ZNNs with a Varying-Parameter Design Formula for Dynamic Sylvester Quaternion Matrix Equation

Lin Xiao, Wenqian Huang, Xiaopeng Li, Fuchun Sun, Qing Liao, Lei Jia, Jichun Li, and Sai Liu

***Abstract*—This paper aims to studying how to solve dynamic Sylvester quaternion matrix equation (DSQME) using the neural dynamic method. In order to solve the DSQME, the complex representation method is firstly adopted to derive the equivalent dynamic Sylvester complex matrix equation (DSCME) from the DSQME. It is proved that the solution to the DSCME is the same with that of the DSQME in essence. Then, a state-of-the-art neural dynamic method is presented to generate a general dynamic-varying parameter zeroing neural network (DVPZNN) model with its global stability being guaranteed by Lyapunov theory. Specifically, when the linear activation function is utilized in the DVPZNN model, the corresponding model (termed LDVPZNN) achieves finite-time convergence, and time range is theoretically calculated. When the nonlinear power-sigmoid activation function is utilized in the DVPZNN model, the corresponding model (termed PSDVPZNN) achieves the better convergence as compared with the LDVPZNN model, which is proved in detail. At last, three examples are presented to compare the solution performance of different neural models for the DSQME and the equivalent DSCME, and the results verify the correctness of the theories, and the superiority of the proposed two DVPZNN models.**

***Index Terms*—Zeroing neural network, dynamic Sylvester quaternion matrix equation, complex representation, finite-time convergence.**

I. Introduction

QUATERNION was proposed by William Luyun Hamilton in the 19th century when studying how to extend complex numbers to higher dimensions [1]. In fact, quaternion is an extension of the field of ordinary complex numbers. The ordinary complex number field is composed of a real part and an imaginary part. When adding two imaginary parts on the basis of the complex number field, the quaternion was formally proposed. Throughout the paper, $\mathbb{C}$ represents the complex number field, and $\mathbb{Q}$ represents the quaternion field. Quaternions are widely used in many aspects, such as image analysis [2], [3], attitude control [4], and signal processing [5], [6].

In recent years, solving the problems related to Sylvester equation has become a hot issue. The Sylvester equation is a very common equation. There have been a lot of researches on the dynamic Sylvester equation. For example, a recurrent neural network (RNN) proposed by Zhang *et al.* [7], which is also termed zeroing neural network (ZNN), was used to solve the Sylvester equation with dynamic coefficients; the generalized Sylvester equation was solved by using generalized Schur methods [8]; and a semi-supervised multi-label learning was worked by solving the Sylvester equation [9]. Generally, the dynamic Sylvester equation is:

$$D(t)U(t) + U(t)G(t) = J(t) \in \mathbb{R}^{n\times n}, \tag{1}$$

where $D(t)$, $G(t)$ and $J(t)$ are known, and $U(t)$ is unknown. By the previous three values, the value of $U(t)$ can be obtained. Solving the Sylvester equation is of great significance in some fields, such as linear system [10], [11], signal processing [12], [13], pole configuration [14], and observer design [15].

Combining the quaternion and the Sylvester equation can get the Sylvester quaternion equation. The Sylvester quaternion equation means that four elements in the equation are all quaternions, including three known quaternions and one unknown quaternion to be solved. The Sylvester quaternion equations include the static and dynamic Sylvester quaternion equations, which have been widely used in the fields of robot [16], [17], human body image [18], [19] and so on. The solution of the static Sylvester quaternion equation will not change with time, while the solution of the dynamic Sylvester quaternion equation will change with the change of time. Generally, the dynamic Sylvester quaternion equation is an extension of the static Sylvester quaternion equation. Therefore, the dynamic Sylvester quaternion matrix equation (DSQME) is a more general case. When all three imaginary parts are equal to 0 in the quaternion, the DSQME becomes a real number Sylvester matrix equation. When two imaginary parts are equal to 0 and the others are not equal to 0, the DSQME becomes a complex number Sylvester matrix equation.

ZNN as a kind of neural network [20] can effectively solve time-varying problems, including the DSQME. In the beginning, many researchers improved its convergence by changing the activation function. Guo *et al.* [21] proposed a novel ZNN model activated by Li-function; based on the adaptive design coefficients of the symbolic dual-power nonlinear activation function, Jian *et al.* [22] proposed three new adaptive ZNN models; in [23], the proposed complex ZNN models are activated with various complex activation functions respectively, such as sign function and Li-function. Later, researchers proposed different design formulas to get

L. Xiao, W. Huang, X. Li, L. Jia and S. Liu are the Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing, and MOE-LCSM, Hunan Normal University, Changsha, Hunan 410081, China. (e-mail: xiaolin5@hunnu.edu.cn; Wqsunshine823@163.com; 478084904@qq.com; LJia@smail.hunnu.edu.cn; 2453069767@qq.com).

F. Sun is with the Department of Computer Science, Tsinghua University, Beijing 100000, China. (e-mail: fcsun@tsinghua.edu.cn).

Q. Liao is with the College of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518005, China. (e-mail: liaoqing@hit.edu.cn).

J. Li is with the School of Science, Engineering and Design, Teesside University, Middlesbrough TS1 3BX, U.K. (e-mail: jl20340@essex.ac.uk).

better performance of the ZNN. Jin *et al.* [24] proposed a noise-tolerant ZNN design formula; Shi *et al.* [25] used integral-type error function and twice ZNN formula. Finally, researchers found that adding varying parameters can also improve convergence and robustness of the ZNN. Tan *et al.* [26] presented new varying-parameter ZNN models; Zhang *et al.* [27] proposed a varying-gain RNN, and other studies on the RNN are proposed [28], [29]. However, the studied varying parameters of the ZNN increase over time, which will increase the calculation complexity of the model and waste a lot of resources. Therefore, in this paper, by combining the design formula with dynamic-varying parameters, the dynamic-varying parameter ZNN (DVPZNN) design formula is proposed. The DVPZNN model can be obtained on the basis of the DVPZNN design formula correspondingly.

Using different activation functions for the proposed DVPZNN model can get different ZNN models. When using the linear activation function in the DVPZNN model, linear DVPZNN (LDVPZNN) model is obtained; when using the nonlinear power-sigmoid activation function, power-sigmoid DVPZNN (PSDVPZNN) model is obtained. The main idea of this paper is to put forward two novel DVPZNN models, named LDVPZNN and PSDVPZNN models, based on a varying design parameter for solving the DSQME.

The organization structure of this paper is as below. Section II introduces the concepts related to the quaternion and the description of the problem to be solved. In Section III, two DVPZNN models to solve the DSQME are proposed. In Section IV, the stability and convergence of the proposed models are analyzed theoretically. In Section V, three simulative experiments are provided to display the excellent attributes of the presented two models. Section VI concludes this paper globally. The main contributions of this paper are as follows.

- The dynamic Sylvester quaternion matrix equation (DSQME) is studied in this work for the first time. Based on the complex representation method, the DSQME is transformed into the dynamic Sylvester complex matrix equation (DSCME), and the correctness of the complex representation of the quaternion matrix is proven.
- A novel design formula which can adapt to the change of the error is applied to the ZNN model, with two dynamic-varying parameter zeroing neural network (DVPZNN) models being further proposed. The global stability and the finite-time convergence are theoretically proven and analyzed.
- Through three examples and comparing with the classic ZNN model activated with the sign-bi-power function, it is concluded that two proposed DVPZNN models in this paper have superior convergence.

II. Problem Description

Quaternion has certain rules and methods of operation. In this section, the rules and methods will be introduced in detail. The quaternion consists of one real part and three imaginary parts. Generally, a quaternion $\hat{q}$ is:

$$\hat{q} = q_0 + q_1 i + q_2 j + q_3 k, \tag{2}$$

where q_0 denotes the real part and $q_1 i + q_2 j + q_3 k$ denotes the imaginary part; i, j and k are all imaginary units. The relationships between three imaginary units are as follows: $i^2 = j^2 = k^2 = ijk = -1, i^3 = -i, j^3 = -j, k^3 = -k, -ij = ji = -k, -jk = kj = -i, -ki = ik = -j$.

There are two ways to express a quaternion in a matrix, which are real representation and complex representation. Since this paper deals with complex representation, the following contents about complex representation will be introduced in detail. Through the above rules, quaternion (2) can be transformed into: $\hat{q} = (q_0 + q_1 i) + (q_2 + q_3 i)j$. Let $h_0 = q_0 + q_1 i, h_1 = q_2 + q_3 i$, we can get: $\hat{q} = h_0 + h_1 j$. Generally, the method of expressing a quaternion with a second-order complex number matrix is called the complex representation. For the convenience of representation, we define $\Phi(\cdot)$ as the complex representation of the quaternion. Next, the complex representation of the quaternion $\hat{q}$ can be expressed as [30]:

$$\Phi(\hat{q}) = \begin{bmatrix} q_0 - q_3 i & -q_2 - q_1 i \\ q_2 - q_1 i & q_0 + q_3 i \end{bmatrix} \in \mathbb{C}^{2\times 2}.$$

For the quaternion matrix $\hat{Q} \in \mathbb{Q}^{2\times 2}$, $\hat{Q} = Q_0 + Q_1 i + Q_2 j + Q_3 k$. Express $\hat{Q}$ as the complex representation of quaternion matrix:

$$\Phi(\hat{Q}) = \begin{bmatrix} Q_0 - Q_3 i & -Q_2 - Q_1 i \\ Q_2 - Q_1 i & Q_0 + Q_3 i \end{bmatrix} \in \mathbb{C}^{4\times 4}.$$

It has been proved that the result of multiplying two quaternions is the same as the result of multiplying their complex representations [30]. In principle, since the complex representation of the quaternion matrix is derived from the complex representation of the quaternion, the quaternion matrix also has the above properties. Below, a theorem is given to prove this point.

Theorem 1. *The complex representation of the product of two quaternion matrices is equal to the product of the complex representations of two quaternion matrices.*

Proof: Give four quaternions $a = a_0 + a_1 i + a_2 j + a_3 k$, $b = b_0 + b_1 i + b_2 j + b_3 k$, $c = c_0 + c_1 i + c_2 j + c_3 k$, and $d = d_0 + d_1 i + d_2 j + d_3 k$. These four quaternion arrays form a quaternion matrix $X \in \mathbb{Q}^{2\times 2}$:

$$X = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \tag{3}$$

and the complex representation of the quaternion matrix X is:

$$\Phi(X) = \begin{bmatrix} X_0 - X_3 i & -X_2 - X_1 i \\ X_2 - X_1 i & X_0 + X_3 i \end{bmatrix}, \tag{4}$$

where

$$X_0 = \begin{bmatrix} a_0 & b_0 \\ c_0 & d_0 \end{bmatrix}, \quad X_1 = \begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix}, \quad X_2 = \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix}, \quad X_3 = \begin{bmatrix} a_3 & b_3 \\ c_3 & d_3 \end{bmatrix}. \tag{5}$$

Similarly, give four quaternions $e = e_0 + e_1 i + e_2 j + e_3 k$, $f = f_0 + f_1 i + f_2 j + f_3 k$, $g = g_0 + g_1 i + g_2 j + g_3 k$, and $h = h_0 + h_1 i + h_2 j + h_3 k$. These four quaternion arrays form a quaternion matrix $Y \in \mathbb{Q}^{2\times 2}$:

$$Y = \begin{bmatrix} e & f \\ g & h \end{bmatrix}, \tag{6}$$

and the complex representation of the quaternion matrix Y is:

$$\Phi(Y)=\begin{bmatrix} Y_0-Y_3 i & -Y_2-Y_1 i \\ Y_2-Y_1 i & Y_0+Y_3 i \end{bmatrix}, \tag{7}$$

where

$$\begin{aligned} Y_0&=\begin{bmatrix} e_0 & f_0 \\ g_0 & h_0 \end{bmatrix}, \quad Y_1=\begin{bmatrix} e_1 & f_1 \\ g_1 & h_1 \end{bmatrix}, \\ Y_2&=\begin{bmatrix} e_2 & f_2 \\ g_2 & h_2 \end{bmatrix}, \quad Y_3=\begin{bmatrix} e_3 & f_3 \\ g_3 & h_3 \end{bmatrix}. \end{aligned} \tag{8}$$

The product of two quaternion matrices is as follows:

$$XY=\begin{bmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{bmatrix}. \tag{9}$$

Let

$$\begin{aligned}
&a_0e_0-a_1e_1-a_2e_2-a_3e_3+b_0g_0-b_1g_1-b_2g_2-b_3g_3=\alpha_0,\\
&a_0e_1+a_1e_0+a_2e_3-a_3e_2+b_0g_1+b_1g_0+b_2g_3-b_3g_2=\alpha_1,\\
&a_0e_2-a_1e_3+a_2e_0+a_3e_1+b_0g_2-b_1g_3+b_2g_0+b_3g_1=\alpha_2,\\
&a_0e_3+a_1e_2-a_2e_1+a_3e_0+b_0g_3+b_1g_2-b_2g_1+b_3g_0=\alpha_3,\\
&a_0f_0-a_1f_1-a_2f_2-a_3f_3+b_0h_0-b_1h_1-b_2h_2-b_3h_3=\beta_0,\\
&a_0f_1+a_1f_0+a_2f_3-a_3f_2+b_0h_1+b_1h_0+b_2h_3-b_3h_2=\beta_1,\\
&a_0f_2-a_1f_3+a_2f_0+a_3f_1+b_0h_2-b_1h_3+b_2h_0+b_3h_1=\beta_2,\\
&a_0f_3+a_1f_2-a_2f_1+a_3f_0+b_0h_3+b_1h_2-b_2h_1+b_3h_0=\beta_3,\\
&c_0e_0-c_1e_1-c_2e_2-c_3e_3+d_0g_0-d_1g_1-d_2g_2-d_3g_3=\eta_0,\\
&c_0e_1+c_1e_0+c_2e_3-c_3e_2+d_0g_1+d_1g_0+d_2g_3-d_3g_2=\eta_1,\\
&c_0e_2-c_1e_3+c_2e_0+c_3e_1+d_0g_2-d_1g_3+d_2g_0+d_3g_1=\eta_2,\\
&c_0e_3+c_1e_2-c_2e_1+c_3e_0+d_0g_3+d_1g_2-d_2g_1+d_3g_0=\eta_3,\\
&c_0f_0-c_1f_1-c_2f_2-c_3f_3+d_0h_0-d_1h_1-d_2h_2-d_3h_3=\rho_0,\\
&c_0f_1+c_1f_0+c_2f_3-c_3f_2+d_0h_1+d_1h_0+d_2h_3-d_3h_2=\rho_1,\\
&c_0f_2-c_1f_3+c_2f_0+c_3f_1+d_0h_2-d_1h_3+d_2h_0+d_3h_1=\rho_2,\\
&c_0f_3+c_1f_2-c_2f_1+c_3f_0+d_0h_3+d_1h_2-d_2h_1+d_3h_0=\rho_3.
\end{aligned}$$

The above formula can be converted to:

$$XY=\begin{bmatrix} \alpha_0+\alpha_1 i+\alpha_2 j+\alpha_3 k & \beta_0+\beta_1 i+\beta_2 j+\beta_3 k \\ \eta_0+\eta_1 i+\eta_2 j+\eta_3 k & \rho_0+\rho_1 i+\rho_2 j+\rho_3 k \end{bmatrix}.$$

The complex representation of XY is

$$\begin{aligned}&\Phi(XY)\\ &=\begin{bmatrix} \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix}-\begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i & -\begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix}-\begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i \\ \begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix}-\begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i & \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix}+\begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i \end{bmatrix}.\end{aligned} \tag{10}$$

The product of the complex representation of two quaternion matrices X and Y is as follows:

$$\begin{aligned}\Phi(X)\Phi(Y)&=\begin{bmatrix} X_0-X_3 i & -X_2-X_1 i \\ X_2-X_1 i & X_0+X_3 i \end{bmatrix}\begin{bmatrix} Y_0-Y_3 i & -Y_2-Y_1 i \\ Y_2-Y_1 i & Y_0+Y_3 i \end{bmatrix}\\
&=\begin{bmatrix} \begin{bmatrix} a_0 & b_0 \\ c_0 & d_0 \end{bmatrix}-\begin{bmatrix} a_3 & b_3 \\ c_3 & d_3 \end{bmatrix} i & -\begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix}-\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} i \\ \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix}-\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} i & \begin{bmatrix} a_0 & b_0 \\ c_0 & d_0 \end{bmatrix}+\begin{bmatrix} a_3 & b_3 \\ c_3 & d_3 \end{bmatrix} i \end{bmatrix}\cdot\\
&\quad\begin{bmatrix} \begin{bmatrix} e_0 & f_0 \\ g_0 & h_0 \end{bmatrix}-\begin{bmatrix} e_3 & f_3 \\ g_3 & h_3 \end{bmatrix} i & -\begin{bmatrix} e_2 & f_2 \\ g_2 & h_2 \end{bmatrix}-\begin{bmatrix} e_1 & f_1 \\ g_1 & h_1 \end{bmatrix} i \\ \begin{bmatrix} e_2 & f_2 \\ g_2 & h_2 \end{bmatrix}-\begin{bmatrix} e_1 & f_1 \\ g_1 & h_1 \end{bmatrix} i & \begin{bmatrix} e_0 & f_0 \\ g_0 & h_0 \end{bmatrix}+\begin{bmatrix} e_3 & f_3 \\ g_3 & h_3 \end{bmatrix} i \end{bmatrix}\\
&=\begin{bmatrix} \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix}-\begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i & -\begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix}-\begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i \\ \begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix}-\begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i & \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix}+\begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i \end{bmatrix}.\end{aligned} \tag{11}$$

Through a series of calculations, the results of $\Phi(X)\Phi(Y)$ and $\Phi(XY)$ are equal, since (10) and (11) are the same. The proof is completed. ■

In this paper, the following 2×2 DSQME is considered:

$$\hat{D}(t)\hat{U}(t)+\hat{U}(t)\hat{G}(t)=\hat{J}(t), \tag{12}$$

where $\hat{D}(t),\hat{G}(t)$ and $\hat{J}(t)\in\mathbb{Q}^{2\times 2}$ are known quaternion matrices, and $\hat{U}(t)\in\mathbb{Q}^{2\times 2}$ is an unknown quaternion matrix to be solved. Firstly, we get the DSCME based on Theorem 1:

$$\Phi(\hat{D}(t))\Phi(\hat{U}(t))+\Phi(\hat{U}(t))\Phi(\hat{G}(t))=\Phi(\hat{J}(t)). \tag{13}$$

Then, we can get:

$$D(t)U(t)+U(t)G(t)=J(t), \tag{14}$$

where

$$\begin{aligned}
D(t)&=\begin{bmatrix} D(t)_0-D(t)_3 i & -D(t)_2-D(t)_1 i \\ D(t)_2-D(t)_1 i & D(t)_0+D(t)_3 i \end{bmatrix}\in\mathbb{C}^{4\times 4},\\
U(t)&=\begin{bmatrix} U(t)_0-U(t)_3 i & -U(t)_2-U(t)_1 i \\ U(t)_2-U(t)_1 i & U(t)_0+U(t)_3 i \end{bmatrix}\in\mathbb{C}^{4\times 4},\\
G(t)&=\begin{bmatrix} G(t)_0-G(t)_3 i & G(t)_2-G(t)_1 i \\ G(t)_2-G(t)_1 i & G(t)_0+G(t)_3 i \end{bmatrix}\in\mathbb{C}^{4\times 4},\\
J(t)&=\begin{bmatrix} J(t)_0-J(t)_3 i & J(t)_2-J(t)_1 i \\ J(t)_2-J(t)_1 i & J(t)_0+J(t)_3 i \end{bmatrix}\in\mathbb{C}^{4\times 4}.
\end{aligned} \tag{15}$$

So far, if we want to solve the DSQME problem, we can solve DSCME (14). The following model derivation and theoretical analysis will be carried out based on equation (14).

III. ZNN MODEL

In this section, the related ZNN models will be introduced in detail. This section is mainly divided into three small chapters for description.

A. The Existing ZNN Design Formula

Define an error function based on DSCME (14), which is obtained in the previous study:

$$L(t)=D(t)U(t)+U(t)G(t)-J(t), \tag{16}$$

where $D(t),G(t)$, $J(t)$ and $U(t)\in\mathbb{C}^{4\times 4}$. In terms of [31], an evolutionary design formula is utilized:

$$\dot{L}(t)=-\gamma\Psi(\nu_1 L(t)+\nu_2 L^{\frac{\varrho}{\sigma}}(t)), \tag{17}$$

where $\Psi(\cdot)$ represents monotonically increasing odd activation function that activates each element in $L(t)$; γ, ν_1 and ν_2 are positive real numbers; ϱ and σ are positive integers. Some classic monotonically increasing odd functions are listed as below.

- The linear activation function:

$$\psi(x)=x. \tag{18}$$

- The power-sigmoid activation function:

$$\psi(x)=\begin{cases}x^{\alpha}, & \text{if } |x|\geq 1,\\ \frac{1+\exp(-\eta)}{1-\exp(-\eta)}\cdot\frac{1-\exp(-\eta x)}{1+\exp(-\eta x)}, & \text{otherwise},\end{cases} \tag{19}$$

where $\eta>2$ and $\alpha\geq 3$.

- The sign-bi-power activation function:

$$\psi(x)=\frac{\text{sgn}^{\epsilon}(x)}{2}+\frac{\text{sgn}^{1/\epsilon}(x)}{2}, \tag{20}$$

with design parameter $\epsilon\in(0,1)$ and $\text{sgn}^{\epsilon}(\cdot)$ defined as

$$\text{sgn}^{\epsilon}(x)=\begin{cases}|x|^{\epsilon}, & \text{if} \quad x>0,\\ 0, & \text{if} \quad x=0,\\ -|x|^{\epsilon}, & \text{if} \quad x<0.\end{cases} \tag{21}$$

Remark 1. *The power-sigmoid activation function not only guarantees the superior convergence when the absolute value of the error is greater than 1, but also guarantees the exponential convergence when the absolute value of the error is less than 1. Compared with the linear activation function, the power-sigmoid activation function is more flexible and autonomous. Due to the switching term sgn($\cdot$) in the sign-bi-power activation function, it will inevitably lead to the chattering of the neural network. Thus the power-sigmoid activation function without switching term sgn($\cdot$) is more stable than the sign-bi-power activation function.*

B. The New ZNN Design Formula

Adaptive thinking has been deeply embedded in neural networks [32], and varying parameters are one of them. Moreover, in order to get better convergence, add varying parameters on the basis of design formula (17) to get a novel design formula:

$$\dot{L}(t)=-\gamma\exp(||L(t)||)\varPsi(\nu_1 L(t)+\nu_2 L^{\frac{\varrho}{\sigma}}(t)), \tag{22}$$

where $||L(t)||=\sqrt{\sum_{s=1}^{4}\sum_{w=1}^{4}|l_{\text{sw}}(t)|^2}$ denotes Frobenius norm of modulus and $|l_{\text{sw}}(t)|$ represents the modulus of $l_{\text{sw}}(t)$. When the error is relatively large, we hope that the parameter is relatively large to increase the step size of the ODE solver to achieve the purpose of rapid convergence of the state solution. However, a larger step size will lead to the consumption of computing resources and the increase in computing time. When the error is small, the situation is opposite to the above. Therefore, we obtain varying parameter $\exp(||L(t)||)$ to ensure the convergence effect while avoiding the unnecessary waste of resources.

C. Two DVPZNN Models

Next, we get the derivation of $L(t)$ based on equation (16):

$$\dot{L}(t)=D(t)\dot{U}(t)+\dot{D}(t)U(t)+\dot{U}(t)G(t)+U(t)\dot{G}(t)-\dot{J}(t). \tag{23}$$

Algorithm 1 Implementation of the DVPZNN model (25)

Input: $D(t)\in\mathbb{C}^{4\times 4}$, $G(t)\in\mathbb{C}^{4\times 4}$, $J(t)\in\mathbb{C}^{4\times 4}$, initial value $U(0)\in\mathbb{C}^{4\times 4}$, parameter $\gamma\in\mathbb{R}^{+}$, total time $T\in\mathbb{R}^{+}$, and step size $\tau\in\mathbb{R}^{+}$.
Output: State solutions $U_{\text{s}}\in\mathbb{C}^{\frac{T}{\tau}\times 16}$.

Define $U_{\text{s}}\in\mathbb{C}^{\frac{T}{\tau}\times 16}$ and $M\in\mathbb{C}^{16\times 16}$.
$\boldsymbol{u}_0$=reshape($U(0)$,16,1) ,$U_{\text{s}}(1,:)=\boldsymbol{u}_0^{\text{T}}$.
Take the time derivatives of $D(t)$, $G(t)$ and $J(t)$ to get $\dot{D}(t)$, $\dot{G}(t)$ and $\dot{J}(t)$.
for $\eta=0:\tau:T$ **do**
  Use η to update $D(t)$, $G(t)$, $J(t)$, $\dot{D}(t)$, $\dot{G}(t)$ and $\dot{J}(t)$ to get $D(\eta)$, $G(\eta)$, $J(\eta)$, $\dot{D}(\eta)$, $\dot{G}(\eta)$ and $\dot{J}(\eta)$.
  $L(\eta)=D(\eta)U(\eta)-U(\eta)G(\eta)-J(\eta)$.
  $v=\exp(||L(\eta)||)$.
  $\boldsymbol{k}$=reshape($J(\eta)$,16,1), $\dot{\boldsymbol{k}}$=reshape($\dot{J}(\eta)$,16,1).
  $\boldsymbol{l}$=reshape($L(\eta)$,16,1).
  $M=I\otimes D(\eta)+G^{\text{T}}(\eta)\otimes I$.
  Find the differential equation $M\boldsymbol{u}_{\eta+1}=-\gamma v\varPsi(\boldsymbol{l})-(I\otimes\dot{D}(\eta)+\dot{G}^{\text{T}}(\eta)\otimes I)\boldsymbol{u}_{\eta}-\dot{\boldsymbol{k}}$ to get $\boldsymbol{u}_{\eta+1}$.
  $U_{\text{s}}(\eta+2,:)=\boldsymbol{u}_{\eta+1}^{\text{T}}$.
end
return U_{s}.

Combining equations (22) and (23), the following result is obtained:

$$\begin{aligned}&D(t)\dot{U}(t)+\dot{U}(t)G(t)\\ =&-\gamma\exp(||L(t)||)\varPsi(\nu_1 L(t)+\nu_2 L^{\frac{\varrho}{\sigma}}(t))\\ &-\dot{D}(t)U(t)-U(t)\dot{G}(t)+\dot{J}(t).\end{aligned} \tag{24}$$

Substitute equation (16) into equation (24) and the DVPZNN model can be acquired as below:

$$\begin{aligned}&D(t)\dot{U}(t)+\dot{U}(t)G(t)\\ =&-\gamma\exp(||L(t)||)\varPsi(\nu_1(D(t)U(t)+U(t)G(t)-J(t))\\ &+\nu_2(D(t)U(t)+U(t)G(t)-J(t))^{\frac{\varrho}{\sigma}})\\ &-\dot{D}(t)U(t)-U(t)\dot{G}(t)+\dot{J}(t).\end{aligned} \tag{25}$$

Its implementation is shown in Algorithm 1. When linear activation function is applied to the DVPZNN model, the LDVPZNN model can be obtained, and when power-sigmoid activation function is applied, the PSDVPZNN model can be gained.

Remark 2. *In DVPZNN model* (25)*, there is a power function: $L^{\frac{\varrho}{\sigma}}(t)$. Since the power function is to exponentiate each element in $L(t)$ and the elements in $L(t)$ may have negative numbers, we must ensure that each exponentiation is meaningful when designing $\frac{\varrho}{\sigma}$, that is, $\frac{\varrho}{\sigma}$ is an odd root.*

IV. Stability and Convergence Analysis

For the purpose of better highlighting the advantages of the proposed models, the corresponding analysis will be made from two aspects of global stability and finite-time convergence.

A. Global Stability Analysis

Theorem 2. *For DSCME* (14)*, if the DVPZNN model is used to solve this problem, given an initial state $U(0)$ arbitrarily, the state solution $U(t)$ will globally converge to the exact solution and remain stable.*

Proof: First, a Lyapunov function candidate is defined: $V(t) = ||L(t)||^2/2$. Obviously, $V(t) \geq 0$. To prove the above theorem, we require deriving the Lyapunov function candidate:

$$\begin{aligned} \dot{V}(t) &= \frac{1}{2}\mathrm{Tr}\left(\dot{L}^{\mathrm{T}}(t)\,L(t) + L^{\mathrm{T}}(t)\,\dot{L}(t)\right) \\ &= -\frac{1}{2}\gamma\exp\left(||L(t)||_{\mathrm{F}}\right)\mathrm{Tr}\left(\varXi\right), \end{aligned} \tag{26}$$

where $\mathrm{Tr}(A) = \sum_{s=1}^{n} a_{\mathrm{ss}}$ denotes the trace of matrix $A \in \mathbb{R}^{n\times n}$; $\varXi = \varPsi\left(\nu_1 L^{\mathrm{T}}(t) + \nu_2\left(L^{\mathrm{T}}\right)^{\frac{\varrho}{\sigma}}(t)\right)L(t) + \varPsi\left(\nu_1 L(t) + \nu_2 (L)^{\frac{\varrho}{\sigma}}(t)\right)L^{\mathrm{T}}(t)$. The activation function of DVPZNN model (17) introduced above is monotonically increasing, so the use of the activation function does not affect the symbols of the elements in $L(t)$ or $\dot{L}(t)$. So $\mathrm{Tr}\left(L(t)L^{\mathrm{T}}(t)\right) \geq 0$ and $\mathrm{Tr}\left(L^{\mathrm{T}}(t)L(t)\right) \geq 0$ always hold.

For $\left(L^{\mathrm{T}}\right)^{\frac{\varrho}{\sigma}}(t)\,L(t)$, as mentioned in Remark 2, $\frac{\varrho}{\sigma}$ is an odd power function, and it will not change the sign of the elements in $L(t)$, so $\left(L^{\mathrm{T}}\right)^{\frac{\varrho}{\sigma}}(t)\,L(t) \geq 0$ and $(L)^{\frac{\varrho}{\sigma}}(t)\,L^{\mathrm{T}}(t) \geq 0$ are true.

As pointed out in the previous section, ν_1 and ν_2 are positive integers, so $\varXi \geq 0$ holds. Besides, $\gamma > 0$ and $\exp(||E(t)||_{\mathrm{F}}) \geq 0$, so $\dot{V}(t) \leq 0$. In summary, the DVPZNN model satisfies the Lyapunov stability theorem [33], so its state solution can globally converge to exact solution and remain stable when solving the DSQME problem. Thus, the proof is completed. ■

B. Finite-time Convergence Analysis

Theorem 3. *Assume that $l^+(t)$ and $l^-(t)$ are the largest and smallest elements in error matrix $L(t)$, respectively. If the LDVPZNN model is utilized to solve the DSQME with random initial value, the exact solution of the DSQME will be obtained by the LDVPZNN model within finite-time:*

$$\frac{\ln\left(\frac{\nu_2}{\min(l_{\mathrm{sw}}(0))\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)} < t_{\mathrm{f}} < \frac{\ln\left(\frac{\nu_2}{\max(l_{\mathrm{sw}})(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1}. \tag{27}$$

Proof: When the LDVPZNN model is used to solve the DSQME, there is the following formula:

$$\dot{L}(t) = -\gamma\exp(||L(t)||)(\nu_1 L(t) + \nu_2 L^{\frac{\varrho}{\sigma}}(t)). \tag{28}$$

Supposing $0 < ||L(t)|| \leq \delta$, then we have:

$$\begin{cases} \dot{L}(t) \geq -\gamma\exp(\delta)\varPsi(\nu_1 L(t) + \nu_2 L^{\frac{\varrho}{\sigma}}(t)), \\ \dot{L}(t) < -\gamma\exp(0)\varPsi(\nu_1 L(t) + \nu_2 L^{\frac{\varrho}{\sigma}}(t)). \end{cases} \tag{29}$$

Solve these two differential inequalities separately.

1) $\dot{L}(t) \geq -\gamma\exp(\delta)\varPsi(\nu_1 L(t) + \nu_2 L^{\frac{\varrho}{\sigma}}(t))$.
Use the Hadamard product to multiply $L^{-\frac{\varrho}{\sigma}}(t)$ on both sides, and

$$L^{-\frac{\varrho}{\sigma}}(t)\circ\dot{L}^{\frac{\sigma-\varrho}{\sigma}}(t) \geq -\gamma\exp(\delta)\varPsi(\nu_1 L(t) + \nu_2\mathscr{Q}_{\mathrm{I}}). \tag{30}$$

where $\mathscr{Q}_{\mathrm{I}}$ is a matrix with all elements being 1. Let $Y(t) = L^{\frac{\sigma-\varrho}{\sigma}}(t)$, and we get

$$\frac{\mathrm{d}Y(t)}{\mathrm{d}t} + \frac{\sigma-\varrho}{\sigma}\gamma\nu_1\exp(\delta)Y(t) \geq \frac{\varrho-\sigma}{\sigma}\gamma\nu_2\exp(\delta)\mathscr{Q}_{\mathrm{I}}. \tag{31}$$

Since the matrix $Y(t)$ is composed by elements $y_{\mathrm{sw}}(t)$, by the first order differential theory, we can get:

$$y_{\mathrm{sw}}(t) \geq \left(y_{\mathrm{sw}}(0) + \frac{\nu_2}{\nu_1}\right)\exp\left(\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)t\right) - \frac{\nu_2}{\nu_1}. \tag{32}$$

Obviously, $Y(t_{\mathrm{f}}) = L(t_{\mathrm{f}}) = 0$. In this case, the time t_{f_1} for the LDVPZNN model to attain exact solution is:

$$t_{\mathrm{f}_1} \geq \frac{\ln\left(\frac{\nu_2}{l_{\mathrm{sw}}(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)}. \tag{33}$$

2) $\dot{L}(t) < -\gamma\exp(0)\varPsi(\nu_1 L(t) + \nu_2 L^{\frac{\varrho}{\sigma}}(t))$.
Similar to the above process, we can get:

$$y_{\mathrm{sw}}(t) < \left(y_{\mathrm{sw}}(0) + \frac{\nu_2}{\nu_1}\right)\exp\left(\frac{\varrho-\sigma}{\sigma}\gamma\nu_1 t\right) - \frac{\nu_2}{\nu_1}, \tag{34}$$

$$t_{\mathrm{f}_2} < \frac{\ln\left(\frac{\nu_2}{l_{\mathrm{sw}}(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1}. \tag{35}$$

The $l_{\mathrm{sw}}(0)$ in t_{f_1} and t_{f_2} represents the swth entry of $L(0)$.

Let $l_{\max}(t) = \max\left(l_{\mathrm{sw}}(t)\right)$ and $l_{\min}(t) = \min\left(l_{\mathrm{sw}}(t)\right)$, then we have

$$t_{\mathrm{f}_1} \geq \frac{\ln\left(\frac{\nu_2}{l_{\mathrm{sw}}(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)} > \frac{\ln\left(\frac{\nu_2}{\min(l_{\mathrm{sw}}(0))\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)}, \tag{36}$$

$$t_{\mathrm{f}_2} < \frac{\ln\left(\frac{\nu_2}{l_{\mathrm{sw}}(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1} < \frac{\ln\left(\frac{\nu_2}{\max(l_{\mathrm{sw}})(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1}. \tag{37}$$

In summary, let t_{f} be the convergence time related to the initial state. t_{f} is between t_{f_1} and t_{f_2}:

$$t_{\mathrm{f}_1} < t_{\mathrm{f}} < t_{\mathrm{f}_2}, \tag{38}$$

$$\frac{\ln\left(\frac{\nu_2}{\min(l_{\mathrm{sw}}(0))\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1\exp(\delta)} < t_f < \frac{\ln\left(\frac{\nu_2}{\max(l_{\mathrm{sw}})(0)\nu_1+\nu_2}\right)}{\frac{\varrho-\sigma}{\sigma}\gamma\nu_1}. \tag{39}$$

■

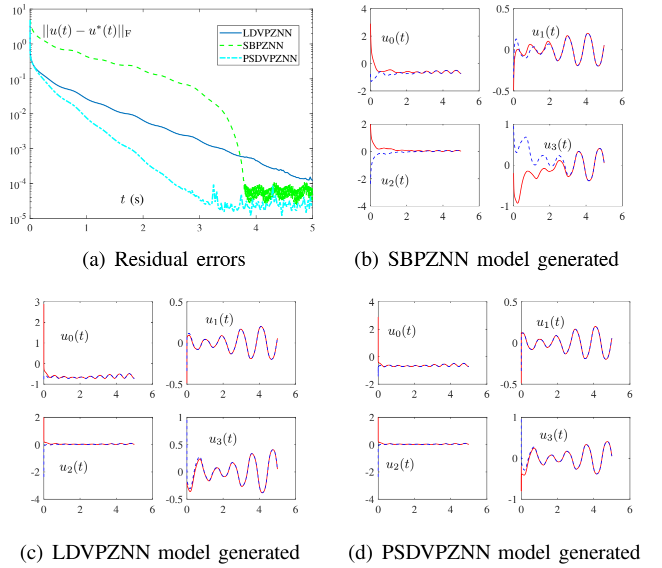
Theorem 4. *If the PSDVPZNN model is utilized to solve the DSQME with random initial value, the PSDVPZNN model will have better convergence performance in solving the DSQME than using the LDVPZNN model.*

Proof: Since the activation function in the DVPZNN model activates each element in $L(t)$, we can get ZNN design formula (22) as the following element form.

$$\dot{l}_{\mathrm{sw}}(t) = -\gamma\exp(||L(t)||)\psi\left(\nu_1 l_{\mathrm{sw}}(t) + \nu_2 {l_{\mathrm{sw}}}^{\frac{\varrho}{\sigma}}(t)\right). \tag{40}$$

Define a Lyapunov function candidate based on the above formula:

$$V\left(l_{\mathrm{sw}}(t), t\right) = \frac{1}{2}l_{\mathrm{sw}}^2(t). \tag{41}$$



(a) Residual errors (b) SBPZNN model generated

(c) LDVPZNN model generated (d) PSDVPZNN model generated

Fig. 1. Error norm comparison chart and trajectories of real and imaginary parts of $\hat{u}(t)$ generated by the LDVPZNN, SBPZNN and PSDVPZNN models in Example I, where x-axis denotes t (s).

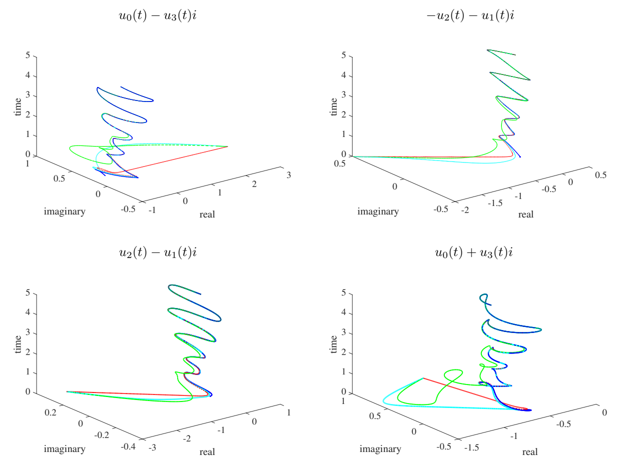


Fig. 2. Trajectories of the complex representation solution of the one-dimensional dynamic Sylvester quaternion equation generated by the LDVPZNN, SBPZNN and PSDVPZNN models in Example I.

Next, we use $\dot{V}_{\text{linear}}(l_{\text{sw}}(t),t)$ and $\dot{V}_{\text{p-s}}(l_{\text{sw}}(t),t)$ to represent the Lyapunov function candidate defined based on the LDVPZNN model and the PSDVPZNN model, respectively. Then, take the derivation of time t of the Lyapunov function candidate:

$$\dot{V}(l_{\text{sw}}(t),t) = \dot{l}_{\text{sw}}(t) l_{\text{sw}}(t). \tag{42}$$

When the LDVPZNN model is used, the above equation becomes:

$$\dot{V}_{\text{linear}}(l_{\text{sw}}(t),t) = -\gamma \exp(||L(t)||) l_{\text{sw}}(t) \left(\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)\right). \tag{43}$$

When the PSDVPZNN model is used, we need to discuss it separately because power-sigmoid activation function is a piecewise function.

CASE I: $|\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)| \geq 1$. In this case, equation (40) becomes:

$$\dot{l}_{\text{sw}}(t) = -\gamma \exp(||L(t)||) \left(\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)\right)^{\alpha}, \tag{44}$$

where $\alpha \geq 3$ and is an odd integer. So take the derivation of the Lyapunov function candidate:

$$\begin{aligned} &\dot{V}_{\text{p-s}}(l_{\text{sw}}(t),t) \\ &\quad = -\gamma \exp(||L(t)||) l_{\text{sw}}(t) \left(\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)\right)^{\alpha} \\ &\quad \begin{cases} = \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t), & \text{if } \ \nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t) = 1; \\ < \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t), & \text{if } \ \nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t) > 1; \\ = \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t), & \text{if } \ \nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t) = -1; \\ < \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t), & \text{if } \ \nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t) < -1; \end{cases} \end{aligned} \tag{45}$$

CASE II: $|\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)| < 1$. In this case, it would be a bit inconvenient to directly bring the activation function into (40). So discuss the sigmoid function firstly:

$$\Psi(u) = \frac{1+\exp(-\eta)}{1-\exp(-\eta)} \cdot \frac{1-\exp(-\eta u)}{1+\exp(-\eta u)} \tag{46}$$

where $u \in (-1,1)$ due to $|\nu_1 l_{\text{sw}}(t) + \nu_2 l_{\text{sw}}{}^{\frac{\varrho}{\sigma}}(t)| < 1$.

Let

$$F(u) = u - \frac{1+\exp(-\eta)}{1-\exp(-\eta)} \cdot \frac{1-\exp(-\eta u)}{1+\exp(-\eta u)} \tag{47}$$

and take the derivation of u:

$$\dot{F}(u) = 1 - \frac{1+\exp(-\eta)}{1-\exp(-\eta)} \cdot \frac{2\eta \exp(-\eta u)}{(1+\exp(-\eta u))^2}. \tag{48}$$

Then let

$$G(u) = \frac{2\eta \exp(-\eta u)}{(1+\exp(-\eta u))^2} \tag{49}$$

and take the derivation of u:

$$\dot{G}(u) = \frac{2\eta^2 \exp(-\eta u)(1-\exp(-2\eta u))}{(1+\exp(-\eta u))^4}. \tag{50}$$

From the above formula, we can obtain that $\dot{G}(u) > 0$ when $u > 0$, $\dot{G}(u) < 0$ when $u < 0$. This means that $G(u)$ takes the minimum value at $u = 0$. Since

$$\begin{aligned} \dot{F}_{\text{max}}(u) &= 1 - G_{\text{min}}(u) \\ &= 1 - G(0) \\ &= 1 - \frac{\eta}{2} \cdot \frac{1+\exp(-\eta)}{1-\exp(-\eta)} < 0, \end{aligned} \tag{51}$$

we can conclude that $F(u)$ is monotonically decreasing. Then when $u \in (-1,1)$,

$$\begin{aligned} F(u) < F(-1) &= -1 - \frac{1+\exp(-\eta)}{1-\exp(-\eta)} \cdot \frac{1-\exp(\eta)}{1+\exp(\eta)} \\ &= -1 - \frac{-\exp(\eta)+\exp(-\eta)}{-\exp(-\eta)+\exp(-\eta)} \\ &= 0. \end{aligned} \tag{52}$$

After the above analysis, we can summarize that when $u \in (-1,1)$, the power-sigmoid function is larger than the linear function. Obviously, in such case,

$$\dot{V}_{\text{p-s}}(l_{\text{sw}}(t),t) < \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t). \tag{53}$$

Combining the above two cases, we can get $\dot{V}_{\text{p-s}}(l_{\text{sw}}(t),t) < \dot{V}_{\text{linear}}(l_{\text{sw}}(t),t)$. In the sense of Lyapunov

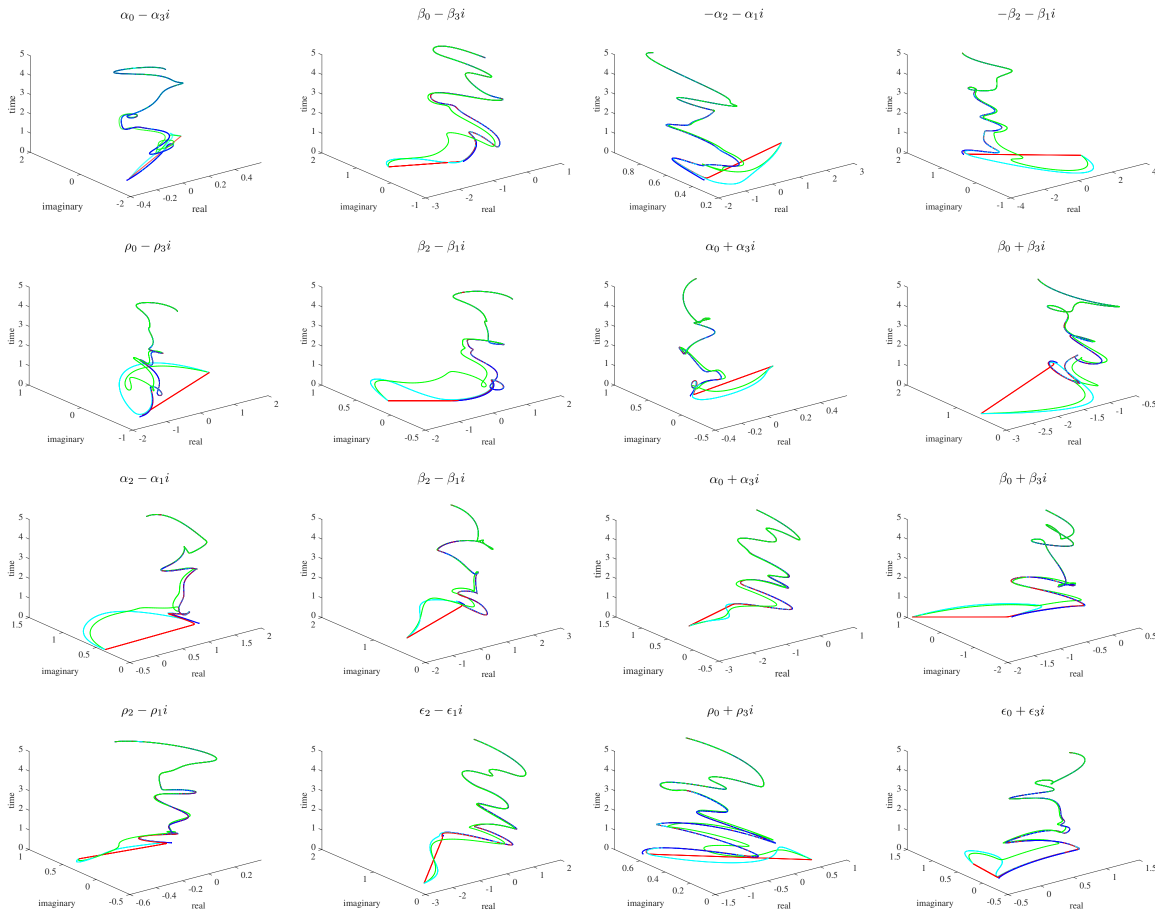


Fig. 3. Trajectories of the complex representation solution of the two-dimensional DSQME generated by the LDVPZNN, SBPZNN and PSDVPZNN models.

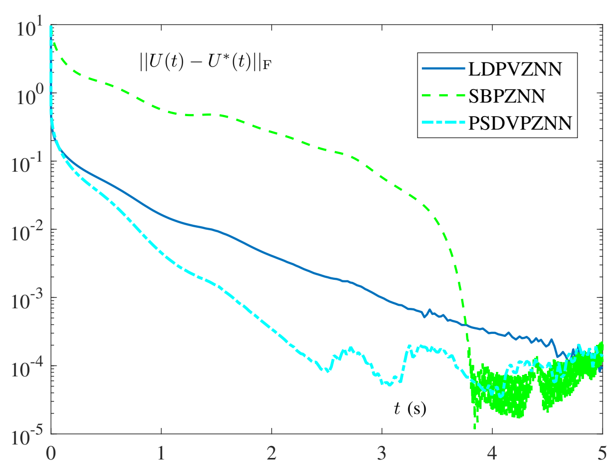


Fig. 4. Error norm comparison charts in Example II.

stability theorem [33], the PSDVPZNN model will have better convergence performance in solving the DSQME than using the LDVPZNN model. Thus the proof is completed. ■

V. Experimental simulation

In this part, three simulative experiments are afforded to prove the stability and convergence performance of the proposed DVPZNN models, and the sign-bi-power function activated original ZNN (SBPZNN) model is used for comparison. The detailed SBPZNN model is omitted here. The experiments include three experimental examples: the dynamic Sylvester quaternion equation with the matrix dimension being one; the dynamic Sylvester quaternion equation with the matrix dimension being 2 × 2; and the static Sylvester quaternion matrix equation with the matrix dimension being 2 × 2.

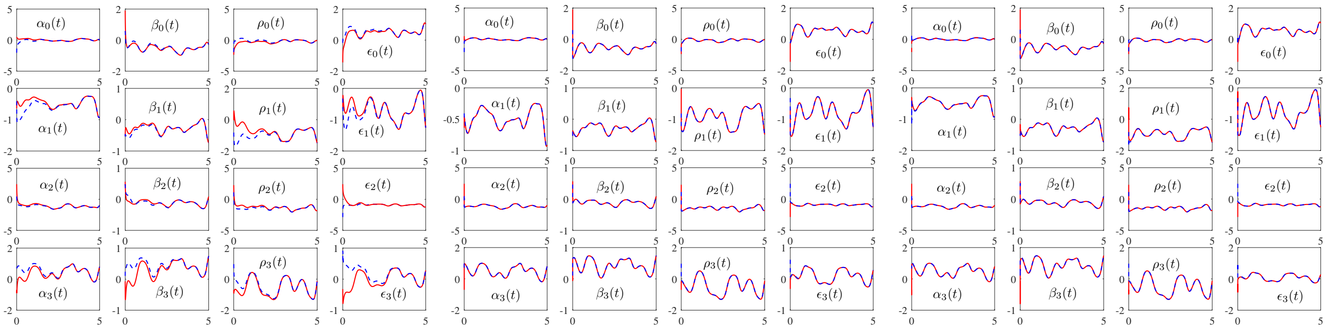
In order to ensure the consistency of the parameters in the experiments, the parameters of all experiments are set as: $\eta=4$, α=3, ϵ=0.4, ν_1=ν_2=1, ϱ=1, and σ=3. In each example, the initial values of the state matrices are randomly generated, the real part of each element ranges from -3 to 3, and the imaginary part ranges from 0 to 1.

A. *Example I: One-dimensional dynamic Sylvester quaternion equation*

Considering that the known quaternions in formula (12) are all one-dimensional quaternions, the following example is given:

$$\hat{d}(t)=\cos(6t)+4i+6k,$$
$$\hat{g}(t)=\cos(6t)-\sin(6t)j+3k,$$
$$\hat{j}(t)=\sin(5t)+3i-2k.$$

Through the values of the above three items, the value of $\hat{u}(t)$ can be calculated by using the ZNN models to calculate



(a) SBPZNN model generated (b) LDVPZNN model generated (c) PSDVPZNN model generated

Fig. 5. Trajectories of real and imaginary parts of $\hat{U}(t)$ generated by the LDVPZNN, SBPZNN and PSDVPZNN model in Example II, where x-axis denotes t (s).

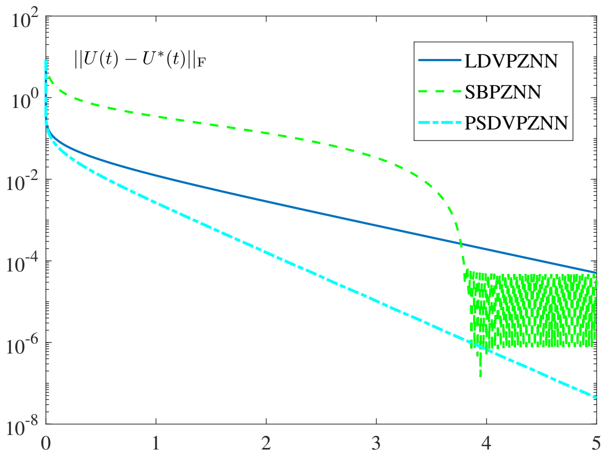


Fig. 6. Error norm comparison charts in Example III.

DSCME (14). Here, we give the complex representation of $\hat{u}$:

$$\Phi(\hat{u}(t)) = u(t) = \begin{bmatrix} u_0(t) - u_3(t)i & -u_2(t) - u_1(t)i \\ u_2(t) - u_1(t)i & u_0(t) + u_3(t)i \end{bmatrix}. \tag{54}$$

In order to know the convergence of the LDVPZNN and PSDVPZNN models more significantly than the SBPZNN model, the error graphs of three models are shown in Fig. 1(a). From Fig. 1(a), we can learn: the LDVPZNN and PSDVPZNN models tend to converge at a faster rate and are relatively stable, but the SBPZNN model tends to stabilize at a slower rate and has greater volatility. Fig. 1(b), Fig. 1(c) and Fig. 1(d) respectively show the trajectories of $u_0(t)$ (real part), $u_1(t)$, $u_2(t)$ and $u_3(t)$ (imaginary parts) in the quaternion solution $u(t)$ produced by three models. Two trajectories in each subgraph respectively represent two $u_0(t)$, $u_1(t)$, $u_2(t)$ and $u_3(t)$ in the complex representation of $u(t)$ (54). When two trajectories coincide, the ZNN models solve the exact solution of the equation. In other words, the state solution generated by the ZNN models converges to the exact solution. At the same time, the trajectories of each element of $U(t)$ in the complex representation are shown in Fig. 2 where the blue trajectory is the exact solution; the red trajectory is the solution produced by the LDVPZNN model; the green trajectory is the solution produced by the SBPZNN model; and the cyan trajectory is the solution produced by the PSDVPZNN model.

B. Example II: *Two-dimensional dynamic Sylvester quaternion matrix equation*

When the known matrices in formula (12) are all two-dimensional quaternion matrices, the following example is considered:

$$\hat{D}(t) = \begin{bmatrix} \cos(5t) + 2i - 3j - 12k & \sin(4t) - 3i - 3j + 5k \\ \sin(-3t) - 4i - 6j + 3k & \cos(2t) + 4i + 5j + 2k \end{bmatrix},$$
$$\hat{G}(t) = \begin{bmatrix} \cos(6t) - 3i + 2j + 2k & \cos(3t) + i + 2j + 3k \\ \sin(2t) + 4i - 3j - 5k & \sin(-5t) + 3i - 2j - 5k \end{bmatrix},$$
$$\hat{J}(t) = \begin{bmatrix} \sin(-3t) - 2i + j - 3k & \cos(-6t) + 3i - 2j + 4k \\ \cos(4t) - 3i - 2j + 5k & \sin(2t) + 4i + 5j - 3k \end{bmatrix}.$$

Similarly, we use the complex representation of the quaternion matrix to find $\hat{U}(t)$ of which complex representation is

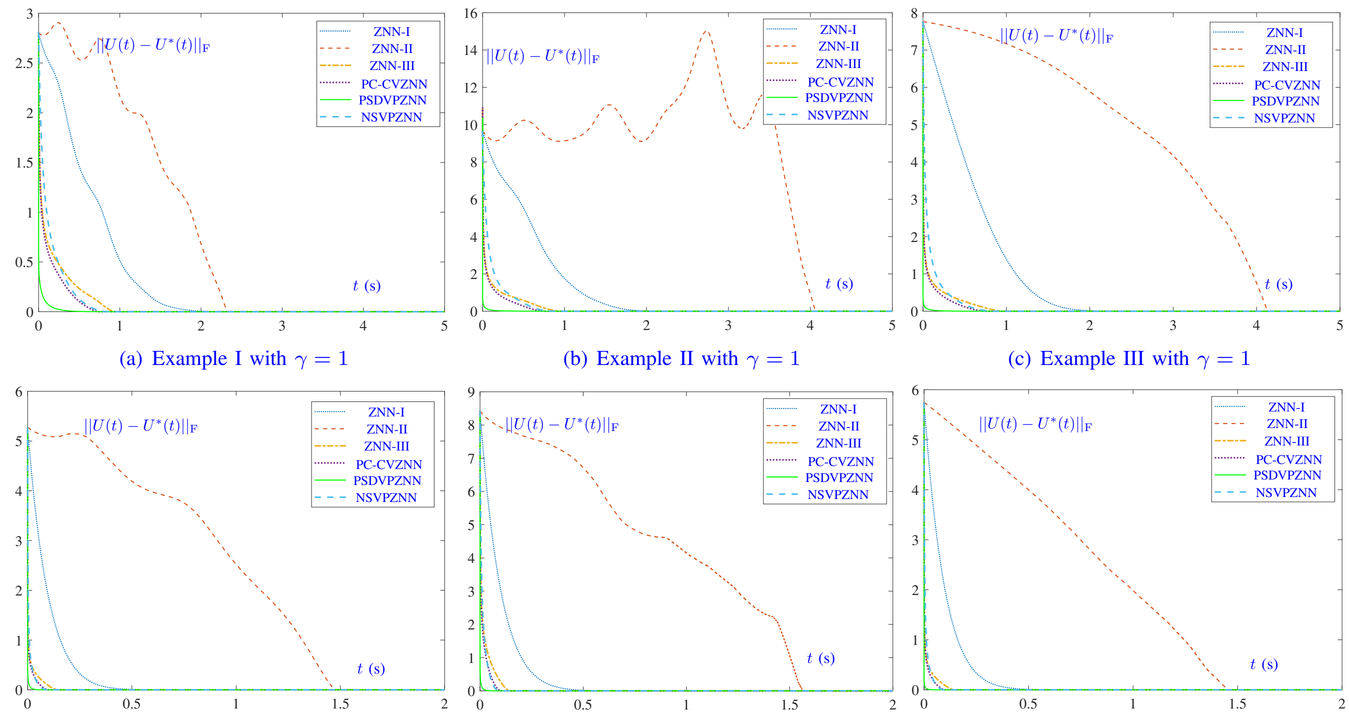
$$\begin{aligned} &\Phi(\hat{U}(t)) = U_0(t) + U_1(t)i + U_2(t)j + U_3(t)k \\ &= \begin{bmatrix} U_0(t) - U_3(t)i & -U_2(t) - U_1(t)i \\ U_2(t) - U_1(t)i & U_0(t) + U_3(t)i \end{bmatrix} \\ &= \begin{bmatrix} \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix} - \begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i & -\begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix} - \begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i \\ \begin{bmatrix} \alpha_2 & \beta_2 \\ \eta_2 & \rho_2 \end{bmatrix} - \begin{bmatrix} \alpha_1 & \beta_1 \\ \eta_1 & \rho_1 \end{bmatrix} i & \begin{bmatrix} \alpha_0 & \beta_0 \\ \eta_0 & \rho_0 \end{bmatrix} + \begin{bmatrix} \alpha_3 & \beta_3 \\ \eta_3 & \rho_3 \end{bmatrix} i \end{bmatrix} \\ &= \begin{bmatrix} \alpha_0 - \alpha_3 i & \beta_0 - \beta_3 i & -\alpha_2 - \alpha_1 i & -\beta_2 - \beta_1 i \\ \eta_0 - \eta_3 i & \rho_0 - \rho_3 i & -\eta_2 - \eta_1 i & -\rho_2 - \rho_1 i \\ \alpha_2 - \alpha_1 i & \beta_2 - \beta_1 i & \alpha_0 + \alpha_3 i & \beta_0 + \beta_3 i \\ \eta_2 - \eta_1 i & \rho_2 - \rho_1 i & \eta_0 + \eta_3 i & \rho_0 + \rho_3 i \end{bmatrix}, \end{aligned} \tag{55}$$

where $\alpha_k, \beta_k, \eta_k, \rho_k \in U_k(t), k = 0, 1, 2, 3$.

Fig. 3 and Fig. 4 respectively show the trajectories of each element of $\hat{U}(t)$ in the complex representation and the error change of the LDVPZNN, SBPZNN and PSDVPZNN models in solving the DSCME. The meanings of the trajectory colors in Fig. 3 are the same as those in Fig. 2. Fig. 5(a), Fig. 5(b) and Fig. 5(c) show the trajectories of the real and imaginary parts in the quaternion matrix $\hat{U}(t)$ generated by the ZNN models. When two trajectories coincide, $\hat{U}(t)$ converges to the exact solution. Each column of three subgraphs represents

TABLE I
CONVERGENCE TIMES OF THE ZNN-I, ZNN-II, ZNN-III [23], PC-CVZNN [34], NSVPZNN [35] AND PSDVPZNN MODELS SOLVING EXAMPLES I, II, AND III WHEN $\gamma = 1$ AND $\gamma = 10$.

ZNN models	Design formulas	Convergence time						Speed
	Parameters	$\gamma = 1$			$\gamma = 10$			
	Examples	I	II	III	I	II	III	
ZNN-I [23]	$\dot{L}(t) = -\gamma \exp(t) L(t)$	2.43s	2.01s	2.02s	1.50s	0.52s	0.51s	slow
ZNN-II [23]	$\dot{L}(t) = -\gamma \exp(t) \varPsi_{\mathrm{s}}(L(t))$	2.22s	4.03s	4.10s	0.52s	1.56s	1.48s	very slow
ZNN-III [23]	$\dot{L}(t) = -\gamma \exp(t) \varPsi_{\mathrm{li}}(L(t))$	0.95s	0.90s	0.98s	0.16s	0.14s	0.14s	fast
PC-CVZNN [34]	$\dot{L}(t) = -\gamma (p^t + 2pt + p) \varPsi_1(L(t))$	0.71s	0.72s	0.72s	0.11s	0.11s	0.11s	fast
NSVPZNN [35]	$\dot{L}(t) = -\gamma (t^2 + m) \varPsi_2(L(t))$	0.84s	0.86s	0.86s	0.13s	0.12s	0.13s	fast
PSDVPZNN (this paper)	$\dot{L}(t) = -\gamma (\lVert L(t) \rVert_{\mathrm{F}}) \varPsi_{\mathrm{ps}}(L(t))$	0.53s	0.38s	0.37s	0.07s	0.07s	0.06s	very fast



(a) Example I with $\gamma = 1$ (b) Example II with $\gamma = 1$ (c) Example III with $\gamma = 1$

(d) Example I with $\gamma = 10$ (e) Example II with $\gamma = 10$ (f) Example III with $\gamma = 10$

Fig. 7. Comparisons of error trajectories generated by the ZNN-I, ZNN-II, ZNN-III [23], PC-CVZNN [34], NSVPZNN [35] and PSDVPZNN models when $\gamma = 1$ and $\gamma = 10$.

elements in matrix $U_k(t)$. From these two figures, obviously, the PSDVPZNN model has the best convergence, followed by the LDVPZNN model, and the SBPZNN model has the worst convergence.

C. Example III: Two-dimensional static Sylvester quaternion matrix equation

Given that the known matrices are two-dimensional static quaternion matrices, the following example is given:

$$\hat{D}(t) = \begin{bmatrix} 6 + 3i - 2j + 7k & 2 + i + j - k \\ -2 - 3i - 5j - k & 3 + 5i + j - 3k \end{bmatrix},$$
$$\hat{G}(t) = \begin{bmatrix} 7 - 2i - 4j + 5k & 2 - 2i + 3j + k \\ 4 + i + 3j + k & 7 - 3i - 5j - 5k \end{bmatrix},$$
$$\hat{J}(t) = \begin{bmatrix} 5 - 3i - j + 2k & 3 - 2i + 2j + 3k \\ 1 + 2i + 3j + 4k & 5 + 4i - 9j - 2k \end{bmatrix},$$

of which complex representation is (55). Through this static example, we can not only verify the superiority of our proposed LDVPZNN model and PSDVPZNN model compared with the SBPZNN model, but also verify that the ZNN models can solve both time-varying problems and time-invariant problems. In addition, the error graph of three models is shown in Fig. 6. The graph shows the effectiveness of the proposed model for static problems, and has better performance.

D. Comparative experiment

In order to further verify the effectiveness of the PSDVPZNN model proposed in this paper, the ZNN models in [23], [34], [35] are used for comparison. The compared design formulas and convergence times when solving the DSQME problem in Examples I, II and III are shown in TABLE I, where $\varPsi_{\mathrm{s}}, \varPsi_{\mathrm{li}}, \varPsi_1, \varPsi_2$ and $\varPsi_{\mathrm{ps}}$ represent the sign, Li [23], new sign-bi-power [34], new improved [35], and power-sigmoid activation functions, respectively. By using the design formulas in TABLE I and the modeling technique in Section III-C, the corresponding models for solving the DSQME can be obtained.

In the process of comparison, relative parameters are set as: $\eta = 4$, α=3, ϵ=0.4, ν_1=ν_2=1, ϱ=1, σ=3, p=1.2, m=2, and especially $\gamma = 1$ or 10. The experimental results are shown in Fig. 7. It can be seen from Fig. 7 that whether $\gamma = 1$ or $\gamma = 10$, the PSDVPZNN model solves the DSQME problem in Examples I, II and III with the fastest convergence rate. It indicates that the convergence performance of the PSVPZNN model is the best. Similarly, TABLE I also verified this result. From TABLE I, more detailed data shows that compared to other ZNN models, the PSDVPZNN model has the fastest convergence speed in various situations.

VI. Conclusion

In this paper, to solve the DSQME problem, we use the equivalence of quaternion complex representation to convert the DSQME to the DSCME. Then two DVPZNN models (i.e., the LDVPZNN model and the PSDVPZNN model) are designed to further address the DSQME problem. On the basis of the novel varying parameter, the DVPZNN models can adapt to error changes to reduce waste of resources and have better convergence, with the convergence time range being calculated. Finally, we verify that the DVPZNN models can effectively solve the DSCME problem through quaternion complex representation and perform better than the SBPZNN model through three numerical experiments. More importantly, the use of new dynamic varying parameter to make the DVPZNN models have better performance provides a very good idea for the design of varying parameter in the future.

References

[1] W. R. Hamilton, "On a new species of imaginary quantities, connected with the theory of quaternions," *Proc. Roy. Irish Acad.*, vol. 2, no. 1, pp. 424–434, Nov. 1840.

[2] W. L. Chan, H. Choi, and R. Baraniuk, "Quaternion wavelets for image analysis and processing," in *Proc. IEEE Int. Conf. Image Processing*, vol. 5, Singapore, Oct. 2004, pp. 3057–3060.

[3] B. Chen, H. Shu, G. Coatrieux, G. Chen, X. Sun, and J. L. Coatrieux, "Color image analysis by quaternion-type moments," *J. Math. Imag. Vis.*, vol. 51, no. 1, pp. 124–144, Jan. 2015.

[4] R. Kristiansen and P. J. Nicklasson, "Satellite attitude control by quaternion-based backstepping," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 1, pp. 227–232, Jan. 2009.

[5] N. Le Bihan and J. Mars, "Singular value decomposition of quaternion matrices: a new tool for vector-sensor signal processing," *Signal Process.*, vol. 84, no. 7, pp. 1177–1199, Jul. 2004.

[6] M. D. Jiang, Y. Li, and W. Liu, "Properties of a general quaternion-valued gradient operator and its applications to signal processing," *Front. Inform. Technol. Electron. Eng.*, vol. 17, pp. 83–95, Feb. 2016.

[7] Y. Zhang, D. Jiang, and J. Wang, "A recurrent neural network for solving Sylvester equation with time-varying coefficients," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1053–1063, 2002.

[8] B. Kagstrom and L. Westin, "Generalized schur methods with condition estimators for solving the generalized Sylvester equation," *IEEE Trans. Autom. Control*, vol. 34, no. 7, pp. 745–751, 1989.

[9] G. Chen, Y. Song, F. Wang, and C. Zhang, "Semi-supervised multi-label learning by solving a Sylvester equation," in *Proc. SIAM Int. Conf. Data Mining*, 2008, pp. 410–419.

[10] P. Benner, T. Damm, and Y. R. R. Cruz, "Dual pairs of generalized Lyapunov inequalities and balanced truncation of stochastic linear systems," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 782–791, Feb. 2016.

[11] R. Chteoui, A. F. Aljohani, and A. B. Mabrouk, "Lyapunov–Sylvester computational method for numerical solutions of a mixed cubic-superlinear schrödinger system," *Eng. Comput.*, pp. 1–14, 2021.

[12] H. R. Shaker and M. Tahavori, "Control configuration selection for bilinear systems via generalised hankel interaction index array," *Int. J. Control*, vol. 88, no. 1, pp. 30–37, 2015.

[13] A. S. Hodel and P. Misra, "Solution of underdetermined Sylvester equations in sensor array signal processing," *Linear Algebra Appl.*, vol. 249, no. 1-3, pp. 1–14, 1996.

[14] T.-x. Li and E. K.-W. Chu, "Pole assignment for linear and quadratic systems with time-delay in control," *Numer. Linear Algebra Appl.*, vol. 20, no. 2, pp. 291–301, 2013.

[15] C.-C. Tsui, "A complete analytical solution to the equation TA-FT= LC and its applications," *IEEE Trans. Autom. Control*, vol. 32, no. 8, pp. 742–744, 1987.

[16] L. Ding, L. Xiao, K. Zhou, Y. Lan, and Y. Zhang, "A new RNN model with a modified nonlinear activation function applied to complex-valued linear equations," *IEEE Access*, vol. 6, pp. 62 954–62 962, 2018.

[17] Z. He, Q. Wang, and Y. Zhang, "A system of quaternary coupled Sylvester-type real quaternion matrix equations," *Automatica*, vol. 87, pp. 25–31, 2018.

[18] H. Kusamichi, T. Isokawa, N. Matsui, Y. Ogawa, and K. Maeda, "A new scheme for color night vision by quaternion neural network," in *Proc. 2nd Int. Conf. Auto. Robots Agents*. Palmerston North, New Zealand, Dec. 2004, pp. 101–106.

[19] J. Biggs, "A quaternion-based attitude tracking controller for robotic systems," in *IMA Conference on Mathematics of Robotics*, 2015.

[20] J. Sun, H. Zhang, S. Xu, and Y. Liu, "Full information control for switched neural networks subject to fault and disturbance," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–12, 2021.

[21] D. Guo and Y. Zhang, "Li-function activated ZNN with finite-time convergence applied to redundant-manipulator kinematic control via time-varying Jacobian matrix pseudoinversion," *Appl. Soft Comput*, vol. 24, pp. 158–168, Nov. 2014.

[22] Z. Jian, L. Xiao, K. Li, Q. Zuo, and Y. Zhang, "Adaptive coefficient designs for nonlinear activation function and its application to zeroing neural network for solving time-varying Sylvester equation," *J Frankl Inst*, vol. 357, no. 14, pp. 9909–9929, 2020.

[23] Q. Ma, S. Qin, and T. Jin, "Complex Zhang neural networks for complex-variable dynamic quadratic programming," *Neurocomputing*, vol. 330, pp. 56–69, 2019.

[24] L. Jin, Y. Zhang, S. Li, and Y. Zhang, "Noise-tolerant ZNN models for solving time-varying zero-finding problems: A control-theoretic approach," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 992–997, Feb. 2016.

[25] Y. Shi and Y. Zhang, "Solving future equation systems using integral-type error function and using twice ZNN formula with disturbances suppressed," *J. Franklin Inst.*, vol. 356, no. 4, pp. 2130–2152, Mar. 2019.

[26] Z. Tan, W. Li, L. Xiao, and Y. Hu, "New varying-parameter ZNN models with finite-time convergence and noise suppression for time-varying matrix Moore–Penrose inversion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2980–2992, 2019.

[27] W. Li, Z. Su, and Z. Tan, "A variable-gain finite-time convergent recurrent neural network for time-variant quadratic programming with unknown noises endured," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5330–5340, Sep. 2019.

[28] S. Li, B. Liu, and Y. Li, "Selective positive–negative feedback produces the winner-take-all competition in recurrent neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 2, pp. 301–309, 2012.

[29] P. S. Stanimirović, I. S. Živković, and Y. Wei, "Recurrent neural network for computing the drazin inverse," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2830–2843, 2015.

[30] F. Zhang, "Quaternions and matrices of quaternions," *Linear Algebra Appl.*, vol. 251, pp. 21–57, 1997.

[31] L. Xiao, "A new design formula exploited for accelerating Zhang neural network and its application to time-varying matrix inversion," *Theor. Comput. Sci.*, vol. 647, pp. 50–58, 2016.

[32] Y. Li, Y. Liu, and S. Tong, "Observer-based neuro-adaptive optimized control of strict-feedback nonlinear systems with state constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, 2021.

[33] Y. Zhang and S. S. Ge, "Design and analysis of a general recurrent neural network model for time-varying matrix inversion," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1477–1490, 2005.

[34] L. Xiao, J. Tao, J. Dai, Y. Wang, L. Jia, and Y. He, "A parameter-changing and complex-valued zeroing neural-network for finding solution of time-varying complex linear matrix equations in finite time," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 6634–6643, 2021.

[35] L. Xiao and Y. He, "A noise-suppression ZNN model with new variable parameter for dynamic Sylvester equation," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7513–7522, 2021.