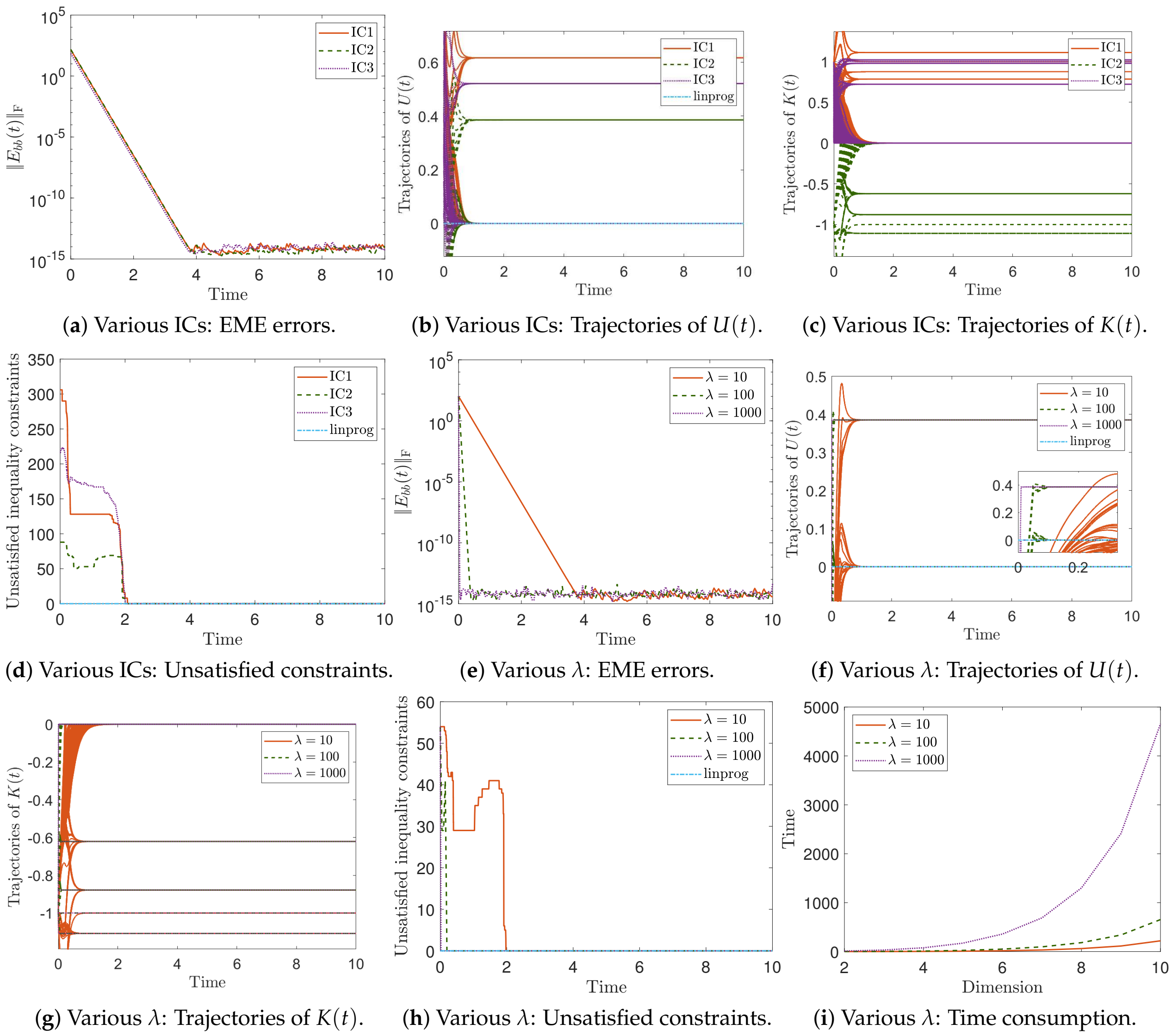
and $\mathcal{B} = \left(n, \sigma^B, \{M^B_{x_i},\ i = 1,2\}, \tau^B\right)$ *defined by*

$$\sigma^B = 2\cdot\mathbf{1}_k^{\mathrm{T}}, \quad \tau^B = -2\cdot\mathbf{1}_k, \quad M^B_{x_1} \ = -\mathbf{1}_{k,k}, \quad M^B_{x_2} = -2\cdot\mathbf{1}_{k,k}.$$

Furthermore, the design parameter of ZNN is set to $\lambda = 10, 100, 1000$, *and the following ICs are used:*

- *IC1:* $\mathbf{x}(0) = \mathbf{1}_{702}$,
- *IC2:* $\mathbf{x}(0) = -\mathbf{1}_{702}$,
- *IC3:* $\mathbf{x}(0) = \mathrm{rand}(702, 1)$.

The results of the ZNN-bb model are presented in Figure 5.



(**a**) Various ICs: EME errors. (**b**) Various ICs: Trajectories of $U(t)$. (**c**) Various ICs: Trajectories of $K(t)$. (**d**) Various ICs: Unsatisfied constraints. (**e**) Various λ: EME errors. (**f**) Various λ: Trajectories of $U(t)$. (**g**) Various λ: Trajectories of $K(t)$. (**h**) Various λ: Unsatisfied constraints. (**i**) Various λ: Time consumption.

Figure 5. Errors and trajectories in Example 6.

Results Discussion

This part discusses the findings from the four numerical examples that look at how effectively the ZNN models perform.

More precisely, in Example 1, we obtain the next outcomes for the ZNN-fs model by IC1, IC2, and IC3 for $\lambda = 10$. Figure 1e shows the ZNN-fs model's EMEs. All instances start from a huge error price at $t = 0$, and all EMEs conclude in the interval $[10^{-8}, 10^{-7}]$ with a negligible error price at $t = 2$. Put another way, the ZNN-fs model validates Theorem 3 by converging to a value close to zero for three distinct ICs. The trajectories of $U(t)$ and $K(t)$, i.e., the model's solutions, are shown in Figures 1f,g, respectively. These results indicate that $U(t)$ and $K(t)$ do not have similar trajectories via IC1, IC2, and IC3, but their convergence speeds are similar. Therefore, the ZNN-fs model appears to give different solutions for a range of ICs, and its solutions' convergence pattern is proven to be matched up with the convergence pattern of the linked EMEs. Moreover, given that the ZNN-fs model must satisfy $z = 23$ in number inequality constraints, Figure 1h illustrates the number of inequality constraints that remain unsatisfied during the ZNN learning process. This number equals 0 when all of the inequality constraints are satisfied. In this example, this number becomes 0 at $t = 0.5$ for IC1, at $t = 0.8$ for IC2, and at $t = 1.3$ for IC3. Therefore, for a variety of ICs, the ZNN-fs model seems to have varying convergence speeds when it comes to satisfying the inequality constraints. Comparing the ZNN-fs model to the `linprog`, we see in Figure 1f that the `linprog` yields different $U(t)$ trajectories than ZNN. Furthermore, we see in Figure 1h that 2 of the 23 inequality constraints are not satisfied by the `linprog` solution. As a result, the ZNN-fs model outperforms the `linprog` in this particular example.

In Example 2, under $\lambda = 10, 100, 1000$, the next outcomes for the ZNN-bs model are obtained. Figure 1a shows the ZNN-bs model's EMEs. All instances in this figure start with a large error price at $t = 0$ and conclude at $[10^{-10}, 10^{-7}]$ at $t = 0.02$ for $\lambda = 1000$, at $t = 0.2$ for $\lambda = 100$, and at $t = 2$ for $\lambda = 10$, with a negligible error price. Put another way, the ZNN approach's convergence features are confirmed by the ZNN-bs model's EME, which is dependent on λ, and the ZNN-bs model validates Theorem 5 by converging to a value close to zero. The trajectories of $U(t)$ and $K(t)$, i.e., the model's solutions, are shown in Figure 1b,d, respectively. These results indicate that the convergence speed of the trajectories of $U(t)$ and $K(t)$ is much faster via $\lambda = 1000$ than via $\lambda = 100$, while the convergence speed of the trajectories of $U(t)$ and $K(t)$ is much faster via $\lambda = 100$ than via $\lambda = 10$. Also, it is observable that $U(t)$ and $K(t)$ have similar trajectories with each other via $\lambda = 10, 100, 1000$, respectively. So, the ZNN-bs model appears to give the same $U(t)$ and $K(t)$ solutions for a range of λ values, and its solutions' convergence pattern is proven to be matched up with the convergence pattern of the linked EMEs. Moreover, given that the ZNN-bs model must satisfy $z = 30$ in the number inequality constraints, Figure 1d illustrates the number of inequality constraints that remain unsatisfied during the ZNN learning process. This number becomes 0 at $t = 0.3$ for $\lambda = 10$, at $t = 0.05$ for $\lambda = 100$, and at $t = 0.005$ for $\lambda = 1000$. Therefore, for higher values of λ, the ZNN-bs model seems to have faster convergence speeds when it comes to satisfying the inequality constraints. Comparing the ZNN-bs model to the `linprog`, we see in Figure 1b that the `linprog` yields different $U(t)$ trajectories than ZNN. Furthermore, we see in Figure 1d that 3 of the 30 inequality constraints are not satisfied by the `linprog` solution. As a result, the ZNN-bs model outperforms the `linprog` in this example.

In Examples 3–6, we obtain the next outcomes for the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models by IC1, IC2, and IC3 for $\lambda = 10$. Figures 2a, 3a, 4a and 5a show the ZNN model's EMEs. All instances start from a huge error price at $t = 0$, and all EMEs conclude in the interval $[10^{-15}, 10^{-13}]$ with a negligible error price at $t = 3.6$. Put another way, the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models validate Theorems 3, 5, 7 and 9, respectively, by converging to a value close to zero for two distinct ICs. The trajectories of $U(t)$, generated by the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models, are shown in Figures 2b, 3b, 4b and 5b, and the trajectories of $K(t)$ are shown in Figures 2c, 3c, 4c and 5c, respectively. For each case, these results indicate that $U(t)$ and $K(t)$ do not have similar trajectories via IC1, IC2, and IC3, but their convergence speeds are similar. Therefore, all ZNN models appear to give different solutions for a range of ICs, and their solutions' convergence pattern is proven to be matched up with the convergence pattern of the linked EMEs. Moreover, given that the ZNN-fs and ZNN-bs models must satisfy $z = 320$ in the number inequality constraints and the ZNN-fb and ZNN-bb models must satisfy $y = 592$ in the number inequality constraints, Figures 2d, 3d, 4d and 5d illustrate the number of inequality constraints that remain unsatisfied during the ZNN learning process. This number becomes 0 at around $t = 2$ for all ICs. Therefore, for a variety of ICs, the ZNN models seem to have varying convergence rates when it comes to satisfying the inequality constraints.

Additionally, the next outcomes for the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models are obtained in Examples 3–6 under $\lambda = 10, 100, 1000$. Figures 2e, 3e, 4e and 5e show the ZNN models' EMEs. All instances in these figures start with a large error price at $t = 0$ and conclude at $[10^{-16}, 10^{-14}]$ at $t = 0.04$ for $\lambda = 1000$, at $t = 0.4$ for $\lambda = 100$, and at $t = 3.8$ for $\lambda = 10$, with a negligible error price. Put another way, the ZNN approach's convergence features are confirmed by the ZNN models' EME, which is dependent on λ, and the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models validate Theorems 3, 5, 7 and 9, respectively, by convergence to a value close to zero. The trajectories of $U(t)$ generated by the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models are shown in Figures 2f, 3f, 4f and 5f, and the trajectories of $K(t)$ are shown in Figures 2g, 3g, 4g and 5g, respectively. These results indicate that the convergence speed of the trajectories of $U(t)$ and $K(t)$ is much faster via $\lambda = 1000$ than via $\lambda = 100$, while the convergence speed of the trajectories of $U(t)$ and $K(t)$ is much faster via $\lambda = 100$ than via $\lambda = 10$. Also, it is observable that $U(t)$ and $K(t)$ have similar trajectories via $\lambda = 10, 100$, and 1000, respectively. So, for each case, the ZNN model appears to give the same $U(t)$ and $K(t)$ solutions for a range of λ values, and its solutions' convergence pattern is proven to be matched up with the convergence pattern of the linked EMEs. Moreover, given that the ZNN-fs and ZNN-bs models must satisfy $z = 320$ in the number of inequality constraints and the ZNN-fb and ZNN-bb models must satisfy $y = 592$ in the number of inequality constraints, Figures 2h, 3h, 4h and 5h illustrate the number of inequality constraints that remain unsatisfied during the ZNN learning process. This number becomes 0 at $t = 1.9$ for $\lambda = 10$, at $t = 0.3$ for $\lambda = 100$, and at $t = 0.03$ for $\lambda = 1000$. Therefore, for higher values of λ, the ZNN models seem to have faster convergence speeds when it comes to satisfying the inequality constraints.

Comparing the ZNN models in Examples 3–6 to the `linprog`, we see in Figures 2b,f, 3b,f, 4b,f and 5b,f that the `linprog` yields different $U(t)$ trajectories than ZNN. It is important to note that the zero solution is produced by the `linprog` in Example 6 and that 10 of the 320 inequality constraints are not satisfied by the `linprog` solution in Example 3. Thus, the ZNN model performs similarly to the `linprog` in Examples 4–6, while the ZNN model outperforms the `linprog` in Example 3. Furthermore, Figures 2i, 3i, 4i and 5i show the time consumption of the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models in Examples 3–6, respectively, using the MATLAB R2022a environment on an Intel® Core™ i5-6600K CPU 3.50 GHz, 16 GB RAM, running on Windows 10 64 bit Operating System. In these figures, as the dimensions of the matrices (i.e., the value of k) rise, we find that the ZNN models' time consumption increases considerably more via $\lambda = 1000$ than via $\lambda = 100$, and that the ZNN models' time consumption increases considerably more via $\lambda = 100$ than via $\lambda = 10$. Therefore, for higher values of λ, the ZNN models seem to have a higher time consumption.

When everything is considered, the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models perform admirably in finding the solution of Equations (3)–(6), respectively. Upon comparing the ZNN models to the `linprog`, it is discovered that each ZNN model exhibits comparable or superior performance to the `linprog`. Additionally, all ZNN model performances are affected by the value of λ, and their solutions are affected by the value of the ICs. Keep in mind that the values of λ and the ICs in the experiments of this section were chosen at random. As a corollary, the approximation to the TSOL, $\mathbf{x}^*(t)$, in the ZNN-fs, ZNN-bs, ZNN-fb, and ZNN-bb models, is achieved faster via $\lambda = 1000$ than via $\lambda = 100$ and $\lambda = 10$, while the time consumption is higher via $\lambda = 1000$ than via $\lambda = 100$ and $\lambda = 10$.

5. Concluding Remarks

Practically, this research is focused on solving the equivalence problem (determining whether two automata determine the same word function) or solving the containment problem (determining whether the word function of one WFA is bounded from above by the word function of another). Our intention was to unify two important topics, namely, the zeroing neural network (ZNN) and the existence of forward and backward simulations and bisimulations for weighted finite automata (WFA) over the field of real numbers $\mathbb{R}$. Two types of quantitative simulations and two types of bisimulations were defined as solutions to particular systems of matrix and vector inequations over $\mathbb{R}$. This research was aimed at the development and analysis of two novel ZNN models, termed as ZNN-bs and ZNN-fs, for addressing the systems of matrix and vector inequations involved in simulations as well as at bisimulations between WFA and two novel ZNN models, termed ZNN-fb and ZNN-bb, for addressing the systems of matrix and vector inequalities involved in bisimulations between WFA. The problem considered in this paper requires solving a system of two vector inequalities and a couple of matrix inequalities. Using positive slack matrices, the required matrix and vector inequations were transformed into corresponding equations which are solvable by the proposed ZNN dynamical systems. A detailed convergence analysis was considered. Numerical examples were performed with different initial state matrices. A comparison with a known LP approach proposed in [7] was presented, and the better performance of the ZNN design was confirmed. The models solved in current research utilized the development of ZNN dynamics based on several inequations and Zhang error functions. The derived models can be viewed as extensions from equations to inequations of ZNN algorithms established upon a few error functions. Such models have been investigated in several papers, such as [31–34].

Seen more generally, the research described in this paper shows that the ZNN design is usable in solving systems of matrix and vector inequations in linear algebra. Further research can be aimed at solving the minimization problems (determining an automaton with the minimal number of states equivalent to a given automaton).

Simulations and bisimulations have already been studied through solving systems of matrix inequations in the context of fuzzy finite automata [2,3], nondeterministic automata [4], WFA over an additively idempotent semiring [5], and max-plus automata [6]. The methodology used there, based on the concept of residuation, is fundamentally different from the methodology applied in this article to WFA over the field of real numbers. Perhaps some general ideas of this article could be applied to solving systems of matrix inequalities in the context of fuzzy finite automata, for example, the use of neuro-fuzzy systems (fuzzy neural networks), which could be the topic of our future research. On the other hand, the proposed methodology could be more directly applied to some special WFA over a field of real numbers, such as WFA over a semiring of nonnegative real numbers and probabilistic automata. This will also be one of the topics of our future research.

Author Contributions: Conceptualization, M.Ć., P.S.S. and S.D.M.; methodology, P.S.S. and S.D.M.; software, S.D.M.; validation, M.Ć., P.S.S. and D.K.; formal analysis, P.S.S., M.Ć., P.B. and D.K.; investigation, P.S.S., S.D.M. and D.K.; resources, S.D.M.; data curation, S.D.M. and D.K.; writing—original draft preparation, M.Ć. and P.S.S.; writing—review and editing, P.S.S., S.D.M. and M.Ć.; visualization, S.D.M.; supervision, M.Ć. and P.S.S.; project administration, P.B., P.S.S. and D.K.; funding acquisition, P.B. and D.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No. 075-15-2022-1121).

Data Availability Statement: The data that support the findings of this study are available on request to the authors.

Acknowledgments: Predrag Stanimirović and Miroslav Ćirić acknowledge the support by the Science Fund of the Republic of Serbia, Grant No. 7750185, Quantitative Automata Models: Fundamental Problems and Applications—QUAM. Predrag Stanimirović and Miroslav Ćirić are also supported by the Ministry of Science, Technological Development and Innovation, Republic of Serbia, Contract No. 451-03-65/2024-03/200124.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ćirić, M.; Ignjatović, J.; Stanimirović, P.S. Bisimulations for weighted finite automata over semirings. *Res. Sq.* **2022** . [CrossRef]
2. Ćirić, M.; Ignjatović, J.; Damljanović, N.; Bašić, M. Bisimulations for fuzzy automata. *Fuzzy Sets Syst.* **2012**, *186*, 100–139. [CrossRef]
3. Ćirić, M.; Ignjatović, J.; Jančić, I.; Damljanović, N. Computation of the greatest simulations and bisimulations between fuzzy automata. *Fuzzy Sets Syst.* **2012**, *208*, 22–42. [CrossRef]
4. Ćirić, M.; Ignjatović, J.; Bašić, M.; Jančić, I. Nondeterministic automata: Equivalence, bisimulations, and uniform relations. *Inf. Sci.* **2014**, *261*, 185–218. [CrossRef]
5. Damljanović, N.; Ćirić, M.; Ignjatović, J. Bisimulations for weighted automata over an additively idempotent semiring. *Theor. Comput. Sci.* **2014**, *534*, 86–100. [CrossRef]
6. Ćirić, M.; Micić, I.; Matejić, J.; Stamenković, A. Simulations and bisimulations for max-plus automata. *Discret. Event Dyn. Syst.* **2024**, *34*, 269–295. [CrossRef]
7. Urabe, N.; Hasuo, I. Quantitative simulations by matrices. *Inf. Comput.* **2017**, *252*, 110–137. [CrossRef]
8. Stanković, I.; Ćirić, M.; Ignjatović, J. Bisimulations for weighted networks with weights in a quantale. *Filomat* **2023**, *37*, 3335–3355.
9. Doyen, L.; Henzinger, T.A.; Raskin, J.F. Equivalence of labeled Markov chains. *Int. J. Found. Comput. Sci.* **2008**, *19*, 549–563. [CrossRef]
10. Balle, B.; Gourdeau, P.; Panangaden, P. Bisimulation metrics and norms for real weighted automata. *Inf. Comput.* **2022**, *282*, 104649. [CrossRef]
11. Boreale, M. Weighted bisimulation in linear algebraic form. In *CONCUR 2009—Concurrency Theory, 20th International Conference, Bologna, Italy, 1–4 September 2009* ; Bravetti, M., Zavattaro, G., Eds.; LNCS; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5710, pp. 163–177.
12. Xiao, L.; Jia, L. *Zeroing Neural Networks: Finite-time Convergence Design, Analysis and Applications*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2022.
13. Zhang, Y.; Yi, C. *Zhang Neural Networks and Neural-Dynamic Method*; Nova Science Publishers, Inc.: New York, NY, USA, 2011.
14. Zhang, Y.; Ge, S.S. Design and analysis of a general recurrent neural network model for time-varying matrix inversion. *IEEE Trans. Neural Netw.* **2005**, *16*, 1477–1490. [CrossRef] [PubMed]
15. Wu, W.; Zheng, B. Two new zhang neural networks for solving time-varying linear equations and inequalities systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 4957–4965. [CrossRef] [PubMed]
16. Xiao, L.; Tan, H.; Dai, J.; Jia, L.; Tang, W. High-order error function designs to compute time-varying linear matrix equations. *Inf. Sci.* **2021**, *576*, 173–186. [CrossRef]
17. Li, X.; Yu, J.; Li, S.; Ni, L. A nonlinear and noise-tolerant ZNN model solving for time-varying linear matrix equation. *Neurocomputing* **2018**, *317*, 70–78. [CrossRef]
18. Dai, J.; Li, Y.; Xiao, L.; Jia, L. Zeroing neural network for time-varying linear equations with application to dynamic positioning. *IEEE Trans. Ind. Inform.* **2022**, *18*, 1552–1561. [CrossRef]
19. Wang, T.; Zhang, Z.; Huang, Y.; Liao, B.; Li, S. Applications of Zeroing Neural Networks: A Survey. *IEEE Access* **2024**, *12*, 51346–51363. [CrossRef]
20. Guo, D.; Yan, L.; Zhang, Y. Zhang Neural Networks for online solution of time-varying linear inequalities. In *Artificial Neural Networks*; Rosa, J.L.G., Ed.; IntechOpen: Rijeka, Crooatia, 2016. [CrossRef]
21. Sun, J.; Wang, S.; Wang, K. Zhang neural networks for a set of linear matrix inequalities with time-varying coefficient matrix. *Inf. Process. Lett.* **2016**, *116*, 603–610. [CrossRef]
22. Xiao, L.; Zhang, Y. Two new types of Zhang Neural Networks solving systems of time-varying nonlinear inequalities. *IEEE Trans. Circuits Syst. Regul. Pap.* **2012**, *59*, 2363–2373. [CrossRef]
23. Xiao, L.; Zhang, Y. Zhang Neural Network versus Gradient Neural Network for solving time-varying linear inequalities. *IEEE Trans. Neural Netw.* **2011**, *22*, 1676–1684. [CrossRef]
24. Xiao, L.; Zhang, Y. Different Zhang functions resulting in different ZNN models demonstrated via time-varying linear matrix–vector inequalities solving. *Neurocomputing* **2013**, *121*, 140–149. [CrossRef]
25. Zeng, Y.; Xiao, L.; Li, K.; Li, J.; Li, K.; Jian, Z. Design and analysis of three nonlinearly activated ZNN models for solving time-varying linear matrix inequalities in finite time. *Neurocomputing* **2020**, *390*, 78–87. [CrossRef]

26. Guo, D.; Zhang, Y. A new variant of the Zhang neural network for solving online time-varying linear inequalities. *Proc. R. Soc.* **2012**, *2255*, 2271. [CrossRef]
27. Zheng, B.; Yue, C.; Wang, Q.; Li, C.; Zhang, Z.; Yu, J.; Liu, P. A new super-predefined-time convergence and noise-tolerant RNN for solving time-variant linear matrix–vector inequality in noisy environment and its application to robot arm. *Neural Comput. Appl.* **2024**, *36*, 4811–4827. [CrossRef]
28. Zhang, Y.; Yang, M.; Huang, H.; Xiao, M.; Hu, H. New discrete solution model for solving future different level linear inequality and equality with robot manipulator control. *IEEE Trans. Ind. Inform.* **2019**, *15*, 1975–1984. [CrossRef]
29. Guo, D.; Zhang, Y. A new inequality-based obstacle-avoidance MVN scheme and its application to redundant robot manipulators. *IEEE Trans. Syst. Man Cybern. Part (Appl. Rev.)* **2012**, *42*, 1326–1340. [CrossRef]
30. Kong, Y.; Jiang, Y.; Lou, J. Terminal computing for Sylvester equations solving with application to intelligent control of redundant manipulators. *Neurocomputing* **2019**, *335*, 119–130. [CrossRef]
31. Katsikis, V.; Mourtas, S.; Stanimirović, P.; Zhang, Y. Solving complex-valued time-varying linear matrix equations via QR decomposition with applications to robotic motion tracking and on angle-of-arrival localization. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 3415–3424. [CrossRef] [PubMed]
32. Li, X.; Lin, C.L.; Simos, T.; Mourtas, S.; Katsikis, V. Computation of time-varying 2,3- and 2,4-inverses through zeroing neural networks. *Mathematics* **2022**, *10*, 4759. [CrossRef]
33. Simos, T.; Katsikis, V.; Mourtas, S.; Stanimirović, P. Unique non-negative definite solution of the time-varying algebraic Riccati equations with applications to stabilization of LTV system. *Math. Comput. Simul.* **2022**, *202*, 164–180. [CrossRef]
34. Stanimirović, P.; Mourtas, S.; Mosić, D.; Katsikis, V.; Cao, X.; Li, S. Zeroing Neural Network approaches for computing time-varying minimal rank outer inverse. *Appl. Math. Comput.* **2024**, *465*. [CrossRef]
35. Buchholz, P. Bisimulation relations for weighted automata. *Theor. Comput. Sci.* **2008**, *393*, 109–123. [CrossRef]
36. Hua, C.; Cao, X.; Xu, Q.; Liao, B.; Li, S. Dynamic neural network models for time-varying problem solving: A survey on model structures. *IEEE Access* **2023**, *11*, 65991–66008. [CrossRef]
37. Jin, L.; Li, S.; Liao, B.; Zhang, Z. Zeroing neural networks: A survey. *Neurocomputing* **2017**, *267*, 597–604. [CrossRef]
38. Graham, A. *Kronecker Products and Matrix Calculus with Applications*; Courier Dover Publications: Mineola, NY, USA, 2018.