Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

A Dynamically Adaptive Error-Based Varying Gain for Recurrent Neural Network to Solve Linear Time-Varying Equation

Yuhuan Chen, Junliu Zhong, Wei He, Chengli Sun[*]

School of Artificial Intelligence, Guangzhou Maritime University, Guangzhou, 510275, China
E-mail: sunchengli@gzmtu.edu.cn

Received: 15 September 2025; **Revised:** 3 October 2025; **Accepted:** 22 October 2025

Abstract: Convergence performance is a very important index for the Recurrent Neural Networks (RNN). Network model structure, activation function and learning rate (also named as gain) are the general ways to improve the convergence performance, in which, it is a common and effective method to design a suitable learning rate. Specially, the recent work has already presented the varying learning rate schemes for the superior convergence. However, these schemes have no relationship with the error function of the solved problem. It means that the learning rate would not change along with the error function. This would lead us to adjust the learning rate without purpose. To address this issue, we present a dynamically and adaptively error-based varying gain for the Zeroing Neural Networks (ZNN) to solve the linear time varying equation, together with its theoretical analysis on the convergence performance. The theoretical and experimental results shows that the error-based varying gain can be used to accelerate the convergence speed, and to achieve a superior convergence performance for the ZNN models.

Keywords: convergence performance, error-based varying gain, time varying, linear matrix equation, Zeroing Neural Network (ZNN)

MSC: 68T07, 65F45

Abbreviation

RNN	Recurrent Neural Networks
ZNN	Zeroing Neural Network
GLNN	Gradient-based Liquid Neural Network
FCRNN	Fully Connected Recurrent Neural Networks
LTVMVE	Linear Time Varying Matrix-Vector Equation
TVQP	Time Varying Quadratic Programming

1. Introduction

Intelligent computing is widely applied in many research investigations and practical engineering field. For example, the robot control problem $r(t) = f(\vartheta(t))$ can be synthesized as a quadratic programming problem solving with the end effector $r(t) \in R^m$, joint angle place $\vartheta(t) \in R^n$ and the nonlinear mapping $f(\cdot)$, which is also simulated in Section 4. [1–6]. In [7], the authors present the influence of local thermal nonequilibrium and thermal gradients on magneto-darcy-Rayleigh-Bénard convective stability by mathematical model analysis. In linear control system, the pole placement can be realized by the Sylvester equation $AX - XB = C$ solving [8–10], where $A \in R^{n*n}$, $B \in R^{n*n}$, $C \in R^{n*n}$, and X is the unknown matrix. Lyapunov matrix equation is used to judge the stability for the linear dynamic control system [11, 12], and linear matrix-vector equation is one of the basic formula for the deep learning [13, 14]. Because of the parallel-computing, convenient hardware realization and generalization ability, Recurrent Neural Networks (RNN) have already become a popular intelligent computing and optimization tool [15–18].

Generally speaking, the gradient-based neural network is broadly applied for the time-invariant intelligent computing problem. In [19], a robust Gradient-based Liquid Neural Network (GLNN) framework utilized ordinary differential equation-based liquid neurons is proposed to solve the beamforming problem, in which, the matrix/vector coefficients are static and fixed, which would not be changed with time t. However, in the real engineering applications, the parameters of the system are generally time varying, which is a function of time t. For instance, in the control process of the robot arm, the end-effector $r(t)$ and the joint angle $\vartheta(t)$ are certainly changed along with the time t [1–3]. To address these issues, recently, Zeroing Neural Networks (ZNN) are found to be good at the time-varying problem solving [4–6, 11, 12] and attracted many attentions.

It is noted that convergence performance is one of the key basic index to measure the neural networks, which could be improved from the network structure, activation function and learning rate (or say, gain). In [20], the authors present a survey on the structure of ZNN models for superior convergence and robustness, including the different ZNN structure model with the fixed/varying gain and activation functions. In [21], a novel Fully Connected Recurrent Neural Network (FCRNN) structure is proposed for the identification of unknown dynamics of nonlinear systems by imparting necessary memory property to the structure and improves its ability in handling the dynamical systems. Activation function is another effective method for a better convergence and robustness. Jin et al. in [22] presented a fuzzy activation function for the ZNN models to compute the currents in circuits with the improvement of the convergence and noise resistance ability by introducing the fuzzy logic technique. A novel Variable-Parameter Variable-Activation-Function Finite-Time Neural Network (VPA-FTNN) to deal with joint-angle drift issues of redundant-robotic arms [23], in which, the finite-time convergence can be achieved without the use of the special activation function. In a summary, much work is devoted to improve the convergence and robustness by designing and exploiting the novel activation functions [20–22].

Recently, as another versatile and effective technique for the improvement of the convergence, varying gain can also be encapsulated with ZNN models for the fast convergence speed. In [24], Xiao et al. presented a finite time ZNN model for the generalized Lyapunov equation, together with the comparisons with the fixed and varying gains. Its experimental results reveal that, in the same uniformly bounded perturbation environment, the residual error of the varying gain ZNN would fast converge to zero, while the fixed gain ZNN model obtains a bounded residual error. A fuzzy complex gain ZNN model is presented for the time-variant complex Sylvester equation problem, in which, the fuzzy parameter can be adaptively adjust the convergence rate according to the residual error from the fuzzy viewpoint [25]. Therefore, the varying gain can improve the convergence speed for the ZNN models [5, 6, 26–28].

However, it is contradictory that the close to zero residual error would be easily oscillated for the big gain, while the varying gain would be bigger and bigger along with the increase time t. This implies that it might be not suitable for the long deployment of the ZNN models if the varying gain increases with the increasement of time t as well. Therefore, to overcome this problem, in this paper, we present a balance varying gain scheme, which can be dynamically and adaptively suitable for the change of the residual error as time goes on. In the beginning, a big gain is provided for ZNN model since the beginning residual error is big, and when the residual error is close to zero near to the end, a relative small gain is required for ZNN model even if $t \to +\infty$. By this way, the beginning error would be a sharp decent in a short time, and then the small error close zero would perfectly and smoothly converge to zero. The theoretical analysis and experimental results further substantiate this conclusion.

The rest of the article are organized as follows. Section 2 lists the Linear Time Varying Matrix-Vector Equation (LTVMVE) problem solving, and presents the related ZNN models with the error-based varying gain, scheduled varying gain, error-related varying gain and the fixed gain. In Section 3, two theorems on the convergence of the proposed

ZNN models are discussed and proved. The validated experimental results are provided in Section 4, together with their comparisons under the same experimental environment and parameter settings. The conclusions are summarized in the final section. Before ending this section, the main contributions of this article are listed as follows.

(1) A novel error-based varying gain is proposed for the ZNN models, which can dynamically and adaptively adjust the convergence speed according to the residual error. Note that, in the beginning, the value of the varying gain is big to accelerate the convergence speed for the relative big residual error, and to avoid the oscillation for the small error close to zero, the value of the presented varying gain would be small automatically.

(2) Different from other varying gain scheme, this error-based gain scheme could be found a balance for the big or small residual error, since it is adaptively changed by the error. In other words, if the residual error is big, the neural model needs a big gain to accelerate the convergence speed, and if the error is small, the model needs a small gain to avoid the oscillation. Then our proposed error-based scheme can satisfied with these requirements.

(3) The theoretical analyses and experimental comparisons further show that the error-based varying scheme can be achieved a superior convergence performance for the ZNN models, and the application to the robot manipulator is also validated the effectiveness and exactness of our proposed scheme.

2. Problem formulation and related models

Suppose that the Linear Time-Varying Matrix-Vector Equation (LTVMVE) can be written as follows:

$$A(t)x(t) = b(t) \tag{1}$$

where $A(t) \in R^{m\times n}$ is a full-rank time varying matrix with $m \geq n$, $b(t) \in R^m$ is a time-varying vector, and $x(t) \in R^n$ is unknown to be solved. Our main objective is to find the solution $x(t)$ to satisfied with the LTVMVE (1). For analysis convenience in the latter sections, $x^*(t)$ is supposed to be the theoretical solution of LTVMVE (1), which is used to compare to the solution $x(t)$, and it can be computed easily by Matrix Laboratory (MATLAB) tool.

Following by the design method of ZNN models [5, 8, 18, 29], an indefinite (i.e., positive, negative or zero) error function is defined as

$$e(t) = A(t)x(t) - b(t), \tag{2}$$

which is corresponding to the problem (1), our goal is to find a neural solution $x(t) \in R^n$ to make $e(t) \in R^n$ equaling zero. Therefore, let

$$\dot{e}(t) = -\gamma(e(t))f(e(t)), \tag{3}$$

where $\gamma(e(t))$ is a varying gain based on the error function $e(t)$, which is exploited to control the convergence rate of the network model, and $f(\cdot)$ is a non-monotonically increasing activation function, which can make the network model be able to fit the complex functions. In this article, the following activation functions are used.

(1) The linear activation function:

$$f(u) = u. \tag{4}$$

(2) The power-sigmoid function

$$f(u) = \begin{cases} u^p, & \text{if } |u| \geq 1, \\ \dfrac{1+\exp(-\varepsilon)}{1-\exp(-\varepsilon)} \cdot \dfrac{1-\exp(-\varepsilon u)}{1+\exp(-\varepsilon u)}, & \text{other} \end{cases} \tag{5}$$

with $\varepsilon > 2$ and $p \geq 3$.

(3) And the sign-bi-power activation function:

$$f(u) = (|u|^{\xi} + |u|^{1/\xi})\operatorname{sgn}(u)/2, \quad \xi > 0 \tag{6}$$

with $\operatorname{sgn}(\cdot)$ denoting the following signum function:

$$\operatorname{sgn}(u) \begin{cases} = 1, & \text{if } u > 0, \\ = 0, & \text{if } u = 0, \\ = -1, & \text{if } u < 0. \end{cases}$$

Remark 1 The varying gain $\gamma(e(t))$ can be set as the following different expression under the different environments.

(1) The gain is a constant $\gamma(e(t)) = \gamma$, which is certainly non-relation with the error.

(2) The gain is only relation with time t, i.e., $\gamma(e(t)) = \gamma(t)$.

(3) The gain would be related with both time t and error function $e(t)$, i.e., $\gamma(e(t))$.

Therefore, by the the case (3) of Remark 1, we propose a varying gain based on the residual error as follows

$$\gamma(e(t)) = \left(\gamma_0 + \gamma_1 \exp\left(-\frac{t}{\tau}\right)\right)(1 + k\|e(t)\|_2), \tag{7}$$

where $\gamma_0 > 0$ is a constant initial varying gain, $\gamma_1 > 0$ is a constant used for the adjustment of varying gain, $\tau > 0$ is an attenuation factor on time t, which is used to control attenuation speed of t. If the bigger τ is, the more slowly the varying gain attenuates. k is a sensitivity coefficient, which is used to control the effect of error $e(t)$ on the varying gain $\gamma(e(t))$. The bigger $k > 0$ is, the more strongly the varying gain responds to changes in error $e(t)$.

Remark 2 As for the proposed error-based varying gain scheme (7), the time attenuation item $\exp\left(-\frac{t}{\tau}\right)$ shows a downward trend for the varying gain during the whole iteration process, which is expected for a much small varying gain as the error $\|e(t)\|_2$ decreases. The item $1 + k\|e(t)\|_2$ can make the varying gain $\gamma(e(t))$ dynamically and adaptively adjust to a suitable value according to the change of the error $\|e(t)\|_2$. The varying gain will automatically be big or small, which is corresponding to the big or small error, respectively. Therefore, in the early stage, although t is very small, the network model can still have a relative big varying gain and can accelerate the convergence speed of the network model till the residual error is relative small. In the later stages, the residual error would be very small, and it requires that the value of the gain cannot be big, since there are fluctuations in the error if the gain is big. Therefore, by the proposed error-based scheme (7), although t is big, the varying gain will not be too large, and thus the residual error of the network will not oscillate. The error will smoothly converge to zero.

By combining problem (1) and Eqs. (2)-(7), we can get the ZNN model encapsulated with the error-based scheme (7)

$$\dot{A}(t)x(t) = -\gamma(\|A(t)x(t)-b(t)\|_2)f(A(t)x(t)-b(t)) - A(t)\dot{x}(t) + \dot{b}(t), \tag{8}$$

where $\|\cdot\|_2$ is denoted by the 2-norm.

For comparison with our proposed ZNN model (8), another two gain schemes are also used for the solution of problem (1), which are presented as follows.

In [5], the authors presented a scheduled-varying-gain-based ZNN model

$$\dot{A}(t)x(t) = -(\gamma_0 + \gamma_1 \exp(-\gamma_2 t))f(A(t)x(t)-b(t)) - A(t)\dot{x}(t) + \dot{b}(t), \tag{9}$$

where the scheduled varying gain

$$\gamma(t) = \gamma_0 + \gamma_1 \exp(-\gamma_2 t) \tag{10}$$

which is corresponding to the case (2) in Remark 1. γ_0 and γ_1 are the same as that of scheme (7), γ_2 is the known positive constant used to adjust the value of the varying gain to improve the network convergence performance.

In [6], Wu et al. presented an adaptive error-related ZNN model

$$\dot{A}(t)x(t) = -(\gamma_0 + \|e(t)\|_2)f(A(t)x(t)-b(t)) - A(t)\dot{x}(t) + \dot{b}(t), \tag{11}$$

where

$$\gamma(t) = \gamma_0 + \|e(t)\|_2 \tag{12}$$

with the same constant $\gamma_0 > 0$ as the above-mentioned value. Without loss of generality, the conventional ZNN model is also presented as follows

$$\dot{A}(t)x(t) = -\gamma_0 f(A(t)x(t)-b(t)) - A(t)\dot{x}(t) + \dot{b}(t) \tag{13}$$

with the same

$$\gamma(t) = \gamma_0. \tag{14}$$

Remark 3 Compared to the four ZNN models (8), (9), (11), and (13), we know that the evident difference is that their gains are different. But we can use the error-based varying gain (7) to express other three gains, which exhibit that the gain will be a relation of exponential attenuation on time t to accelerate the convergence speed, and it is relation with the error feedback to dynamically and adaptively to improve the gain along with the change of error $e(t)$.

Remark 4 Compared with the four gain schemes (7), (10), (12), and (14), γ_0, γ_1, γ_2, τ, and k are the positive constants. Generally speaking, if their values are big, the value of the gain will also be big, and the resultant convergence speed is big. For example, as for the ZNN model (13), $\gamma_0 = 5$ or $\gamma_0 = 10$. We can set the different γ_0 by the requirements of the network model. In addition, (14) is the simplest scheme, there is no any adjustable parameters for the convergence. The scheduled varying gain (10) can get the superior convergence performance for a long deployment. However, this scheme is nothing to do with the residual error $e(t)$, and thus it lacks an adjustable parameter. Although the scheme (12) is related to the residual error, it is just used to adjust the value of γ_0, and does not magnify the role of the residual error. Therefore, the proposed error-based varying gain scheme (7) is a comprehensive embodiment of other three schemes, combining their advantages to design.

3. Theoretical analysis

In this section, we would like to address the theoretical analysis on the convergence performance of ZNN model (8).

Theorem 1 Consider the problem (1) solved by ZNN model (8). Any non-monotonically increasing activation function $f(\cdot)$ is used. The neural state $x(t)$ will globally converge to the theoretical solution $x^*(t)$. In addition, if the linear activation function $f(u) = u$ is used, then the exponential rate is at least $2\gamma_0$.

Proof. Define a Lyapunov candidate function $l_i(t) = e_i^2(t)/2$ for the ZNN model (8), together with $e_i(t)$ is denoted by the element-wise of $e(t) \in R^m$. Therefore,

$$\dot{l}_i(t) = e_i(t)\dot{e}_i(t) = -\gamma(e_i(t))e_i(t)f(e_i(t)). \tag{15}$$

Since $f(\cdot)$ is a non-monotonically increasing activation function, it is evident that $l_i(t) \geq 0$ and $\dot{l}_i(t) \leq 0$. By the Lyapunov stability, the error $e_i(t)$ would converge to zero, which means that the neural state $x_i(t) \to x_i^*(t)$. Note that, iff $e_i(t) = 0$, then we have $l_i(t) = 0$. Moreover, from (15), if ZNN model (8) is encapsulated with the linear function, we have

$$\dot{l}_i(t) = -\gamma(e_i(t))e_i^2(t) = -2\gamma(e_i(t))l_i(t).$$

By substituting (7) into the above equation and when time $t \to 0$, we have

$$\dot{l}_i(t) \leq -2\gamma_0 l_i(t).$$

Thus, solving the above inequality, we have

$$l_i(t) \leq l_i(0)\exp(-2\gamma_0 t).$$

The Lyapunov function $l_i(t)$ can be viewed as an energy function on the error function $e_i(t)$. From the above inequality, we know that the $l_i(t)$ would converge to zero with the exponential rate $2\gamma_0$, which also means that the error $e_i(t) \to 0$ with the same convergence rate. In other words, $A(t)(x(t) - x^*(t))$ would also converge to zero with the exponential rate $2\gamma_0$; that is, $x(t) \to x^*(t)$. The proof is completed. □

Theorem 2 Consider ZNN model (8) solving to the problem (1). A faster convergence speed can be obtained when the power-sigmoid function (5) is used, compared to the use of linear function (4).

Proof. Since the rate of change for the Lyapunov function $l_i(t)$ (i.e., $\dot{l}_i(t)$) can be used to express the convergence speed of the ZNN model (8), then we can use $\dot{l}_{i-lin}(t)$ and $\dot{l}_{i-ps}(t)$ to compare with the convergence speed of the linear function and the power-sigmoid function, in which, $\dot{l}_{i-lin}(t)$ and $\dot{l}_{i-ps}(t)$ are denoted by the rates of change when linear function and power-sigmoid function are used, respectively. Therefore, by following (5), the following two cases are considered.

Case 1: When $|e_i(t)| \geq 1$, from Theorem 1, we know that $\dot{l}_i(t) = -\gamma(e_i(t))e_i(t)f(e_i(t))$, thus

$$\begin{aligned}\dot{l}_{i-lin}(t) - \dot{l}_{i-ps}(t) &= -\gamma(e_i(t))e_i(t)(e_i(t) - e_i^p(t)) \\ &= -\gamma(e_i(t))e_i^2(t)(1 - e_i^{p-1}(t)) \\ &\geq 0\end{aligned}$$

That is, $\dot{l}_{i-lin}(t) \geq \dot{l}_{i-ps}(t)$, which means that the convergence speed for the use of power-sigmoid is faster than that of the linear function along with the negative direction.

Case 2: When $|e_i(t)|$ is other value, we have

$$\begin{aligned}\dot{l}_{i-lin}(t) - \dot{l}_{i-ps}(t) &= -\gamma(e_i(t))e_i(t)\left(e_i(t) - \frac{1+\exp(-\varepsilon)}{1-\exp(-\varepsilon)} \cdot \frac{1-\exp(-\varepsilon e_i(t))}{1+\exp(-\varepsilon e_i(t))}\right) \\ &= -\gamma(e_i(t))e_i(t)\frac{(e_i(t)+1)\exp(-\varepsilon)\left(\exp\left(\dfrac{e_i(t)}{\varepsilon}\right) - 1\right) + (e_i(t)-1)(1-\exp(-\varepsilon(1+e_i(t)))}{(1-\exp(-\varepsilon))(1+\exp(-\varepsilon e_i(t))}\end{aligned}$$

Consider $|e_i(t)| < 1$ and $\varepsilon > 2$. Therefore, $\exp\left(\dfrac{e_i(t)}{\varepsilon}\right) - 1 < 0$ and $(e_i(t)-1)(1-\exp(-\varepsilon(1+e_i(t))) < 0$. That is, $\dot{l}_{i-lin}(t) - \dot{l}_{i-ps}(t) \geq 0$. Similarly, we can draw the same conclusion as Case 1. The proof is thus completed. □

4. Experiments and comparisons

In this section, we would like to carried out the experiments for the ZNN models (8), (9), (11), and (13), together with their comparisons on the convergence performance under the same simulation environment.

Example 1 LTVMVE problem solving.

Consider the LTVMVE problem (1) with the following time varying coefficients

$$A(t) = \begin{bmatrix} 0.5\sin t + 2 & \cos t & \sin(4t) \\ \cos t & 0.5\cos t + 2 & \cos(4t) \\ \sin(4t) & \cos(4t) & 0 \end{bmatrix}, \quad b(t) = \begin{bmatrix} \sin(3t) \\ -\cos(3t) \\ \cos(2t) \end{bmatrix}.$$

To validate the solution exactness of the ZNN models, the theoretical solution $x^*(t)$ can be computed by using MATLAB (MATLAB version: R2016a; MATLAB tool: ODE45) running on the common computer (11th Gen Intel(R), Core(TM) i7-1165G7, @2.80 GHz, 16 G RAM). Since the theoretical solution is very complex in expression form, and thus it is ignored for reading convenience.

Suppose that $\gamma_0 = \gamma_1 = \gamma_2 = k = 10$, and $\tau = 300$. When the power-sigmoid activation function is used for the ZNN model (8) with $p = 3$ and $\varepsilon = 4$, we can obtain the simulation results as shown in Figure 1. The neural state $x(t)$ is denoted by the solid blue lines, and the theoretical solution is denoted by the red dash-dotted lines, which can be seen at Figure 1a. Evidently, $x(t)$ is coincided quickly with $x^*(t)$, even if the rate of change for this curve is large. It means that the residual error $e(t) = \|A(t)x(t) - b(t)\|_2$ is also fast convergent to zero, as shown in Figure 1b within 1 s. This implies that the neural state $x(t)$ is convergent to $x^*(t)$.

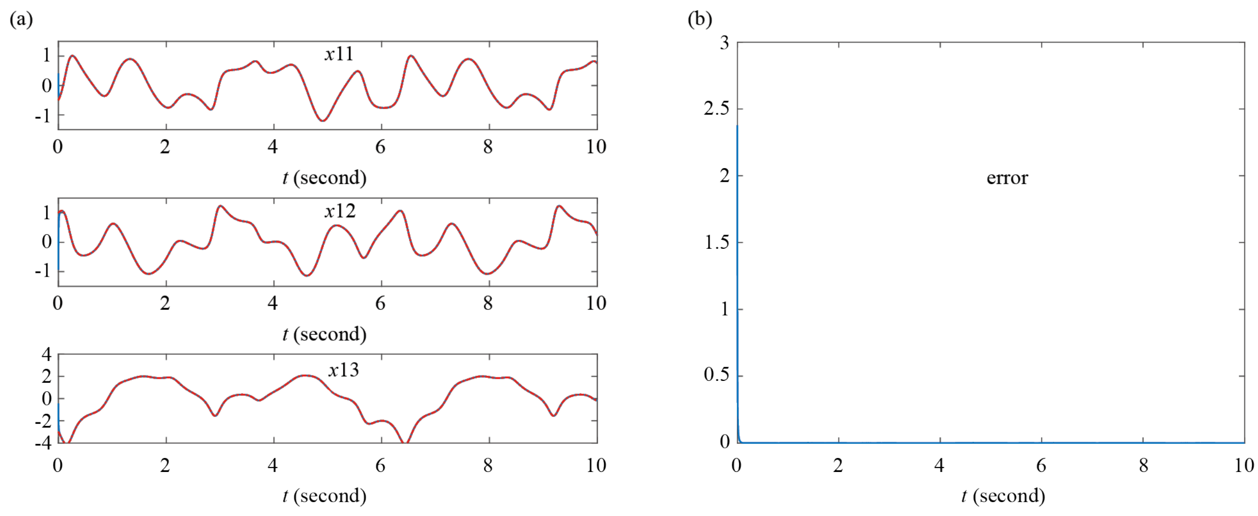

Figure 1. Online solution of ZNN model (8) with power-sigmoid function (5). (a) Neural state $x(t)$; (b) Norm error $\|A(t)x(t) - b(t)\|$

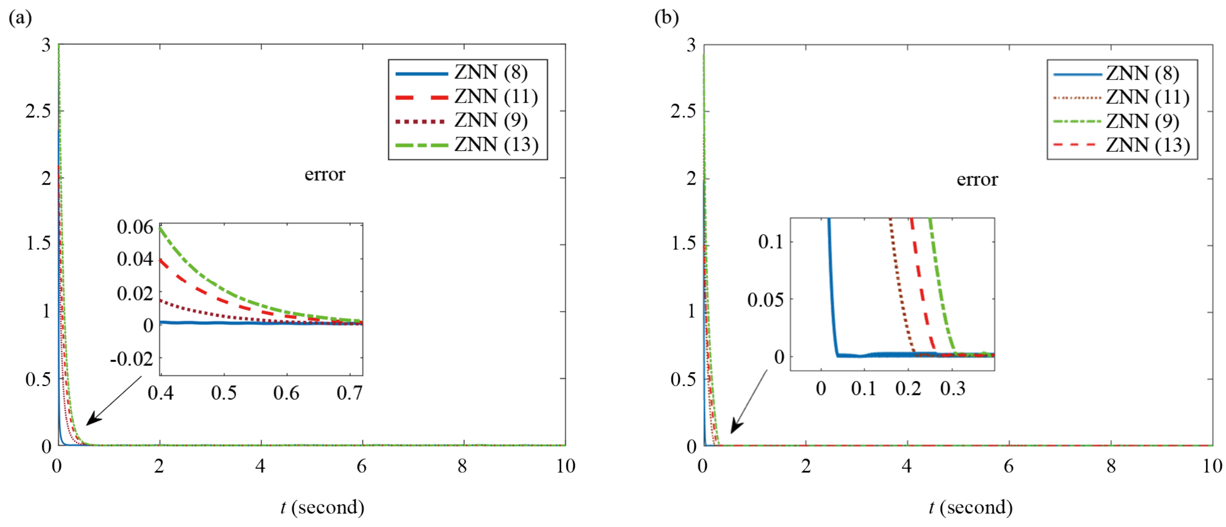

Figure 2. Convergence performance of ZNN models (8)-(13) with different activation functions. (a) Activated by linear function (6); (b) Activated by sign-bi-power function (4)

As shown in Figure 2, if used different activation functions, the ZNN models (8), (9), (11), and (13) would be achieved different convergence performance. if activated by the same linear function, ZNN model (8) would achieve the best convergence performance, as shown in Figure 2a. The same conclusion could be drawn from Figure 2. It is noted that, ZNN (13) get the worst convergence performance among these four ZNN models, this is because its gain value is the minimum. Compared to the ZNN model (13), the convergence speed would be slower since the error $e(t)$ infinitely closes to zero, its varying gain is also close to the constant γ_0 used in ZNN model (13), and then it need more compute time.

Table 1 shows the comparisons on the convergence performance for the four ZNN models. Evidently, as for the same ZNN model, different activation function can obtain the different convergence performance. For example, when linear function is encapsulated on the ZNN (8), the time is 0.1191 s when the residual error is 10^{-3} at the first time, while it is 0.03516 s when the sign-bi-power function is used. The steady state error would almost the same since the final convergence results would be same as for the same ZNN model. In addition, as for the residual error arriving at 10^{-3} at first time, ZNN model (8) would get the best convergence effect. This is because the varying gain value is the biggest among the four schemes, which further validates that the convergence effect is closely related to the varying gain value. Moreover, as for the steady-state error, (9) would get the best convergence effect. This is because the varying gain value in this scheme is the biggest among the four varying gain schemes. However, this big gain would take the oscillating convergence phenomenon.

Table 1. Comparisons on convergence performance of the four ZNN models with the same experimental environment

ZNN model	Activation function	Time (s)	Steady-state error $e(t)$
ZNN (8)	Linear function	0.1191	1.652×10^{-4}
	Sign-bi-power	0.03516	2.231×10^{-4}
ZNN (9)	Linear function	0.45	7.406×10^{-5}
	Sign-bi-power	0.212	2.422×10^{-5}
ZNN (11)	Linear function	0.5789	5.69×10^{-4}
	Sign-bi-power	0.3072	1.032×10^{-4}
ZNN (13)	Linear function	0.5533	6.815×10^{-5}
	Sign-bi-power	0.2576	2.795×10^{-4}

Note: time (s) is the time when the residual error arrives at 10^{-3} at the first time

Example 2 Application to robot arm.

Robot arm can be used in many engineering fields, e.g., trajectory tracking and repetitive motion [2, 17, 29, 30]. In this section, we use the PUMA560 robot arm [29] to track a desired star trajectory with error-free manner. To do this, the mathematical model of PUMA560 arm should be built first, which is expressed as the following forward-kinematics notion equation Cartesian space [17, 29, 30]

$$r(t) = f(\vartheta(t)) \tag{16}$$

where $r(t) \in R^m$ is denoted by the position of the end effector, which is similar to a pen to draw the trajectory in real time in this simulation. $\vartheta(t) \in R^n$ is denoted by the joint angle space, which shows the angle of the end effector in Cartesian space. The mapping $f(\cdot)$ is evidently a nonlinear function, which expresses the relationship between the joint angle $\vartheta(t)$ and the position $r(t)$. Note that, $r(t)$, $\vartheta(t)$, and $f(\cdot)$ could be differentiable, which can promise Eq. (16) to be solved by using differential technique, since Eq. (16) is difficultly solved directly. Therefore, by differentiating Eq. (16), we can get the expression of PUMA560 in velocity level with the following dynamic equation

$$\dot{r}(t) = J(\vartheta(t))\dot{\vartheta}(t) \tag{17}$$

where $J(\vartheta(t)) = \partial f(\vartheta(t))/\partial\vartheta(t) \in R^{m\times n}$ is a Jacobian matrix [29, 31]. Similarly, it is not easy to solve the motion Eq. (17) in velocity level. To ensure the joint angle velocity $\vartheta(t)$ changes smoothly and minimize wear and energy consumption of the robotic arm, we can take the sum of the squares of joint velocities as the optimization objective. However, the solution of the optimization problem should be subjected to Eq. (17). In other words, the solution of the optimization problem is also the solution of the dynamic Eq. (17). Therefore, the dynamic equation in velocity level can be transformed into the following optimization problem to be solved

$$\begin{cases} \min & \frac{1}{2}\|\dot{\vartheta}(t)\|_2^2, \\ \text{s. t.} & J(\vartheta(t))\dot{\vartheta}(t) = \dot{r}(t). \end{cases} \tag{18}$$

Consider $\|\dot{\vartheta}(t)\|_2^2 = \dot{\vartheta}(t)\dot{\vartheta}^T(t)$. By using the replacement technique, Eq. (18) can be further become into the Time Varying Quadratic Programming (TVQP) problem

$$\begin{cases} \min & \frac{1}{2}\mathrm{x}^T(t)W(t)\mathrm{x}(t) + q^T(t)\mathrm{x}(t), \\ \text{s. t.} & C(t)\mathrm{x} = d(t), \end{cases} \tag{19}$$

with $\mathrm{x} := \dot{\vartheta}(t) \in \mathrm{R}^n$, $W(t) := I \in \mathrm{R}^{n\times n}$, $C(t) := J(\vartheta(t)) \in \mathrm{R}^{m\times n}$, $d(t) := \dot{r}(t) \in \mathrm{R}^m$, and $q := 0 \in \mathrm{R}^n$. As for the solution the similar TVQP problem solving, the recent work [5, 6, 29] have already present many solution scheme. Further more, TVQP problem (19) can be changed into the following LTVMVE problem $A(t)x(t) = b(t)$ with the following time varying coefficients

$$A(t) := \begin{bmatrix} W(t) & C^T(t) \\ C(t) & \mathbf{0}_{m\times m} \end{bmatrix} \in R^{(n+m)\times(n+m)},$$

$$x(t) := \begin{bmatrix} \mathrm{x}(t) \\ \nu(t) \end{bmatrix} \in R^{n+m}, \quad b(t) := \begin{bmatrix} -q(t) \\ d(t) \end{bmatrix} \in R^{n+m}.$$

As a result, ZNN model (8) can be used to solve the above time varying problem. To show the solution exactness, the desired trajectory $r_d(t)$ is given out as follows

$$r_d(t) := \begin{bmatrix} 0.4(\cos(2t)/25) + \cos((4t)/3)/10) + r_x(0) - 0.056 \\ 0.4(\sin(2t)/25 - (3\sin((4t)/3))/50) + r_y(0) \\ r_z(0) \end{bmatrix},$$

which can be viewed as the theoretical solution, and is the trajectory tracking objective of $r(t)$ obtained by ZNN model (8).

The tracking effect can be seen in Figure 3. When ZNN model (8) is used for the tracking of PUMA560, all of the error-wise can be arrived at 10^{-10} in x, y, and z axis; for example, the max value of x-error is about 10×10^{-10}, as shown Figure 3a. This experimental result is validated again in Figure 3b. The blue solid line is denoted by the trajectory $r(t)$ of ZNN model (8), and the red dash-dotted line is denoted by the desired trajectory $r_d(t)$ (it is also called as theoretical solution for problem (1)). Evidently, both are coincided with each other perfectly, and thus the robot arm PUMA560 can efficiently track the given desired object trajectory $r_d(t)$, which can be shown in Figure 3c.

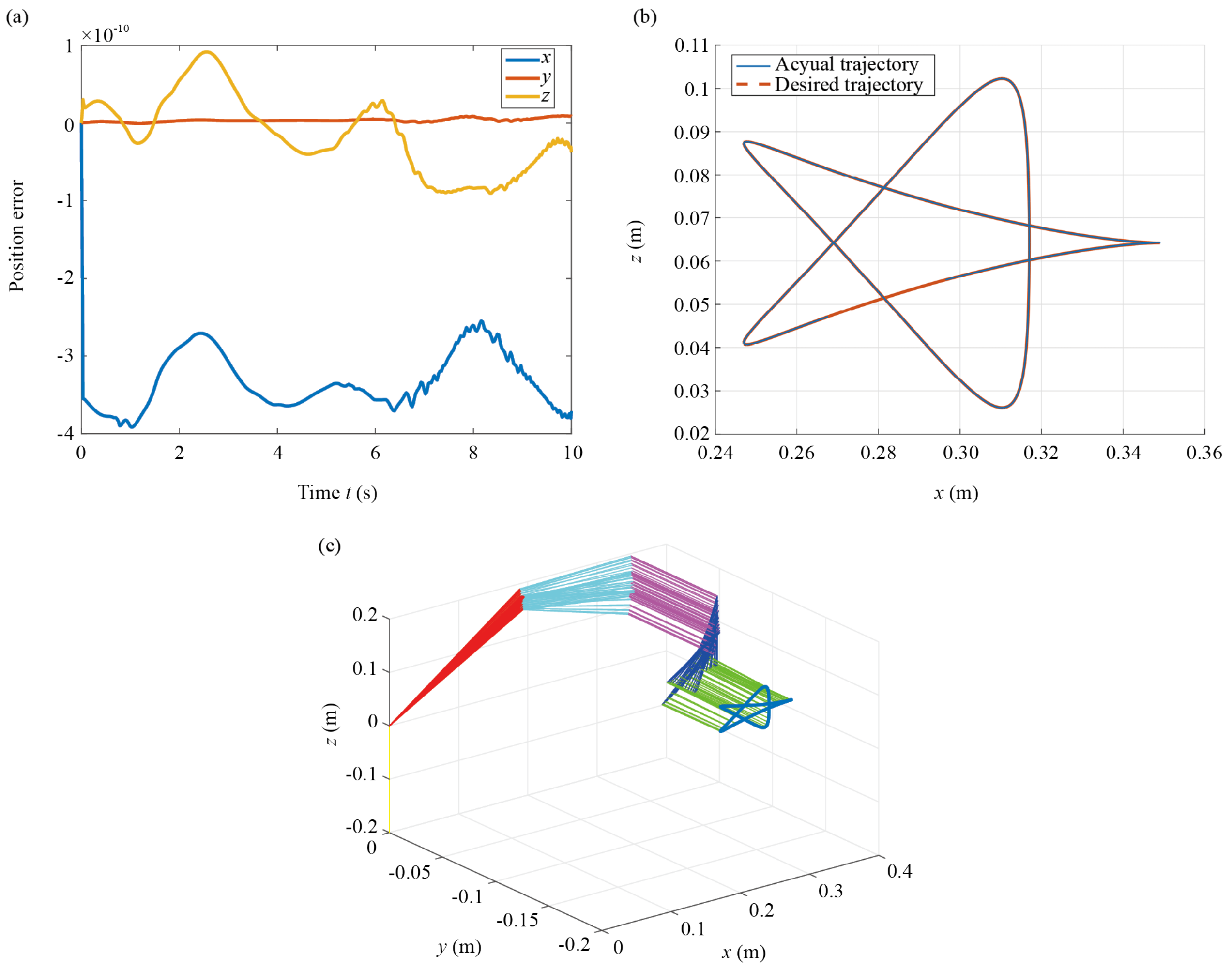

Figure 3. Trajectory tracking of PUMA560 solved by ZNN model (8). (a) Tracking error in x, y, and z axis; (b) Trajectories $r(t)$ and $r_d(t)$; (c) Trajectory of PUMA560

5. Conclusions

Varying gain is a key basic technique to improve the convergence performance for the network models. However, the traditional varying gain scheme would become bigger and bigger along with the time t. In this case, the smaller residual error is, the bigger the value of the varying gain is. Evidently, it is unreasonable because the oscillation of residual error easily happen. Therefore, in this paper, a dynamically and adaptively error-based varying gain scheme is proposed to balance the relationship between residual error and time t. In other words, if the residual error is big, this proposed scheme can provide a relative big varying gain to accelerate the convergence, and also can provide a small varying gain for the small error to smoothly converge to zero. The theoretical analysis and the experimental results further validate the effectiveness of this proposed error-based varying gain scheme. In the near future work, we would like to use this proposed error-based varying gain scheme to many scientific applications (e.g., robot arm control, chaotic control, and other areas), together with the detailed analysis and the comparisons.

Author contributions

Conceptualization, Y. C.; methodology, Y. C.; software, Y. C.; validation, C. S.; formal analysis, C. S.; investigation, Y. C. and C. S.; writing—original draft preparation, Y. C.; writing—review and editing, C. S.; visualization, C. S.; supervision, C. S.; project administration, C. S.; funding acquisition, C. S. All authors have read and agreed to the published version of the manuscript

Acknowledgement

This work was supported by Guangdong basic and applied basic research foundation under Grant (2025A151501 0466), Guangdong Key Construction Discipline Scientific Research Ability Enhancement Project (2024ZDJS056) and the Guangzhou Programs (2024312151, 2024312000, and 2024312374).

Conflict of interest

The authors declare no conflict of interest.

References

[1] Bambade A, Schramm F, El-Kazdadi S, Caron S, Taylor A, Carpentier J. Proxqp: An efficient and versatile quadratic programming solver for real-time robotics applications and beyond. *IEEE Transactions on Robotics*. 2025; 1-19. Available from: https://doi.org/10.1109/TRO.2025.3577107.

[2] Balandi L, Giordano PR, Tognon M. QP-based inner-loop control for constraint-safe and robust trajectory tracking for aerial robots. *IEEE Robotics and Automation Letters*. 2025; 10(7): 6640-6647. Available from: https://doi.org/10.1109/LRA.2025.3568319.

[3] Li W, Wu H, Jin L. A lower dimension zeroing neural network for time-variant quadratic programming applied to robot pose control. *IEEE Transactions on Industrial Informatics*. 2024; 20(10): 11835-11843. Available from: https://doi.org/10.1109/TII.2024.3413317.

[4] Yan D, Li C, Wu J, Deng J, Zhang Z, Yu J, et al. A novel error-based adaptive feedback zeroing neural network for solving time-varying quadratic programming problems. *Mathematics*. 2024; 12(13): 2090. Available from: https://doi.org/10.3390/math12132090.

[5] Fu J, Zhang Y, Geng G, Liu Z. Recurrent neural network with scheduled varying gain for solving time-varying QP. *IEEE Transactions on Circuits and Systems II: Express Briefs*. 2023; 71(2): 882-886. Available from: https://doi.org/10.1109/TCSII.2023.3313554.

[6] Wu W, Zhang Y, Tan N. Adaptive ZNN model and solvers for tackling temporally variant quadratic program with applications. *IEEE Transactions on Industrial Informatics*. 2024; 20(11): 13015-13025. Available from: https://doi.org/10.1109/TII.2024.3431046.

[7] Narayanappa M, Balaji VK, Munirathnam L, Ramakrishna S. Influence of local thermal nonequilibrium and thermal gradients on magneto-Darcy-Rayleigh-Bénard convective stability with heat generation. *Heat Transfer*. 2025. Available from: https://doi.org/10.1002/htj.70057.

[8] Cai J, Zhang W, Zhong S, Yi C. A super-twisting algorithm combined zeroing neural network with noise tolerance and finite-time convergence for solving time-variant Sylvester equation. *Expert Systems with Applications*. 2024; 248: 123380. Available from: https://doi.org/10.1016/j.eswa.2024.123380.

[9] Guo P, Zhang Y, Li S. Reciprocal-kind Zhang neurodynamics method for temporal-dependent Sylvester equation and robot manipulator motion planning. *IEEE Transactions on Neural Networks and Learning Systems*. 2025; 36(8): 15110-15124. Available from: https://doi.org/10.1109/TNNLS.2025.3525747.

[10] Casulli A, Robol L. An efficient block rational Krylov solver for Sylvester equations with adaptive pole selection. *SIAM Journal on Scientific Computing*. 2024; 46(2): A798-A824. Available from: https://doi.org/10.1137/23M1548463.

[11] Li S, Zhou B, Duan G. Performance recovery for a class of nonlinear systems by a parametric Lyapunov equation based observer. *International Journal of Control*. 2025; 98: 2506-2515. Available from: https://doi.org/10.1080/00207179.2025.2463573.

[12] Xiao L, Yan X, He Y, Luo B, Song Q. A nonlinear noise-resistant zeroing neural network model for solving time-varying quaternion generalized Lyapunov equation and applications to color image processing. *IEEE Transactions on Neural Networks and Learning Systems*. 2025; 36(8): 14371-14383. Available from: https://doi.org/10.1109/TNNLS.2025.3526620.

[13] Moralis-Pegios M, Mourgias-Alexandris G, Tsakyridis A, Giamougiannis G, Totovic A, Dabos G, et al. Neuromorphic silicon photonics and hardware-aware deep learning for high-speed inference. *Journal of Lightwave Technology*. 2022; 40(10): 3243-3254. Available from: https://doi.org/10.1109/JLT.2022.3171831.

[14] Panagakis Y, Kossaifi J, Chrysos GG, Oldfield J, Nicolaou MA, Anandkumar A, et al. Tensor methods in computer vision and deep learning. *Proceedings of the IEEE*. 2021; 109(5): 863-890. Available from: https://doi.org/10.1109/JPROC.2021.3074329.

[15] Tokarev K, Lebed N, Prokofiev P, Volobuev S, Yudaev I, Daus Y, et al. Monitoring and intelligent management of agrophytocenosis productivity based on deep neural network algorithms. In: *Proceedings of the 5th International Conference on Intelligent Computing and Optimization 2022 (ICO2022)*. Springer; 2022. p.686-694. Available from: https://doi.org/10.1007/978-3-031-19958-5_65.

[16] Sandamirskaya Y, Kaboli M, Conradt J, Celikel T. Neuromorphic computing hardware and neural architectures for robotics. *Science Robotics*. 2022; 7(67): eabl8419. Available from: https://doi.org/10.1126/scirobotics.abl8419.

[17] Chen J, Pan Y, Zhang Y, Li S, Tan N. Inverse-free zeroing neural network for time-variant nonlinear optimization with manipulator applications. *Neural Networks*. 2024; 178: 106462. Available from: https://doi.org/10.1016/j.neunet.2024.106462.

[18] Chen Y, Chen J, Yi C. A pre-defined finite time neural solver for the time-variant matrix equation $E(t)X(t)G(t) = D(t)$. *Journal of the Franklin Institute*. 2024; 361(6): 106710. Available from: https://doi.org/10.1016/j.jfranklin.2024.106710.

[19] Wang X, Zhu F, Huang C, Alhammadi A, Bader F, Zhang Z, et al. Robust beamforming with gradient-based liquid neural network. *IEEE Wireless Communications Letters*. 2024; 13(11): 3020-3024. Available from: https://doi.org/10.1109/LWC.2024.3436576.

[20] Zhou X, Liao B. Advances in zeroing neural networks: convergence optimization and robustness in dynamic systems. *Mathematics*. 2025; 13(11): 1801. Available from: https://doi.org/10.3390/math13111801.

[21] Shobana R, Jaint B, Kumar R. Design of a novel robust recurrent neural network for the identification of complex nonlinear dynamical systems. *Soft Computing*. 2024; 28(3): 2737-2751. Available from: https://doi.org/10.1007/s00500-023-09187-5.

[22] Jin J, Chen W, Ouyang A, Liu H. Toward fuzzy activation function activated zeroing neural network for currents computing. *IEEE Transactions on Circuits and Systems II: Express Briefs*. 2023; 70(11): 4201-4205. Available from: https://doi.org/10.1109/TCSII.2023.3269060.

[23] Deng J, Li C, Chen R, Zheng B, Zhang Z, Yu J, et al. A novel variable-parameter variable-activation-function finite-time neural network for solving joint-angle drift issues of redundant-robot manipulators. *IEEE/ASME Transactions on Mechatronics*. 2024; 30(2): 1578-1589. Available from: https://doi.org/10.1109/TMECH.2024.3425325.

[24] Zuo Q, Li K, Xiao L, Li K. Robust finite-time zeroing neural networks with fixed and varying parameters for solving dynamic generalized Lyapunov equation. *IEEE Transactions on Neural Networks and Learning Systems*. 2021; 33(12): 7695-7705. Available from: https://doi.org/10.1109/TNNLS.2021.3086500.

[25] Kong Y, Zeng X, Jiang Y, Sun D. Comprehensive study on a fuzzy parameter strategy of zeroing neural network for time-variant complex Sylvester equation. *IEEE Transactions on Fuzzy Systems*. 2024; 32(8): 4470-4481. Available from: https://doi.org/10.1109/TFUZZ.2024.3401109.

[26] Gerontitis D, Behera R, Tzekis P, Stanimirović P. A family of varying-parameter finite-time zeroing neural networks for solving time-varying Sylvester equation and its application. *Journal of Computational and Applied Mathematics*. 2022; 403: 113826. Available from: https://doi.org/10.1016/j.cam.2021.113826.

[27] Zhang Y, Wang L, Zhong G. Design and analysis of a variable-parameter noise-tolerant ZNN for solving time-variant nonlinear equations and applications. *Applied Intelligence*. 2025; 55(6): 460. Available from: https://doi.org/10.1007/s10489-025-06304-9.

[28] Chen Y, Chen J, Yi C. Time-varying learning rate for recurrent neural networks to solve linear equations. *Mathematical Methods in the Applied Sciences*. 2022. Available from: https://doi.org/10.1002/mma.8835.

[29] Zhang Y, Yi C. *Zhang Neural Networks and Neural-Dynamic Method*. Nova Science Publishers, Inc.; 2011.

[30] Guo D, Zhang C, Cang N, Jia Z, Xue S, Zhang W, et al. Harmonic noise rejection zeroing neural network for time-dependent equality-constrained quadratic program and its application to robot arms. *IEEE Transactions on Industrial Informatics*. 2024; 21(2): 1279-1288. Available from: https://doi.org/10.1109/TII.2024.3476530.

[31] Cao M, Xiao L, Zuo Q, Li L, Gao X. Data-based model-free predictive control system under the design philosophy of MPC and zeroing neurodynamics for robotic arm pose tracking. *IEEE Transactions on Neural Networks and Learning Systems*. 2025; 36(9): 17477-17490. Available from: https://doi.org/10.1109/TNNLS.2025.3553195.