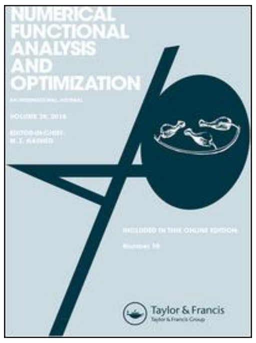

Numerical Functional Analysis and Optimization

ISSN: 0163-0563 (Print) 1532-2467 (Online) Journal homepage: https://www.tandfonline.com/loi/lnfa20

Computing Time-Varying ML-Weighted Pseudoinverse by the Zhang Neural Networks

Sanzheng Qiao, Yimin Wei & Xuxin Zhang

To cite this article: Sanzheng Qiao, Yimin Wei & Xuxin Zhang (2020): Computing Time-Varying ML-Weighted Pseudoinverse by the Zhang Neural Networks, Numerical Functional Analysis and Optimization, DOI: 10.1080/01630563.2020.1740887

To link to this article: https://doi.org/10.1080/01630563.2020.1740887

Published online: 17 Mar 2020.

NUMERICAL FUNCTIONAL ANALYSIS AND OPTIMIZATION
https://doi.org/10.1080/01630563.2020.1740887

Computing Time-Varying ML-Weighted Pseudoinverse by the Zhang Neural Networks

Sanzheng Qiao[a], Yimin Wei[b], and Xuxin Zhang[c]

[a]Department of Computing and Software, McMaster University, Hamilton, Canada; [b]School of Mathematical Sciences and Shanghai Key Laboratory of Contemporary Applied Mathematics, Fudan University, Shanghai, P. R. of China; [c]School of Mathematical Sciences, Fudan University, Shanghai, P. R. of China

ABSTRACT
The Zhang neural network (ZNN), a recurrent neural network, proposed in 2001, is particularly effective in solving time-varying problems. It has shown high efficiency and excellent performance in various applications. The weighted pseudoinverse is a useful tool in solving and analyzing the constrained least-squares problems. In this paper, we propose a ZNN model for computing the weighted pseudoinverse of a time-varying matrix. We show that our model converges globally and exponentially to the solution and our system is robust at the presence of small errors. A Matlab Simulink implementation of our model is presented. Our convergence analysis is verified by our experiments on testing matrices. A comparison study shows that our model has superior performance over the conventional gradient-based neural networks.

ARTICLE HISTORY
Received 27 July 2017
Revised 19 January 2018
Accepted 1 May 2018

KEYWORDS
ML-weighted pseudoinverse; time-varying matrix; Zhang neural network

1. Introduction

The generalized inverses have numerous applications, such as control, networks, statistics, econometrics. See the monograph [5] edited by Nashed and published in 1976 and recently reprinted by Elsevier for theory, analysis, computational methods and applications. In particular, the weighted pseudoinverse can be used in solving the constrained least-squares problems [2], which have wide applications. Algorithms for computing the weighted generalized inverses have been proposed, for example, Galba, Deineka, and Sergienko presented iterative methods for calculating the weighted pseudoinverses [3]. Note that these methods are designed for time-invariant matrices, that is, matrices with constant entries. However, there are applications such as online frequency domain identification processes where the weighted pseudoinverses of time-varying matrices are involved [2, 11, 12]. The algorithms in [3] are not suitable for the time-varying applications [8, 9, 26, 28].

CONTACT Yimin Wei ymwei@fudan.edu.cn; yimin.wei@gmail.com
Dedicated to Prof. M. Nashed's 85th Birthday

Due to its parallel processing nature, high performance in large-scale online applications, and the convenience of hardware implementations, the neural dynamics approach in the form of recurrent neural networks (RNNs) has been investigated intensively and regarded as a powerful tool for solving online linear and nonlinear problems. See [14–18] for recent progresses.

Since 2001, a special class of RNNs, called the Zhang neural networks (ZNNs), has been proposed for online solutions of various time-varying problems [4, 22–25]. The ZNNs differ from the conventional gradient neural networks (GNNs) in the types of the problems to be solved, error functions, design formulas, dynamic equations, and the utilization of time derivatives. By using the Zhang functions (ZFs), the basis for the ZNN, and the time-derivative information of the time-varying coefficients, the ZNN models guarantee the exponential global convergence.

In this paper, we present a ZNN model for computing the weighted pseudoinverse of a time-varying matrix and prove its exponential global convergence. The computer simulation of our ZNN model demonstrates its efficiency and superiority over the conventional GNNs.

This paper is organized as follows. In Section 2 introduces some preliminaries for the time-varying ML-weighted pseudoinverse. The design procedure of our ZNN models is given in Section 3 and the simulations are presented in Section 4.

2. Preliminaries and problem formulation

This section presents some necessary preliminaries of the time-varying weighted pseudoinverse, including a definition, properties, and a limit representation.

We modify the definition of the weighted, or more precisely ML-weighted, pseudoinverse of a matrix K with constant (time-invariant) entries in [2] into that of a matrix $K(t)$ with time-varying entries [1, 2, 6, 10, 12, 13].

Definition 2.1. Given an $m \times n$ time-varying matrix $K(t)$ and and two weight matrices $M(t)$, $s \times m$, and $L(t)$, $l \times n$, whose entries are continuously differentiable real-valued functions of time t, $t \geq 0$. Let $P(t) = I_n - [M(t)K(t)]^{\dagger} M(t)K(t)$, then the time-varying ML-weighted pseudoinverse matrix of $K(t)$ is defined by

$$K^{\dagger}_{ML}(t) = \left[I_n - (L(t)P(t))^{\dagger} L(t)\right] [M(t)K(t)]^{\dagger} M(t),$$

where $[M(t)K(t)]^{\dagger}$ denotes the Moore-Penrose inverse of $M(t)K(t)$.

The following theorem shows a relationship between the the ML-weighted pseudoinverse and the ML-weighted least-squares (ML-WLS) problem, which is similar to [2, Theorem 2.1].

Theorem 2.2. *Given an $m \times n$ time-varying real matrix K(t) and two weight matrices M(t), $s \times m$, and L(t), $l \times n$, for $t \geq 0$, the ML-weighted least-squares (ML-WLS) problem is*

$$\min_{f(t)\in\mathscr{B}} ||f(t)||_{L(t)}, \mathscr{B} = \{f(t) |\ ||K(t)f(t)-g(t)||_{M(t)} \text{ is minimum}\}, \quad \forall t \geq 0, \tag{2.1}$$

where $||\cdot||_{M(t)}$ and $||\cdot||_{L(t)}$ are the ellipsoid semi-norms defined respectively by

$$||g(t)||^2_{M(t)} = g(t)^{\mathrm{T}}M(t)^{\mathrm{T}}M(t)g(t) \ \textit{and}\ ||f(t)||^2_{L(t)} = f(t)^{\mathrm{T}}L(t)^{\mathrm{T}}L(t)f(t), \forall t \geq 0.$$

The general solution to this problem is given by

$$f(t) = K^{\dagger}_{ML}(t)g(t) + P(t)[I_n - (L(t)P(t))^{\dagger}L(t)P(t)]z(t), \quad \forall t \geq 0, \tag{2.2}$$

where z(t) is a time-varying n-vector. Moreover, the solution is unique if and only if

$$\mathrm{null}(M(t)K(t)) \cap \mathrm{null}(L(t)) = \{\mathbf{0}\}. \tag{2.3}$$

Proof. First, if $M(t)K(t)$ is of full column rank, then $\mathrm{null}(M(t)K(t)) = \{\mathbf{0}\}$, and $P(t) = 0$, and the theorem is true.

For the case $\mathrm{rank}(M(t)K(t)) < n$, we use [7, (3.2.5)] to obtain

$$\mathscr{B} = \{f(t)|f(t) = [M(t)K(t)]^{\dagger}M(t)g(t) + P(t)y(t),\ y(t)\ \textit{arbitrary}\}.$$

Put $\tilde{g}(t) = [M(t)K(t)]^{\dagger}M(t)g(t)$, we have

$$\min_{f(t)\in\mathscr{B}} ||f(t)||_{L(t)} = \min||\tilde{g}(t) + P(t)y(t)||_{L(t)}.$$

It follows from [7] that this least-squares problem has the general solution

$$y(t) = -[L(t)P(t)]^{\dagger}L(t)\tilde{g}(t) + \left[I-(L(t)P(t))^{\dagger}L(t)P(t)\right]z(t),$$

where $z(t)$ is arbitrary. Thus the general solution of (2.1) is

$$\begin{aligned} f(t) &= [M(t)K(t)]^{\dagger}M(t)g(t)-P(t)[L(t)P(t)]^{\dagger}L(t)\tilde{g}(t) + P(t)\left[I-(L(t)P(t))^{\dagger}L(t)P(t)\right]z(t) \\ &= \left[I-P(t)(L(t)P(t))^{\dagger}L(t)\right](M(t)K(t))^{\dagger}M(t)g(t) + P(t)\left[I-(L(t)P(t))^{\dagger}L(t)P(t)\right]z(t). \end{aligned}$$

It is obvious that $P(t)[L(t)P(t)]^{\dagger} = [L(t)P(t)]^{\dagger}$ and thus we prove the first part.

For the second part, if (2.3) is not satisfied, then the solution cannot be unique, for any element of the intersection of the nullspaces can be added to the solution without changing the minimum.

If (2.3) is satisfied. Let $z(t)$ be arbitrary and put $w(t) = P(t)[I-(L(t)P(t)]^{\dagger}L(t)P(t))z(t)$. Then $w(t) \in \text{null}(M(t)K(t))$ and $w(t) \in \text{null}(L(t))$. Thus by (2.3) $w(t) = 0$, and the solution is unique. $\square$

The time-varying ML-weighted pseudoinverse matrix $K_{ML}^{\dagger}(t)$ has the following properties analogous to those of the Moore-Penrose inverse.

Lemma 2.3. *Let $K_{ML}^{\dagger}(t) = X(t), t \geq 0$, then the following four equations hold for all $t \geq 0$:*

1. $M(t)K(t)X(t)K(t) = M(t)K(t)$;
2. $X(t)K(t)X(t) = X(t)$;
3. $[M(t)^{\mathrm{T}}M(t)K(t)X(t)]^{\mathrm{T}} = M(t)^{\mathrm{T}}M(t)K(t)X(t)$;
4. $[L(t)^{\mathrm{T}}L(t)X(t)K(t)]^{\mathrm{T}} = L(t)^{\mathrm{T}}L(t)X(t)K(t)$.

The following lemma gives a limit representation of $K_{ML}^{\dagger}(t)$.

Lemma 2.4. *The limit*

$$\lim_{\lambda \to 0} \left[(M(t)K(t))^{\mathrm{T}}M(t)K(t) + \lambda^2 L(t)^{\mathrm{T}}L(t)\right]^{\dagger}(M(t)K(t))^{\mathrm{T}}M(t) \tag{2.4}$$

exists and equals $K_{ML}^{\dagger}(t)$, for all $t \geq 0$.

Denoting $D_{\lambda}(t) = [M(t)K(t)]^{\mathrm{T}}M(t)K(t) + \lambda^2 L(t)^{\mathrm{T}}L(t)$ and $S(t) = [M(t)K(t)]^{\mathrm{T}}M(t)$, then the limit representation (2.4) becomes

$$K_{ML}^{\dagger}(t) = \lim_{\lambda \to 0} D_{\lambda}(t)^{\dagger}S(t). \tag{2.5}$$

Let $D(t) = \lim_{\lambda \to 0} D_{\lambda}(t)$ and $X(t)$ be an approximation of $K_{ML}^{\dagger}(t)$, then $D(t)^{\dagger}S(t) - X(t)$ gives the approximation error. However, the computation of the error requires $D(t)^{\dagger}$. How do we compute the error without the pseudoinverse $D(t)^{\dagger}$? Indeed,

$$\begin{aligned}
\text{range}(S(t)) &= \text{range}\left[(M(t)K(t))^{\mathrm{T}}M(t)\right] \\
&\subset \text{range}\left[(M(t)K(t))^{\mathrm{T}}\right] \\
&= \text{range}\left[(M(t)K(t))^{\mathrm{T}}M(t)K(t)\right] \\
&= \text{range}(D(t)).
\end{aligned}$$

Thus $D(t)D(t)^{\dagger}S(t) = S(t)$, since $D(t)D(t)^{\dagger}$ is an orthogonal projection onto range$(D(t))$, which includes range$(S(t))$. Consequently, $K_{ML}^{\dagger}(t) - D(t)^{\dagger}S(t) = 0$ implies $D(t)K_{ML}^{\dagger}(t) - S(t) = 0$. That is,

$$D_{\lambda}(t)K_{ML}^{\dagger}(t) - S(t) \to 0, \quad \text{as } \lambda \to 0. \tag{2.6}$$

If $X(t)$ is an approximation of $K_{ML}^{\dagger}$, then $D_{\lambda}(t)X(t) - S(t)$ can be used as an approximation error for small λ.

Remark. The equation (2.6) is used as the error monitoring function in our ZNN model, since it is easier to compute $D_{\lambda}(t)X(t) - S(t)$ than to compute the pseudoinverse form $D_{\lambda}(t)^{\dagger}S(t) - X(t)$. As for numerical experiments, a very small λ should be avoided, as it can cause spurious results.

Remark. Since

$$D_{\lambda}(t) = \begin{bmatrix} M(t)K(t) \\ \lambda L(t) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} M(t)K(t) \\ \lambda L(t) \end{bmatrix},$$

when

$$\operatorname{rank}\left(\begin{bmatrix} M(t)K(t) \\ \lambda L(t) \end{bmatrix} \right) = n, \quad \text{for any} \quad t \geq 0, \text{ as } \lambda \to 0,$$

then $\operatorname{rank}(D_{\lambda}(t)) = n$ and $(D_{\lambda}(t))^{\dagger} = (D_{\lambda}(t))^{-1}$, as $\lambda \to 0$. In this case $D(t)K_{ML}^{\dagger}(t) - S(t) = 0$ is equivalent to $D(t)^{\dagger}S(t) - K_{ML}^{\dagger}(t) = 0$.

Remark. (Regularization) If

$$\operatorname{rank}\left(\begin{bmatrix} M(t_0)K(t_0) \\ \lambda L(t_0) \end{bmatrix} \right) < n, \quad \text{for some } t_0 \geq 0, \text{ as } \lambda \to 0,$$

then $D_{\lambda}(t)$ is not invertible for all t, as $\lambda \to 0$. To alleviate the problem, we introduce a small regularization parameter μ into $D_{\lambda}(t)$:

$$D_{\lambda}(t) = [M(t)K(t)]^{\mathrm{T}} M(t)K(t) + (\lambda^2 + \mu)L(t)^{\mathrm{T}}L(t), \quad \text{as } \lambda \to 0.$$

In our experiments, we chose $\mu \approx 10^{-8}$, the square root of the double precision.

3. Design of the ZNN model

In this section, we present the design procedure of our ZNN model. The ZF is the design basis for a ZNN model, for simplicity, we denote $E(t)$ as the ZF and $\dot{E}(t)$ as the time derivative of $E(t)$.

To achieve global exponential convergence of $E(t)$ to zero, we choose

$$\dot{E}(t) := \frac{dE(t)}{dt} = -\gamma\Phi(E(t)), \tag{3.1}$$

called the ZNN design formula, where the design parameter $\gamma > 0$ corresponds to the inductance parameter or the reciprocal of a capacitance parameter, which should be set as large as the hardware permits, and Φ is an activation array to be specified in subsection 3.2.

3.1. The ZNN model for the time-varying ML-weighted pseudoinverse matrix

We define the ZF

$$E(t) = D_\lambda(t)X(t) - S(t)$$

as the fundamental error-monitoring function, where $X(t)$ is an approximation of $K^\dagger_{ML}(t)$. Its derivative is then

$$\dot{E}(t) = \dot{D}_\lambda(t)X(t) + D_\lambda(t)\dot{X}(t) - \dot{S}(t),$$

as $\lambda \to 0$. Applying the ZNN design formula (3.1), we have the corresponding dynamic equation of the ZNN model:

$$\dot{D}_\lambda(t)X(t) + D_\lambda(t)\dot{X}(t) - \dot{S}(t) = -\gamma\Phi(D_\lambda(t)X(t) - S(t)), \tag{3.2}$$

which is equivalent to:

$$D_\lambda(t)\dot{X}(t) = -\gamma\Phi(D_\lambda(t)X(t) - S(t)) - \dot{D}_\lambda(t)X(t) + \dot{S}(t), \tag{3.3}$$

or

$$\dot{X}(t) = -\gamma\Phi(D_\lambda(t)X(t) - S(t)) - \dot{D}_\lambda(t)X(t) + \dot{S}(t) - D_\lambda(t)\dot{X}(t) + \dot{X}(t). \tag{3.4}$$

The (ij)th-neuron dynamic equation of the ZNN model (3.4) is

$$\begin{aligned}\dot{x}_{ij}(t) = &-\sum_{k=1}^{n}\left(\dot{d}_{ik}(t)x_{kj}(t) - d_{ik}(t)\dot{x}_{kj}(t)\right)\\ &+\dot{s}_{ij}(t) - \gamma\phi\left(\sum_{k=1}^{n}\left(d_{ik}(t)x_{kj}(t) - s_{ij}(t)\right)\right) + \dot{x}_{ij}(t),\end{aligned}$$

where $x_{ij}(t), \dot{x}_{ij}(t), d_{ij}(t),\ \dot{d}_{ij}(t), s_{ij}(t)$, and $\dot{s}_{ij}(t)$ denote the (ij)th entries of their corresponding matrices $X(t)$, $\dot{X}(t)$, $D(t)$, $\dot{D}(t)$, $S(t)$, $\dot{S}(t)$. The block diagram realizing the ZNN model (3.4) is shown in Figure 1.

3.2. Four types of the activation array $\Phi(\cdot)$

The activation array Φ can be one of the following four types:

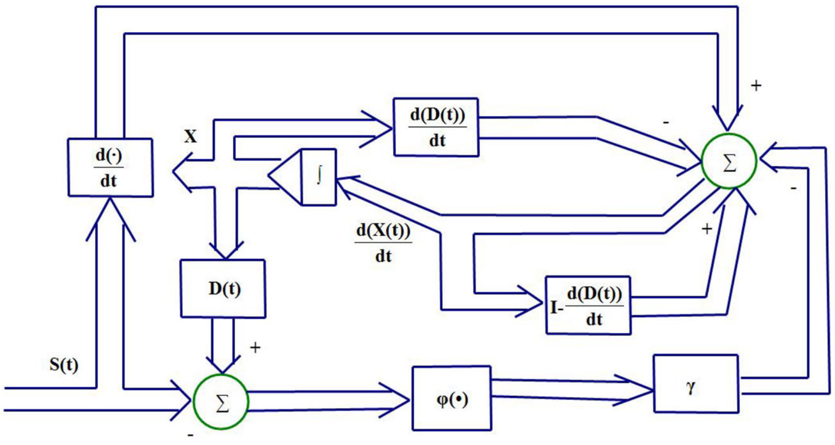

Figure 1. Block Diagram of the ZNN model (3.4).

- linear function: $\phi(u) = u$;
- bipolar-sigmoid function: $\phi(u) = \frac{1-\exp(-qu)}{1+\exp(-qu)}, q \geq 1$;
- power-sigmoid function: $\phi(u) = \frac{1+\exp(q)}{1-\exp(q)} \cdot \frac{1-\exp(-qu)}{1+\exp(-qu)},\ q \geq 1,$ if $|u| < 1$ or $\phi(u) = u^p,\ p \geq 3,$ otherwise.

3.3. Global exponential convergence of the ZNN-ML model (3.4)

Now, we show that the ZNN-ML model (3.4) converges globally and exponentially to the ML-weighted pseudoinverse under the following condition.

Invertibility Condition A. There exists a positive real number $\alpha_1 > 0$ such that

$$\min_{1\leq i\leq n} |\lambda_i(D(t))| \geq \alpha_1, \quad \text{for any } t \geq 0,$$

where $\lambda_i(D(t))$ denotes the ith eigenvalue of matrix $D(t)$ of order n.

Before proving the convergence, we state the following result [26, Lemma 4.2.1].

Lemma 3.1. *If $D_\lambda(t)$ satisfies the Invertibility Condition A and is bounded above by β, that is, $||D_\lambda(t)||_F \leq \beta$, for all $t \geq 0$, as $\lambda \to 0$, then its inverse is uniformly upper bounded, that is,*

$$||D_\lambda^{-1}(t)||_F \leq \varphi(\beta) := \sum_{i=0}^{n-2} (C_n^i \beta^{n-i-1}/\alpha_1^{n-i} + n^{3/2}/\alpha_1), \quad \textit{for } t \geq 0, \text{ as } \lambda \to 0,$$

where $C_n^i = n!/(i!(n-i)!)$, recalling that α_1 is a lower bound for the eigenvalues.

Theorem 3.2. *Given an $m \times n$ smoothly time-varying matrix $K(t)$ and two smoothly time-varying weight matrices $M(t)$, $s \times m$, and $L(t)$, $l \times n$. If $D_\lambda(t)$*

satisfies the Invertibility Condition A, as $\lambda \longrightarrow 0^+$, and $\Phi(\cdot)$ is a monotonically increasing odd function array, then the $n \times m$ state matrix $X(t)$ of the ZNN-ML model (3.4), starting from any initial $X(0)$, globally converges to the time-varying pseudoinverse $K_{ML}^{\dagger}(t)$ of matrix $K(t)$.

Proof. Since $D(t)$ satisfies Invertibility Condition A,

$$\operatorname{rank}(D_\lambda(t)) = \operatorname{rank}\left(\begin{bmatrix} M(t)K(t) \\ \lambda L(t) \end{bmatrix}\right) = n, \quad \text{for any } t \geq 0, \text{ as } \lambda \longrightarrow 0^+.$$

So, (2.6) holds and the ZNN-ML model (3.4) is applicable. Let $X^*(t)$ denote the exact solution $K_{ML}^{\dagger}(t)$ and $\tilde{X}(t) = X(t) - X^*(t)$ the difference between the solution $X(t)$ generated by the ZNN-ML model (3.4) and the exact solution. As $\lambda \longrightarrow 0^+$, we have $D(t)X^*(t) - S(t) = 0$ and its time derivative

$$\dot{D}(t)X^*(t) + D(t)\dot{X}^*(t) - \dot{S}(t) = 0. \tag{3.5}$$

Substituting $X^*(t) = X(t) - \tilde{X}(t)$ into the above equation (3.5), we get

$$\dot{D}(t)\tilde{X}(t) + D(t)\dot{\tilde{X}}(t) = -\gamma\Phi(D(t)\tilde{X}(t)). \tag{3.6}$$

It then follows that the difference $\tilde{X}(t)$ is the solution ensuing the dynamics with the initial state $\tilde{X}(0) = X(0) - \tilde{X}(0)$.

Since $E(t) = D(t)X(t) - S(t) = D(t)(X^*(t) + \tilde{X}(t)) - S(t) = D(t)\tilde{X}(t)$ and $\dot{E}(t) = \dot{D}(t)\tilde{X}(t) + D(t)\dot{\tilde{X}}(t)$, Equation (3.6) can be rewritten as

$$\dot{E}(t) = -\gamma\Phi(E(t)),$$

which is a compact matrix form of the following set of n^2 equations

$$\dot{e}_{ij}(t) = -\gamma\phi(e_{ij}(t)), \quad \text{for all } i,j \in \{1,2,\ldots,n\}. \tag{3.7}$$

Now, we define a Lyapunov function candidate $v_{ij}(t) = e_{ij}^2(t)/2$ for the (ij)th subsystem (3.7) with its time derivative

$$\frac{dv_{ij}(t)}{dt} = e_{ij}(t)\dot{e}_{ij}(t) = -\gamma e_{ij}(t)\phi(e_{ij}(t)). \tag{3.8}$$

Because the activation functions $\phi(u)$ are monotonically increasing odd functions, we have $\phi(-u) = -\phi(u)$ and

$$\phi(u) = \begin{cases} > 0, & \text{if } u > 0; \\ = 0, & \text{if } u = 0; \\ < 0, & \text{if } u < 0, \end{cases}$$

which guarantees the negative definiteness of $\dot{v}_{ij}(t)$, that is, $\dot{v}_{ij}(t) < 0$, for $e_{ij}(t) \neq 0$, and $\dot{v}_{ij}(t) = 0$, for $e_{ij}(t) = 0$. By the Lyapunov stability theory, $e_{ij}(t)$ globally converges to zero for any $i,j \in \{1,2,\ldots,n\}$. Thus from $E(t) = D(t)\tilde{X}(t)$ and

Invertibility Condition A, we have $\tilde{X}(t) \to 0$ as $t \to \infty$, i.e., the neural state $X(t)$ is globally convergent to the exact inverse $X^*(t)$. This completes the proof of the global convergence. Moreover, $E(t) = D(t)\tilde{X}(t) = D(t)[X(t) - X^*(t)]$, Invertibility Condition A, and Lemma 3.1 imply that

$$||X(t) - X^*(t)||_F \le ||D^{-1}(t)||_F ||E(t)||_F \le \varphi(\beta) \left(\sum_i^n \sum_j^n e_{ij}^2(t) \right)^{1/2}$$
$$\le n\varphi(\beta) \left(\max_{1 \le i,j \le n} |e_{ij}(t)| \right),$$

showing that the error and the network convergence can be estimated by the maximum entry error $e_{ij}(t)$ in (3.8). □

For the simple linear case where $\phi(e_{ij}(t)) = e_{ij}(t)$ and $\dot{e}_{ij}(t) = -\gamma e_{ij}(t)$, we have

$$e_{ij}(t) = \exp(-\gamma t) e_{ij}(0).$$

Thus there exists a constant $\zeta > 0$, such that

$$||X(t) - X^*(t)||_F \le \zeta \exp(-\gamma t).$$

This means that the neural network possesses the exponential convergence at the rate γ, when using the linear activation function $\phi(u) = u$.

Next, we consider an alternative invertibility condition.

Invertibility Condition B. There exists a positive real number $\alpha_2 > 0$ such that

$$\sigma_{\min}(D_\lambda(t)) \ge \alpha(t) > \alpha_2, \quad \text{for any } t \ge 0, \text{ as } \lambda \to 0,$$

where $\sigma_{\min}(D_\lambda(t))$ denotes the minimum singular value of matrix $D_\lambda(t)$.

Lemma 3.3. *When $D_\lambda(t)$ satisfies Invertibility Condition B, then*

$$||D_\lambda^{-1}(t)||_2 = \frac{1}{\sigma_{\min}(D_\lambda(t))} \le \frac{1}{\alpha(t)} < \frac{1}{\alpha_2}.$$

Theorem 3.4. *Given an $m \times n$ $K(t)$ and two weight matrices $M(t)$, $s \times m$ and $L(t)$, $l \times n$. If $D_\lambda(t)$ satisfies the Invertibility Condition B, as $\lambda \to 0^+$ and a monotonically increasing odd function array $\Phi(\cdot)$ is used, then the $n \times m$ state matrix $X(t)$ of the ZNN-ML model (3.4), starting from any initial $X(0)$, globally converges to the time-varying pseudoinverse $K_{ML}^\dagger(t)$ of matrix $K(t)$.*

Proof. The proof is similar to that of Theorem 3.2, we only give the error bound:

$$||X(t) - X^*(t)||_F = ||D^{-1}(t)E(t)||_F \le ||D^{-1}(t)||_2 ||E(t)||_F,$$

which implies that

$$||X(t) - X^*(t)||_F \le ||D^{-1}(t)||_2 ||E(t)||_F < \frac{1}{\alpha_2} \cdot n \cdot \max_{i,j} |e_{ij}|.$$

Finally, we consider the third condition.

Invertibility Condition C. The matrix $D_\lambda(t)$ is strictly diagonally dominant.

Lemma 3.5. *Varga gives a bound of $||A^{-1}||_\infty$ for a strictly diagonally dominant matrix A:*

$$||A^{-1}||_\infty \le \max_i \frac{1}{|a_{ii}| - \Lambda_i(A)},$$

where $\Lambda_i(A) = \sum_{k=1,\, k\neq i}^{n} |a_{ik}|$.

Theorem 3.6. *Given an $m \times n$ smoothly time-varying matrix $K(t)$ and two weight matrices $M(t)$, $s \times m$ and $L(t)$, $l \times n$. If $D_\lambda(t)$ satisfies the Invertibility Condition C as $\lambda \to 0^+$ and a monotonically increasing odd function array $\Phi(\cdot)$ is used, then the $n \times m$ state matrix $X(t)$ of the ZNN-ML model (3.4), starting from any initial $X(0)$, globally converges to the time-varying pseudoinverse $K_{ML}^{\dagger}(t)$ of matrix $K(t)$.*

Proof. The proof is similar to that of Theorem 3.2, we give the error bound:

$$||X(t) - X^*(t)||_F \le ||D_\lambda^{-1}(t)||_2 ||E(t)||_F \le \sqrt{n} \cdot ||D_\lambda^{-1}(t)||_\infty ||E(t)||_F$$

which implies that

$$||X(t) - X^*(t)||_F \le n^{3/2} \cdot \max_t \max_i \frac{1}{|d_{ii}(t)| - \Lambda_i(D(t))} \max_{i,j} |e_{ij}|.$$

□

We provide the above three conditions to provide alternative ways of checking the invertibility of $D_\lambda(t)$, depending on the available information.

Remark. Zhang has proven that the bipolar sigmoid activation function, the power activation function, and the power sigmoid activation function have superior convergence over the linear function, but the superior convergence occurs only when the error falls in some special interval. Also, when using these functions, we can only perform the level-1 BLAS operations, instead of the more efficient level-3 BLAS operations.

3.3.1. Robustness of ZNN-ML model (3.3)

In this section, we consider the effect of errors on the ZNN-ML model (3.3). Let $\Delta_B(t)$ $(n \times n)$ be the differentiation error and $\Delta_R(t)$ $(n \times n)$ the model-implementation error, and the ZNN-ML model (3.3) with these errors is

$$D(t)\dot{X}(t) = -(\dot{D}(t) + \Delta_B(t))X(t) + \dot{S}(t) - \gamma\Phi(D(t)X(t) - S(t)) + \Delta_R(t), \tag{3.9}$$

considered for the general robustness properties of the ZNN-ML model [26, Theorem 4.3.2].

Theorem 3.7. *Suppose that* $||\Delta_B(t)||_F \le \epsilon_1$, $||\Delta_R(t)||_F \le \epsilon_2$, *and* $D(t)$ *satisfies the Invertibility Condition A, so Lemma 3.1 holds, then the error* $||X(t) - X^*(t)||_F$ *is bounded.*

Proof. Defining the error matrix

$$E(t) = D(t)(X(t) - X^*(t)),$$

we have $X(t) = D^{-1}(t)E(t) + X^*(t)$, since $X^*(t) = D^{-1}(t)S(t)$. Then (3.9) can be reformulated as

$$\dot{E}(t) = -\Delta_B(t)D^{-1}(t)E(t) - \gamma\Phi(E) + (\Delta_R(t) - \Delta_B(t)X^*(t)),$$

whose equivalent vector form is

$$\dot{e}(t) = -\gamma\Phi(e(t)) + B(t)e(t) + c(t),$$

where

$$\begin{aligned} &e(t) := \mathrm{vec}(E(t)), B(t) := I \otimes (-\Delta_B(t)D^{-1}(t)), \text{ and} \\ &c(t) := \mathrm{vec}(\Delta_R(t) - \Delta_B(t)X^*(t)). \end{aligned}$$

Define the Lyapunov function candidate $v(t) = e^{\mathrm{T}}(t)e(t)/2 \ge 0$ for the error dynamics. The time derivative of $v(t)$ is

$$\begin{aligned} \frac{dv}{dt} &= e^{\mathrm{T}}(t)\dot{e}(t) \\ &= e^{\mathrm{T}}(t)(-\gamma\Phi(e(t)) + B(t)e(t) + c(t)) \\ &= -\gamma e^{\mathrm{T}}(t)\Phi(e(t)) + e^{\mathrm{T}}(t)\frac{B(t) + B^{\mathrm{T}}(t)}{2}e(t) + e^{\mathrm{T}}(t)c(t). \end{aligned}$$

It then follows from the logarithmic norm, the inequality $\max_i |\lambda_i(A)| \le ||A||_2 \le ||A||_F$ and Lemma 3.1 that

$$\begin{aligned} e^{\mathrm{T}}(t)\frac{B(t) + B^{\mathrm{T}}(t)}{2}e(t) &\le e^{\mathrm{T}}(t)e(t) \max_{1 \le i \le n^2} \left(|\lambda_i((B(t) + B^{\mathrm{T}}(t))/2)|\right) \\ &= e^{\mathrm{T}}(t)e(t) \max_{1 \le i \le n^2} \left(|\lambda_i((I \otimes (\Delta_B(t)D^{-1}(t) + (\Delta_B(t)D^{-1}(t))^{\mathrm{T}}))/2)|\right) \\ &= e^{\mathrm{T}}(t)e(t) \max_{1 \le i \le n} \left(|\lambda_i((\Delta_B(t)D^{-1}(t) + (\Delta_B(t)D^{-1}(t))^{\mathrm{T}})/2)|\right) \\ &\le e^{\mathrm{T}}(t)e(t) \; ||\Delta_B(t)||_F||D^{-1}(t)||_F \\ &\le e^{\mathrm{T}}(t)e(t)\epsilon_1\varphi. \end{aligned}$$

Similarly, it follows from $||E(t)||_F^2 = \sum_{i=1}^n \sum_{j=1}^m |e_{ij}(t)|^2$ that $|c_i(t)| \le ||\Delta_R(t) - \Delta_B(t)X^*(t)||_F \le ||\Delta_R(t)||_F + ||\Delta_B(t)X^*(t)||_F = \epsilon_2 + \epsilon_1\varphi$, for $i =$

$1, 2, \ldots, n^2$. Thus, $e^{\mathrm{T}}(t)c(t) \leq (\epsilon_2 + \epsilon_1\varphi) \sum_{i=1}^{n} \sum_{j=1}^{m} |e_{ij}|$. Finally, from the above argument and the symmetry property of $\phi(\cdot)$, we have

$$\begin{aligned} \frac{dv(t)}{dt} &\leq -\gamma e^{\mathrm{T}}(t)\Phi(e(t)) + \epsilon_1\varphi e^{\mathrm{T}}(t)e(t) + e^{\mathrm{T}}(t)c(t) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}|e_{ij}(t)|(\gamma\phi(|e_{ij}(t)|) - \epsilon_1\varphi|e_{ij}(t)| - \epsilon_2 - \epsilon_1\varphi). \end{aligned} \tag{3.10}$$

During the time evolution of $e_{ij}(t)$, the above equation falls into the situation $\gamma\phi(|e_{ij}(t)|) - \epsilon_1\varphi|e_{ij}(t)| - \epsilon_2 - \epsilon_1\varphi \geq 0$ or $\gamma\phi(|e_{ij}(t)|) - \epsilon_1\varphi|e_{ij}(t)| - \epsilon_2 - \epsilon_1\varphi < 0$. If in the time interval $[t_0, t_1)$, the trajectory of the system (3.9) is in the first situation, then $\dot{v}(t) \leq 0$ and (3.10) implies that $X(t)$ converges to $X^*(t)$ as time evolves. For any time t when the trajectory falls into the second situation, the difference between $X(t)$ and $X^*(t)$ may not decrease, however, even in the worst case, the entry error $|e_{ij}(t)|$ is also bounded by the steady-state entry residual error $\bar{e}_{ij}(t) = (\epsilon_2 + \epsilon_1\varphi)/(\gamma\rho - \epsilon_1\varphi)$, where the insensitivity parameter $\rho > 0$ is between $\phi(e_{ij}(0))/e_{ij}(0)$ and $\phi'(0)$, and the design parameter γ satisfies $\gamma > \epsilon_1\varphi/\rho$. It then follows that

$$\lim_{t\to\infty} ||X(t) - X^*(t)||_F \leq n\varphi(\epsilon_2 + \epsilon_1\varphi)/(\gamma\rho - \epsilon_1\varphi).$$

Clearly, this steady-state residual error caused by differentiation and implementation errors can be made arbitrarily small as the design parameter γ increases. □

4. Simulations and verifications

In this section, the related simulation techniques are presented and some illustrative examples are given to verify the efficacy and the superiority of the proposed ZNN model for the time-varying ML-weighted pseudoinverse.

4.1. Kronecker product and vectorization

In the previous sections, we have developed a ZNN model. The model is described in the matrix form, which cannot be directly simulated. Thus, the Kronecker product and vectorization techniques are necessary to transform such matrix-form differential equations into vector-from for simulation purposes.

For the ZNN-ML model (3.3), based on the Kronecker product, denoted by $\otimes$, and vectorization techniques, we transform the model into the following vector-form:

$$(I_m \otimes D(t))\text{vec}(\dot{X}(t)) = -\gamma\Phi((I_m \otimes D(t))\text{vec}(X(t)) - \text{vec}(S(t)))$$
$$-(I_m \otimes \dot{D}(t)\text{vec}(X(t)) + \text{vec}(\dot{S}(t))$$

For simplicity, we write

$$M(t)\dot{x}(t) = -\dot{M}(t)x(t) - \gamma\Phi(M(t)x(t) - \hat{S}(t)) + \dot{\hat{S}}(t), \qquad (4.1)$$

where the activation-function array $\Phi(\cdot)$ is defined in (3.3), except that its dimensions are changed so that $\Phi(\cdot) : \mathbb{R}^{mn\times 1} \to \mathbb{R}^{mn\times 1}$, the so-called mass matrix $M(t) := I \otimes D(t)$, $x(t) := \text{vec}(X(t))$, and $\hat{S}(t) = \text{vec}(S(t))$.

4.2. Computer simulation examples

In this subsection, some computer-simulation examples are demonstrated to verify the efficacy and the superiority of the proposed ZNN-ML model.

All the tests were performed on an OSX 10.9.5 machine, with 4 GB RAM.

Example 4.1. Consider the first group of matrices

$$K_1(t) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad M_1(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & t^2 + 10^{-3} \end{bmatrix}, \quad L_1(t) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

thus $m = 3$, $n = 2$, $s = 3$, and $l = 2$. The (3,3)-entry of $M_1(t)$ is set to $t^2 + 10^{-3}$ to avoid its rank deficiency when t starts from 0.

Then

$$\begin{bmatrix} M_1(t)K_1(t) \\ \lambda L_1(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & t^2 + 10^{-3} \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

and

$$D_\lambda(t) = \begin{bmatrix} M_1(t)K_1(t))^{\mathrm{T}} \lambda(L_1(t))^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} M_1(t)K_1(t) \\ \lambda L_1(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & (t^2 + 10^{-3})^2 \end{bmatrix}$$

is invertible. The ML-weighted pseudoinverse of $K_1(t)$ is

$$(K_1)^{\dagger}_{ML} = \lim_{\lambda \to 0} D_\lambda^{-1}(t)(M_1(t)K_1(t))^{\mathrm{T}} M_1(t) = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The computer simulation is based on the system (3.1) of differential equations. For the parameters, the terminal time was set to 10^{-7} second for this small problem, the relative error tolerance 10^{-9}, the absolute error tolerance 10^{-9}. We started the Matlab function ode45 with the zero vector. The convergence behavior of the network in the time interval $[0, 2 \times 10^{-8}\text{s}]$ is shown in Figure 2. The system converges very quickly and is stabilized after 2×10^{-8} s.

The approximation produced by our ZNN-ML model at $t = 10^{-7}$ s is

$$X_1(10^{-7}) = \begin{bmatrix} 0.5000 & 0.5000 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}.$$

Example 4.2. The second group of matrices

$$K_2(t) = \begin{bmatrix} 1 & 0 \\ t & t \\ 0 & 1 \end{bmatrix}, \quad M_2(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \sqrt{0.2} \end{bmatrix}, \quad L_2(t) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

The exact ML-weighted pseudoinverse of $K_2(t)$ is

$$(K_2(t))^{\dagger}_{ML} = \begin{bmatrix} \dfrac{1}{t^2+1} & \dfrac{t}{t^2+1} & -1 \\ 0 & 0 & -1 \end{bmatrix}.$$

The error behavior of the ZNN-ML model for $0 \le t \le 10^{-7}$ is shown in Figure 3, where the four errors are defined by

$$\begin{aligned}
\text{Error}_1 &= ||M(t)K(t)X(t)K(t) - M(t)K(t)||_F, \\
\text{Error}_2 &= ||X(t)K(t)X(t) - X(t)||_F, \\
\text{Error}_3 &= ||((M(t)K(t))^{\mathrm{T}}(M(t)K(t)) + \lambda^2 L(t)^{\mathrm{T}} L(t))X(t) - (M(t)K(t))^{\mathrm{T}} M(t)||_F, \\
\text{Error}_4 &= ||(K(t))^{\dagger}_{ML} - X(t)||_F,
\end{aligned}$$

Take initial vector as zero vector, $\gamma = 10^9$ and linear function. As shown in Figure 3, all four errors remain small after the convergence. Table 1 lists more precise four errors at $t = 10$ s in Examples 4.1 and 4.2.

Example 4.3. In this example, we consider the following time-varying matrix $K(t)$:

$$K(t) = \begin{bmatrix} \sin(t) + 2 & 0 & 0 \\ 0 & t & 0 \\ 0 & 0 & 3 - t \\ 0 & 0 & \cos(t) \end{bmatrix},$$

and two weight matrices:

$$M(t)=\begin{bmatrix}2&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\quad\text{and}\quad L(t)=\begin{bmatrix}0&0&0\\0&1&0\\0&0&1\end{bmatrix}.$$

Both M(t) and L(t) are rank deficient for all $t\geq 0$, but the augmented matrix

$$G(t)=\begin{bmatrix}M(t)K(t)\\L(t)\end{bmatrix}=\begin{bmatrix}2\sin(t)+4&0&0\\0&0&0\\0&0&3-t\\0&0&\cos(t)\\0&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

is of full column rank for all $t\geq 0$. The exact ML-weighted pseudoinverse $K_{ML}^{\dagger}(t)$ is

$$K_{ML}^{\dagger}(t)=\begin{bmatrix}\dfrac{1}{\sin(t)+2}&0&0&0\\0&0&0&0\\0&0&-\dfrac{t-3}{(t-3)^2+cos^2(t)}&\dfrac{\cos(t)}{(t-3)^2+\cos^2(t)}\end{bmatrix}.$$

At time t = 20, the relative errors in the three nonzero components of the computed pseudoinverse are $8.48\times 10^{-11}, 3.28\times 10^{-8}$, and 1.01×10^{-7}.

Example 4.4. In this example, we compare the four types of activation functions. Consider the symmetric testing matrix

$$S_n(t)=\begin{bmatrix}t+1&t&\dots&\dots&\dots&t&t+1\\t&t-1&t&\dots&\dots&\dots&t\\\vdots&t&t+1&t&\dots&\dots&\vdots\\\vdots&\ddots&t&t-1&t&\dots&\vdots\\\vdots&\ddots&\ddots&t&\ddots&t&\vdots\\t&\ddots&\ddots&\ddots&t&t-1&t\\t+1&t&\dots&\dots&\dots&t&t+1\end{bmatrix}$$

of order n, where $n\geq 3$ is an odd integer. Its rank is n − 1 [27]. The two weight matrices are:

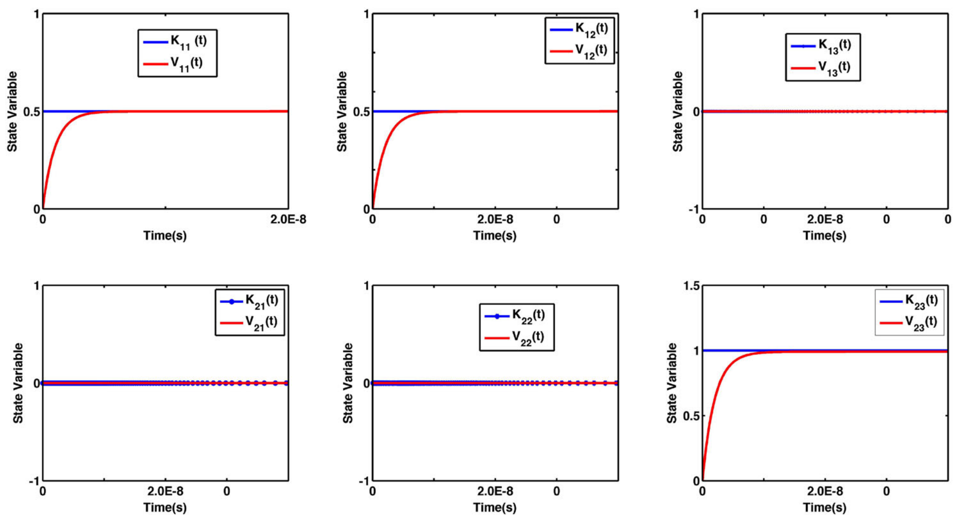

Figure 2. Convergence behavior of the ZNN-ML model in the time interval $[0, 2 \times 10^{-8}\text{s}]$ for Example 4.1, where $K_{ij}(t)$ are the entries of the exact time-varying pseudoinverse and $V_{ij}(t)$ are the entries of the system state matrix.

$$M(t) = I_n \quad \text{and} \quad L(t) = I_n.$$

Thus the ML-weighted pseudoinverse of $S_n(t)$ equals the Moore-Penrose inverse given by

$$S_n(t)^{\dagger}_{ML} = \begin{bmatrix} -t+1 & 2t & -2t & 2t & \dots & 2t & -t+1 \\ 2t & -4t-4 & 4t & -4t & 4t & -4t & 2t \\ -2t & 4t & -4t+4 & \ddots & \ddots & 4t & -2t \\ 2t & -4t & \ddots & \ddots & \ddots & -4t & 2t \\ \vdots & 4t & \ddots & \ddots & \ddots & 4t & \vdots \\ 2t & -4t & 4t & -4t & 4t & -4t-4 & 2t \\ -t+1 & 2t & -2t & 2t & \dots & 2t & -t+1 \end{bmatrix}.$$

We set the components of the initial vectors as zero vectors, $\gamma = 10^9$, and $\lambda = 10^{-3}$. We tested our ZNN-ML model in the time interval $[0, 1 \times 10^{-2}\text{s}]$ on the above matrices with $\text{n} = 1000$ using the linear activation function. Trajectories of Error_3, generated by using the ZNN-ML model, are shown in Figure 4. The terminal time of Examples 4.4 is 900 seconds.

We found that all three types of functions led to the system convergence. The two nonlinear functions slightly accelerated the convergence and they had the same accelerating behavior.

Example 4.5. Finally, we compare our ZNN-ML model with a conventional gradient-based neural networks (GNN) for computing the ML-weighted

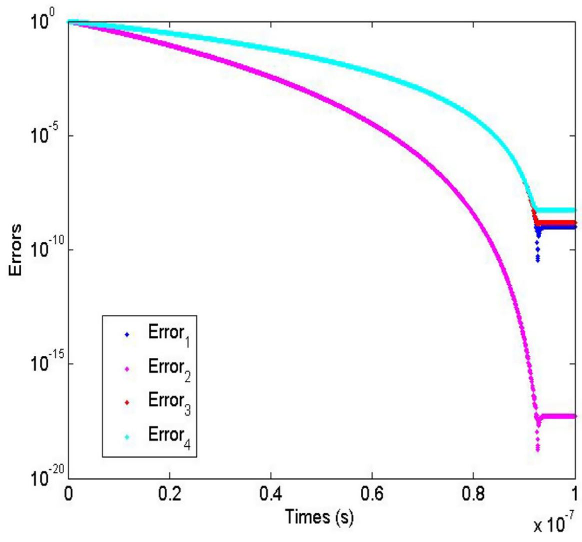

Figure 3. Error behavior of the ZNN-ML model for Example 4.2.

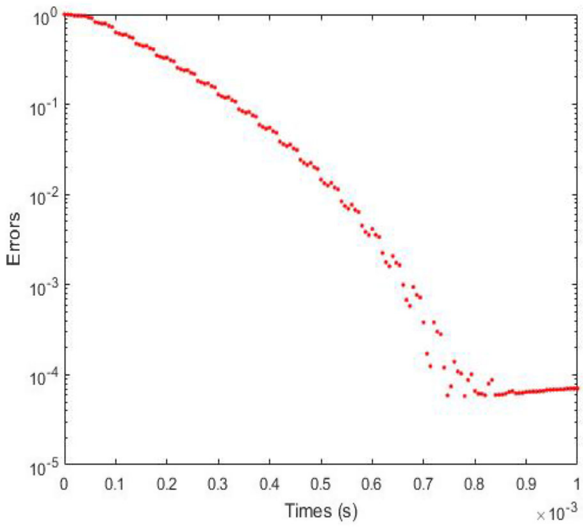

Figure 4. When $n = 1000$, Error$_3$ behaviors of the ZNN-ML model with the linear activation function for Example 4.4.

pseudoinverse of a time-varying matrix. Following the design methods in [19–21] and defining the scalar-valued error functions as

$$\epsilon(t) = \frac{||((M(t)K(t))^{\mathrm{T}}M(t)K(t) + \lambda^2 L(t)^{\mathrm{T}}L(t))V(t) - (M(t)K(t))^{\mathrm{T}}M(t)||_2^2}{2},$$

where V(t) is the system state matrix, we have the following GNN model:

$$\dot{V}(t) = -\gamma G^{\mathrm{T}}(t)(G(t)V(t) - F(t)$$

Table 1. Errors in Examples 4.1 and 4.2.

Example	Terminal Time	Error_1	Error_2	Error_3	Error_4
4.1	10 s	$4.58*10^{-12}$	$1.43*10^{-11}$	$2.07*10^{-13}$	$5.89*10^{-11}$
4.2	10 s	$9.9996*10^{-10}$	$5.0988*10^{-18}$	$1.6214*10^{-9}$	$5.2412*10^{-9}$

for time-varying ML-weighted pseudoinverse, where $G(t) = [M(t)K(t)]^{\mathrm{T}}M(t)K(t) + \lambda^2 L(t)^{\mathrm{T}}L(t)$ and $F(t) = [M(t)K(t))]^{\mathrm{T}}M(t)$. We tested our ZNN-ML and the GNN on the matrix $K(t) = S_n(t)$ in Example 4.4 with $M(t) = L(t) = I_n$, where n = 1001, and measured the logarithm of a typical relative error defined by

$$\text{Error} = \log_{10}(||V(t)K(t)V(t) - V(t)||_F / ||K(t)||_F).$$

Table 2 compares the errors at $t = t_{\text{final}}$ in the two networks and shows the superior performance of our ZNN-ML model over the GNN model.

4.3. Simulink modeling and verification

MATLAB Simulink, a graphical-design based modeling tool, exploits existing function blocks to construct mathematical and logical models as well as process flow. A Simulink model is a representation of the design or implementation of a system satisfying a set of requirements. In Section 3, we described the mathematical model (3.3) of our ZNN-ML system. In this section, we investigate the MATLAB Simulink modeling techniques.

4.3.1. Basic function blocks

MATLAB Simulink includes a comprehensive block library of sinks, sources, linear and nonlinear components, connectors, and so on. The basic function blocks used in the construction of the dynamic system (3.3) are briefly listed as follows.

- The Sine Wave block can generate the sine wave $O(t) = A\sin(ft + \phi) + D$, where A is the amplitude, f the frequency, ϕ the phase, and D the bias;
- The Product block allows two types of multiplications: element-wise and matrix multiplication. In our implementation, the default element-wise multiplication is changed to the matrix multiplication;
- The Math Function block represents mathematical functions, including logarithmic, exponential, power and modulus functions;
- The Integrator block makes continuous-time integration on the input signals.

Table 2. Compare the errors at $t = t_{\text{final}}$ in the GNN with the ZNN-ML.

	GNN	ZNN-ML
Error	−7.3527	−7.2048
t_{final}	497.423s	64.500s

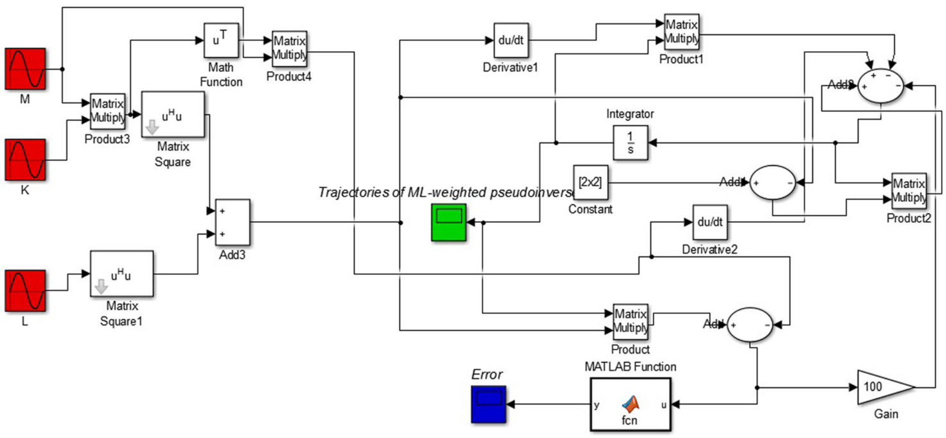

Figure 5. Diagram of the Simulink model of the ZNN-ML model (3.3) for online time-varying ML-weighted pseudoinverse.

4.3.2. Illustrative example

Based on the MATLAB Simulink modeling techniques described in the previous subsection, we implemented our ZNN-ML model (3.3) for computing the ML-weighted pseudoinverse with the following matrices:

$$M(t) = \begin{bmatrix} \sin(t) & \cos(t) \\ \cos(t) & -\sin(t) \end{bmatrix}, \quad K(t) = \begin{bmatrix} \sin(t) & \cos(t) \\ \cos(t) & -\sin(t) \end{bmatrix},$$

$$\text{and } L(t) = \begin{bmatrix} \sin(t) & \cos(t) \\ \cos(t) & -\sin(t) \end{bmatrix}.$$

The exact ML-weighted pseudoinverse is

$$K^{\dagger}_{ML}(t) = \begin{bmatrix} \sin(t) & \cos(t) \\ \cos(t) & -\sin(t) \end{bmatrix}.$$

A block diagram of the Simulink model of the ZNN-ML system (3.3) for online time-varying ML-weighted pseudoinverse is depicted in Figure 5. As expected, the ZNN-ML system converged quickly to the time-varying ML-weighted pseudoinverse.

5. Conclusion

The Zhang neural network is a kind of recurrent neural network that is effective on time-varying matrix problems. This paper presents a Zhang neural network model, called ZNN-ML model, for computing the ML-weighted pseudoinverse of a time-varying matrix. We prove the global and exponential convergence and robustness of the ZNN-ML model. We provide three alternative invertibility conditions for the robustness. In our model, the positive definiteness of the weight matrices is not required. Our numerical experiments on several test matrices demonstrate that our ZNN-ML model produces accurate results quickly. A comparison study shows that our model is superior over the conventional gradient-based neural network. A MATLAB Simulink implementation of our model is also presented.

Acknowledgements

We would like to thank Prof. M. Z. Nashed and two referees for their detailed comments on our presentation. We also thank Dr. Maolin Che and Xuezhong Wang for testing the examples.

Funding

This author was partially supported by Natural Science and Engineering Research Council (NSERC) of Canada under grant RGPIN-2014-04252 and International Cooperation Project of Shanghai Municipal Science and Technology Commission under grant no. 16510711200.This author is supported by the National Natural Science Foundation of China under grant no. 11771099 and International Cooperation Project of Shanghai Municipal Science and Technology Commission under grant no. 16510711200.

ORCID

Yimin Wei http://orcid.org/0000-0001-6192-0546

References

[1] Cai, J., Chen, G. (2002). The expression of $A^\dagger, A^\dagger_{MN}$ and its applications. *Numer Math: J. Chinese Univ*. 24(4):320–326. (in Chinese).
[2] Eldén, L. (1982). A weighted pseudoinverse, generalized singular values, and constrained least squares problems. *BIT*. 22(4):487–502. DOI: 10.1007/BF01934412.
[3] Galba, E.F., Neka, V.S., Sergienko, I.V. (2009). Weighted pseudoinverses and weighted normal pseudosolutions with singular weights. *Comput. Math. And Math. Phys*. . 49(8):1281–1363. DOI: 10.1134/S0965542509080016.
[4] Liao, B., Zhang, Y. (2014). Different complex ZFs leading to different complex ZNN models for time-varying complex generalized inverse matrices. *IEEE Trans. Neural Netw. Learn Syst*. 25(9):1621–1631. DOI: 10.1109/TNNLS.2013.2271779.
[5] M. Z. Nashed, editor. (1973). *Generalized inverses and applications*, Proceedings of an Advanced Seminar Sponsored by the Mathematics Research Center. *The*

University of Wisconsin-Madison, October, p. 8–10. New York, NY: Academic Press, Inc., 1976.

[6] Pian, J., Chen, G. (2004). A unified approach for the recursive computation of weighted generalized inverses. *J. Lanzhou Univ. (Nat. Sci.)*. 40(6):13–17.

[7] Rao, C. R., Mitra, S. K. (1971). *Generalized Inverse of Matrices and Its Applications*. New York: John Wiley & Sons.

[8] Rubini, L., Cancelliere, R., Gallinari, P., Grosso, A., Raiti, A.(2014) Computational experience with pseudoinversion-based training of neural networks using random projection matrices. In Agre Gennady, Hitzler Pascal, A. Krisnadhi Adila, and O. Kuznetsov Sergei, editors, *Artificial Intelligence: Methodology, Systems, and Applications*. New York, NY: Springer International Publishing, pp. 236–245.

[9] Wang, X., Wei, Y., Stanimirović, P. S. (2016). Complex neural network models for time-varying Drazin inverse. *Neural Comput.* 28(12):2790–2824. DOI: 10.1162/NECO_a_00866.

[10] Wei, M. (2006). *Theory and Computation of Generalized Least Squares Problem*. Beijing: Science Press (in Chinese).

[11] Wei, M., De Pierro, A.R. (2000). Upper perturbation bounds of weighted projections, weighted and constrained least squares problems. *SIAM J. Matrix Anal. & Appl.* 21(3):931–951. DOI: 10.1137/S0895479898336306.

[12] Wei, M., Zhang, B. (1994). Structures and uniqueness conditions of MK-weighted pseudoinverses. *BIT Numerical Mathematics*. 34(3):437–450. DOI: 10.1007/BF01935652.

[13] Wei, Y. (2000). Recurrent neural networks for computing weighted Moore-Penrose inverse. *Appl. Math. Comput.* 116(3):279–287. DOI: 10.1016/S0096-3003(99)00147-2.

[14] Xiao, L. (2015). A finite-time convergent neural dynamics for online solution of time-varying linear complex matrix equation. *Neurocomputing*. 167(1):254–259. DOI: 10.1016/j.neucom.2015.04.070.

[15] Xiao, L. (2016). A nonlinearly activated neural dynamics and its finite-time solution to time-varying nonlinear equation. *Neurocomputing*. 173(3):1983–1988. DOI: 10.1016/j.neucom.2015.08.031.

[16] Xiao, L. (2016). A nonlinearly-activated neurodynamics model and its finite-time solution to equality constrained quadratic optimization with nonstationary coefficients. *Appl. Soft Comput.* 40:252–259. DOI: 10.1016/j.asoc.2015.11.023.

[17] Xiao, L., Liao, B. (2016). A convergence-accelerated Zhang neural network and its solution application to Lyapunov equation. *Neurocomputing*. 193:213–218. DOI: 10.1016/j.neucom.2016.02.021.

[18] Xiao, L., Lu, R. (2015). Finite-time solution to nonlinear equation using recurrent neural dynamics with a specially-constructed activation function. *Neurocomputing*. 151(1):246–251. DOI: 10.1016/j.neucom.2014.09.047.

[19] Zhang, Y. (2005). Revisit the analog computer and gradient-based neural system for matrix inversion. In Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium, Mediterrean Conference on Control and Automation, Limassol, Cyprus, pp. 1411–1416.

[20] Zhang, Y., Chen, K. (2008). Global exponential convergence and stability of Wang neural network for solving online linear equations. *Electron. Lett.* 44(2):145–146. DOI: 10.1049/el:20081928.

[21] Zhang, Y., Chen, K., Ma, W., Li, X. (2007). MATLAB simulation of gradient-based neural network for online matrix inversion. In Third International Conference on Intelligent Computing (ICIC 2007), Proceedings, Qingdao, China, Volume 4682, pp. 98–109.

[22] Zhang, Y., Ge, S. (2005). Design and analysis of a general recurrent neural network model for time-varying matrix inversion. *IEEE Trans. Neural Netw.* 16(6): 1477–1490. DOI: 10.1109/TNN.2005.857946.

[23] Zhang, Y., Jiang, D., Wang, J. (2002). A recurrent neural network for solving Sylvester equation with time-varying coefficients. *IEEE Trans Neural Netw.* 13(5): 1053–1063. DOI: 10.1109/TNN.2002.1031938.

[24] Zhang, Y., Wang, J. (2002). Global exponential stability of recurrent neural networks for synthesizing linear feedback control systems via pole assignment. *IEEE Trans Neural Netw.* 13(3):633–644. DOI: 10.1109/TNN.2002.1000129.

[25] Zhang, Y., Yang, Y., Tan, N., Cai, B. (2011). Zhang neural network solving for time-varying full-rank matrix Moore-Penrose inverse. *Computing.* 92(2):97–121. DOI: 10.1007/s00607-010-0133-9.

[26] Zhang, Y., Yi, C. (2011). *Zhang Neural Networks and Neural-Dynamic Method.* New York, NY: Nova Science Publishers, Inc.

[27] Zielke, G. (1986). Report on test matrices for generalized inverses. *Computing.* 36(1-2): 105–162. DOI: 10.1007/BF02238196.

[28] Zivković, I. S., Stanimirović, P. S., Wei, Y. (2016). Recurrent neural network for computing outer inverse. *Neural Comput.* 28(5):970–998. DOI: 10.1162/NECO_a_00821.