From Zhang Neural Network to Newton Iteration for Matrix Inversion

Yunong Zhang, *Member, IEEE*, Weimu Ma, and Binghuang Cai

***Abstract*—Different from gradient-based neural networks, a special kind of recurrent neural network (RNN) has recently been proposed by Zhang *et al.* for online matrix inversion. Such an RNN is designed based on a matrix-valued error function instead of a scalar-valued error function. In addition, it was depicted in an implicit dynamics instead of an explicit dynamics. In this paper, we develop and investigate a discrete-time model of Zhang neural network (termed as such and abbreviated as ZNN for presentation convenience), which is depicted by a system of difference equations. Comparing with Newton iteration for matrix inversion, we find that the discrete-time ZNN model incorporates Newton iteration as its special case. Noticing this relation, we perform numerical comparisons on different situations of using ZNN and Newton iteration for matrix inversion. Different kinds of activation functions and different step-size values are examined for superior convergence and better stability of ZNN. Numerical examples demonstrate the efficacy of both ZNN and Newton iteration for online matrix inversion.**

***Index Terms*—Activation function, initial state, matrix inversion, Newton iteration, recurrent neural network (RNN), step size.**

I. Introduction

The PROBLEM of matrix inversion is considered to be one of the basic problems widely encountered in science and engineering fields. It is usually an essential part of many solutions, e.g., as preliminary steps for optimization [1], [2], signal processing [3], [4], electromagnetic systems [5], robotic control [6], [7], statistics [4], [8], and physics [9]. In many engineering applications, the online (or real-time) inversion of matrices is usually desired. Since the mid1980s, efforts have been directed toward computational aspects of fast matrix inversion, and many algorithms have thus been proposed [10]–[12]. Due to the in-depth research in artificial neural networks, various dynamic and analog solvers based on recurrent neural networks (RNNs) have been developed, investigated, and implemented on specific architectures [2], [3], [6], [13]–[26]. Suited for analog VLSI implementation [27] and based on potential high-speed parallel-processing capability, the neural-dynamic approach is now regarded as a powerful alternative for online problem solving [6], [7], [14]–[22], [26]–[28].

Manuscript received July 19, 2007; revised May 28, 2008 and August 10, 2008. First published November 11, 2008; current version published July 01, 2009. This work was supported by the National Science Foundation of China under Grants 60643004 and 60775050. This paper was presented in part at the 2008 IEEE International Conference on Industrial Technology, Chengdu, China, April 2008. This paper was recommended by Associate Editor A. Kuh.

The authors are with the Department of Electronics and Communication Engineering, Sun Yat-Sen University, Guangzhou 510275, China (e-mail: ynzhang@ieee.org; maweimu@mail2.sysu.edu.cn; caibh@mail2.sysu.edu.cn).

Digital Object Identifier 10.1109/TCSI.2008.2007065

Recently, a special kind of RNN has been proposed by Zhang *et al.*[6], [23]–[25] for online matrix inversion. Zhang neural network (ZNN) is designed based on a matrix-valued error function instead of a scalar-valued error function associated with gradient-based neural networks (GNNs) [1], [6], [14]–[21], [29], [30]. In addition, ZNN is depicted in an implicit dynamics instead of an explicit dynamics that usually depicts a GNN model. In this paper, we will develop and investigate a discrete-time model of ZNN for online matrix inversion. General nonlinear activation functions and different learning step sizes could be used for such a model. When the linear activation function and step size $h \equiv 1$ are used, the discrete-time ZNN model reduces exactly to Newton iteration for matrix inversion. Noticing the relation between the two approaches, we will perform numerical comparisons on different situations of using ZNN and Newton iteration for matrix inversion. Different kinds of activation functions and different step-size values will also be tested in order to investigate the convergence and stability of ZNN.

The remainder of this paper is organized as follows. Section II presents the matrix-inverse problem formulation, its online solution based on ZNN, and the link to Newton iteration. Corresponding to different kinds of activation functions, Sections III and IV present appropriate choices of initial state and step size, respectively, which could assure the convergence and stability of the discrete-time ZNN model. Section V presents the computer simulation results for online matrix inversion based on the discrete-time ZNN model. Section VI concludes this paper with final remarks.

II. ZNN

To solve for a matrix inverse, the neural-network design methods are usually based on the definition equation $AX - I = XA - I = 0$, where $A \in \mathbb{R}^{n\times n}$ is assumed nonsingular and $I \in \mathbb{R}^{n\times n}$ denotes an identity matrix.

A. Continuous-Time ZNN Model

Following the design method of Zhang *et al.*[6], [23]–[25], to monitor the matrix-inversion process, we could define the following matrix-valued error function (instead of scalar-valued error function $\|AX(t) - I\|^2/2$ associated with GNN):

$$E(X(t),t) := AX(t) - I.$$

To make $E(X(t),t)$ converge to zero (specifically, every entry $e_{ij}(t)$ converges to zero, $i,j=1,2,\ldots,n$), its time derivative $\dot{E}(X(t),t)$ could generally be chosen as

$$\frac{\mathrm{d}E(X(t),t)}{\mathrm{d}t} = -\Gamma\mathcal{F}\left(E\left(X(t),t\right)\right) \tag{1}$$

where $\Gamma \in \mathbb{R}^{n\times n}$ is a positive-definite matrix used to scale the convergence rate of the inversion process and $\mathcal{F}(\cdot):\mathbb{R}^{n\times n}\to\mathbb{R}^{n\times n}$ denotes an activation-function matrix mapping of neural networks. In general, any monotonically increasing odd activation function $f(\cdot)$, being the ijth element of matrix mapping $\mathcal{F}(\cdot)$, can be used for RNN construction. To demonstrate the idea, the following four types of $f(\cdot)$ are investigated:

1) linear activation function $f(e_{ij}) = e_{ij}$;
2) sigmoid activation function (with $\xi \geqslant 2$)

$$f(e_{ij}) = \frac{1-\exp(-\xi e_{ij})}{1+\exp(-\xi e_{ij})};$$

3) power activation function $f(e_{ij}) = e_{ij}^p$ with odd integer $p \geqslant 3$;
4) power–sigmoid activation function

$$f(e_{ij}) = \begin{cases} e_{ij}^p, & \text{if } |e_{ij}| \geqslant 1 \\ \frac{1+\exp(-\xi)}{1-\exp(-\xi)} \cdot \frac{1-\exp(-\xi e_{ij})}{1+\exp(-\xi e_{ij})}, & \text{otherwise} \end{cases}$$

with suitable design parameters $\xi \geqslant 1$ and $p \geqslant 3$.

Expanding (1) leads to the following implicit dynamic equation of ZNN:

$$A\dot{X}(t) = -\Gamma\mathcal{F}\left(AX(t)-I\right)$$

where $X(t)$, starting from the initial state $X(0) := X_0 \in \mathbb{R}^{n\times n}$, is the activation state matrix corresponding to the theoretical inverse A^{-1}. Similar to usual neural-network approaches, the matrix-valued design parameter $\Gamma \in \mathbb{R}^{n\times n}$ in the aforementioned ZNN model, being a set of reciprocals of capacitance parameters, should be set as large as the hardware permits (e.g., in analog circuits or VLSI [27]) or selected appropriately for experimental/simulative purposes. To keep the differential equations well conditioned, it is desired to keep Γ well conditioned. That is, the eigenvalues of Γ should be in the same scale. For simplicity, we assume $\Gamma = \gamma I$ with scalar-valued design parameter $\gamma > 0$. As a consequence, we could keep every $e_{ij}(t)$ $(\forall i,j=1,\ldots,n)$ converge at the same rate and at the same time, which also simplifies the neural-network design and analysis. ZNN could thus be depicted in the following specific form:

$$A\dot{X}(t) = -\gamma\mathcal{F}\left(AX(t)-I\right). \tag{2}$$

Evidently, the aforesaid continuous-time ZNN model is depicted in an implicit dynamics [23]–[25] instead of an explicit dynamics usually associated with conventional Hopfield-type neural networks [1]–[3], [6], [13]–[18], [20], [27], [29], [30]. In addition, the following theorem can be derived [6], [23]–[25].

Theorem 1: Given nonsingular matrix $A \in \mathbb{R}^{n\times n}$, if a monotonically increasing odd function array $\mathcal{F}(\cdot)$ is used, the state matrix $X(t)$ of ZNN (2), starting from any initial state $X_0 \in \mathbb{R}^{n\times n}$, converges to the theoretical inverse A^{-1} of matrix A. Moreover, ZNN (2) has the following properties.

1) If the linear activation function is used, the exponential convergence with rate γ can be achieved for (2).
2) If the sigmoid activation function is used, superior convergence can be achieved for (2) on error range $[-\varepsilon,\varepsilon]$, $\exists\varepsilon > 0$, as compared to the situation of using the linear activation function.
3) If the power activation function is used, superior convergence can be achieved for (2) on error ranges $(-\infty,-1)$ and $(1,+\infty)$ as compared to the linear situation.
4) If the power–sigmoid activation function is used, superior convergence can be achieved for the whole error range $(-\infty,+\infty)$ as compared to the linear situation.

Remark 1: For comparison, by using the negative-gradient design method, we could have a conventional GNN [3], [6], [24], [30] for matrix inversion $\dot{X}(t) = -\gamma A^T(AX(t)-I)$ and its general nonlinear form

$$\dot{X}(t) = -\gamma A^T\mathcal{F}\left(AX(t)-I\right). \tag{3}$$

The main differences between ZNN (2) and GNN (3) could be summarized as follows.

1) ZNN (2) is designed based on the elimination of every entry of matrix-valued error function $E(X) = AX(t)-I$, whereas GNN (3) is based on the elimination of norm-based scalar-valued error function $\|AX(t)-I\|^2/2$.
2) ZNN (2) is depicted generally in an implicit dynamics $A\dot{X}(t) = \cdots$, whereas GNN (3) is depicted usually in an explicit dynamics $\dot{X}(t) = \cdots$.
3) ZNN (2) is designed intrinsically for time-varying matrix inversion [23], [25], with its original dynamics being

$$A(t)\dot{X}(t) = -\dot{A}(t)X(t) - \gamma\mathcal{F}\left(AX(t)-I\right)$$

which shows that ZNN could systematically and methodically exploit the time-derivative information, and thus, its state could be guaranteed to globally exponentially converge to the exact time-varying inverse $A^{-1}(t)$. In contrast, GNN (3) is designed essentially for constant matrix inversion and could only approximately solve for the theoretical matrix inverse if exploited in the time-varying situation.

B. Discrete-Time ZNN Model

For possible hardware implementation of ZNN based on digital circuits [10], [11], [27], we could discretize the continuous-time model (2) by using the Euler forward-difference rule $\dot{X}(t) \approx (X_{k+1}-X_k)/\tau$, where $\tau > 0$ denotes the sampling time and X_k corresponds to the kth iteration/sampling of $X(t=k\tau)$, with $k=1,2,\ldots$. We could thus generate the discrete-time model of ZNN (2) as follows:

$$AX_{k+1} = AX_k - h\mathcal{F}(AX_k - I)$$

where $h = \tau\gamma > 0$ is the step size that should appropriately be selected for the convergence of X_k to the theoretical inverse

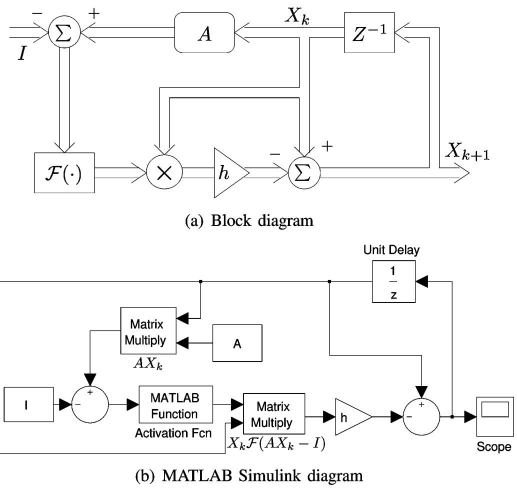


Fig. 1. Diagrams of discrete-time ZNN (5) inverting online matrix A.

A^{-1}. As matrix A is nonsingular, the aforementioned implicit discrete-time ZNN model can be rewritten as

$$X_{k+1} = X_k - hA^{-1}\mathcal{F}(AX_k - I). \quad (4)$$

In view of the convergence of state $X(t)$ to A^{-1} in the continuous-time ZNN model (2), the state of the corresponding discrete-time ZNN model $X_k = X(t = k\tau)$ could be very close to A^{-1} after an appropriate number of iterations k. As A^{-1} is actually unknown, in order to make ZNN (4) more computable, replacing A^{-1} with X_k appears to be a simple choice in generating an applicable discrete-time ZNN model. This replacement yields the following explicit difference equation of discrete-time ZNN for matrix inversion (of which the convergence of state X_k to A^{-1} could be proved in the ensuing two sections):

$$X_{k+1} = X_k - hX_k\mathcal{F}(AX_k - I). \quad (5)$$

The block diagram realizing ZNN (5) is shown in Fig. 1(a), which can be modeled and simulated via Fig. 1(b) in the MATLAB Simulink environment [31]. In view of (5) and Fig. 1, different choices of h and $\mathcal{F}(\cdot)$ may lead to different performances of such a discrete-time ZNN model.

C. Link to Newton Iteration

Look at the discrete-time ZNN model (5). When we use the linear activation function $f(e_{ij}) = e_{ij}$, ZNN (5) reduces to

$$X_{k+1} = X_k - hX_k(AX_k - I). \quad (6)$$

For $h = 1$, ZNN (6) further becomes

$$X_{k+1} = X_k - X_k(AX_k - I) = 2X_k - X_kAX_k \quad (7)$$

which is exactly Newton iteration for matrix inversion [32]. In other words, we have discovered that the general form of Newton iteration for matrix inversion could be given by the discrete-time ZNN model (5).

Remark 2: The discrete-time ZNN model (5) allows us to have many more choices of different step-size values and activation functions. In addition, it may be necessary to investigate the impact of different activation functions in the neural network, in view of the fact that nonlinear phenomena may appear, even in the hardware implementation of a linear activation function, e.g., in the form of saturation and/or inconsistency of the linear slope and in the form of truncation and round-off errors in digital realization [27], [28]. The investigation of different activation functions (such as the sigmoid, power, and power–sigmoid functions) may show us an answer to the problems and side effects of imprecise implementation and nonlinearities occurring to linear activation functions (including the situation of using Newton iteration for matrix inversion) [6], [27], [28].

It is worth mentioning that Newton iteration (7) is strongly stable in the sense that it computes a good approximation of A^{-1}, even if the computation is performed with finite precision [32]. In the ensuing sections, we will show that ZNN (5) [including (7) as a special case] is strongly stable as well, provided that $\mathcal{F}(\cdot)$ and h are appropriately chosen.

III. Choices of Initial State X_0

Theorem 2 in this section addresses the quadratic convergence of the discrete-time ZNN model (7) (namely, Newton iteration), which starts from the initial state $X_0 = \alpha A^T$, with $\alpha = 2/\mathrm{tr}(AA^T)$ for general nonsingular matrix A, or preferably starts from $X_0 = \alpha I$, with $\alpha = 2/\mathrm{tr}(A)$ for positive- or negative-definite matrix A. At the end of this section, we apply such choices of initial state X_0 to the general situation of discrete-time ZNN (5). Before presenting such results, four lemmas are given in the following with proofs in the Appendix.

Lemma 1: Given nonsingular $A \in \mathbb{R}^{n\times n}$, AA^T is positive definite and has only real positive eigenvalues [33], [34].

Lemma 2: Given nonsingular matrix $A \in \mathbb{R}^{n\times n}$, let $\lambda_{\max}(AA^T)$ and $\lambda_{\min}(AA^T)$ represent the largest and smallest eigenvalues of AA^T, respectively. If $0 < \alpha < \beta = 2/(\lambda_{\max}(AA^T) + \lambda_{\min}(AA^T))$, we have

$$1 - \frac{2}{\mathrm{cond}_2^2(A) + 1} = \|I - \beta AA^T\|_2 < \|I - \alpha AA^T\|_2 < 1 \quad (8)$$

where the two-norm of matrix A, $\|A\|_2 := \sqrt{\lambda_{\max}(AA^T)}$, and $\mathrm{cond}_2(A) = \|A\|_2\|A^{-1}\|_2 = \sqrt{\lambda_{\max}(AA^T)/\lambda_{\min}(AA^T)} \geqslant 1$ denotes the two-norm-based condition number of matrix A.

Lemma 3: Given nonsingular matrix $A \in \mathbb{R}^{n\times n}$ and $\alpha = 2/\mathrm{tr}(AA^T)$, we have

$$1 - \frac{2}{\mathrm{cond}_2^2(A) + n - 1} \leqslant q := \|\alpha AA^T - I\|_2 \leqslant 1 - \frac{2}{(n-1)\mathrm{cond}_2^2(A) + 1}. \quad (9)$$

Lemma 4: For positive- or negative-definite matrix $A \in \mathbb{R}^{n\times n}$, given $\alpha = 2/\mathrm{tr}(A)$, we have

$$1 - \frac{2}{\mathrm{cond}_2(A) + n - 1} \leqslant q := \|\alpha A - I\|_2 \leqslant 1 - \frac{2}{(n-1)\mathrm{cond}_2(A) + 1}. \quad (10)$$

Theorem 2: For general nonsingular matrix $A \in \mathbb{R}^{n\times n}$, we could choose the initial state $X_0 = \alpha A^T$, with $\alpha = 2/\mathrm{tr}(AA^T)$, for ZNN (7) which, starting with $\|AX_0 - I\|_2 < 1$, quadratically converges to the theoretical inverse A^{-1}. In addition, for positive- or negative-definite matrix A, we could choose the initial state $X_0 = \alpha I$, with $\alpha = 2/\mathrm{tr}(A)$, for (7) which, starting with a smaller value of $\|AX_0 - I\|_2$, quadratically converges to A^{-1}.

Proof: Let $E_k := AX_k - I$ denote the residual-error matrix at the kth iteration during the matrix-inverse computation. Premultiplying (7) by A yields

$$AX_{k+1} = AX_k - AX_k(AX_k - I).$$

The residual error E_{k+1} could thus be written as

$$\begin{aligned} E_{k+1} &= AX_{k+1} - I = AX_k - AX_k(AX_k - I) - I \\ &= (AX_k - I) - AX_k(AX_k - I) \\ &= E_k - (E_k + I)(E_k) = -E_k^2 \\ &= -E_0^{2^{k+1}} = -(AX_0 - I)^{2^{k+1}} \end{aligned} \tag{11}$$

which implies that sequence $\{E_k\}$ quadratically converges to zero, provided that the initial state X_0 is chosen such that $\|AX_0 - I\|_2 < 1$. This implies as well that sequence $\{X_k\}$ quadratically converges to A^{-1} if $\|AX_0 - I\|_2 < 1$.

Now, we come to choose the initial state X_0 satisfying $\|AX_0 - I\|_2 = q < 1$. It follows from Lemma 2 that $X_0 = \alpha A^T$, with $0 < \alpha \leqslant 2/(\lambda_{\max}(AA^T) + \lambda_{\min}(AA^T))$, is a reasonably good initial state that starts ZNN (7) and ensures $\lim_{k\to\infty} X_k = A^{-1}$. The larger the value of α, the better the initial state X_0 in the sense that $\|AX_0 - I\|_2$ is smaller.

However, to avoid the computation of $\lambda_{\min}(AA^T)$ and $\lambda_{\max}(AA^T)$ (which is usually time consuming), for general nonsingular matrix A, we can exploit Lemma 3 to choose $X_0 = \alpha A^T$, with $\alpha = 2/\mathrm{tr}(AA^T)$, as the initial state that starts ZNN (7) and ensures $\lim_{k\to\infty} X_k = A^{-1}$. In addition, for positive- or negative-definite matrix A, we exploit Lemma 4 to choose a better initial state $X_0 = 2I/\mathrm{tr}(A)$ that starts ZNN (7) with a smaller value of $\|AX_0 - I\|_2$ and ensures $\lim_{k\to\infty} X_k = A^{-1}$. The proof is thus complete. ■

Remark 3: If the largest and smallest eigenvalues of matrices are known or can be estimated, then we could choose the initial state $X_0 = 2A^T/(\lambda_{\max}(AA^T) + \lambda_{\min}(AA^T))$ for general nonsingular matrix A or preferably choose $X_0 = 2I/(\lambda_{\max}(A) + \lambda_{\min}(A))$ for positive- or negative-definite matrix A. These choices could have smaller values of $\|AX_0 - I\|_2$ as compared to Theorem 2. For example, in the former case, $\|AX_0 - I\|_2 = 1 - 2/(\mathrm{cond}_2^2(A) + 1)$, and in the latter case, $\|AX_0 - I\|_2 = 1 - 2/(\mathrm{cond}_2(A) + 1)$. However, it is known that eigenvalues are usually difficult to compute, and thus, Theorem 2 may be a more practical way to choose X_0.

Remark 4: There could be many other choices of α in the design of initial state $X_0 = \alpha A^T$ for general nonsingular matrix A. For example, we could set $\alpha = 1/\|A\|_1\|A\|_\infty$[32]. In addition, in the design of initial state $X_0 = \alpha I$ for positive-definite matrix A, we could set $\alpha = 1/\|A\|_1$, which generates $\|AX_0 - I\|_2 \leqslant 1 - 1/(\sqrt{n}\mathrm{cond}_2(A))$[32]. However, for circuit implementation purposes, we prefer to use $\alpha = 2/\mathrm{tr}(AA^T) = 2/\left(\sum_{i=1}^n \sum_{j=1}^n a_{ij}^2\right)$ [or $\alpha = 2/\mathrm{tr}(A) = 2/\left(\sum_{i=1}^n a_{ii}\right)$ for positive- or negative-definite matrix A], which could be implemented more easily than other norm-based choices. The lower complexity of circuit implementation of trace-based X_0 motivates Theorem 2.

It is worth mentioning that the aforementioned results can be applied to the general situation of the discrete-time ZNN model (5), which includes the linear situation (6) as a special case. Specifically speaking, to start a general discrete-time ZNN model (5), the initial state X_0 could be αA^T, with $\alpha = 2/\mathrm{tr}(AA^T)$, for inverting general nonsingular matrix A or preferably be αI, with $\alpha = 2/\mathrm{tr}(A)$, for inverting positive- or negative-definite matrix A. The analysis will be given in the ensuing section.

Remark 5: The number of iterations required for each process of matrix inversion using ZNN (7) could be estimated via the following procedure. According to (9) and (11), after k iterations, we could have the error bound

$$\|E_k\|_2 \leqslant \left(1 - \frac{2}{(n-1)\mathrm{cond}_2^2(A) + 1}\right)^{2^k}.$$

If the prescribed accuracy is $\epsilon > 0$ (i.e., with $\|E_k\|_2 \leqslant \epsilon$ required), it follows from the aforesaid inequality that the necessary number of iterations k satisfies

$$k \leqslant \log_2\left(\log_2 \epsilon \Big/ \log_2\left(1 - \frac{2}{(n-1)\mathrm{cond}_2^2(A) + 1}\right)\right).$$

For example, given nonsingular matrix $A \in \mathbb{R}^{100\times 100}$ with $\mathrm{cond}_2(A) = 100$, to achieve $\epsilon = 10^{-8}$, we know that the number of iterations $k \leqslant 22.9277$, i.e., ZNN (7) needs at most 23 iterations for inverting such a matrix A.

Remark 6: In comparison with the aforesaid full parallel-processing situation, if we prefer to use fewer processing units for matrix multiplication, we can have the following observations by citing the analysis results of [32] and the references therein. That is, let $M(n)$ processors [with $M(n) < n^3$] suffice to perform two n-by-n matrices' multiplication in $O(\log_2 n)$ parallel time, and then, for such an n-dimensional nonsingular matrix inversion, the iterative process of (7) needs $O(\log_2 n)$ iterations and totally $O(\log_2^2 n)$ parallel time [32].

IV. Choices of Step Size h

When we exploit discrete-time ZNN (5) to solve for a matrix inverse, after the initial state X_0 is chosen appropriately as in the preceding section, the next important issue to be discussed here is the appropriate choices of step size h. In the following two subsections, with respect to different activation functions, we present the rough bounds and optimal values of step size h, which are analyzed for the convergence and superior convergence of the discrete-time ZNN model (5) for matrix inversion purposes, respectively.

A. Bounds of Step Size h

Theorem 3 in this section addresses the bounds of step size h that guarantee the convergence of discrete-time ZNN (6) (namely, the linear situation) starting from the initial state X_0 given in Section III. Specifically, the bounds of step size h

in ZNN (6) are $0 < h \leqslant 1$. At the end of this section, we extend such analysis of step-size bounds to the general situation of discrete-time ZNN (5), which starts from the initial state X_0 given in Section III as well. Before presenting such main results, a lemma is given as follows.

Lemma 5: Given nonsingular matrix $A \in \mathbb{R}^{n\times n}$, for unitary matrices U and V (i.e., satisfying $U^TU = I$ and $V^TV = I$), we have $\|UA\|_2 = \|AV\|_2 = \|UAV\|_2 = \|A\|_2$[33], [34].

Theorem 3: Given nonsingular matrix $A \in \mathbb{R}^{n\times n}$, if step size h satisfies $0 < h \leqslant 1$, discrete-time ZNN (6), starting from the initial state $X_0 = \alpha A^T$, with $\alpha = 2/\mathrm{tr}(AA^T)$, will converge to the theoretical inverse A^{-1}. In addition, for positive- or negative-definite matrix A, if step size h satisfies $0 < h \leqslant 1$, discrete-time ZNN (6), starting from the initial state $X_0 = \alpha I$, with $\alpha = 2/\mathrm{tr}(A)$, will converge to the theoretical inverse A^{-1}.

Proof: Premultiplying (6) by A yields

$$AX_{k+1} = AX_k - hAX_k(AX_k - I).$$

The residual error E_{k+1} could thus be written as

$$\begin{aligned} E_{k+1} &= AX_{k+1} - I = AX_k - hAX_k(AX_k - I) - I \\ &= (AX_k - I) - hAX_k(AX_k - I) \\ &= E_k - h(E_k + I)E_k. \end{aligned} \tag{12}$$

Evidently, error matrix $E_0 = AX_0 - I$ is symmetric (i.e., $E_0^T = E_0$), because $X_0 = \alpha A^T$, with $\alpha = 2/\mathrm{tr}(AA^T)$, for general nonsingular matrix A or $X_0 = \alpha I$, with $\alpha = 2/\mathrm{tr}(A)$, for positive- or negative-definite matrix A. Then, by (12), we know that the next-iteration error matrix E_1 is also symmetric

$$\begin{aligned} E_1^T &= (E_0 - h(E_0 + I)E_0)^T = E_0^T - hE_0^T(E_0 + I)^T \\ &= E_0^T - hE_0^T\left(E_0^T + I^T\right) = E_0 - hE_0(E_0 + I) \\ &= E_0 - h\left(E_0^2 + E_0\right) = E_0 - h(E_0 + I)E_0 = E_1. \end{aligned}$$

Similarly, error matrix E_k can be proved to be symmetric for any $k = 0, 1, 2, \ldots$. Thus, E_k can be diagonalized by the following transformation: $E_k = U\Lambda_kU^T$[33], where $U \in \mathbb{R}^{n\times n}$ is a unitary matrix, with $U^T = U^{-1}$, and matrix Λ_k is diagonal. It follows from (12) that

$$\begin{aligned} E_{k+1} &= U\Lambda_kU^T - h(U\Lambda_kU^T + I)U\Lambda_kU^T \\ &= U\Lambda_kU^T - h\left(U\Lambda_k^2U^T + U\Lambda_kU^T\right) \\ &= U\Lambda_kU^T - hU\left(\Lambda_k^2 + \Lambda_k\right)U^T \\ &= U\left(\Lambda_k - h\left(\Lambda_k^2 + \Lambda_k\right)\right)U^T. \end{aligned} \tag{13}$$

Let us define diagonal matrix $\Lambda_{k+1} = \Lambda_k - h(\Lambda_k^2 + \Lambda_k)$, and thus, error matrix E_{k+1} is written as $E_{k+1} = U\Lambda_{k+1}U^T$. It follows from Lemma 5 that $\|E_{k+1}\|_2 = \|U\Lambda_{k+1}U^T\|_2 \leqslant \|E_k\|_2 = \|U\Lambda_kU^T\|_2$ if and only if $\|\Lambda_{k+1}\|_2 \leqslant \|\Lambda_k\|_2$. Thus, the convergence analysis of E_k becomes the convergence analysis of Λ_k. Now, we come to investigate $\Lambda_{k+1} = \Lambda_k - h(\Lambda_k^2 + \Lambda_k)$, with $\Lambda_k := \mathrm{diag}(\sigma_{1,k}, \ldots, \sigma_{i,k}, \ldots, \sigma_{n,k})$. Thus, the ith decoupled subsystem of $\Lambda_{k+1} = \Lambda_k - h(\Lambda_k^2 + \Lambda_k)$ is

$$\sigma_{i,k+1} = \sigma_{i,k} - h\left(\sigma_{i,k}^2 + \sigma_{i,k}\right) \tag{14}$$

with $-1 < \sigma_{i,0} < 1$ due to $\|AX_0 - I\|_2 < 1$ for $i = 1, 2, \ldots, n$ and $k = 0, 1, 2, \ldots$.

To ensure that $\|E_k\|_2$ and $\|\Lambda_k\|_2$ decrease to zero as k tends to positive infinity (i.e., $\lim_{k\to\infty} E_k = 0$ and $\lim_{k\to\infty} \Lambda_k = 0$), we could keep $|\sigma_{i,k+1}| < |\sigma_{i,k}|$ (if $\sigma_{i,k} \neq 0$) for each $i = 1, 2, \ldots, n$. That is

$$|\sigma_{i,k+1}| = \left|\sigma_{i,k} - h\left(\sigma_{i,k}^2 + \sigma_{i,k}\right)\right| < |\sigma_{i,k}|. \tag{15}$$

Noting that $-1 < \sigma_{i,0} < 1$ and $h > 0$, the solution to (15) is

$$0 < h < \frac{2}{1 + \sigma_{i,k}}, \qquad -1 < \sigma_{i,k} < 1, \qquad \forall i = 1, 2, \ldots, n.$$

Since $-1 < \sigma_{i,k} < 1$, we have $1 < 2/(1 + \sigma_{i,k})$ and $2/(1 + \sigma_{i,k}) \to 1$ when $\sigma_{i,k} \to 1$. Thus, we can take $0 < h \leqslant 1$ satisfying the aforementioned inequalities as well as convergence condition (15), which guarantees $\lim_{k\to\infty} \Lambda_k = 0$ and $\lim_{k\to\infty} E_k = \lim_{k\to\infty} U\Lambda_kU^T = 0$. From the definition of error matrix $E_k = AX_k - I$, it follows that $\lim_{k\to\infty} X_k = A^{-1}$. The proof is thus complete. ∎

Following the previous analysis procedure, we could investigate further the bounds of step size h in the general discrete-time ZNN model (5). Continuing from (13) and (14), to simplify the analysis, let us consider the residual-error matrix $E_k = AX_k - I$ in the diagonal form $E_k = \mathrm{diag}(\sigma_{1,k}, \ldots, \sigma_{i,k}, \ldots, \sigma_{n,k})$. Thus, the ith decoupled subsystem of $E_{k+1} = E_k - h(E_k + I)\mathcal{F}(E_k)$ is described as $\sigma_{i,k+1} = \sigma_{i,k} - h(\sigma_{i,k} + 1)f(\sigma_{i,k})$, $\forall i = 1, 2, \ldots, n$ and $k = 0, 1, 2, \ldots$, with $-1 < \sigma_{i,0} < 1$ due to $\|AX_0 - I\|_2 < 1$. Similarly, for the convergence of E_k, we could require

$$|\sigma_{i,k+1}| = |\sigma_{i,k} - h(\sigma_{i,k} + 1)f(\sigma_{i,k})| < |\sigma_{i,k}|. \tag{16}$$

Because $f(\cdot)$ is an odd function, $f(-\sigma_{i,k}) = -f(\sigma_{i,k})$ and

$$f(\sigma_{i,k}) \begin{cases} > 0, & \text{if } \sigma_{i,k} > 0 \\ = 0, & \text{if } \sigma_{i,k} = 0 \\ < 0, & \text{if } \sigma_{i,k} < 0. \end{cases}$$

Noting that $-1 < \sigma_{i,0} < 1$ and $h > 0$, the solution to (16) is

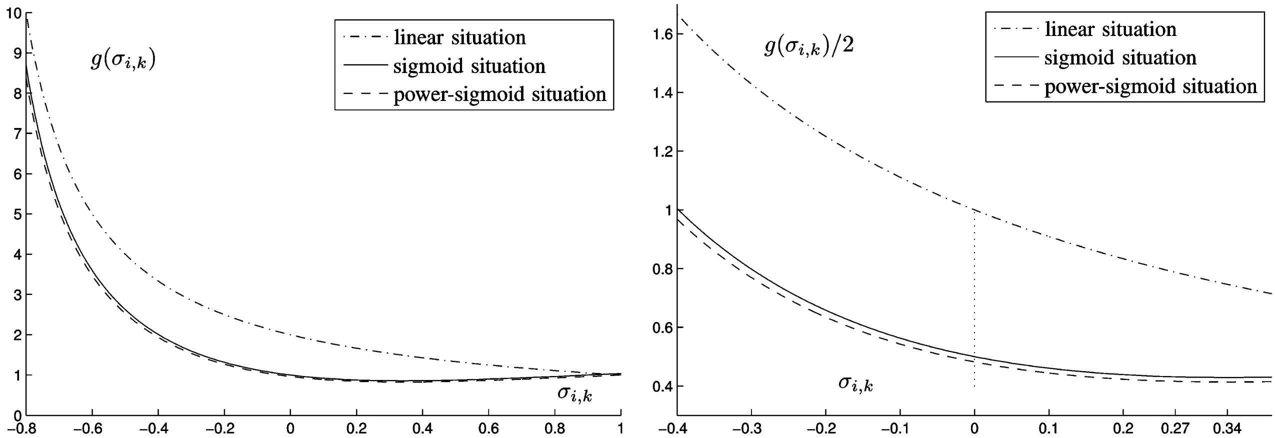
$$0 < h < \frac{2\sigma_{i,k}}{(1 + \sigma_{i,k})f(\sigma_{i,k})}, \qquad -1 < \sigma_{i,k} < 1, \qquad \forall i = 1, \ldots, n.$$

Thus, to satisfy the aforementioned inequalities as well as convergence condition (16), we can take

$$0 < h < \min_{|\sigma_{i,k}|<1} g(\sigma_{i,k}), \quad \text{with } g(\sigma_{i,k}) := \frac{2\sigma_{i,k}}{(1 + \sigma_{i,k})f(\sigma_{i,k})}$$

which ensures that $E_k \to 0$ and $X_k \to A^{-1}$ as $k \to +\infty$.

Remark 7: For monotonically increasing odd function $f(\cdot)$, the resultant function $g(\sigma_{i,k})$ is generally no longer monotonically increasing or decreasing. Therefore, $\min_{|\sigma_{i,k}|<1} g(\sigma_{i,k})$ has to be evaluated corresponding to a specific type of $f(\cdot)$. With the aid of Fig. 2(a), the bounds of step size h can be listed hereinafter for the diagonal situation, within which the general



(a) Profile of $g(\sigma_{i,k})$ over $\sigma_{i,k}$ (b) Profile of $g(\sigma_{i,k})/2$ over $\sigma_{i,k}$

Fig. 2. Profiles of $g(\sigma_{i,k})$ and $g(\sigma_{i,k})/2$ resulting from different activation function $f(\cdot)$.

discrete-time ZNN model (5), starting from the initial state X_0 satisfying $\|AX_0 - I\|_2 = q < 1$, converges to A^{-1}.

1) When the linear activation function $f(e_{ij}) = e_{ij}$ is used, the bound is $0 < h \leqslant 1$, the same as in Theorem 3.
2) When the sigmoid activation function is used with $\xi = 4$, the minimum value of $g(\sigma)$, namely, $\min_{|\sigma_{i,k}|<1} g(\sigma_{i,k}) = 0.8579$, is achieved, with $\sigma_{i,k} = 0.3416$. Thus, the bound of step size h in this situation is $0 < h < 0.8579$.
3) When the power activation function $f(e_{ij}) = e_{ij}^p$ is used with odd integer $p \geqslant 3$, the minimum value of $g(\sigma_{i,k})$, namely, $\min_{|\sigma_{i,k}|<1} g(\sigma_{i,k}) = 1$, is achieved, with $\sigma_{i,k} \to 1$. Thus, the bound of step size h in this case is $0 < h \leqslant 1$.
4) When the power–sigmoid activation function is used with $\xi = 4$ and $p = 3$, the minimum value $\min_{|\sigma_{i,k}|<1} g(\sigma_{i,k}) = 0.8270$ is achieved, with $\sigma_{i,k} = 0.3416$. The bound in this situation is thus $0 < h < 0.8270$.
5) When other types of activation functions are used, the bounds of step size h can be obtained in a similar way.

Note that, for the general nonsingular situation, the upper bounds of step size h for (5) are usually smaller than that in the previous (except for the situation of using the linear activation function).

B. Optimal or Nearly Optimal Value of Step Size h

For the superior convergence of the discrete-time ZNN model (5), it is desired to find the optimal (or nearly optimal) value of step size h in order to minimize $\max_{1\leqslant i\leqslant n} |\sigma_{i,k+1}|$ of (16)

$$\begin{aligned} h^o &= \arg\min_h \max_{1\leqslant i\leqslant n} |\sigma_{i,k+1}| \\ &= \arg\min_h \max_{1\leqslant i\leqslant n} |\sigma_{i,k} - h(\sigma_{i,k}+1)f(\sigma_{i,k})|. \end{aligned}$$

The ideal situation of h^o is to make $|\sigma_{i,k+1}| = 0$, i.e., $|\sigma_{i,k} - h(\sigma_{i,k}+1)f(\sigma_{i,k})| = 0\ \forall\, i = 1, 2, \ldots, n$. However, it is difficult to determine h^o from the aforementioned relation since the values of $\sigma_{i,k}$, $i = 1, 2, \ldots, n$, are unknown. Evidently, if step size h is too small, the convergence process may be very slow, whereas if h is too large, the neural network may have zigzag trajectories or even diverge. Typical trajectories of state matrix X_k of ZNN (5) are shown in Fig. 3, where different values of step size h are tried, in addition to using sigmoid activation functions. Specifically speaking, if $h = 0.3$, the convergence is slow, and 25 iterations are needed to generate the solution. If $h = 0.5$, ZNN (5) converges smoothly to the solution within 14 iterations. In comparison, if $h = 0.8$, the neural-network solution becomes zigzag.

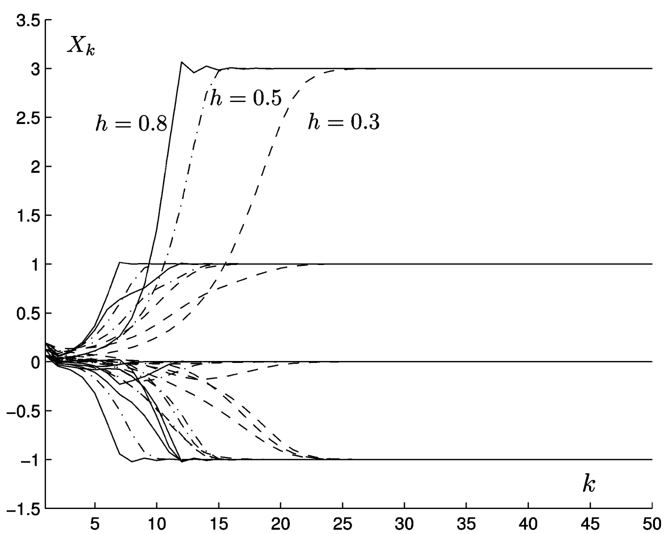


Fig. 3. State trajectories of ZNN (5) using sigmoid activation functions with different values of step size h for online matrix inversion.

To approach the ideal situation of finding h^o, we need to make $|\sigma_{i,k+1}| = |\sigma_{i,k} - h(\sigma_{i,k}+1)f(\sigma_{i,k})| = 0$, i.e.,

$$h = h(\sigma_{i,k}) = \frac{\sigma_{i,k}}{(\sigma_{i,k}+1)f(\sigma_{i,k})} = \frac{g(\sigma_{i,k})}{2}. \tag{17}$$

Note that the value of $\sigma_{i,k}$ for $i = 1, 2, \ldots, n$ could be different from each other, and thus, the value of h is theoretically different with respect to different $\sigma_{i,k}$. If $\|E_k\|_2 = q$, then $0 \leqslant |\sigma_{i,k}| \leqslant q$ holds for each $i = 1, 2, \ldots, n$. In view of Fig. 2(b), if $\|E_k\|_2 =$

q is small enough (e.g., with $q < 0.3$), then $\forall\, \sigma_{i,k}$, with $i = 1, 2, \ldots, n$

$$\frac{q}{(q+1)f(q)} = \frac{g(q)}{2} \leqslant h(\sigma_{i,k}) \leqslant \frac{q}{(-q+1)f(q)} = \frac{g(-q)}{2} \tag{18}$$

in the context of using linear, sigmoid (with $\xi = 4$), or power–sigmoid (with $\xi = 4$ and $p = 3$) activation functions. It follows from the convergence result (i.e., $q \to 0$) that bound (18) shrinks finally to $h = h(\sigma_{i,k}) = g(0)/2$, $\forall\, \sigma_{i,k}$, with $i = 1, 2, \ldots, n$, which could be exploited as the practical substitute for optimal step size h^o. In addition, $h^o \approx h := g(0)/2$ agrees relatively well with Fig. 3, which was already mentioned in the preceding paragraph. With respect to different types of activation functions, the detailed discussion on the optimal (or nearly optimal) choices of h is given in the following.

1) When the linear activation function $f(e_{ij}) = e_{ij}$ is used in ZNN (5) [i.e., linear situation (6)], we have $h^o \approx h := g(0)/2 = 1$, which also satisfies the bound $0 < h \leqslant 1$ given in Theorem 3.
2) When the sigmoid activation function is used in ZNN (5) with $\xi = 4$, we have $h^o \approx h := g(0)/2 = 0.5$, satisfying the rough bound $0 < h < 0.8579$ discussed in Remark 7.
3) When the power–sigmoid activation function is used in ZNN (5) with $\xi = 4$ and $p = 3$, we have $h^o \approx h := g(0)/2 = 0.4820$, which also satisfies the rough bound $0 < h < 0.8270$ discussed in Remark 7.

V. Computer Simulation Results

In the previous sections, we have presented the theoretical results about the discrete-time ZNN model (5), in addition to showing that the general nonlinear form of Newton iteration for matrix inversion is given as in (5). In this section, we present several numerical illustrative examples, which could substantiate further these theoretical results.

A. Example 1

For illustration and comparison purposes, we are now applying ZNN (5) [including Newton iteration (7) as a special case] to compute the inverse of the following matrix $A \in \mathbb{R}^{3\times 3}$[35]:

$$A = \begin{bmatrix} 1 & 1 & 2 \\ 3 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix}.$$

Different activation function mapping $\mathcal{F}(\cdot)$ and step size h are tried in the ZNN simulation. Starting from the initial state $X_0 = 2A^T/\mathrm{tr}(AA^T)$, discrete-time ZNN (5) could generate the inverse of A denoted by X^* within a few iterations

$$X^* = \begin{bmatrix} -1.0000 & 1.0000 & -1.0000 \\ 0.0000 & -1.0000 & 3.0000 \\ 1.0000 & 0.0000 & -1.0000 \end{bmatrix}$$

rounded to four decimal places. We have the following facts.

1) The X_k trajectories of ZNN (5) using sigmoid activation functions with different step size h are shown in Fig. 3. It shows that, by using optimal step size $h = 0.5$, the state of ZNN (5) converges to X^* within 14 iterations.
2) The situation of using other types of activation functions to construct ZNN (5) is simulated and depicted as well, e.g., in Fig. 4, where the trajectories of solution error $\|X_k - A^{-1}\|_2$ are shown. We can see that, by using linear activation functions with step size $h = 1$, sigmoid activation functions with $h = 0.7$, or power–sigmoid activation functions with $h = 0.7$, the states of ZNN (5) could all converge to X^* within 14 iterations.
3) For comparison with GNNs, two GNN models [with one similar to the linear discrete-time model of GNN (3)] were applied to solving the same matrix-inverse problem as that presented previously, which, however, took roughly 1000 iterations [35].

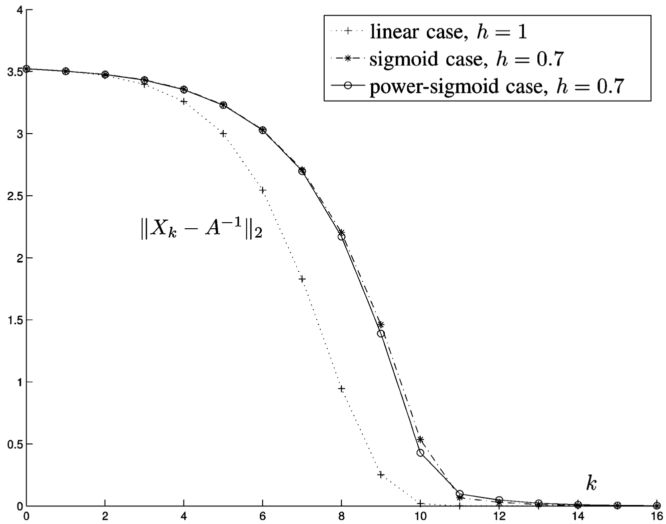


Fig. 4. Solution-error trajectories of ZNN (5) for online matrix inversion.

B. Example 2

In this example, we would like to compare the performance of ZNN (5) when using different kinds of activation functions and/or different values of step size. To do so, four groups of 1000 nonsingular random matrices $A \in \mathbb{R}^{3\times 3}$ are tested. Note that the matrix entries are the uniformly distributed random numbers in $[-5, 5]$ generated by using MATLAB [31], and the initial state of ZNN (5) is chosen as $X_0 = 2A^T/\mathrm{tr}(AA^T)$. The test results are shown in Table I, where the following notations are introduced.

1) The results related to ZNN (5) using linear activation functions with step size $h = 1$ (namely, Newton iteration) are denoted by $\mathrm{ZNN}_{\mathrm{lin}}^{1.0}$.
2) The results related to ZNN (5) using sigmoid activation functions with $h = 0.50$ are denoted by $\mathrm{ZNN}_{\mathrm{sig}}^{0.50}$.
3) The results related to ZNN (5) using power–sigmoid functions with $h = 0.60$ are denoted by $\mathrm{ZNN}_{\mathrm{ps}}^{0.60}$.

As seen from Table I, when the prescribed accuracy becomes higher, better performance is achieved by using linear activation functions with optimal step size $h = 1$. On the other hand, when using sigmoid activation functions, better performance can be

TABLE I
COMPARISONS ON ZNN PERFORMANCE FOR INVERTING ONLINE FOUR GROUPS OF 1000 RANDOM MATRICES

	Average number of iterations							
Prescribed accuracy	$\mathrm{ZNN}_{\mathrm{lin}}^{0.80}$	$\mathrm{ZNN}_{\mathrm{lin}}^{1.0}$	$\mathrm{ZNN}_{\mathrm{sig}}^{0.45}$	$\mathrm{ZNN}_{\mathrm{sig}}^{0.50}$	$\mathrm{ZNN}_{\mathrm{sig}}^{0.60}$	$\mathrm{ZNN}_{\mathrm{ps}}^{0.45}$	$\mathrm{ZNN}_{\mathrm{ps}}^{0.482}$	$\mathrm{ZNN}_{\mathrm{ps}}^{0.60}$
$\|AX_k - I\|_2 < 10^{-12}$	24.4970	10.800	21.402	12.872	22.884	19.265	13.876	24.761
$\|AX_k - I\|_2 < 10^{-8}$	18.9150	10.038	17.496	11.949	17.076	15.776	12.158	17.917
$\|AX_k - I\|_2 < 10^{-4}$	13.2880	9.104	13.153	10.875	11.224	12.754	11.224	11.919
$\|AX_k - I\|_2 < 10^{-2}$	10.2560	8.133	11.195	10.359	8.430	10.919	10.153	8.548

TABLE II
PERFORMANCE COMPARISON ON ZNN (5) STARTING FROM DIFFERENT INITIAL STATES TO INVERT ONLINE POSITIVE-DEFINITE MATRICES

	Average number of iterations	
Prescribed accuracy	$X_0 = 2A^T/\operatorname{tr}(AA^T)$	$X_0 = 2I/\operatorname{tr}(A)$
$\|AX_k - I\|_2 < 10^{-12}$	25.639	19.015
$\|AX_k - I\|_2 < 10^{-8}$	15.634	9.994
$\|AX_k - I\|_2 < 10^{-4}$	14.795	9.066
$\|AX_k - I\|_2 < 10^{-2}$	13.943	8.167

achieved with step size $h = 0.5$, and when using power–sigmoid activation functions, step size $h = 0.482$ works better. Note that if the prescribed accuracy is not very high, ZNN (5) with larger values of h may perform better.

C. Example 3

This example compares the ZNN performance by using different initial states for positive-definite matrix inversion. To do so, four groups of 1000 positive-definite random matrices $A \in \mathbb{R}^{3\times 3}$ are tested, and the matrix entries are uniformly distributed in $[-5, 5]$ as well. According to the theoretical analysis presented in the previous sections, the initial state X_0 of ZNN (5) could be chosen as $2A^T/\operatorname{tr}(AA^T)$ or, preferably, $X_0 = 2I/\operatorname{tr}(A)$. The testing results are shown in Table II, which could summarize that $X_0 = 2I/\operatorname{tr}(A)$ is a better initial state for ZNN (5) to invert positive-definite matrices with fewer iterations. These substantiate the theoretical analysis for positive-definite matrix inversion (i.e., Theorems 2 and 3).

D. Example 4

In this example, we compare the ZNN performance on inverting matrices with different condition numbers. To do so, four groups of 1000 nonsingular random matrices $A \in \mathbb{R}^{3\times 3}$ with entries in $[-5, 5]$ are tested. The initial state of ZNN (5) is chosen as $X_0 = 2A^T/\operatorname{tr}(AA^T)$, and the prescribed accuracy is 10^{-4}. Different types of activation functions with their corresponding optimal (or nearly optimal) values of step size h are used. Simulation results are shown in Table III, from which we can see that the closer to one the condition number, the better the performance the ZNN model can achieve. These substantiate the theoretical results about the number of iterations presented in Section III (particularly Remark 5). In addition, even for the situation with a large condition number [e.g., $\operatorname{cond}_2(A) > 10\,000$], the matrix inverse can be obtained by ZNN (5) with a relatively small number of iterations (e.g., with about 50 iterations).

TABLE III
PERFORMANCE COMPARISON ON ZNN (5) INVERTING ONLINE FOUR GROUPS OF 1000 RANDOM MATRICES WITH DIFFERENT CONDITION NUMBERS

	Average number of iterations		
Condition-number range	$\mathrm{ZNN}_{\mathrm{lin}}^{1.0}$	$\mathrm{ZNN}_{\mathrm{sig}}^{0.50}$	$\mathrm{ZNN}_{\mathrm{ps}}^{0.482}$
$[1, 100)$	8.680	10.362	10.362
$[100, 1000)$	18.539	25.830	25.832
$[1000, 10000)$	25.206	37.279	37.283
$[10000, +\infty)$	32.482	50.134	50.135

TABLE IV
PERFORMANCE COMPARISON ON ZNN (5) INVERTING ONLINE FOUR GROUPS OF 1000 RANDOM 100-DIMENSIONAL MATRICES

	Average number of iterations		
Prescribed accuracy	$\mathrm{ZNN}_{\mathrm{lin}}^{1.0}$	$\mathrm{ZNN}_{\mathrm{sig}}^{0.50}$	$\mathrm{ZNN}_{\mathrm{ps}}^{0.482}$
$\|AX_k - I\|_2 < 10^{-12}$	32.785	36.163	36.490
$\|AX_k - I\|_2 < 10^{-8}$	25.956	29.345	29.369
$\|AX_k - I\|_2 < 10^{-4}$	24.887	28.259	28.260
$\|AX_k - I\|_2 < 10^{-2}$	23.746	27.154	27.156

E. Example 5

To show the ZNN performance on larger dimensional matrix inversion (as compared to the previous examples), four groups of 1000 nonsingular random matrices $A \in \mathbb{R}^{100\times 100}$ with entries in $[-5, 5]$ are tested. The initial state of ZNN (5) is also chosen as $X_0 = 2A^T/\operatorname{tr}(AA^T)$, and different types of activation functions with their corresponding optimal (or nearly optimal) values of step size h are used. The testing results are shown in Table IV, from which we can see that the prescribed accuracy can be achieved by using the ZNN model with a relatively small number of iterations (e.g., with about 37 iterations). This example substantiates that ZNN (5) could work well for larger dimensional matrix inversion as compared to the previous 3-D examples.

F. Example 6

In this example, we evaluate the ZNN performance in the context of imprecise implementation (i.e., with environmental noises). For this evaluating purpose, random noises (with their bounds specified in the first column of Table V) are added to the right-hand side of the ZNN model (5). The initial state of ZNN (5) is still chosen as $X_0 = 2A^T/\operatorname{tr}(AA^T) \in \mathbb{R}^{3\times 3}$, and the evaluating results are shown in Table V. As we can see, when ZNN (5) is imprecisely implemented and used, the matrix inverse can be generated with the prescribed accuracy 10^{-3} for most of

TABLE V
ZNN PERFORMANCE FOR ONLINE INVERTING 1000 RANDOM MATRICES IN NOISY IMPLEMENTATION ENVIRONMENT

Noise bounds		$\text{ZNN}_{\text{lin}}^{1.0}$	$\text{ZNN}_{\text{sig}}^{0.50}$	$\text{ZNN}_{\text{ps}}^{0.482}$
$\pm 10.0 \times 10^{-5}$	Average iterations	7.544	9.299	9.300
	Successful number	847	853	853
$\pm 7.5 \times 10^{-5}$	Average iterations	8.505	10.350	10.352
	Successful number	981	984	984
$\pm 5.0 \times 10^{-5}$	Average iterations	8.737	10.708	10.708
	Successful number	1000	1000	1000
$\pm 2.5 \times 10^{-5}$	Average iterations	8.604	10.492	10.493
	Successful number	1000	1000	1000

the tests. For example, about 850 out of 1000 tests could succeed in achieving the accuracy, even if the added noises are on $[-10^{-4}, 10^{-4}]$. In addition, if the noises are small enough (e.g., within $[-5 \times 10^{-5}, 5 \times 10^{-5}]$), the imprecisely implemented ZNN model (5) could always successfully converge to the inverse with the prescribed accuracy 10^{-3} satisfied. These simulation results substantiate the analysis and observations presented in Section II-C about the imprecise implementation of neural networks.

Moreover, before ending this discussion, it is worth providing the following supplements about nonlinear and linear methods (i.e., using nonlinear and linear activation functions).

1) As shown in Theorem 1, nonlinear methods work better in continuous-time ZNN models as compared to the linear method.
2) As discussed further in Remark 2, nonlinear phenomena may appear in the hardware implementation of the linear method (e.g., Newton's method), and thus, the investigation of nonlinear methods/phenomena becomes more necessary and practical than considering the linear method alone.
3) As we can see from Table V, for the imprecisely implemented ZNN model, nonlinear methods could have more successful tests than the linear method, e.g., 853 versus 847 over 1000 tests with noise bound $\pm 10^{-4}$.

VI. Concluding Remarks

We have developed and investigated the discrete-time ZNN model for online matrix inversion. Such a discrete-time ZNN model could be regarded as a generalized form of Newton iteration for matrix inversion. Corresponding to the use of different types of activation functions, the initial state and step size were analyzed for such a neural network so as to assure superior convergence and better stability. Computer simulation and comparison were presented as well, which have demonstrated the efficacy of the discrete-time ZNN models (including Newton iteration as a special linear case with $h = 1$) for online matrix inversion.

Appendix

Proof of Lemma 2: We could first start from the proof of the rightmost part of (8). For ease of writing, we abbreviate here $\lambda_{\max}(AA^T)$ and $\lambda_{\min}(AA^T)$ as $\lambda_{\max}$ and $\lambda_{\min}$, respectively. Let $\mu_i \in \mathbb{R}$ and $\xi \in \mathbb{R}^n$ denote the ith eigenvalue and its corresponding eigenvector of $I - \alpha AA^T$, respectively, for $i = 1, 2, \ldots, n$. That is, $(I - \alpha AA^T)\xi = \xi - \alpha AA^T\xi = \mu_i\xi$ holds true. We thus have $AA^T\xi = ((1-\mu_i)/\alpha)\xi := \lambda_i\xi$, where $\lambda_i = (1-\mu_i)/\alpha$ and ξ are the ith eigenvalue and eigenvector of AA^T, respectively. In view of A being nonsingular and Lemma 1, we have $\lambda_{\max} \geqslant \lambda_i \geqslant \lambda_{\min} > 0\ \forall i \in \{1, \ldots, n\}$. It is known that, for symmetric matrix $I - \alpha AA^T$

$$\|I - \alpha AA^T\|_2 = \max_{1 \leqslant i \leqslant n} |\mu_i| = \max_{1 \leqslant i \leqslant n} (|1 - \alpha\lambda_i|). \quad (19)$$

For the purpose of analyzing the aforementioned equation, it follows from $0 < \alpha < \beta = 2/(\lambda_{\max} + \lambda_{\min})$ that

$$\begin{aligned} 0 < \alpha\lambda_{\max} + \alpha\lambda_{\min} < 2 \\ \alpha\lambda_{\min} - 1 < 1 - \alpha\lambda_{\max} \leqslant 1 - \alpha\lambda_{\min} < 1 \\ |1 - \alpha\lambda_{\max}| \leqslant 1 - \alpha\lambda_{\min} < 1. \end{aligned}$$

From (19) and the aforementioned inequalities, we have

$$\begin{aligned} \|I - \alpha AA^T\|_2 &= \max_{1 \leqslant i \leqslant n} (|1 - \alpha\lambda_i|) \\ &= \max(|1 - \alpha\lambda_{\min}|, |1 - \alpha\lambda_{\max}|) \\ &= 1 - \alpha\lambda_{\min} < 1 \end{aligned}$$

which completes the proof of the rightmost part of (8).

Second, following the aforesaid result, we can readily complete the proof of the left part of (8) by considering $0 < \alpha < \beta = 2/(\lambda_{\max} + \lambda_{\min})$

$$\begin{aligned} \|I - \alpha AA^T\|_2 &= 1 - \alpha\lambda_{\min} > 1 - \beta\lambda_{\min} = \|I - \beta AA^T\|_2 \\ &= 1 - \frac{2\lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} = 1 - \frac{2}{\lambda_{\max}/\lambda_{\min} + 1} \\ &= 1 - \frac{2}{\text{cond}_2^2(A) + 1}. \end{aligned}$$

The proof of Lemma 2 is thus complete. ■

Proof of Lemma 3: Following the notations used in the proof of Lemma 2, we know that $\text{tr}(AA^T) = \sum_{i=1}^n \lambda_i \geqslant \lambda_{\max} + \lambda_{\min}$, and thus, $0 < \alpha = 2/\text{tr}(AA^T) \leqslant 2/(\lambda_{\max} + \lambda_{\min})$. It then follows from the proof of Lemma 2 that

$$\|\alpha AA^T - I\|_2 = 1 - \alpha\lambda_{\min} = 1 - \frac{2\lambda_{\min}}{\text{tr}(AA^T)} < 1. \quad (20)$$

By deriving the lower bound of $\text{tr}(AA^T)/\lambda_{\min}$ as

$$\begin{aligned} \frac{\text{tr}(AA^T)}{\lambda_{\min}} &= \sum_{k=1}^n \frac{\lambda_k}{\lambda_{\min}} \geqslant \frac{\lambda_{\max}}{\lambda_{\min}} + \sum_{i=1}^{n-1} \frac{\lambda_{\min}}{\lambda_{\min}} \\ &= \text{cond}_2^2(A) + n - 1 \end{aligned}$$

and the upper bound of $\text{tr}(AA^T)/\lambda_{\min}$ as

$$\begin{aligned} \frac{\text{tr}(AA^T)}{\lambda_{\min}} &= \sum_{k=1}^n \frac{\lambda_k}{\lambda_{\min}} \leqslant \sum_{i=1}^{n-1} \frac{\lambda_{\max}}{\lambda_{\min}} + \frac{\lambda_{\min}}{\lambda_{\min}} \\ &= (n-1)\text{cond}_2^2(A) + 1 \end{aligned}$$

we thus have

$$\frac{2}{(n-1)\text{cond}_2^2(A) + 1} \leqslant \frac{2\lambda_{\min}}{\text{tr}(AA^T)} \leqslant \frac{2}{\text{cond}_2^2(A) + n - 1}$$

which, together with (20), gives rise to (9) and thus completes the proof of Lemma 3. ■

Proof of Lemma 4: For positive-definite matrix A, we use here λ_i, $\lambda_{\max}$, and $\lambda_{\min}$ to denote the ith, largest, and smallest eigenvalues of A, respectively. We know that $\mathrm{tr}(A) = \sum_{i=1}^{n}\lambda_i \geqslant \lambda_{\max}+\lambda_{\min}$, and thus, $0<\alpha=2/\mathrm{tr}(A)\leqslant 2/(\lambda_{\max}+\lambda_{\min})$. It then follows from the proof of Lemma 2 that

$$\|\alpha A - I\|_2 = 1-\alpha\lambda_{\min} = 1-\frac{2\lambda_{\min}}{\mathrm{tr}(A)} < 1. \tag{21}$$

Similar to the proof of Lemma 3, we could have

$$\frac{2}{(n-1)\mathrm{cond}_2(A)+1} \leqslant \frac{2\lambda_{\min}}{\mathrm{tr}(A)} \leqslant \frac{2}{\mathrm{cond}_2(A)+n-1}$$

which, together with (21), gives rise to (10) and thus completes the proof of Lemma 4. ∎

ACKNOWLEDGMENT

The authors would like to thank the editors and the anonymous reviewers for their time and efforts in providing many constructive and inspiring comments.

REFERENCES

[1] Y. Zhang, "Towards piecewise-linear primal neural networks for optimization and redundant robotics," in *Proc. IEEE Int. Conf. Netw. Sens. Control*, Ft. Lauderdale, FL, 2006, pp. 374–379.

[2] Y. Zhang and J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Phys. Lett. A*, vol. 298, no. 4, pp. 271–278, Jun. 2002.

[3] R. J. Steriti and M. A. Fiddy, "Regularized image reconstruction using SVD and a neural network method for matrix inversion," *IEEE Trans. Signal Proces.*, vol. 41, no. 10, pp. 3074–3077, Oct. 1993.

[4] Y. Zhang, W. E. Leithead, and D. J. Leith, "Time-series Gaussian process regression based on Toeplitz computation of $O(N^2)$ operations and $O(N)$ -level storage," in *Proc. 44th IEEE Conf. Decision Control*, Seville, Spain, 2005, pp. 3711–3716.

[5] T. Sarkar, K. Siarkiewicz, and R. Stratton, "Survey of numerical methods for solution of large systems of linear equations for electromagnetic field problems," *IEEE Trans. Antennas Propag.*, vol. 29, no. 6, pp. 847–856, Nov. 1981.

[6] Y. Zhang, "Revisit the analog computer and gradient-based neural system for matrix inversion," in *Proc. IEEE Int. Symp. Intell. Control*, Limassol, Cyprus, 2005, pp. 1411–1416.

[7] R. H. Sturges, Jr., "Analog matrix inversion (robot kinematics)," *IEEE J. Robot. Autom.*, vol. 4, no. 2, pp. 157–162, Apr. 1988.

[8] W. E. Leithead and Y. Zhang, "$O(N^2)$-operation approximation of covariance matrix inverse in Gaussian process regression based on quasi-Newton BFGS methods," *Commun. Stat.—Simul. Comput.*, vol. 36, no. 2, pp. 367–380, Mar. 2007.

[9] S. Van Huffel and J. Vandewalle, "Analysis and properties of the generalized total least squares problem $AX \approx B$ when some or all columns in A are subject to errors," *SIAM J. Matrix Anal. Appl.*, vol. 10, pp. 294–315, Jul. 1989.

[10] K. S. Yeung and F. Kumbi, "Symbolic matrix inversion with application to electronic circuits," *IEEE Trans. Circuits Syst.*, vol. 35, no. 2, pp. 235–238, Feb. 1988.

[11] A. El-Amawy, "A systolic architecture for fast dense matrix inversion," *IEEE Trans. Comput.*, vol. 38, no. 3, pp. 449–455, Mar. 1989.

[12] Y. Q. Wang and H. B. Gooi, "New ordering methods for space matrix inversion via diagonalization," *IEEE Trans. Power Syst.*, vol. 12, no. 3, pp. 1298–1305, Aug. 1997.

[13] J. Xu, D. Pi, Y.-Y. Cao, and S. Zhong, "On stability of neural networks by a Lyapunov functional-based approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 4, pp. 912–924, Apr. 2007.

[14] R. K. Manherz, B. W. Jordan, and S. L. Hakimi, "Analog methods for computation of the generalized inverse," *IEEE Trans. Autom. Control*, vol. AC-13, no. 5, pp. 582–585, Oct. 1968.

[15] X. Xue and W. Bian, "Subgradient-based neural networks for nonsmooth convex optimization problems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 8, pp. 2378–2391, Sep. 2008.

[16] D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction*. Chichester, U.K.: Wiley, 2001.

[17] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals Through Simulations*. Chichester, U.K.: Wiley, 2000.

[18] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Chichester, U.K.: Wiley, 1998.

[19] J.-S. Wang and Y.-P. Chen, "A fully automated recurrent neural network for unknown dynamic system identification and control," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 6, pp. 1363–1372, Jun. 2006.

[20] J. J. Rubio and W. Yu, "Stability analysis of nonlinear system identification via delayed neural networks," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 2, pp. 161–165, Feb. 2007.

[21] S. J. Yoo, Y. H. Choi, and J. B. Park, "Generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: Adaptive learning rates approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 6, pp. 1381–1394, Jun. 2006.

[22] G. Setti, P. Thiran, and C. Serpico, "An approach to information propagation in 1-D cellular neural networks—Part II: Global propagation," *IEEE Trans. Circuits Syst. I, Fundam. Theory and Appl.*, vol. 45, no. 8, pp. 790–811, Aug. 1998.

[23] 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, Sep. 2002.

[24] Y. Zhang and S. S. Ge, "A general recurrent neural network model for time-varying matrix inversion," in *Proc. 42nd IEEE Conf. Decision Control*, HI, Maui, 2003, pp. 6169–6174.

[25] 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, Nov. 2005.

[26] H. Zhang, Z. Wang, and D. Liu, "Global asymptotic stability and robust stability of a class of Cohen–Grossberg neural networks with mixed delays," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 3, pp. 616–629, Mar. 2009.

[27] C. Mead, *Analog VLSI and Neural Systems*. Reading, MA: Addison-Wesley, 1989.

[28] N. C. F. Carneiro and L. P. Caloba, "A new algorithm for analog matrix inversion," in *Proc. 38th Midwest Symp. Circuits Syst.*, Rio de Janeiro, Brazil, 1995, vol. 1, pp. 401–404.

[29] J. Jang, S. Lee, and S. Shin, "An optimization network for matrix inversion," in *Neural Information Processing and Computing*. New York: Amer. Inst. Phys., 1988, pp. 397–401.

[30] J. Wang, "A recurrent neural network for real-time matrix inversion," *Appl. Math. Comput.*, vol. 55, no. 1, pp. 89–100, Apr. 1993.

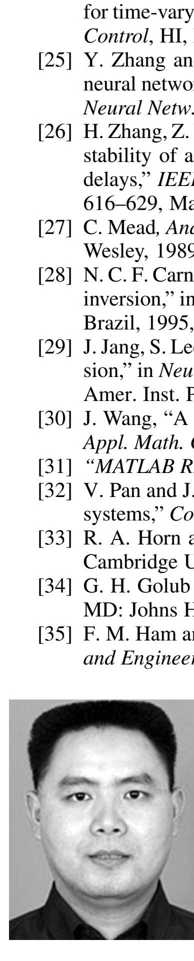
[31] *"MATLAB R2007a,"* MathWorks Inc., Natick, MA, 2007.

[32] V. Pan and J. Reif, "Fast and efficient parallel solution of dense linear systems," *Comput. Math. Appl.*, vol. 17, no. 11, pp. 1481–1491, 1989.

[33] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

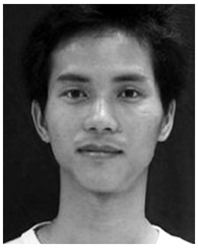
[34] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1996.

[35] F. M. Ham and I. Kostanic, *Principles of Neurocomputing for Science and Engineering*. New York: McGraw-Hill, 2001.



Yunong Zhang (S'02–M'03) was born in Henan, China, in 1973. He received the B.S. degree from Huazhong University of Science and Technology, Wuhan, China, in 1996, the M.S. degree from South China University of Technology, Guangzhou, China, in 1999, and the Ph.D. degree from The Chinese University of Hong Kong, Shatin, Hong Kong, in 2003.

He is currently a Professor with the School of Information Science and Technology, Sun Yat-Sen University (SYSU), Guangzhou. Before joining SYSU in 2006, he had been with the National University of Ireland, Maynooth, the University of Strathclyde, Glasgow, U.K., and the National University of Singapore, Singapore, since 2003. His main research interests include neural networks, robotics, and Gaussian processes.



Weimu Ma was born in Guangdong, China, in 1983. He received the B.S. degree in applied mathematics and the M.S. degree in measuring technology and automated equipment from Sun Yat-Sen University, Guangzhou, China, in 2006 and 2008, respectively.

He is currently with the Department of Electronics and Communication Engineering, Sun Yat-Sen University. His current research interests include neural networks, nonlinear systems, machine learning, and their hardware implementation.

Binghuang Cai was born in Guangdong, China, in 1981. He received the B.S. degree in electronic information engineering and the M.S. degree in signal and information processing from Shantou University, Shantou, China, in 2004 and 2007, respectively. He is currently working toward the Ph.D. degree in communication and information systems in the School of Information Science and Technology, Sun Yat-Sen University, Guangzhou, China.

His current research interests include neural networks, robotics, and intelligent information processing.