DOI: 10.2298/CSIS120121043Z

Superior Performance of Using Hyperbolic Sine Activation Functions in ZNN Illustrated via Time-Varying Matrix Square Roots Finding

Yunong Zhang, Long Jin, and Zhende Ke

School of Information Science and Technology, Sun Yat-sen University
Guangzhou 510006, Guangdong, China
zhynong@mail.sysu.edu.cn, spaformind@qq.com, kezhende@126.com

Abstract. A special class of recurrent neural network, termed Zhang neural network (ZNN) depicted in the implicit dynamics, has recently been proposed for online solution of time-varying matrix square roots. Such a ZNN model can be constructed by using monotonically-increasing odd activation functions to obtain the theoretical time-varying matrix square roots in an error-free manner. Different choices of activation function arrays may lead to different performance of the ZNN model. Generally speaking, ZNN model using hyperbolic sine activation functions may achieve better performance, as compared with those using other activation functions. In this paper, to pursue the superior convergence and robustness properties, hyperbolic sine activation functions are applied to the ZNN model for online solution of time-varying matrix square roots. Theoretical analysis and computer-simulation results further demonstrate the superior performance of the ZNN model using hyperbolic sine activation functions in the context of large model-implementation errors, in comparison with that using linear activation functions.

Keywords: Zhang neural network, global exponential convergence, hyperbolic sine activation functions, time-varying matrix square roots, implementation errors.

1. Introduction

The problem of solving for matrix square roots is considered to be an important special case of nonlinear matrix equation problem, which widely arises in many scientific and engineering fields; e.g., control theory [1], optimization [2], and signal processing [3]. In general, the solution of matrix square roots, which can usually be a fundamental part of many solutions, can be achieved via matrix equations solving. Thus, many numerical algorithms/methods have been presented and developed for online solution of matrix square roots [1–6]. Generally speaking, it may not be efficient enough for most numerical algorithms due to their serial-processing nature performed on digital computers [2, 3]. For large-scale online or real-time applications, the minimal arithmetic operations of such numerical algorithms are usually proportional to the cube of the matrix dimension n, i.e., $O(n^3)$ operations [7]. To remedy inherent weaknesses of such numerical approaches, many parallel-processing computational methods, including various dynamic-system approaches, have been developed and implemented on specific architectures [8–14]. The neural-dynamic approach is thus now regarded as a powerful alternative to online computation in view of its high-speed parallel-distributed processing property and the convenience of hardware realization [8]. Besides, it is worth mentioning that most reported computational schemes are theoretically/intrinsically designed for time-invariant (or termed, static, constant) problems solving currently, which are usually related to the traditional gradient-based methods [8, 14–18].

Since March 2001, a special recurrent neural network, termed Zhang neural network (ZNN), has been formally proposed by Zhang *et al* [9, 11, 12] for time-varying problems solving (e.g., time-varying Sylvester equation solving, time-varying matrix inversion and optimization). The design of the ZNN is based on a matrix/vector-valued error-function, instead of a norm-based scalar-valued energy-function usually associated with gradient-based neural networks (GNN). In addition, ZNN is depicted in an implicit dynamics which arises frequently in analog electric circuits and systems due to Kirchhoff's rules [11], instead of an explicit dynamics that usually depicts a GNN model.

In the hardware implementation of neural networks, there always exist some realization errors, which are more complicated than the ideal situation. For example, the incapacity of electronic components would limit the performance of the ZNN, and generate various errors (e.g., differentiation error and model-implementation error [15]). Due to these realization errors, the solution of the circuit-implemented ZNN may not be accurate. In this case, robustness analysis of the proposed ZNN would be important and necessary. This paper presents the general framework of the ZNN model solving for time-varying matrix square roots, and, by using a special type of monotonically-increasing odd activation functions (e.g., hyperbolic sine activation functions), the ZNN model with superior robustness in the context of large implementation errors, is analyzed and investigated, which is compared with the ZNN model activated by linear functions.

The rest of this paper is organized as follows. Section 3 introduces the novel problem formulation and investigates the convergence properties of the ZNN model for online solution of time-varying matrix square roots. Section 4 presents the robustness analysis of the ZNN model with hyperbolic sine activation functions in the context of (very) large model-implementation errors. In Section 5, illustrative simulative results are shown to verify the superior convergence and robustness of the ZNN model using hyperbolic sine activation functions for time-varying matrix square roots finding, which further substantiate the theoretical analysis. Finally, conclusions are drawn in Section 6. To the best of the authors' knowledge, there is almost no others' literature dealing with such a specific problem of online solution of time-varying matrix square roots at present stage, and the main contributions of the paper lie in the following facts.

- This paper presents a recurrent neural network (i.e., ZNN) for online solution of time-varying matrix square roots. To pursue the superior convergence and robustness properties, a special type of activation functions (i.e., hyperbolic sine activation functions), which is compared with the linear activation functions, is applied to the ZNN model. This is also the main motivation of the work.

- Together with the theoretical analysis, this paper presents the convergence properties of the proposed ZNN model, which substantiate the superior performance of the ZNN model using hyperbolic sine activation functions, in comparison with that using linear activation functions.
- For potential hardware/circuit implementation and considering its related uncertain realization errors, the robustness property of ZNN model is investigated in the context of large implementation errors, which demonstrates the efficacy and superiority of the proposed ZNN model activated by hyperbolic sine functions.
- The simulative results of different illustrative and representative examples are presented, where different activation functions [i.e., hyperbolic sine functions and linear functions] are exploited in the ZNN model for online solution of time-varying matrix square roots, and the superior convergence and robustness of the proposed ZNN model are thus verified evidently.

2. Related work

Newton iteration [2] has been investigated as an useful method for matrix square roots finding. Similar iterations have been designed and studied differently, such as the Denman and Beavers (DB) method [19] based on the matrix sign function iteration, the Meini iteration based on a cyclic reduction (CR) algorithm [1] and the iteration derived from Newton (IN) method [20]. However, all of those methods, which are designed intrinsically for static matrix square roots finding, may generate large lagging errors in time-varying applications. As a novel class of recurrent neural network (RNN), ZNN has been formally proposed and investigated for the online solution of various time-varying problems. Detailed comparisons between the ZNN model and the four numerical methods (i.e., Newton iteration, DB iteration, CR iteration, and IN iteration) for online solution of static matrix square roots can be found in [21]. Furthermore, the traditional GNN model, which is also designed intrinsically for static matrix square roots finding, is simulated and compared with the ZNN model in [22] for online solution of time-varying matrix square roots.

Moreover, this paper investigates the matrix-valued nonlinear time-varying problem, i.e., the time-varying matrix square roots finding, which is quite different from the authors' previous work (e.g., the linear time-varying problems solving [9, 11, 12, 23, 24]). To further pursue the superior convergence and robustness properties, a special type of activation functions, i.e., hyperbolic sine activation functions (which is compared with the linear activation functions), is applied to the ZNN model. It is worth pointing out that the method of using hyperbolic sine activation functions has seldom been proposed and studied before, which is one of the main motivations of the work.

3. Problem formulation and neural-network solver

In the ensuing subsections, the problem formulation of time-varying matrix square roots is introduced firstly, and then the ZNN model is developed and analyzed for time-varying matrix square roots finding.

3.1. Problem formulation

Let us consider the following time-varying matrix square roots (TVMSR) problem (which can also be viewed as a time-varying nonlinear matrix equation problem):

$$X^2(t) - A(t) = 0, \;\; t \in [0, +\infty), \tag{1}$$

where $A(t) \in R^{n\times n}$ denotes a smoothly time-varying positive-definite matrix, which, together with its time derivative $\dot{A}(t)$, are assumed to be known numerically or could be measured accurately. In addition, $X(t)$ is the time-varying unknown matrix to be solved for, and our objective in this work is to find $X(t) \in R^{n\times n}$ so that (1) holds true for any $t \geqslant 0$.

Before solving (1) in real time, the following preliminaries [1, 3, 25, 26] are provided as a basis for further discussion.

DEFINITION 2.1. *Given a smoothly time-varying matrix $A(t) \in R^{n\times n}$, if matrix $X(t) \in R^{n\times n}$ satisfies the time-varying nonlinear equation $X^2(t) = A(t)$, then $X(t) \in R^{n\times n}$ is a time-varying square root of matrix $A(t) \in R^{n\times n}$ [or to say, $X(t)$ is a time-varying solution to nonlinear equation (1)].*

Square-root existence condition. *If smoothly time-varying matrix $A(t) \in R^{n\times n}$ is positive-definite (in general sense [26]) at any time instant $t \in [0, +\infty)$, then there exists a time-varying matrix square root $X(t) \in R^{n\times n}$ for $A(t)$.*

In addition, it follows from Kronecker-product and vectorization technique [9, 27] that time-varying nonlinear matrix equation (1) could be written as

$$\big(I \otimes X(t)\big)\text{vec}\big(X(t)\big) - \text{vec}\big(A(t)\big) = 0, \tag{2}$$

where symbol $\otimes$ denotes the Kronecker product [9, 27], and operator vec$(\cdot)$: $R^{n\times n} \to R^{n^2\times 1}$ [e.g., vec$(A(t))$] generates a column vector obtained by stacking all column vectors of the input matrix argument [e.g., $A(t)$] together.

3.2. Zhang neural network

To solve for time-vary matrix square root $A^{1/2}(t)$, by Zhang *et al*'s method [9, 11, 12], the following matrix-valued error function could be defined firstly:

$$E(t) = X^2(t) - A(t) \in R^{n\times n}.$$

Secondly, the error-function's time-derivative $\dot{E}(t) \in R^{n\times n}$ could be made such that every entry $e_{ij}(t) \in R$ of $E(t) \in R^{n\times n}$ converges to zero, $i, j =$

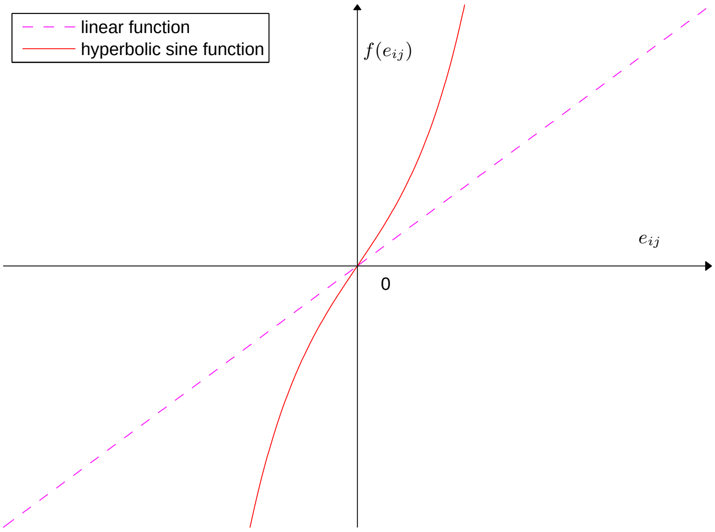


Fig. 1. Activation function $f(\cdot)$ used in the neural network.

$1, 2, \cdots, n$; in mathematics, to choose $\dot{e}_{ij}(t)$ such that $\lim_{t\to\infty} e_{ij}(t) = 0$, $\forall i, j \in \{1, 2, \cdots, n\}$. A general form of $\dot{E}(t)$ is given by Zhang *et al* as

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = -\Gamma\mathcal{F}\big(E(t)\big), \tag{3}$$

where design parameter Γ and activation-function array $\mathcal{F}(\cdot)$ are described as follows.

- $\Gamma \in R^{n\times n}$ is a positive-definite (diagonal) matrix used to scale the convergence rate of the neural network. For simplicity, Γ can be γI with scalar $\gamma > 0 \in R$ and I denoting the identity matrix. Γ (or γI), being a set of reciprocals of capacitance parameters in the hardware implementation, should be set as large as the hardware would permit (e.g., in analog circuits or VLSI [15]), or selected appropriately for experimental and/or simulative purposes.
- $\mathcal{F}(\cdot) : R^{n\times n} \to R^{n\times n}$ denotes an activation-function matrix array of the neural network. In general, any monotonically-increasing odd activation function $f(\cdot)$, being the ijth element of $\mathcal{F}(\cdot)$, can be used for the construction of the neural network. In this paper, the following two types of activation functions are discussed and compared (which are shown in Figure 1):
  - linear activation function $f(e_{ij}) = e_{ij}$, and
  - hyperbolic sine activation function $f(e_{ij}) = \exp(\xi e_{ij})/2 - \exp(-\xi e_{ij})/2$ with design parameter $\xi \geqslant 1$.

Thirdly, expanding ZNN design formula (3) leads to the following implicit dynamic equation of the ZNN model for online matrix square roots finding via nonlinear time-varying equation (1) solving:

$$X(t)\dot{X}(t) + \dot{X}(t)X(t) = -\gamma\mathcal{F}\big(X^2(t) - A(t)\big) + \dot{A}(t), \quad (4)$$

where $X(t)$, starting from an initial condition $X(0) \in R^{n\times n}$, is the activation state matrix corresponding to theoretical time-varying matrix square root $X^*(t)$ of $A(t)$. It is worth mentioning that, when using a linear activation function array $\mathcal{F}(\cdot)$, the general nonlinearly-activated ZNN (4) reduces to the following linearly-activated one:

$$X(t)\dot{X}(t) + \dot{X}(t)X(t) = -\gamma X^2(t) + \gamma A(t) + \dot{A}(t). \quad (5)$$

For simulative purposes, based on the Kronecker-product and vectorization technique [9, 27], the matrix differential equation (4) can be transformed to a vector differential equation. We thus obtain the following theorem.

THEOREM 2.1. *The matrix-form differential equation (4) can be reformulated as the following vector-form differential equation:*

$$(I \otimes X + X^T \otimes I)\mathit{vec}(\dot{X}) = -\gamma\mathcal{F}\big((I \otimes X)\mathit{vec}(X) - \mathit{vec}(A)\big) + \mathit{vec}(\dot{A}), \quad (6)$$

where superscript ^{T} denotes the transpose of a matrix or a vector, and activation-function array $\mathcal{F}(\cdot)$ in (6) is defined as before except that its dimensions are changed to be $R^{n^2\times 1} \to R^{n^2\times 1}$. In the simulation, $M(t, x) := I \otimes X + X^T \otimes I$ denotes the nonsingular mass matrix of a standard ordinary-differential-equation (ODE) problem.

Proof. See Appendix A for details.

In addition, for ZNN (4) which solves for the time-varying matrix square root of $A(t)$, we have the following theorems on its convergence.

THEOREM 2.2. *Consider smoothly time-varying matrix $A(t) \in R^{n\times n}$ in nonlinear equation (1), which satisfies the square-root existence condition. If a monotonically increasing odd activation-function array $\mathcal{F}(\cdot)$ is used, then error-function $E(t) = X^2(t) - A(t) \in R^{n\times n}$ of ZNN (4), starting from randomly-generated positive-definite (or negative-definite) diagonal initial state-matrix $X(0) \in R^{n\times n}$, can converge to zero [which implies that state matrix $X(t) \in R^{n\times n}$ of ZNN (4) can converge to theoretical positive-definite (or negative-definite) time-varying matrix square root $X^*(t)$ of $A(t)$].*

Proof. From the compact form of ZNN design formula $\dot{E}(t) = -\Gamma\mathcal{F}(E(t))$, a set of n^2 decoupled differential equations can be written equivalently as follows:

$$\dot{e}_{ij}(t) = -\gamma f\big(e_{ij}(t)\big), \quad (7)$$

for any $i \in \{1, 2, 3, \cdots, n\}$ and $j \in \{1, 2, 3, \cdots, n\}$. Thus, to analyze the equivalent ijth subsystem (7), we define a Lyapunov function candidate $v_{ij}(t) = e_{ij}^2(t)/2 \geqslant 0$ with its time-derivative

$$\frac{\mathrm{d}v_{ij}(t)}{\mathrm{d}t} = e_{ij}(t)\dot{e}_{ij}(t) = -\gamma e_{ij}(t)f\big(e_{ij}(t)\big).$$

As mentioned previously, $f(\cdot)$ is a monotonically-increasing odd activation function; i.e., $f(-e_{ij}(t)) = -f(e_{ij}(t))$. Then, the following result is obtained:

$$e_{ij}(t)f(e_{ij}(t)) \begin{cases} > 0, & \text{if } e_{ij}(t) \neq 0, \\ = 0, & \text{if } e_{ij}(t) = 0, \end{cases}$$

which guarantees the final negative-definiteness of $\dot{v}_{ij}$ (i.e., $\dot{v}_{ij} < 0$ for $e_{ij} \neq 0$ while $\dot{v}_{ij} = 0$ for $e_{ij} = 0$ only). By Lyapunov theory [28], equilibrium point $e_{ij} = 0$ of (7) is asymptotically stable; i.e., $e_{ij}(t)$ converges to zero, for any $i \in \{1, 2, 3, \cdots, n\}$ and $j \in \{1, 2, 3, \cdots, n\}$. In other words, the matrix-valued error-function $E(t) = [e_{ij}(t)] \in R^{n\times n}$ is convergent to zero. In addition, we have $E(t) = X^2(t) - A(t)$; or equivalently, $X^2(t) = A(t) + E(t)$. Since $E(t) \to 0$ as $t \to +\infty$, we have $X^2(t) \to A(t)$ [i.e., $X(t) \to X^*(t)$] as $t \to +\infty$. That is, state matrix $X(t)$ of ZNN model (4) can converge to the theoretical time-varying matrix square root $X^*(t)$ of $A(t)$.

Furthermore, when state matrix $X(t)$ of ZNN model (4) starting from a randomly-generated positive-definite diagonal initial state-matrix $X(0)$, it can converge to the positive-definite time-varying matrix square root $A^{1/2}(t)$ [i.e., a form of $X^*(t)$]. This can be proofed by contradiction. Suppose that state matrix $X(t)$ starting from a positive-definite diagonal initial state-matrix $X(0)$ converges to the negative-definite TVMSR $-A^{1/2}(t)$ [i.e., the other form of $X^*(t)$], then state-matrix $X(t)$ must pass through at least one 0-eigenvalue, which leads to the contradiction that the left and right hand sides of ZNN (4) can not hold. So, starting from a randomly-generated positive-definite diagonal initial state-matrix $X(0)$, state matrix $X(t)$ of ZNN model (4) can converge to the positive-definite time-varying matrix square root $A^{1/2}(t)$. Similarly, it can proved that, starting from a randomly-generated negative-definite diagonal initial state-matrix $X(0)$, state matrix $X(t)$ of ZNN model (4) can converge to the negative-definite time-varying matrix square root $-A^{1/2}(t)$. The proof is thus complete.

THEOREM 2.3. *In addition to Theorem 2.2, if a linear activation function array $\mathcal{F}(\cdot)$ is used, then the matrix-valued error-function $E(t) = X^2(t) - A(t) \in R^{n\times n}$ of ZNN (4), starting from a randomly-generated positive-definite (or negative-definite) diagonal initial-state-matrix $X(0) \in R^{n\times n}$, can exponentially converge to zero with convergence rate γ, which corresponds to the convergence of state matrix $X(t) \in R^{n\times n}$ of ZNN (4) to $X^*(t)$. Moreover, if the hyperbolic sine activation function array is used, then the superior convergence can be achieved for ZNN (4), as compared to the linear activation case.*

Proof. 1) If the linear activation function $f(e_{ij}) = e_{ij}$ is used, from the ijth subsystem (7), we have $\dot{e}_{ij} = -\gamma e_{ij}$, and thus $e_{ij}(t) = \exp(-\gamma t)e_{ij}(0)$. In other words, the matrix-valued error-function $E(t) \in R^{n\times n}$ can be expressed explicitly as

$$E(t) = \begin{bmatrix} e_{11}(0) & e_{12}(0) & \cdots & e_{1n}(0) \\ e_{21}(0) & e_{22}(0) & \cdots & e_{2n}(0) \\ \vdots & \vdots & \ddots & \vdots \\ e_{n1}(0) & e_{n2}(0) & \cdots & e_{nn}(0) \end{bmatrix} \exp(-\gamma t) = E(0)\exp(-\gamma t).$$

This evidently shows that the error function $E(t)$ exponentially converges to zero with convergence rate γ for the ZNN (4) activated by the linear activation function array. In addition, we have $E(t) = X^2(t) - A(t)$ and then $X^2(t) = A(t) + E(t)$, i.e., $X^2(t) = A(t) + E(0)\exp(-\gamma t)$. Since $E(0)\exp(-\gamma t) \to 0$ exponentially as $t \to +\infty$, we have again $X^2(t) \to A(t)$ and $X(t) \to X^*(t)$ as $t \to +\infty$. That is, state matrix $X(t)$ of ZNN (4) can converge to the theoretical time-varying matrix square root $X^*(t)$ of $A(t)$.

2) Define a Lyapunov function candidate $V := \|E(t)\|_{\mathsf{F}}^2/2 = \mathsf{trace}(E^{\mathsf{T}}(t)E(t))/2 = \mathsf{vec}^{\mathsf{T}}(E(t))\mathsf{vec}(E(t))/2 \geqslant 0$ for ZNN (4). Since

$$\mathsf{vec}\big(E(t)\big) := [e_{11}, \cdots, e_{n1}, e_{12}, \cdots, e_{n2}, \cdots, e_{1n}, \cdots, e_{nn}]^{\mathsf{T}} \to 0$$

is equivalent to $E(t) \to 0$, then we use $\mathsf{vec}(E(t))$ instead of $E(t)$ to analyze the Lyapunov function candidate and its time derivative. Thus, defining e_i as the ith element of $\mathsf{vec}(E(t))$, we have

$$V(t) = \sum_{i=1}^{n^2} e_i^2(t)/2, \text{ and } \dot{V}(t) = \sum_{i=1}^{n^2} e_i(t)\dot{e}_i(t) = -\gamma \sum_{i=1}^{n^2} e_i(t) f(e_i(t)).$$

So, if linear activation functions are used, we have

$$V(t)_{\mathsf{lin}} = \sum_{i=1}^{n^2} e_i^2(t)/2 \geqslant 0, \text{ and } \dot{V}(t)_{\mathsf{lin}} = -\gamma \sum_{i=1}^{n^2} e_i^2(t) \leqslant 0.$$

This implies that the matrix-valued error-function $E(t)$ of linearly-activated ZNN model (5) can converge to zero according to the aforementioned discussion and Lyapunov theory [28].

On the other hand, if hyperbolic sine functions are used, the corresponding Lyapunov function candidate is still $v(t)_{\mathsf{hs}} = v(t)_{\mathsf{lin}} \geqslant 0$. By Taylor series expansion, the aforementioned hyperbolic sine function is formulated as

$$\begin{aligned} f(e_i) &= [\mathsf{exp}(\xi e_i) - \mathsf{exp}(-\xi e_i)]/2 \\ &= [2(\xi e_i) + 2 \cdot \frac{(\xi e_i)^3}{3!} + 2 \cdot \frac{(\xi e_i)^5}{5!} + \cdots]/2 \\ &= \xi e_i + \frac{(\xi e_i)^3}{3!} + \frac{(\xi e_i)^5}{5!} + \cdots \\ &= \sum_{r=1}^{+\infty} \frac{\xi^{2r-1}(e_i)^{2r-1}}{(2r-1)!}. \end{aligned}$$

Thus, we have the following derivation:

$$\begin{aligned}\dot{v}(t)_{\mathsf{hs}} &= -\gamma\sum_{i=1}^{n^2} e_i f(e_i)\\ &= -\gamma\sum_{i=1}^{n^2} e_i \sum_{r=1}^{+\infty}\frac{\xi^{2r-1}(e_i)^{2r-1}}{(2r-1)!}\\ &= -\gamma\sum_{i=1}^{n^2}\sum_{r=1}^{+\infty}\frac{\xi^{2r-1}(e_i)^{2r}}{(2r-1)!}\\ &\ll -\gamma\sum_{i=1}^{n^2} e_i^2 = \dot{v}(t)_{\mathsf{lin}} \leqslant 0.\end{aligned}$$

From the aforementioned analysis procedure, we see that Lyapunov function candidate $v(t)_{\mathsf{hs}}$ can diminish to zero when hyperbolic sine activation functions are used, with much faster convergence rate than that using linear activation functions. This implies that, when hyperbolic sine functions are exploited, nonlinearly-activated ZNN model (4) possesses superior convergence in comparison with linearly-activated ZNN model (5). The proof is thus completed.

REMARK 2.1. The construction of ZNN (4) allows us to have many more choices of different activation functions. In many engineering applications, it may be necessary to investigate the impact of different activation functions in the RNN, in view of the fact that nonlinearity always exists. Even if the linear activation function is used, the nonlinear phenomenon may appear in its hardware implementation, e.g., in the form of saturation and/or inconsistency of the linear slope, or due to truncation and round-off errors of digital realization [15]. The investigation of different activation functions may give more insights into the imprecise-implementation problem of neural networks.

REMARK 2.2. One more advantage of using the hyperbolic sine function over the linear function lies in the extra design parameter ξ, which is an effective factor of the convergence rate. When there is an upper bound on γ due to hardware implementation, the parameter ξ will be another effective factor expediting the ZNN (4) convergence. The convergence for the hyperbolic sine activation functions can be much faster than that for the linear activation functions, when using the same level of design parameters ξ and γ. This is because the error signal $e_{ij} = [X^2 - A]_{ij}$ in (4) is amplified by the hyperbolic sine activation function for the whole error range $(-\infty, +\infty)$ (i.e., the larger slope and absolute value of the hyperbolic sine activation function in the whole error range as shown in Figure 1).

4. Robustness analysis

In the analog implementation or simulation of recurrent neural networks, we usually assume that it is under ideal conditions. However, there always exist some realization errors in hardware implementation. The differentiation errors of $A(t)$, and the model-implementation error appear most frequently in the hardware realization. For these realization errors possibly appearing in model (4), we investigate the ZNN robustness by considering the following matrix-valued ZNN design formula perturbed with a large model-implementation error:

$$X(t)\dot{X}(t) + \dot{X}(t)X(t) = -\gamma\mathcal{F}(X^2(t) - A(t)) + \dot{A}(t) + \Delta\omega, \tag{8}$$

where $\Delta\omega \in R^{n\times n}$ denotes the general model-implementation errors [including the differentiation errors of matrix $A(t)$ as a part]. For these errors, the following lemmas on the robustness of largely-perturbed ZNN model (8) could be achieved [23, 24].

LEMMA 3.1. *Consider the above perturbed ZNN model with a large model implementation error $\Delta\omega \in R^{n\times n}$ finally depicted in equation (8). If $0 \leqslant \|\Delta\omega\|_F \leqslant \varepsilon < \infty$ for any $t \in [0, +\infty]$, then the steady-state residual error $\lim_{t\to\infty} \|E(t)\|_F$ is always uniformly upper bounded by some positive scalar, provided that the design parameter $\gamma > 0$ is large enough (the so-called design-parameter requirement). Furthermore, the steady-state residual error $\lim_{t\to\infty} \|e(t)\|_F$ decreases to zero as γ tends to positive infinity.*

LEMMA 3.2. *In addition to the general robustness results given in Lemma 3.1, the largely-perturbed ZNN model (8) possesses the following properties.*

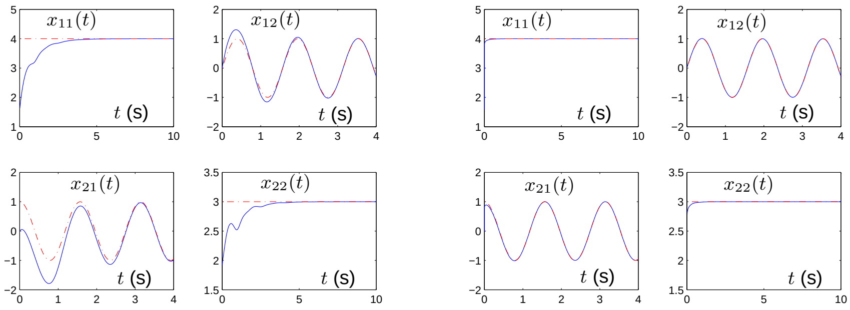
– *With linear activation functions used, the steady-state error $\lim_{t\to\infty} \|E(t)\|_F$ can be written out as a positive scalar under the design parameter requirement.*
– *With hyperbolic sine activation functions used, the superior convergence and robustness properties exist for the whole error range $(-\infty, +\infty)$; i.e., the design-parameter requirement can be removed in this case and the steady-state residual error $\lim_{t\to\infty} \|E(t)\|_F$ can be made (much) smaller [which can be further done by increasing γ and/or ξ], as compared to the situation of using linear activation functions.*

5. Illustrative examples

The previous sections have presented the convergence and robustness results of ZNN model (4) [together with a largely perturbed ZNN model (8)] for online solution of time-varying matrix square root problem (1). In this section, computer-simulation results and observations are provided to verify the superior characteristics of using hyperbolic sine activation functions to those of using linear activation functions.

Example 1. For illustration and comparison, let us consider equation (1) with the following time-varying matrix $A(t)$:

$$A(t) = \begin{bmatrix} 16 + \sin 4t \cos 4t & 7 \sin 4t \\ 7 \cos 4t & 9 + \sin 4t \cos 4t \end{bmatrix}. \tag{9}$$



(a) linear activation functions (b) hyperbolic sine functions ($\xi = 3$)

Fig. 2. Online solution of time-varying matrix square root problem (9) by ZNN (4) with $\gamma = 1$, where the neural-network solutions $X(t)$ are denoted by solid blue curves, and the theoretical time-varying square root $A^{1/2}(t)$ is denoted by red dash-dotted curves.

Simple manipulations may verify that a theoretical time-varying matrix square root $X^*(t)$ of $A(t)$ could be

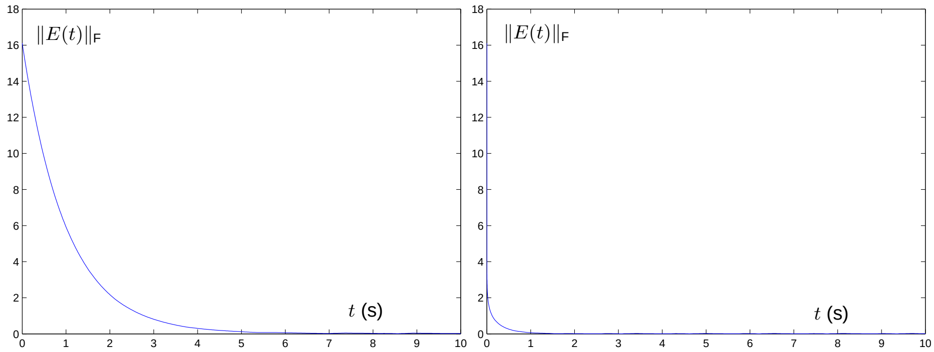
$$X^*(t) = \begin{bmatrix} 4 & \sin 4t \\ \cos 4t & 3 \end{bmatrix},$$

which is used to check the correctness of the neural-network solution $X(t)$. Based on the aforementioned vector-form ZNN model (6) proposed in Theorem 2.1, we can obtain the simulated ZNN state $X(t)$ by using ODE routine “ode23t” [25]. As illustrated in Figure 2, starting from randomly-generated positive-definite diagonal initial-state $X(0) \in [0, 2]^{2\times 2}$, neural state $X(t)$ of ZNN (4) using hyperbolic sine activation functions converges to the theoretical time-varying solution much faster, compared to that using linear activation functions. Furthermore, in order to further investigate the convergence performance, we monitor and show the residual error $\|E(t)\|_{\mathrm{F}}$ during the problem solving process of ZNN. As seen from Figure 3, the residual errors $\|E(t)\|_{\mathrm{F}}$ of ZNN (4) all decrease rapidly to zero, where the convergence rate of ZNN (4) using hyperbolic sine activation functions appears to be 5 times faster than that using linear activation functions. From these figures, we can confirm well the theoretical results given in Theorems 2.2 and 2.3.

To show the robustness characteristics of the largely-perturbed ZNN model, the following large model implementation error $\Delta\omega$ is specially added in (8):

$$\Delta\omega = \begin{bmatrix} 10^2 & 10^2 \\ 10^2 & 10^2 \end{bmatrix}.$$

As we can see from Figure 4, with the large model-implementation error, the steady-state residual error $\lim_{t\to\infty} \|E(t)\|_{\mathrm{F}}$ of the perturbed ZNN (8) is still

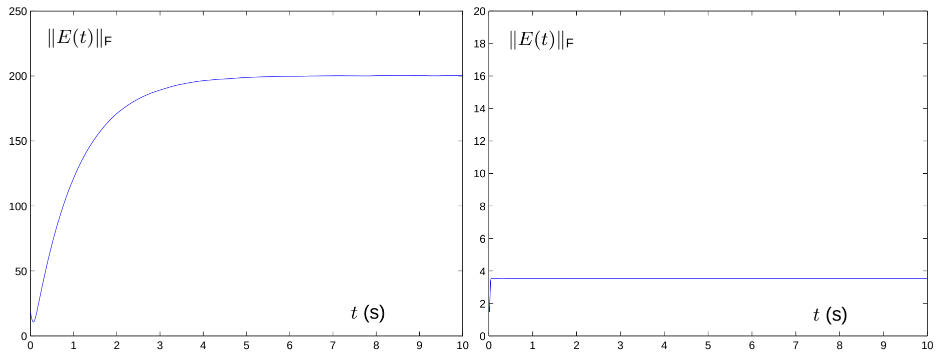


(a) linear activation functions (b) hyperbolic sine functions ($\xi = 3$)

Fig. 3. Residual errors of ZNN (4) with $\gamma = 1$ for TVMSR finding of (9).

bounded. In addition, with $\gamma = 1$, ZNN (8) using linear activation functions results in large residual errors. This is shown evidently in Figure 4(a). In contrast, as shown comparatively in Figure 4(a) and (b), when hyperbolic sine activation functions are used, the convergence time of the perturbed ZNN (8) is faster than that using linear activation functions, and the steady-state residual error of the perturbed ZNN (8) is around 50 times smaller than that using linear activation functions. Furthermore, Figure 4(b) is about using hyperbolic sine activation functions with design parameter $\xi = 3$, while Figure 5(a) and (b) are about $\xi = 5$ and $\xi = 7$, respectively. It is observed that, by increasing ξ, the steady-state residual error $\lim_{t\to\infty} \|E(t)\|_\mathsf{F}$ of ZNN (8) is decreased very effectively (e.g., for $\xi = 7$, which is around 133 times smaller than that using linear activation functions). In addition, comparing Figure 4(b) and Figure 6, we can see that, as the design parameter γ increases form 1 to 10 and then to 100, the upper bound of the steady-state residual error is decreased effectively from around 3.53 to 1.99 and then to 0.59. Note that, when γ is much larger (e.g., 10^6 or 10^9), the steady-state residual error would become tiny. Besides, if the very large model-implementation error (e.g., $\Delta\omega_{ij} = 10000$ with $i, j = 1, 2$) is added to the ZNN model, the robustness results are shown in Figure 7, from which the same conclusion can be drawn; i.e., using hyperbolic sine activation functions results in a much smaller steady-state residual error than using linear activation functions [e.g., around 5000 times smaller, as seen comparatively from Figure 7(a) and (b)].

Example 2. In order to further investigate the efficacy of ZNN models [including (4) and (8)] using hyperbolic sine activation functions, let us consider equation (1) with the following symmetric positive-definite time-varying matrix $A(t)$ with its theoretical time-varying square root $X^*(t)$ given as well for com-



(a) linear activation functions (b) hyperbolic sine functions ($\xi = 3$)

Fig. 4. Residual errors of largely-perturbed ZNN (8) for time-varying matrix square roots finding of (9) (with $\gamma = 1$ and $\varDelta\omega_{ij} = 10^2, i, j \in \{1, 2\}$).

parison purposes:

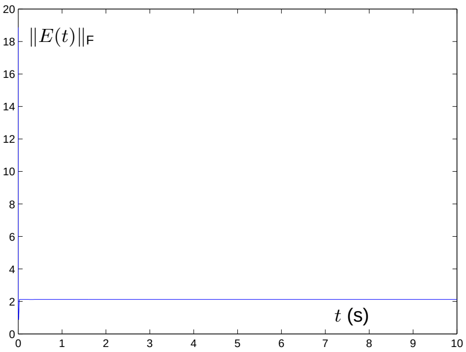
$$A(t) = \begin{bmatrix} 5 + 0.25s^2 & 2s + 0.5c & 4 + 0.25s \times c \\ 2s + 0.5c & 4.25 & 2c + 0.5s \\ 4 + 0.25s \times c & 2c + 0.5s & 5 + 0.25c^2 \end{bmatrix}, \tag{10}$$

$$X^*(t) = \begin{bmatrix} 2 & 0.5s & 1 \\ 0.5s & 2 & 0.5c \\ 1 & 0.5c & 2 \end{bmatrix},$$

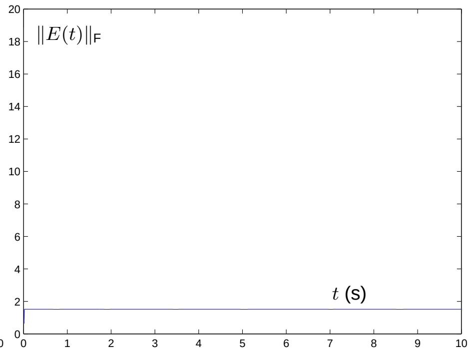
where s and c denote $\sin(6t)$ and $\cos(6t)$, respectively.

As illustrated in Figure 8, starting from randomly-generated positive-definite diagonal initial-state $X(0) \in [0, 2]^{3\times3}$, state matrix $X(t)$ of ZNN (4) using hyperbolic sine activation functions converges faster than that using linear activation functions. In order to further investigate the convergence performance, we monitor and show the residual error $\|E(t)\|_{\mathsf{F}}$ during the problem solving process of ZNN. From Figure 9, we observe that the residual errors $\|E(t)\|_{\mathsf{F}}$ of ZNN (4) all decrease rapidly to zero, where the convergence rate of ZNN (4) using hyperbolic sine activation functions is also 5 times faster than that using linear activation functions. These figures substantiate again the theoretical results given in Theorems 2.2 and 2.3.

To comparatively show the robustness characteristics of the largely-perturbed ZNN model, we exploit once more the large model-implementation error $\varDelta\omega_{ij} = 10^2, i, j \in \{1, 2, 3\}$, which is added in ZNN model (8). With design parameter $\gamma = 1$ and two types of activation functions used, the robustness performance of the largely-perturbed ZNN model (8) is shown in Figure 10, where the steady-state residual errors $\lim_{t\to\infty} \|E(t)\|_{\mathsf{F}}$ are all still bounded. In addition, as shown in Figure 10(a) and (b), when hyperbolic sine activation functions with



(a) $\xi = 5$



(b) $\xi = 7$

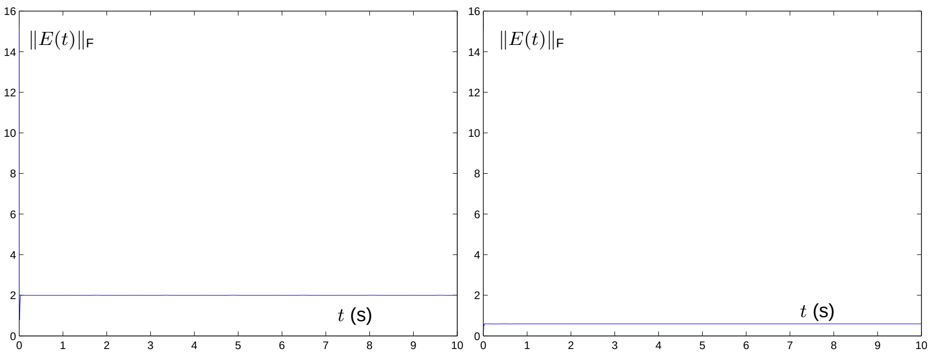
Fig. 5. Residual errors of largely-perturbed ZNN (8) using hyperbolic sine activation functions with different values of ξ (still with $\gamma = 1$ and $\Delta\omega_{ij} = 10^2, i, j \in \{1, 2\}$).

$\xi = 5$ are used, the convergence time of the largely-perturbed ZNN (8) is faster than that using linear activation functions, and the steady-state residual error of the largely-perturbed ZNN (8) is around 100 times smaller than that using linear activation functions. In summary, compared to the situation of using linear activation functions, superior robustness performance is achieved for the ZNN models by using hyperbolic sine activation functions.

Example 3. In order to further investigate the efficacy of ZNN model (4) using hyperbolic sine activation functions for larger-dimension matrices, let us consider equation (1) with the following time-varying Toeplitz matrix $A(t)$:

$$A(t) = \begin{bmatrix} a_1(t) & a_2(t) & a_3(t) & \cdots & a_n(t) \\ a_2(t) & a_1(t) & a_2(t) & \cdots & a_{n-1}(t) \\ a_3(t) & a_2(t) & a_1(t) & \cdots & a_{n-2}(t) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_n(t) & a_{n-1}(t) & a_{n-2}(t) & \cdots & a_1(t) \end{bmatrix} \in R^{n\times n}. \tag{11}$$

Let $a_1(t) = n+\sin(5t)$, and $a_k(t) = \cos(5t)/(k-1)$ with $k = 2, 3, \cdots, n$. Figure 11 shows the simulation results of ZNN model (4) using hyperbolic sine activation functions for time-varying square roots finding of the above Toeplitz matrix $A(t)$ in the situation of $n = 4$ and $n = 10$. As seen from Figure 11(a) and (b), the residual errors $\|E(t)\|_F$ of ZNN (4) for factorizing Toeplitz matrices with different dimensions (i.e., $R^{4\times 4}$ and $R^{10\times 10}$) both diminish to zero, which implies that their corresponding state matrices always converge to the time-varying square roots of $A(t)$. These further substantiate the efficacy of the ZNN model (4) using hyperbolic sine activation functions on solving for time-varying square roots of larger-dimension matrices.



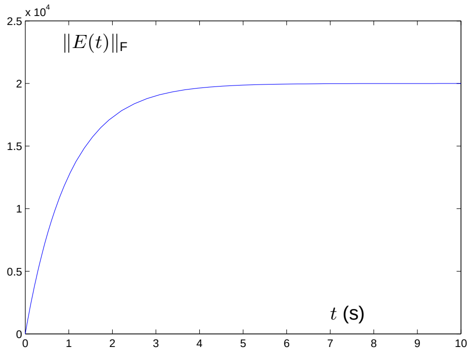
(a) $\gamma = 10$ (b) $\gamma = 100$

Fig. 6. Residual errors of largely-perturbed ZNN (8) using hyperbolic sine activation functions with different values of γ (with $\xi = 3$ and $\Delta\omega_{ij} = 10^2, i, j \in \{1, 2\}$).

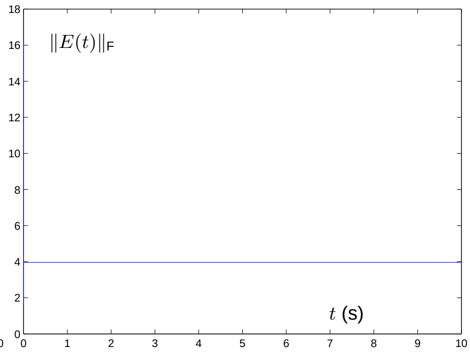
Before ending this section, it is worth noting that, because of the similarity of the results and figures to the above ones, the corresponding simulations of ZNN models starting from negative-definite initial states are not presented (though they have done successfully and consistently with the theoretical results given in the theorems and lemmas of the paper). Besides, comparisons between ZNN model (4) and other methods can be seen in Appendix B.

6. Conclusions

In this paper, the convergence and robustness properties of Zhang neural network using hyperbolic sine activation functions have been investigated, analyzed and verified for the online solution of time-varying matrix square roots. Other computer-simulation results of using different activation functions (e.g., sigmoid activation functions, hard-limiting activation functions, piecewise-linear activation functions, and hyperbolic tangent activation functions) have been omitted due to the similarity and the space limitation, and thus hyperbolic activation functions have only been compared with linear activation functions in this paper. Theoretical analysis has demonstrated that superior convergence and robustness can be achieved readily for ZNN models even in the context of (very) large model-implementation errors by using hyperbolic sine activation functions, as compared to those using linear activation functions. Computer-simulation results have further substantiated the efficacy and superiority of the ZNN models using hyperbolic sine activation functions for time-varying matrix square roots finding.



(a) linear activation functions



(b) hyperbolic sine functions ($\xi = 5$)

Fig. 7. Residual errors of very largely perturbed ZNN (8) for time-varying matrix square roots finding of (9) (with $\gamma = 1$ and $\Delta\omega_{ij} = 10^4, i, j \in \{1, 2\}$).

Appendix A

Proof of Theorem 2.1

By vectorizing ZNN model (4) based on the Kronecker-product and vectorization technique, the left hand side of equation (4) is

$$\begin{aligned}\text{vec}(X\dot{X} + \dot{X}X) &= \text{vec}(X\dot{X}) + \text{vec}(\dot{X}X)\\ &= (I \otimes X)\text{vec}(\dot{X}) + (X^{\mathsf{T}} \otimes I)\text{vec}(\dot{X})\\ &= (I \otimes X + X^{\mathsf{T}} \otimes I)\text{vec}(\dot{X}),\end{aligned}$$

where argument t is dropped for presentation convenience. The right hand side of (4) is

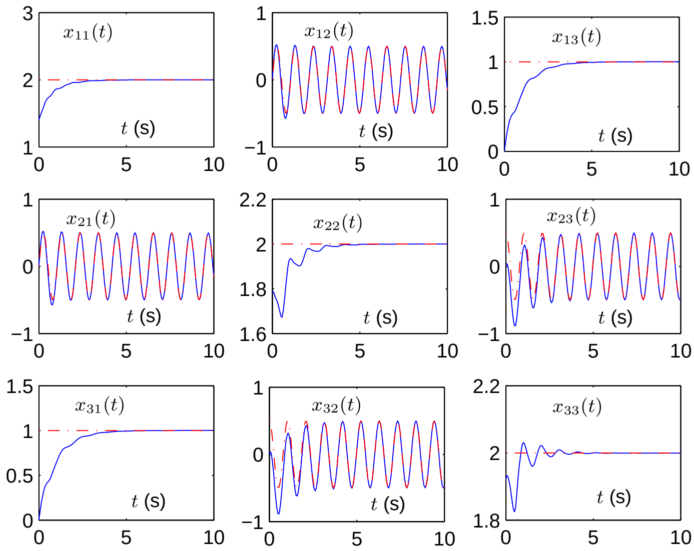
$$\begin{aligned}\text{vec}\big(-\gamma\mathcal{F}(X^2 - A) + \dot{A}\big) &= -\gamma\text{vec}\big(\mathcal{F}(X^2 - A) + \dot{A}\big)\\ &= -\gamma\text{vec}\big(\mathcal{F}(X^2 - A)\big) + \text{vec}(\dot{A}).\end{aligned} \tag{12}$$

Note that the aforementioned activation function array $\mathcal{F}(\cdot)$ could also be vectorized, i.e., being from $R^{n^2\times 1}$ to $R^{n^2\times 1}$. Thus, we have

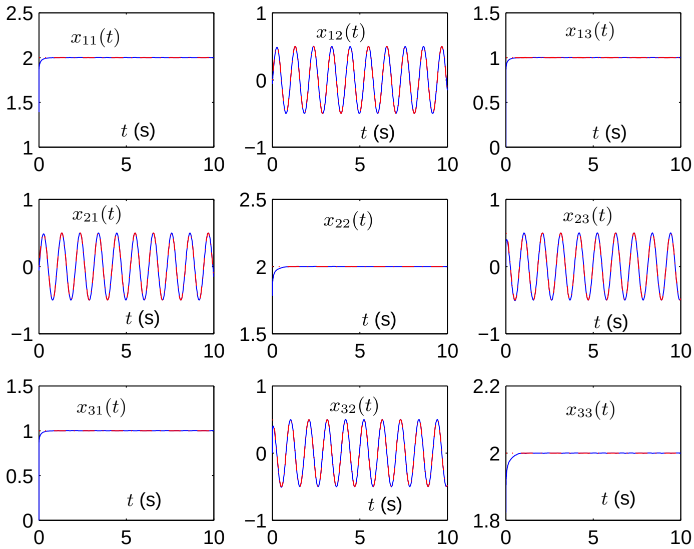
$$\begin{aligned}\text{vec}\big(\mathcal{F}(X^2 - A)\big) &= \mathcal{F}\big(\text{vec}(X^2 - A)\big)\\ &= \mathcal{F}\big(\text{vec}(X^2) - \text{vec}(A)\big)\\ &= \mathcal{F}\big((I \otimes X)\text{vec}(X) - \text{vec}(A)\big).\end{aligned} \tag{13}$$

Combining (12) and (13) yields the vectorization of the right hand side of equation (4):

$$\text{vec}\big(-\gamma\mathcal{F}(X^2 - A) + \dot{A}\big) = -\gamma\mathcal{F}\big((I \otimes X)\text{vec}(X) - \text{vec}(A)\big) + \text{vec}(\dot{A}).$$

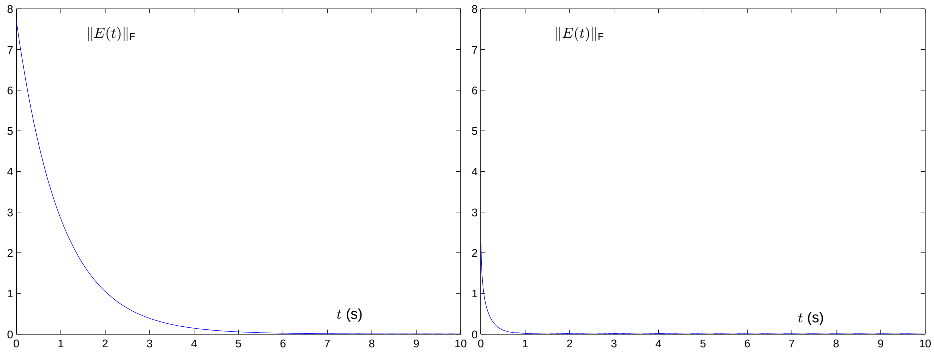


(a) linear activation functions



(b) hyperbolic sine functions with $\xi = 5$

Fig. 8. Online solution of time-varying matrix square root problem (10) by ZNN (4) with $\gamma = 1$, where the neural-network solutions $X(t)$ are denoted by solid blue curves, and the theoretical time-varying square root $A^{1/2}(t)$ is denoted by red dash-dotted curves.



(a) linear activation functions (b) hyperbolic sine functions ($\xi = 5$)

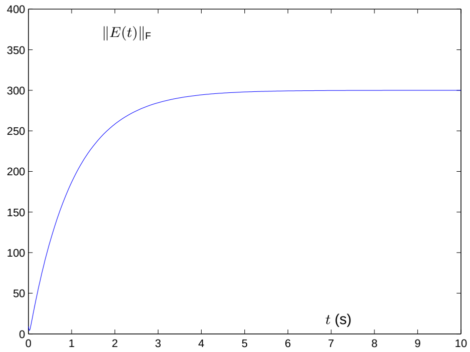
Fig. 9. Residual errors of ZNN (4) with $\gamma = 1$ for TVMSR finding of (10).

Evidently, the vectorization of both sides of matrix-form differential equation (4) should be equal, which generates the vector-form differential equation (6). The proof of Theorem 2.1 is thus completed.

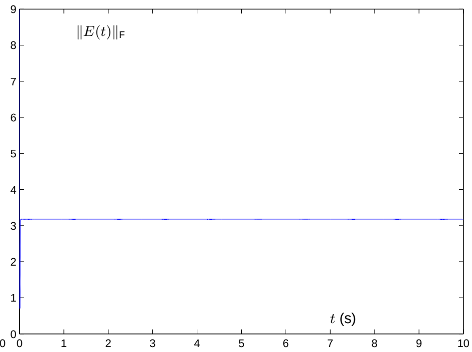
Appendix B

To keep the completeness of the paper and make interesting comparisons with ZNN model (4), numerical methods are presented and investigated here for the online solution of time-varying matrix square roots as well. Based on the results in [5, 6], four numerical methods (i.e., Newton iteration, DB iteration, CR iteration, and IN iteration) are used frequently to solve for static matrix square roots, which is the constant case of (1). For comparison, the important iterative formulas of the numerical methods are listed as follows in order to solve for the time-varying matrix square roots.

– Newton iteration:
$$\begin{cases} X_k H_k + H_k X_k = A_k - X_k^2, \\ X_{k+1} = X_k + H_k, \end{cases} \tag{14}$$
where iteration index $k = 0, 1, 2, 3, \cdots$. In addition, $A(t)$ is discretized by the standard sampling method, of which the sampling gap is denoted by $\tau = t_{k+1} - t_k$. For convenience and for consistency with X_k, we use A_k standing for $A(t = k\tau)$.
– DB iteration:
$$\begin{cases} X_{k+1} = (X_k + Y_k^{-1})/2, \quad X_0 = A_0, \\ Y_{k+1} = (Y_k + X_k^{-1})/2, \quad Y_0 = I, \end{cases} \tag{15}$$
where $A_0 = A(t = 0)$.



(a) linear activation functions



(b) hyperbolic sine functions ($\xi = 5$)

Fig. 10. Residual errors of largely-perturbed ZNN (8) for time-varying matrix square roots finding of (10) (with $\gamma = 1$ and $\Delta\omega_{ij} = 10^2, i, j \in \{1, 2, 3\}$).

- CR iteration:

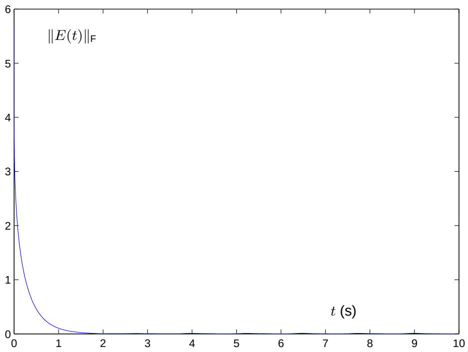
$$\begin{cases} Y_{k+1} = -Y_k Z_k^{-1} Y_k, & Y_0 = I - A_0, \\ Z_{k+1} = Z_k + 2Y_{k+1}, & Z_0 = 2(I + A_0), \\ X_{k+1} = Z_{k+1}/4. \end{cases} \tag{16}$$

- IN iteration:

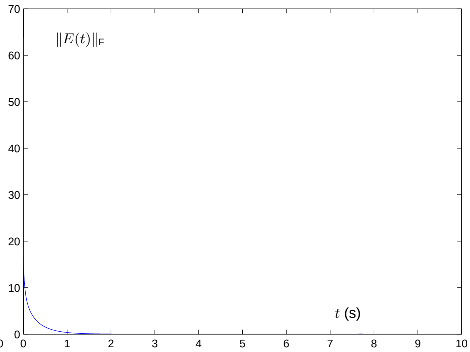
$$\begin{cases} X_{k+1} = X_k + E_k, & X_0 = A_0, \\ E_{k+1} = -E_k X_{k+1}^{-1} E_k/2, & E_0 = (I - A_0)/2. \end{cases} \tag{17}$$

According to the formulas above, the solving process of DB iteration (15), CR iteration (16) and IN iteration (17) do not contain the information of A_k in the iteration process [i.e., without A_k in the formulas], which implies that the sequences $\{X_k\}$ generated by these methods converge to the matrix square roots of A_0 (in view of the initial conditions) and then do not change [though the matrix $A(t)$ changes]. The three numerical methods are theoretically/intrinsically designed for online solution of time-invariant (or termed, static, constant) matrix square roots, but not aimed at finding the time-varying matrix square roots in real time t. The three methods may be approximately effective, when the time-varying problem can be divided into many static problems to solve via a short-time invariance assumption [i.e., to find the matrix square roots of A_k separately and successively], but this may not be accurate and may not be what solving a time-varying problem in real time t means. Therefore, DB iteration (15), CR iteration (16) and IN iteration (17) are, generally speaking, not applicable to online solution of time-varying matrix square roots.

Different with the aforementioned three iterations, Newton iteration (14) contains A_k in the formula, so the sequence $\{X_k\}$ generated by Newton iteration



(a) $n = 4$

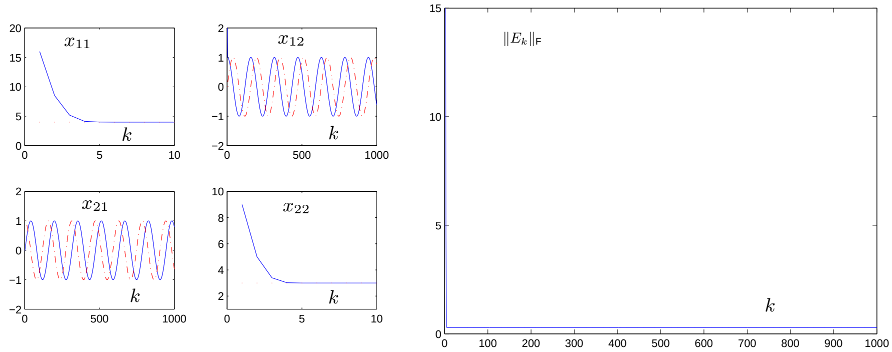


(b) $n = 10$

Fig. 11. Residual errors of ZNN (4) using hyperbolic sine activation functions for time-varying square roots finding of the Toeplitz matrix $A(t)$ with different dimensions.

(14) can change with A_k correspondingly [e.g., X_2 corresponds to the estimation of the matrix square root of $A_2 = A(t = 2\tau)$] and try to find the time-varying matrix square roots in real time t. In addition, the relationship between the ZNN model and Newton iteration for online solution of static matrix square roots is presented in [21]. Considering Example 1 again, we apply Newton iteration (14) to online solution of the time-varying matrix square root of (9). As illustrated in Figure 12(a), the sequence $\{X_k\}$ generated by Newton iteration (14) with the sampling gap $\tau = 0.01$ could not fit well with the theoretical square root even after a long period of time. Its residual error $\|E_k\|_F$ is depicted in Figure 12(b), which shows that the error of the solution computed by Newton iteration (14) is considerably large. Comparing Figures 2 and 3 with Figure 12, we can see that the ZNN model (4) is more effective and more accurate than Newton iteration (14) for online time-varying matrix square roots finding.

In addition to the above, the detailed comparison between the ZNN model and the four numerical methods (i.e., Newton iteration, DB iteration, CR iteration, and IN iteration) for online solution of static matrix square roots can refer to [21]. Furthermore, the gradient neural network (GNN) model for online solution of time-varying matrix square roots is simulated and compared with the ZNN model in [22]. These results substantiate that the ZNN model (4), which is theoretically/intrinsically designed for online solution of time-varying matrix square roots, is thus more effective than the four numerical methods (i.e., Newton iteration, DB iteration, CR iteration, and IN iteration) and the GNN model for online time-varying matrix square roots finding (or termed, time-varying matrix quadratic factorization).



(a) sequence $\{X_k\}$ of Newton iteration (b) $\|E_k\|_\mathrm{F}$ of Newton iteration

Fig. 12. Convergence performance of Newton iteration (14) for online solution of time-varying matrix square root problem (9).

References

1. Higham, N.J.: Stable iterations for the matrix square root. Numerical Algorithms 15(2), 227–242 (1997)
2. Long, J., Hu, X., Zhang, L.: Newton's method with exact line search for the square root of a matrix. Journal of Physics: Conference Series 96(1), 1–5 (2008)
3. Hasan, M.A., Rahman, S.: Fixed point iterations for computing square roots and the matrix sign function of complex matrices. In: Proceedings of the 39th IEEE Conference on Decision and control. pp. 4253–4258. Sydeny, Australia (2000)
4. Johnson, C.R., Okubo, K., Reams, R.: Uniqueness of matrix square roots and an application. Linear Algebra and Its Applications 323(1–3), 51–60 (2001)
5. Higham, N.J.: Functions of Matrices: Theory and Computation. Society for Industrial and Applied Mathematics, SIAM, England (2008)
6. Higham, N.J.: Accuracy and Stability of Numerical Algorithms. Second Edition. Society for Industrial and Applied Mathematics, SIAM, England (2002)
7. Zhang, Y., Leithead, W.E., Leith, D.J.: Time-series Gaussian process regression based on Toeplitz computation of $O(N^2)$ operations and $O(N)$-level storage. In: Proceedings of the 44th IEEE Conference on Decision and Control. pp. 3711–3716. Seville, Spain (2005)
8. Zhang, Y.: Revisit the analog computer and gradient-based neural system for matrix inversion. In: Proceedings of the IEEE International Symposium on Intelligent control. pp. 1411–1416. Limassol, Cyprus (2005)
9. Zhang, Y., Jiang, D., Wang, J.: A recurrent neural network for solving Sylvester equation with time-varying coefficients. IEEE Transactions on Neural Networks 13(5), 1053–1063 (2002)
10. Belair, J., Campell, S.A., Driessche, P.V.: Frustration, stability, and delay-induced oscillations in a neural network model. SIAM Journal on Applied Mathematics 56(1), 245–255 (1996)

11. Zhang, Y., Ge, S.S.: Design and analysis of a general recurrent neural network model for time-varying matrix inversion. IEEE Transactions on Neural Networks 16(6), 1477–1490 (2005)
12. Ma, W., Zhang, Y., Wang, J.: MATLAB Simulink modeling and simulation of Zhang neural networks for online time-varying Sylvester equation solving. In: Proceedings of IEEE International Joint Conference on Neural Networks. pp. 285–289. Hong Kong, China (2008)
13. Wang, J.: Recurrent neural networks for computing pseudoinverses of rank-deficient matrices. SIAM Journal on Applied Mathematics 18(5), 1479–1493 (1997)
14. Driessche, V.D., Zou, X.: Global attractivity in delayed Hopfield neural network models. SIAM Journal on Applied Mathematics 58(6), 1878–1890 (1998)
15. Mead, C.: Analog VLSI and Neural Systems. Addison-Wesley, Reading, Boston, MA, USA (1989)
16. El Emary, I., Emar, W., Aqel, M.J.: The adaptive fuzzy designed PID controller using wavelet network. Computer Science and Information Systems 6(2), 141–163 (2009)
17. Zhang, Y., Wang, J.: Global exponential stability of recurrent neural networks for synthesizing linear feedback control systems via pole assignment. IEEE Transactions on Neural Networks 13(3), 633–644 (2002)
18. Zhang, Y., Wang, J.: A dual neural network for convex quadratic programming subject to linear equality and inequality constraints. Physics Letters A 298(4), 271–278 (2002)
19. B Meini.: The matrix square root from a new functional perspective: theoretical results and computational issues. SIAM Journal on Matrix Analysis and Applications 26(2), 362–376 (2004)
20. B Iannazzo.: A note on computing the matrix square root. Calcolo 40, 273–283 (2003)
21. Zhang, Y., Yang, Y., Cai, B., Guo, D.: Zhang neural network and its application to Newton iteration for matrix square root estimation. Neural Computing and Applications 21(3) 1–8 (2010)
22. Zhang, Y., Yang, Y., Yi, C., Chen, K.: Simulation and comparison of Zhang neural network and gradient neural network solving for time-varying matrix square roots. In: Proceedings of the 2nd International Symposium on Intelligent Information Technology Application. pp. 966–970. Shanghai, China (2008)
23. Zhang, Y., Peng. H.: Zhang neural network for linear time-varying equation solving and its robotic application. In: Proceedings of International conference on Machine Learning and Cybernetics. pp. 3543–3548 (2007)
24. Zhang, Y., Chen, K., Ma, W.: MATLAB simulation and comparison of Zhang neural network and gradient neural network for online solutin of linear time-varying equations. In: Proceedings of International Conference on Life System Modeling and Simulation. pp. 450–454. Shanghai, China (2007)
25. Chapman, J.S.: MATLAB Programming for Engineers. Thomson Learning, USA (2002)
26. Zhang, Y.: On the LVI-based primal-dual neural network for solving online linear and quadratic programming problems. In: Proceedings of the American Control Conference. pp. 1351–1356 (2005)
27. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis. Cambridge University Press, Cambridge (1991)
28. Zhang, Y.: Dual Neural Networks: Design, Analysis, and Application to Redundant Robotics. Progress in Neurocomputing Research. Nova Science Publishers. pp. 41–81. New York (2007)

Yunong Zhang received the B.S. degree from Huazhong University of Science and Technology, Wuhan, China, in 1996, the M.S. degree from South China University of Technology, Guangzhou, China, in 1999, and the Ph.D. degree from Chinese University of Hong Kong, Shatin, Hong Kong, China, in 2003.

He is currently a professor with the School of Information Science and Technology, Sun Yat-sen University (SYSU), Guangzhou, China. Before joining the SYSU in 2006, he had been with the National University of Ireland, Maynooth, Ireland, the University of Strathclyde, Glasgow, U.K., and the National University of Singapore, Singapore, since 2003. His main research interests include neural networks, robotics, computation and optimization. His web-page is now available at http://sist.sysu.edu.cn/~zhynong.

Long Jin was born in Lanzhou, China, in 1988. He received the B.S. degree in Automation from Sun Yat-sen University, Guangzhou, China, in 2011. He is currently pursuing the Ph.D. degree in Communication and Information Systems at the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China. His main research interests include neural networks, robotics and intelligent information processing.

Zhende Ke received the B.S. degree in Electronic Information Science and Technology and M.S. degree in Communication and Information Systems from Sun Yat-sen University, Guangzhou, China, in 2010 and 2012, respectively. His research interests include neural networks and numerical methods.

Received: January 21, 2012; Accepted: December 1, 2012.