Filomat 31:10 (2017), 2999–3014
https://doi.org/10.2298/FIL1710999S

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

ZNN Models for Computing Matrix Inverse Based on Hyperpower Iterative Methods

Igor Stojanović[a], Predrag S. Stanimirović[b], Ivan S. Živković[c], Dimitrios Gerontitis[d], Xue-Zhong Wang[e]

[a]*Goce Delčev University, Faculty of Computer Science, Goce Delčev 89, 2000 Štip, Macedonia*
[b]*University of Niš, Faculty of Sciences and Mathematics, Višegradska 33, 18000 Niš, Serbia*
[c]*Mathematical Institute of the Serbian Academy of Sciences and Arts, Kneza Mihaila 36, 11001 Belgrade, Serbia*
[d]*Aristoteleion Panepistimion, Thessalonikis, Greece*
[e]*School of Mathematical Sciences, Fudan University, Shanghai, 200433, P.R. China*

Abstract. Our goal is to investigate and exploit an analogy between the scaled hyperpower family (SHPI family) of iterative methods for computing the matrix inverse and the discretization of Zhang Neural Network (ZNN) models. A class of ZNN models corresponding to the family of hyperpower iterative methods for computing generalized inverses is defined on the basis of the discovered analogy. The Simulink implementation in Matlab of the introduced ZNN models is described in the case of scaled hyperpower methods of the order 2 and 3. Convergence properties of the proposed ZNN models are investigated as well as their numerical behavior.

1. Introduction and Preliminaries

Following the usual notation, the set of all $m \times n$ complex matrices is denoted by $\mathbb{C}^{m\times n}$, while the set of $m \times n$ complex matrices of rank r is denoted by $\mathbb{C}_r^{m\times n}$. Further, $\|A\|_1$, A^{H}, $\mathrm{rank}(A)$, $\mathcal{R}(A)$ and $\mathcal{N}(A)$ denote the matrix 2-norm, the conjugate transpose, the rank, the range space and the null space of $A \in \mathbb{C}^{m\times n}$. For $A \in \mathbb{C}^{n\times n}$, the smallest nonnegative integer j such that $\mathrm{rank}(A^{j+1}) = \mathrm{rank}(A^j)$ is called the index of A and is denoted by $\mathrm{ind}(A)$. The inverse of a nonsingular square matrix A is a matrix A^{-1} satisfying $AA^{-1} = A^{-1}A = I$, where I denotes the identity matrix of an appropriate order. Generalized inverses are defined in the case when A is rectangular or singular. They are defined as matrices possessing some properties of the inverse matrix, but not necessarily all of them. The key point in the investigation and computation of generalized inverses of $A \in \mathbb{C}_r^{m\times n}$ are Penrose equations with respect to unknown matrix X:

$$(1)\quad AXA = A \qquad (2)\quad XAX = X \qquad (3)\quad (AX)^{\mathrm{H}} = AX \qquad (4)\quad (XA)^{\mathrm{H}} = XA.$$

2010 *Mathematics Subject Classification*. Primary 15A09; Secondary 15A23, 65F20, 68T05
Keywords. Zhang neural network, matrix inverse, convergence, time-varying complex matrix, iterative methods
Received: 21 September 2016; Accepted: 11 January 2017
Communicated by Dragan Djurčić
The first and second author gratefully acknowledge support from the Project "Applying direct methods for digital image restoring" of the Goce Delčev University. The second author gratefully acknowledge support from the Research Project 174013 of the Serbian Ministry of Science.The third author gratefully acknowledge support from the Research Project III 044006 of the Serbian Ministry of Science. The 4th author is supported by the National Natural Science Foundation of China under grant 11271084 and International Cooperation Project of Shanghai Municipal Science and Technology Commission under grant 16510711200.
Email addresses: igor.stojanovik@ugd.edu.mk (Igor Stojanović), pecko@pmf.ni.ac.rs (Predrag S. Stanimirović), zivkovic.ivan83@gmail.com (Ivan S. Živković), dimitrios_gerontitis@yahoo.gr (Dimitrios Gerontitis), xuezhongwang77@126.com (Xue-Zhong Wang)

The set of $\mathcal{S}$-inverses satisfies some of equations (1)–(4) and it is denoted by $A\{\mathcal{S}\}$. A single element of the set $A\{1,2,3,4\}$ is called the Moore-Penrose inverse $A^\dagger$ of A.

The Drazin inverse of a square matrix $A \in \mathbb{C}^{n\times n}$ is the unique matrix $X \in \mathbb{C}^{n\times n}$ which fulfills the matrix equation (2) in conjunction with

$$(1^k)\ \ A^{l+1}X = A^l,\ l \geq \mathrm{ind}(A), \quad (5)\ \ AX = XA,$$

and it is denoted by $X = A^{\mathrm{D}}$ (for more details see [18]). In the case $\mathrm{ind}(A) = 1$, the Drazin inverse becomes the group inverse $X = A^{\#}$. The outer inverse of $A \in \mathbb{C}_r^{m\times n}$ with prescribed range T and null space S, denoted by $A^{(2)}_{T,S}$, satisfies the matrix equation (2) and two additional properties: $\mathcal{R}(X) = T$ and $\mathcal{N}(X) = S$. Various representations of the generalized inverse $A^{(2)}_{T,S}$ as well as corresponding algorithms for its computation have been investigated frequently in the last years. There exist two categories of the numerical algorithms: direct and iterative methods. The direct method means that the accurate solutions for the problem are computed in finite steps. An iterative method for computing $A^\dagger$ is a set of instructions for generating a sequence $\{Xk\}$ converging to $A^\dagger$. The instructions specify how to select the initial approximation X_0, how to proceed from X_k to X_{k+1} for each k, and when to stop, having obtained a reasonable approximation. Main results can be found at [14, 15, 20, 23].

One of the most important methods for computing the matrix inverse and various generalized inverses is the family of hyperpower iterations. These iterations possess an arbitrary order of the convergence $p \geq 2$, and are given by the standard form

$$X_{k+1} = X_k\left(I + R_k + \cdots + R_k^{p-1}\right) = X_k\sum_{i=0}^{p-1} R_k^i, \quad R_k = I - AX_k. \tag{1.1}$$

The hyperpower iterative family has been investigated extensively in a number of papers. The references [3, 6, 7, 9] can be highlighted as the most important.

All iterative methods, in general, require initial conditions which are strict and sometimes cannot be fulfilled easily. The continuous-time neural learning algorithms have emerged as parallel distributed computational models for real-time applications.

In recent years, many studies have been developed for developing gradient neural network (GNN) models and Zhang neural network (ZNN) models for computing the matrix inverse and various classes of generalized inverses. The authors of the paper [8] investigated five complex-valued ZNN models which are aimed to computation of time-varying complex matrix generalized inverses. ZNN models for online time-varying full-rank matrix pseudoinversion were considered in [24]. An RNN with the linear activation function for the Drazin inverse computation was proposed by Stanimirović, Zivković, and Wei in [17]. The relationship between the Zhang matrix inverse and the Drazin inverse, discovered in [25], leads to the same dynamic state equation which was considered in [17] in the time invariant matrix case. The dynamical equation and corresponding artificial recurrent neural network for computing the Drazin inverse of an arbitrary square real matrix, without any restriction on eigenvalues of its rank invariant powers, were proposed in [16]. A discrete-time model of ZNN for matrix inversion, which is depicted by a system of difference equations, was investigated in [27]. A general recurrent neural network model for online inversion of time-varying matrices was presented in [28]. The simulation and verification of such a ZNN were investigated in [26]. ZNN models for computing online time-varying Moore-Penrose inverse of a full-rank matrix were generalized, investigated and analyzed in [24]. Two complex Zhang neural network (ZNN) models for computing the Drazin inverse of arbitrary time-varying complex square matrix were presented in [21]. The design of the ZNNs defined in [21] is based on corresponding matrix-valued error functions arising from the limit representations of the Drazin inverse.

Our basic motivation is the fact that the scaled Newton method for the usual matrix inversion appears after the discretization of the Zhang Neural Network (ZNN) designed for the matrix inversion. The discretization was introduced in [27]. We intend to generalize the significant result
From Zhang neural network to Newton iteration for matrix inversion,

derived in [27], into the more general goal
From Zhang neural network to scaled hyperpower iterations for matrix inversion and vice versa.

More precisely, main goals in the present paper are summarized as follows.

(1) Generalize the discretization from [27] and consequently define the scaled hyperpower iterative methods (SHPI shortly) of arbitrary order $p \geq 2$.

(2) In addition, our intention is to define a ZNN model (called ZNNCM) whose discretization produces the scaled Chebyshev iterative method introduced in [13].

(3) Numerical behavior as well as the convergence properties of the ZNNCM model are investigated.

(4) A combination of the ZNNCM and the ZNNM model, called the ZNNHM model, is also defined and considered in numerical testing.

The remainder of the paper is organized as follows. Section 2 defines a generalization of the discretization of the ZNN model which was defined in [27]. The generalization describes the ZNN model corresponding to the hyperpower iterative method of an arbitrary order $p \geq 2$ as well as the discretization of this model, corresponding to the scaled hyperpower iterations (SHPI shortly) of an arbitrary order $p \geq 2$. The ZNN model corresponding to the third order hyperpower method, called ZNNCM, is investigated in details in Section 3. The convergence of the complex neural network model ZNNCM based on two activation functions is investigated in Section 4. Simulation results and comparison of the ZNNCM, ZNNNM and the ZNNHM models among themselves and with the GNN model are presented in Section 5.

2. Scaled Hyperpower Iterations as Discretized ZNN Models

It is assumed that the matrix A is a constant $n \times n$ nonsingular matrix. For the sake of completeness, we restate main steps of the discretization which was defined in [27]. The matrix-valued error-monitoring function (ZF) of the form

$$E(X(t), t) := AX(t) - I \tag{2.1}$$

was used to derive the dynamic equation determined by the general pattern

$$\frac{\mathrm{d}E(X(t), t)}{\mathrm{d}t} = -\Gamma \mathcal{H}\left(E(X(t), t)\right), \tag{2.2}$$

where $\Gamma \in \mathbb{R}^{n\times n}$ is a positive-definite matrix used to scale the inversion process and $\mathcal{H}(\cdot) : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ denotes an appropriate matrix-valued activation-function mapping. An application of the general pattern (2.2) on the Zhang error-monitoring function (2.1) in the case $\mathcal{H} = \mathcal{I}$ and $\Gamma = \gamma I$, where $\gamma > 0$ is a scalar-valued design parameter, leads to the following implicit dynamic equation of ZNN:

$$A\dot{X}(t) = -\gamma \left(AX(t) - I)\right). \tag{2.3}$$

Further, assume that the linear activation function $\mathcal{H} = \mathcal{I}$ is used and the discretization of the continuous-time model (2.3) is performed by using the Euler forward-difference rule

$$\dot{X}(t) \approx (X_{k+1} - X_k)/\tau,$$

where τ denotes the sampling time and $X_k = X(t = k\tau)$, $k = 1, 2, \ldots$. Then the discrete-time model of (2.3) is defined by

$$AX_{k+1} = AX_k - \beta\left(AX_k - I\right), \tag{2.4}$$

where $\beta = \tau\,\gamma > 0$ is the step size that should appropriately be selected for the convergence to the theoretical inverse A^{-1}. Since A is nonsingular, the implicit discrete-time ZNN model can be rewritten as

$$X_{k+1} = X_k - \beta A^{-1}\left(AX_k - I\right). \tag{2.5}$$

According to [28], the state matrix $X(t)$ converges to A^{-1} in the continuous-time ZNN model (2.3). Hence, it is justifiably to replace A^{-1} by its approximation X_k. This replacement yields the following explicit difference equation of the discrete-time ZNN for the nonsingular matrix inversion:

$$X_{k+1} = X_k - \beta X_k\ (AX_k - I) = X_k\,(I + \beta\,(I - AX_k))\,. \tag{2.6}$$

The iterative rule (2.6) is exactly of the form of the scaled Newton iteration for computing outer inverses with prescribed range and null space, introduced in [11, 12]:

$$X_{k+1} = (1+\beta)X_k - \beta X_k A X_k, \qquad X_0 = \alpha G, \quad \beta \in (0,1], \tag{2.7}$$

where $G \in \mathbb{C}_r^{m\times n}$ is a given matrix, α, β are real constants and $G \in \mathbb{C}_s^{n\times m}$ is a chosen matrix and $0 < s \leq r$. In the case $\beta = 1$ the iterative process (2.7) produces well known generalization of the Schultz iterative method, intended for computing outer inverses [4, 22].

Charif et all. in [2] developed a new fast online algorithms for motion estimation which is based on the Horn & Schunck algorithm with the Discrete Zhang Neural Networks (DZNN) defined by (2.6) and Simoncelli's matched-pair 5 tap filters. A novel implementation of the multi-dimensional Capon spectral estimator was proposed in [1]. The algorithm is derived using the discrete Zhang neural network for the online covariance matrix inversion.

In order to extend defined discretization, we start from the continuous-time model which is based on the error-monitoring function defined by the second and the third term of the hyperpower iterative process:

$$E_C(X(t),t) := I - AX(t) + (I - AX(t))^2 = 2I - 3AX(t) + (AX(t))^2\,. \tag{2.8}$$

In view of the general ZNN pattern (2.2), the Zhang error-monitoring function (2.8) leads to the following implicit dynamic equation

$$\begin{aligned} \dot{E_C}(X(t),t) &= -A\dot{X}(t) - A\dot{X}(t)\,(I - AX(t)) + (I - AX(t))\left(-A\dot{X}(t)\right) \\ &= -3A\dot{X}(t) + A\dot{X}(t)AX(t) + AX(t)A\dot{X}(t) = -\Gamma\mathcal{H}\,(2I - AX(t)\,(3I - AX(t)))\,. \end{aligned} \tag{2.9}$$

The expected convergence of $X(t)$ to A^{-1} approves the substitution $AX(t) = I$ in the left hand side of (2.9), which in the case $\Gamma = \gamma I$, $\mathcal{H} = I$ leads to

$$\begin{aligned} A\dot{X}(t) &= \gamma\,(2I - AX(t)\,(3I - AX(t))) \\ &= -\gamma\left(-(AX(t))^2 + 3AX(t) - 2I\right), \end{aligned} \tag{2.10}$$

where $X(0)$ is appropriately defined initial point. Further, the discrete-time model of (2.10) based on the Euler forward-difference rule is defined by

$$AX_{k+1} = AX_k + \beta\,(2I - AX_k\,(3I - AX_k))\,,$$

where $\beta = \tau\,\gamma > 0$ is the step size. After the replacement of A^{-1} by X_k, the implicit discrete-time ZNN model for the usual matrix inversion can be stated as

$$X_{k+1} = X_k\left(I + \beta\left(2I - 3AX_k + (AX_k)^2\right)\right), \tag{2.11}$$

i.e. in the form of scaled hyperpower iterative method of the order 3. This method was proposed by Srivastava and Gupta in [13] for estimating the Moore-Penrose inverse. The scaled hyperpower iterative method (2.11) of the order 3 is developed by extending the scaled hyperpower iterative method (2.7) of the order 2.

As a consequence, it is reasonable to investigate the ZNN model defined in (2.10), initiated by the ZF defined in (2.8).

Our intention is to extend just defined principle in the widest sense, which assumes an arbitrary hyperpower method of the order $p \geq 2$. In view of the previously exploited principle, the corresponding continuous-time model starts from the error-monitoring function defined by

$$E_H(X(t), t) := \sum_{i=1}^{p-1} R(t)^i = \sum_{i=1}^{p-1} (I - AX(t))^i .$$

The principle of mathematical induction reveals

$$\dot{R(t)}^i = -\sum_{l=0}^{i-1} (I - AX(t))^l A\dot{X}(t) (I - AX(t))^{i-1-l} .$$

Then the general ZNN design model (2.2) leads to the following implicit dynamic equation in the case $\mathcal{H} = I$:

$$-\sum_{i=1}^{p-1}\sum_{l=0}^{i-1} (I - AX(t))^l A\dot{X}(t) (I - AX(t))^{i-1-l} = -\gamma \sum_{i=1}^{p-1} (I - AX(t))^i . \tag{2.12}$$

After the substitution $AX(t) = I$ in the left hand side of the implicit dynamics (2.12), one can verify

$$A\dot{X}(t) = \gamma \sum_{i=1}^{p-1} (I - AX(t))^i . \tag{2.13}$$

The discretization of the ZNN model (2.13) corresponding to the Euler forward-difference rule is given as

$$AX_{k+1} = AX_k + \beta \sum_{i=1}^{p-1} (I - AX_k)^i , \ \beta = \tau\, \gamma > 0.$$

The inverse A^{-1} can be approximated by X_k, so that the implicit discrete-time ZNN model of (2.13), aimed for the matrix inversion, is given as

$$X_{k+1} = X_k \left(I + \beta \sum_{i=1}^{p-1} (I - AX_k)^i \right). \tag{2.14}$$

The iterative rule (2.14) is referred as the scaled hyperpower iterative methods (SHPI shortly) of an arbitrary order $p \geq 2$.

In conclusion, it is reasonable to define the ZNN model (2.3) as the continuous-time version of the scaled Chebyshev iterative method, in the same way as the ZNN model (2.10) represents the continuous-time version of the scaled Newton iterative method. A comparison between these two concurrent ZNN models will be investigated in the present article.

3. Neural Network Architecture

The graphical editor, customizable block libraries and solvers available in Matlab Simulink are used for modeling and simulating the proposed dynamic systems. As it was mentioned in [29], the ZNN modeling could be readily developed, expanded and finally realized by using Matlab Simulink tool. This fact was our motivation to use the Matlab Simulink tool in the implementation of defined ZNN models. The ZNN model (2.3) will be denoted by ZNNNM. Also, the ZNN model (2.10) is termed as ZNNCM. In addition, we define a hybrid method which starts from the ZNN model (2.3) and finishes with (2.10). Finally, GNN

denotes the gradient based neural network from [30] in the nonsingular case, corresponding to the case $G = A^{\mathrm{T}}$ in the RNN1 model.

The Simulink implementation of the ZNNNM model (2.3), restated in the equivalent form

$$\dot{X}(t) = (I - A)\dot{X}(t) - \gamma\mathcal{H}\left(AX(t) - I\right). \tag{3.1}$$

is presented in Figure 1.

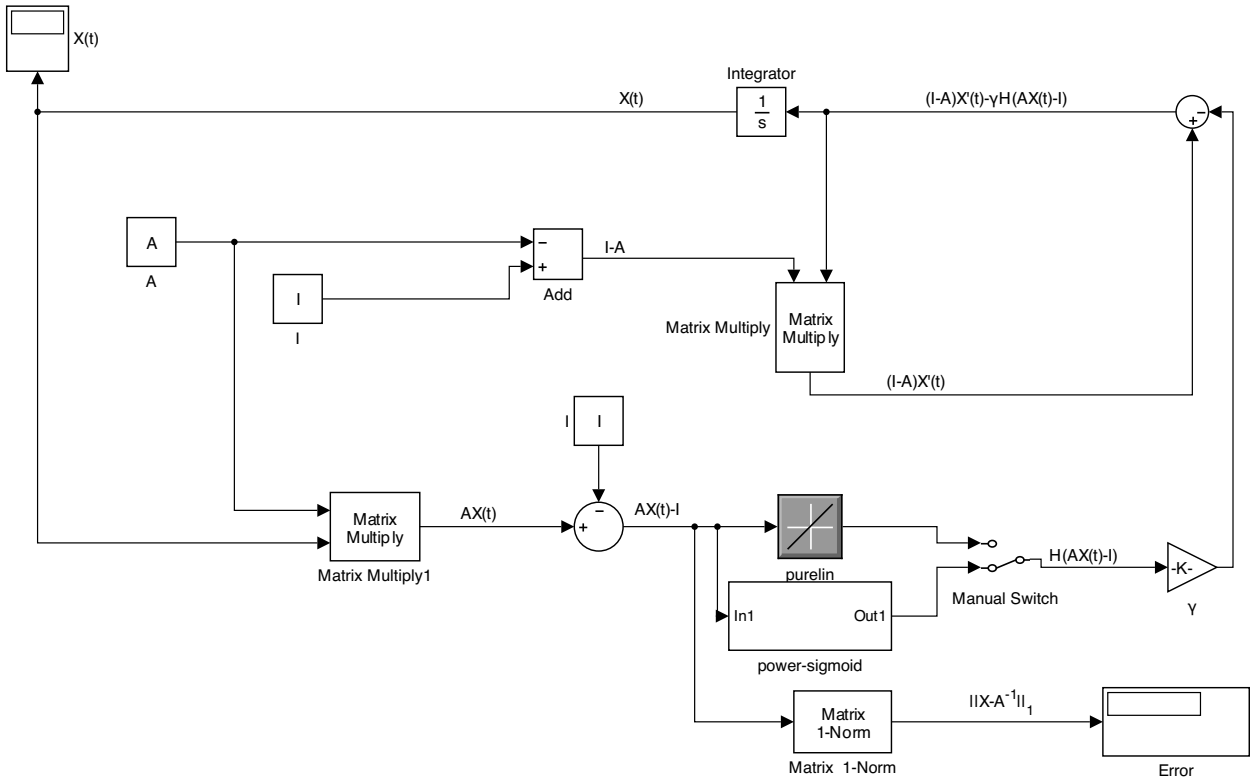

Figure 1: Simulink implementation of the ZNNNM model.

In order to ensure the implementation, (2.10) is transformed into the following equivalent form:

$$\dot{X}(t) = (I - A)\dot{X}(t) - \gamma\mathcal{H}\left(-(AX(t))^2 + 3AX(t) - 2I\right). \tag{3.2}$$

The Matlab Simulink implementation of the ZNNCM model, based on (3.2), is presented in Figure 2.

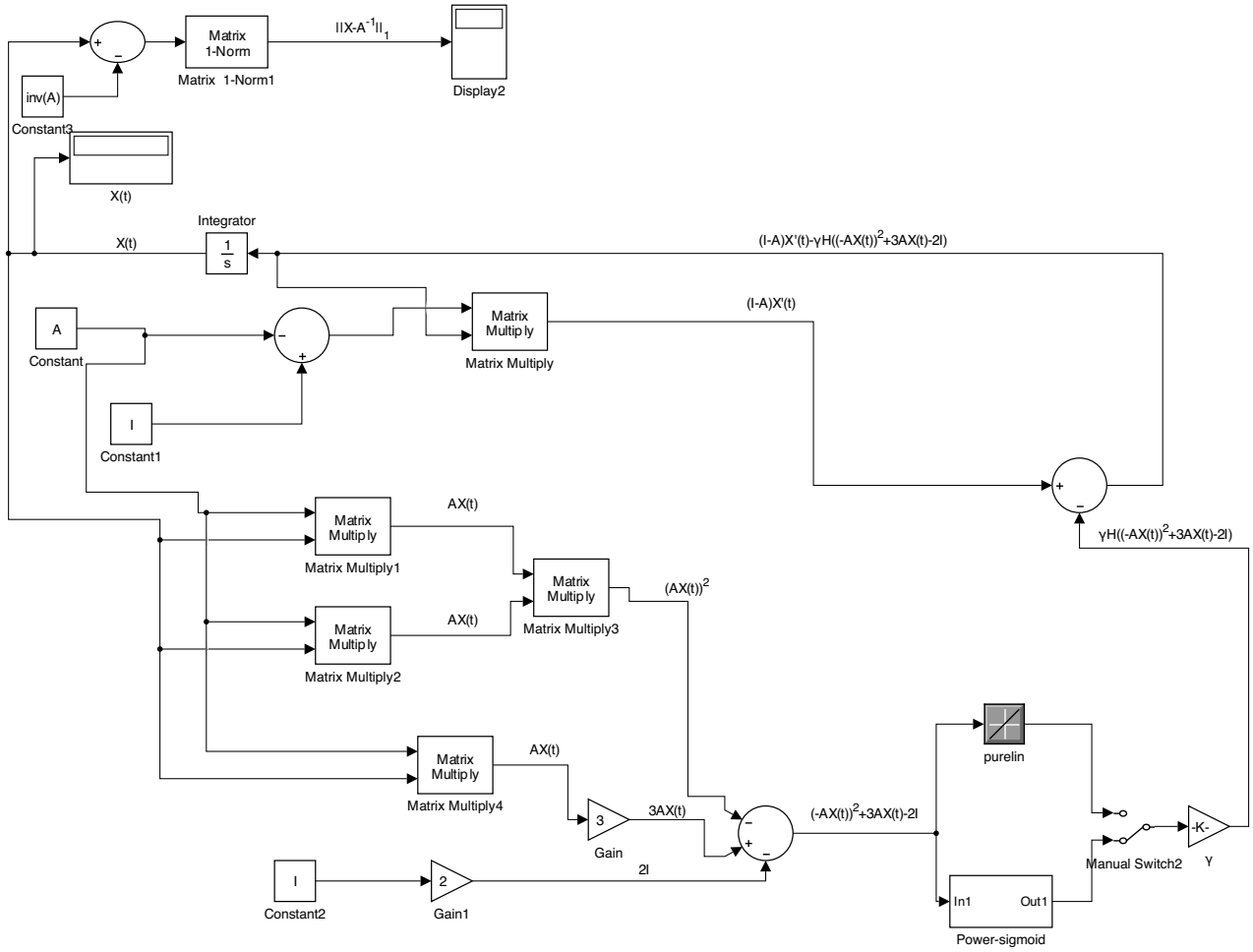

Figure 2: Simulink implementation of the ZNNCM model.

Two appropriate activation functions, introduced in [5], will be exploited in nodes of two developed ZNNs. Their definitions will be restated here in order to complete the presentation. It is assumed that $A \in \mathbb{C}^{n\times n}$ is written as $B + \iota C$, where $\iota = \sqrt{-1}$ denotes the imaginary unit and $B \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{n\times n}$ are two real matrices. The matrices B, C correspond to real and imaginary part of the complex entries of A, respectively. Additionally, let $\mathcal{F}(D)$ be an odd and monotonically increasing function element-wise applicable to elements of $D = (d_{kj}) \in \mathbb{R}^{n\times n}$ according to the rule $\mathcal{F}(D) = (f(d_{kj}))$, where $f(\cdot)$ is an odd and monotonically increasing function.

The type I activation function is defined by

$$\mathcal{H}_1(A) = \mathcal{H}_1(B + \iota C) = \mathcal{F}(B) + \iota\mathcal{F}(C). \tag{3.3}$$

Similarly, the type II activation function exploits the Hadamard product $U \circ V = (u_{kj}v_{kj})$ of matrices $U = (u_{kj})$ and $V = (v_{kj})$,and it is defined as

$$\mathcal{H}_2(A) = \mathcal{H}_2(B + \iota C) = \mathcal{F}(\Gamma) \circ \exp(\iota\Theta), \tag{3.4}$$

where $\Gamma = |B + \iota C| \in \mathbb{R}^{n\times n}$ and $\Theta = \Theta(B + \iota C) \in (-\pi, \pi]^{n\times n}$ denote element-wise modulus and the element-wise arguments, respectively, of the complex matrix $B + \iota C$. In sequel, we use the notation $\mathcal{H}_k$ as a universal replacement for $\mathcal{H}_1$ or $\mathcal{H}_2$.

The hybrid method starts using the ZNNCM method and then continues with the ZNNNM method. The starting point x_0 of the ZNNNM method is just the output of the ZNNCM method and the finishing time of the ZNNCM method is the initial time of the ZNNNM method. The hybrid method will be denoted by ZNNHM(t_0), where t_0 denotes the time when the ZNNNCM stops and ZNNNM continues. More precisely, the ZNNCM method evaluates in the time interval $[0, t_0]$, while the ZNNNM method evaluates in the time interval $[t_0, t]$, where $[0, t]$ denotes the considered time interval of the hybrid method. Since the ZNNNM model is globally exponentially convergent to the exact time-varying inverse $A(t)^{-1}$, the output x_0 of the ZNNCM method could be submitted as the initial point of the ZNNNM method.

4. Convergence of the ZNNCM Model

In this section, it is proven the convergence of the complex neural network model (2.10) based on both the activation functions $\mathcal{H}_1$ and $\mathcal{H}_2$.

Theorem 4.1. *Let the invertible complex matrix $A \in \mathbb{C}^{n\times n}$ be given. Then the state matrix $X(t) \in \mathbb{C}^{n\times n}$ of the complex neural network model* (2.10) *based on the activation function $\mathcal{H}_1$ converges to the matrix inverse A^{-1}, and the solution is stable in the sense of Lyapunov.*

Proof. Let $\tilde{X}(t) := A^{-1} - X(t)$. Then $X(t) = A^{-1} - \tilde{X}(t)$ and $\dot{\tilde{X}}(t) = -\dot{X}(t)$. Substituting the above two equations into (2.10) yields

$$A\dot{\tilde{X}}(t) = \gamma\mathcal{H}_1\left(-\left(I - A\tilde{X}(t)\right)^2 + 3\left(I - A\tilde{X}(t)\right) - 2I\right). \tag{4.1}$$

After substituting $X(t) = A^{-1} - \tilde{X}(t)$ in the ZF defined by (2.8), one can verify

$$E_C(\tilde{X}(t), t) = A\tilde{X}(t) + (A\tilde{X}(t))^2. \tag{4.2}$$

In the view of the definition of activation function $\mathcal{H}_1(\cdot)$, taking into account $E_C(t) = \mathbf{Re}(E_C(t)) + \iota\mathbf{Im}(E_C(t))$, the general model $\dot{E}(t) = -\gamma\mathcal{H}_1(E(t))$ splits into the following two equations in the real domain:

$$\mathbf{Re}(\dot{E_C}(t)) = -\gamma\mathcal{F}(\mathbf{Re}(E_C(t)))$$

and

$$\mathbf{Im}(\dot{E_C}(t)) = -\gamma\mathcal{F}(\mathbf{Im}(E_C(t))).$$

In order to verify the convergence, the Lyapunov function candidate is defined as

$$L(\tilde{X}(t), t) = L(t) = \frac{\|E_C(t)\|_F^2}{2} = \frac{\mathrm{Tr}\left(E_C(t)^{\mathrm{H}}E_C(t)\right)}{2}. \tag{4.3}$$

Then the following identities can be verified:

$$\begin{aligned}
\frac{\mathrm{d}L(t)}{\mathrm{d}t} &= \frac{\mathrm{Tr}\left(\dot{E_C}(t)^{\mathrm{H}}E_C(t) + E_C(t)^{\mathrm{H}}\dot{E_C}(t)\right)}{2} \\
&= -\frac{1}{2}\gamma\mathrm{Tr}\left\{\left(\mathcal{F}\left(\mathbf{Re}\left(E_C(t)\right)\right)^{\mathrm{T}} - \iota\mathcal{F}\left(\mathbf{Im}\left(E_C(t)\right)\right)^{\mathrm{T}}\right)\left(\mathbf{Re}\left(E(t)\right) + \iota\mathbf{Im}\left(E_C(t)\right)\right)\right. \\
&\qquad \left. + \left(\mathbf{Re}\left(E_C(t)\right)^{\mathrm{T}} - \iota\mathbf{Im}\left(E_C(t)\right)^{\mathrm{T}}\right)\left(\mathcal{F}\left(\mathbf{Re}\left(E_C(t)\right)\right)^{\mathrm{T}} + \iota\mathcal{F}\left(\mathbf{Im}\left(E_C(t)\right)\right)\right)\right\} \\
&= -\gamma\mathrm{Tr}\left\{\mathbf{Re}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Re}\left(E_C(t)\right)\right) + \mathbf{Im}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Im}\left(E_C(t)\right)\right)\right\}.
\end{aligned}$$

Since $\mathcal{F}(C) = (f(c_{kj}))$ and $f(\cdot)$ is an odd and monotonically increasing function, it follows that

$$\begin{aligned}
&\mathrm{Tr}\left\{\mathbf{Re}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Re}\left(E_C(t)\right)\right) + \mathbf{Im}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Im}\left(E_C(t)\right)\right)\right\} \\
&= \mathrm{Tr}\left\{\mathbf{Re}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Re}\left(E_C(t)\right)\right)\right\} + \mathrm{Tr}\left\{\mathbf{Im}\left(E_C(t)\right)^{\mathrm{T}}\mathcal{F}\left(\mathbf{Im}\left(E_C(t)\right)\right)\right\} \geq 0.
\end{aligned}$$

To simplify notation, let us denote (i, j)th element of $\mathbf{Re}\,(E_C(t))$ by e_{ij} and (i, j)th element of $\mathbf{Im}\,(E_C(t))$ by e'_{ij}. Then

$$\mathrm{Tr}\left\{\mathbf{Re}\,(E_C(t))^{\mathrm{T}}\,\mathcal{F}\,(\mathbf{Re}\,(E_C(t))) + \mathbf{Im}\,(E_C(t))^{\mathrm{T}}\,\mathcal{F}\,(\mathbf{Im}\,(E_C(t)))\right\} = \sum_j e_{ij} f(e_{ij}) + \sum_j e'_{ij} f(e'_{ij}) \geq 0$$

and finally

$$\frac{\mathrm{d}L(\tilde{X}(t),t)}{\mathrm{d}t} \begin{cases} < 0 & \text{if } E_C(\tilde{X}(t),t) \neq 0, \\ = 0 & \text{if } E_C(\tilde{X}(t),t) = 0. \end{cases}$$

Since $\tilde{X}(t) = 0$ is an equilibrium point of the system (4.1), and $E(0) = 0$ it follows that

$$\frac{\mathrm{d}L(\tilde{X}(t),t)}{\mathrm{d}t} \leq 0, \;\; \forall \tilde{X}(t) \neq 0.$$

As a consequence of the Lyapunov stability theory, the equilibrium state $\tilde{X}(t) = 0$ is stable. Since $\tilde{X}(t) := A^{-1} - X(t)$, we have $X(t) \to A^{-1}, t \to \infty$. □

Theorem 4.2. *Let the invertible complex matrix $A \in \mathbb{C}^{n\times n}$ be given. Then the state matrix $X(t) \in \mathbb{C}^{n\times n}$ of the complex neural network model* (2.10) *based on the activation function $\mathcal{H}_2$ converges to the matrix inverse A^{-1}, and the solution is stable in the sense of the Lyapunov.*

Proof. Analogically as in the proof of Theorem 4.1, the general model is given by

$$\dot{E_C}(t) = -\gamma \mathcal{H}_2(E(t)),$$

where $E(\tilde{X}(t), t) = E_C(t)$ is defined in (4.2). The definition of $\mathcal{H}_2(\cdot)$ implies

$$\mathcal{H}_2(E_C(t)) = \mathcal{F}\,(|E_C(t)|) \circ \exp(\iota\Theta(E_C(t))).$$

The time derivative of the Lyapunov function candidate (4.3) is equal to

$$\begin{aligned}
\frac{\mathrm{d}L(t)}{\mathrm{d}t} &= \frac{\mathrm{Tr}\left(E_C(t)^{\mathrm{H}}\dot{E}(t) + \dot{E_C}(t)^{\mathrm{H}}E_C(t)\right)}{2} \\
&= -\frac{1}{2}\gamma \mathrm{Tr}\left(E(t)^{\mathrm{H}}\mathcal{H}_2\left(E_C(t)\right) + E(t)\mathcal{H}_2\left(E_C(t)\right)^{\mathrm{H}}\right) \\
&= -\frac{1}{2}\gamma \mathrm{Tr}\left(E_C(t)^{\mathrm{H}}\mathcal{H}_2\left(E_C(t)\right) + \left(E_C(t)^{\mathrm{H}}\mathcal{H}_2\left(E_C(t)\right)\right)^{\mathrm{H}}\right) \\
&= -\gamma \mathrm{Tr}\left(\mathbf{Re}\left(E_C(t)^{\mathrm{H}}\mathcal{H}_2\left(E_C(t)\right)\right)\right) \\
&= -\gamma \mathrm{Tr}\left\{\mathbf{Re}\left[E_C(t)^{\mathrm{H}}\mathcal{F}\left(|E_C(t)|\right) \circ \exp\left(\iota\Theta(E_C(t))\right)\right]\right\}.
\end{aligned}$$

Since $E_C(t) = |E_C(t)| \circ \exp(\iota\Theta(E_C(t)))$, it follows that

$$\frac{\mathrm{d}L(t)}{\mathrm{d}t} = -\gamma \mathrm{Tr}\left\{\mathbf{Re}\left[\exp\left(-\iota\Theta\left(E_C(t)^{\mathrm{H}}\right) \circ \left|E_C(t)^{\mathrm{H}}\right|\right)\left(\mathcal{F}\left(|E_C(t)|\right) \circ \exp\left(\iota\Theta\left(E_C(t)\right)\right)\right)\right]\right\}.$$

Again, using that $\mathcal{F}(\cdot)$ is monotonically increasing, it follows the inequality $\mathcal{F}(|E_C(t)|) > 0$, for $E(t) \neq 0$, and $\mathcal{F}(|E_C(t)|) = 0$, for $E_C(t) = 0$ which implies that

$$\frac{\mathrm{d}L(\tilde{X}(t),t)}{\mathrm{d}t} \leq 0, \;\; \forall \tilde{X}(t) \neq 0.$$

According to the Lyapunov stability theory, the equilibrium state $\tilde{X}(t) = 0$ is stable and, $X(t) \to A^{-1}, t \to \infty$. □

5. Simulation Results and its Comparison

Example 5.1. As it was observed in [10], the GNN models are not appropriate for calculating the inverse of a matrix with a big condition number. So this is a reason to apply the ZNNCM model to a matrix with a big condition number. The following matrix A is considered for this purpose:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 & 32 \\ 1 & 3 & 9 & 27 & 81 & 243 \\ 1 & 4 & 16 & 64 & 256 & 1024 \\ 1 & 5 & 25 & 125 & 625 & 3125 \end{bmatrix}$$

with the condition number $\text{cond}(A) = 5.7689e{+}04$. The theoretical inverse of A is equal to

$$A^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{137}{60} & 5 & -5 & \frac{10}{3} & -\frac{5}{4} & \frac{1}{5} \\ \frac{15}{8} & -\frac{77}{12} & \frac{107}{12} & -\frac{13}{2} & \frac{61}{24} & -\frac{5}{12} \\ -\frac{17}{24} & \frac{71}{24} & -\frac{59}{12} & \frac{49}{12} & -\frac{41}{24} & \frac{7}{24} \\ \frac{1}{8} & -\frac{7}{12} & \frac{13}{12} & -1 & \frac{11}{24} & -\frac{1}{12} \\ -\frac{1}{120} & \frac{1}{24} & -\frac{1}{12} & \frac{1}{12} & -\frac{1}{24} & \frac{1}{120} \end{bmatrix}.$$

For $\gamma = 10^6$, with the Power-Sigmoid activation function and *ode45* solver, after $t = 10^{-5}s$ sec, the ZNNCM model gives the results ZNNCM(A) equal to

$$\begin{bmatrix} 0.999999999294662 & -0.000000000000000 & -0.000000000000000 & 0.000000000000000 & -0.000000000000000 & -0.000000000000000 \\ -2.283333331722813 & 4.999999996473304 & -4.999999996473306 & 3.333333330982192 & -1.249999999118328 & 0.199999999858933 \\ 1.874999998677495 & -6.416666662140734 & 8.916666660377395 & -6.499999995415274 & 2.541666664873933 & -0.416666666372777 \\ -0.708333332833721 & 2.958333331246700 & -4.916666663198751 & 4.083333330453185 & -1.708333332128381 & 0.291666666460944 \\ 0.124999999911833 & -0.583333332921884 & 1.083333332569217 & -0.999999999294657 & 0.458333333010054 & -0.083333333274555 \\ -0.008333333327456 & 0.041666666637277 & -0.083333333274555 & 0.083333333274555 & -0.041666666637278 & 0.008333333327456 \end{bmatrix}$$

with the absolute error $\|X(t) - A^{-1}\|_1 = 1.4106399215397e{-}08$. Trajectories of convergence behavior in 10^{-5}s under zero initial conditions in the ZNNCM model are shown in Figure 3.

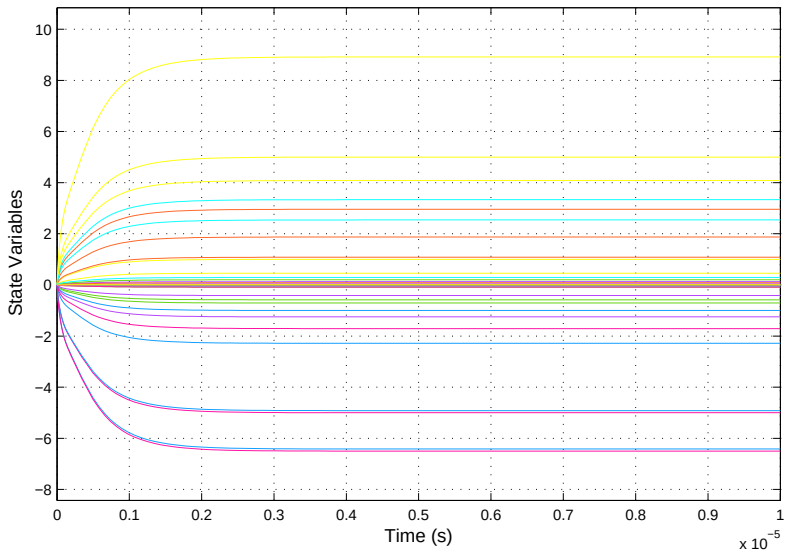

Figure 3: Trajectories in 10^{-5}s under zero initial conditions in the ZNNCM model.

Trajectories of the residual errors $\|X(t) - A^{-1}\|_1$ of the model ZNNCM are illustrated in Figure 4.

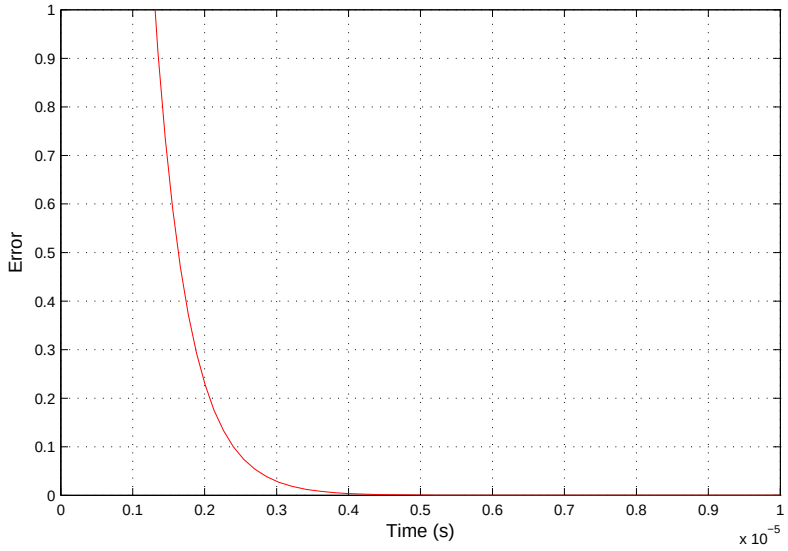

Figure 4: Trajectories of the residual errors of the model ZNNCM.

Trajectories of the residual errors $\|X(t) - A^{-1}\|_1$ of both the ZNNCM and ZNNNM models are illustrated in Figure 5.

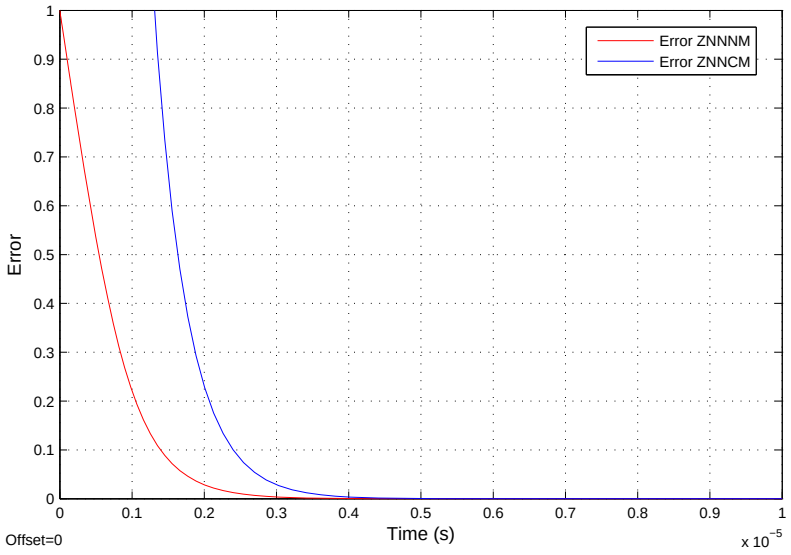

Figure 5: Trajectories of the residual errors of the models ZNNNM and ZNNCM.

The ZNNNM method produces the result with the absolute error $\|X(t) - A^{-1}\|_1 = 3.5446017745966e$–08 while the GNN model corresponding to the usual matrix inversion, from [19] does not achieve the convergence and stops with the absolute error equal to $\|X(t) - A^{-1}\|_1 = 19.509081114657$. So, the ZNNCM model can be used to compute the inverses of ill-conditioned matrices. This is one more advantage for the ZNNCM model over the GNN model.

In the subsequent examples, the matrix A is a randomly generated $n \times n$ matrix and $x(0)$ is a vectorization of a given $n \times n$ matrix $X(0)$. It is assumed that $x(0)$ is the same for all models ZNNNM, ZNNCM and GNN

in the actual table. The ordered triple $(t, n, solver)$ in headings of subsequent tables will include the time t, the dimension n of the input matrix and the used *Matlab* solver. Let us mention that the best results in all tables are marked in bold.

Example 5.2. According to Theorem 4.1 and Theorem 4.2, the solution of the complex neural network model (2.10) is stable in the sense of the Lyapunov. Therefore, it is desirable to choose the zero initial state $X(0)$. In this example, $X(0)$ is randomly generated $n \times n$ matrix in order to test behavior of the ZNNCM model. The activation function $\mathcal{H}$ is linear.

Table 5.1. Comparison of models ZNNNM, ZNNCM and GNN.

$(10^{-9}, 10, ode45)$			$(10^{-11}, 10, ode45)$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
ZNNNM	10^8	9.6055	ZNNNM	10^8	10.6051
ZNNCM	10^8	8.6367	**ZNNCM**	10^8	10.5938
GNN	10^8	10.4382	GNN	10^8	10.6126
ZNNNM	10^9	3.9053	ZNNNM	10^9	10.5101
ZNNCM	10^9	8.7346	**ZNNCM**	10^9	10.3980
GNN	10^9	9.8657	GNN	10^9	10.5877
ZNNNM	10^{10}	$4.8369e-04$	ZNNNM	10^{10}	9.6055
ZNNCM	10^{10}	0.0013	**ZNNCM**	10^{10}	8.6367
GNN	10^{10}	8.4838	GNN	10^{10}	10.4382
ZNNNM	10^{11}	$2.8613e-06$	**ZNNNM**	10^{11}	3.9053
ZNNCM	10^{11}	$7.9558e-05$	ZNNCM	10^{11}	8.7346
GNN	10^{11}	2.1187	GNN	10^{11}	9.8657
ZNNNM	10^{12}	$1.0572e-05$	**ZNNNM**	10^{12}	$4.8369e-04$
ZNNCM	10^{12}	$3.4264e-05$	ZNNCM	10^{12}	0.0013
GNN	10^{12}	$9.8238e-06$	GNN	10^{12}	8.4838
ZNNNM	10^{13}	$2.1264e-05$	**ZNNNM**	10^{13}	$2.8613e-06$
ZNNCM	10^{13}	0.0010	ZNNCM	10^{13}	$7.9558e-05$
GNN	10^{13}	$5.7854e-06$	GNN	10^{13}	2.1187
ZNNNM	10^{14}	$2.0846e-06$	ZNNNM	10^{14}	$1.0572e-05$
ZNNCM	10^{14}	0.0015	ZNNCM	10^{14}	$3.4264e-05$
GNN	10^{14}	$3.3838e-06$	**GNN**	10^{14}	$9.8238e-06$

According to the results arranged in Table 5.1, the following observations could be emphasized.
(a) The property "as large as possible" of the scaling parameter γ is valid for ZNNNM and GNN methods, especially for the GNN method, and it is not applicable in the case of the ZNNCM method.
(b) The ZNNCM method produces better results within the smaller time period $[0, 10^{-11}]$ than in the time period $[0, 10^{-9}]$.
(c) The ZNNCM method produces the best results during the time $[0, 10^{-11}]$ and smaller values of γ: $\gamma = 10^8, 10^9, 10^{10}$.
(d) In the case of a nonzero initial state $X(0)$, the ZNNCM method should be used in a short time $[0, 10^{-11}]$ and with smaller values of $\gamma \leq 10^{10}$.

Example 5.3. In this example, $X(0)$ is randomly generated $n \times n$ matrix and the activation function $\mathcal{H}$ is linear.

Table 5.2. Comparison of models ZNNNM, ZNNCM and ZNNHM.

$(10^{-11}, 30, ode45)$			$(10^{-11}, 30, ode15s)$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$

$(10^{-11}, 30, ode45)$			$(10^{-11}, 30, ode15s)$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
ZNNNM	10^{10}	12.6304	**ZNNNM**	10^{10}	16.3735
ZNNCM	10^{10}	11.9629	ZNNCM	10^{10}	$1.8502e+03$
ZNNHM(10^{-12})	10^{10}	11.2585	ZNNHM(10^{-12})	10^{10}	19.4103
ZNNHM(10^{-14})	10^{10}	11.4939	ZNNHM(10^{-14})	10^{10}	18.2060
ZNNNM	10^{11}	5.1351	**ZNNNM**	10^{11}	6.6561
ZNNCM	10^{11}	$5.1555e+15$	ZNNCM	10^{11}	7.6935
ZNNHM(10^{-12})	10^{11}	4.7521	ZNNHM(10^{-12})	10^{11}	680.4448
ZNNHM(10^{-14})	10^{11}	4.6613	ZNNHM(10^{-14})	10^{11}	7.4102
ZNNNM	10^{12}	$6.3565e-04$	ZNNNM	10^{12}	$8.2654e-04$
ZNNCM	10^{12}	$6.1391e+15$	**ZNNCM**	10^{12}	$4.7134e-04$
ZNNHM(10^{-12})	10^{12}	$2.7946e+11$	ZNNHM(10^{-12})	10^{12}	$7.7231e-04$
ZNNHM(10^{-14})	10^{12}	$5.6655e-04$	ZNNHM(10^{-14})	10^{12}	$9.6796e-04$
ZNNNM	10^{13}	$6.7814e-06$	ZNNNM	10^{13}	$5.4011e-12$
ZNNCM	10^{13}	$1.0634e+16$	**ZNNCM**	10^{13}	$1.7082e-13$
ZNNHM(10^{-12})	10^{13}	$2.7107e-05$	ZNNHM(10^{-12})	10^{13}	$3.7240e-11$
ZNNHM(10^{-14})	10^{13}	$6.2916e-06$	ZNNHM(10^{-14})	10^{13}	$1.9452e-10$
ZNNHM(10^{-16})	10^{13}	$3.5050e-06$	ZNNHM(10^{-16})	10^{13}	$3.0346e-12$
ZNNNM	10^{14}	$1.9529e-05$	**ZNNNM**	10^{14}	$3.2919e-14$
ZNNCM	10^{14}	$5.1555e+15$	ZNNCM	10^{14}	$8.8654e-14$
ZNNHM(10^{-12})	10^{14}	$7.3399e-05$	ZNNHM(10^{-12})	10^{14}	$3.8212e-14$
ZNNHM(10^{-14})	10^{14}	$7.3399e-05$	ZNNHM(10^{-14})	10^{14}	$3.6538e-14$
ZNNHM(10^{-16})	10^{14}	$6.1610e-06$	ZNNHM(10^{-16})	10^{14}	$4.2286e-14$

The following observations rise from the numerical results arranged in Table 5.2.
(a) The hybrid method ZNNHM produces the best results in the case when the underlying solver is `ode45`.
(b) the ZNNNM or CNNCM give the best results in the case when the underlying solver is `ode15s`.
(c) The solver `ode15s` is more appropriate than `ode45` with respect to the ZNNCM method in the case when X_0 is a nonzero value.
(d) ZNNCM model is sensitive on the choice of the initial point X_0 and the best choice is $X_0 = 0$.

Example 5.4. In the left column of Table 5.3 this example, $X(0)$ is randomly generated $n \times n$ matrix and $X(0)$ $n \times n$ zero matrix in the right column. The activation function $\mathcal{H}$ is Power Sigmoid activation function.

Table 5.3. Comparison of models ZNNNM, ZNNCM and ZNNHM with Power Sigmoid activation function $\mathcal{H}$, defined by the parameter $p = 3$.

$(10^{-7}, 20, ode15s)$, X_0 is arbitrary			$(10^{-7}, 30, ode15s)$, $X_0 = 0$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
ZNNNM	10^{12}	$2.7645e-14$	**ZNNNM**	10^{12}	$1.4903e-12$
ZNNCM	10^{12}	$4.3998e-14$	ZNNCM	10^{12}	$6.1530e-12$
ZNNHM(10^{-12})	10^{12}	$2.3267e-14$	ZNNHM(10^{-12})	10^{12}	$1.6522e-12$
ZNNHM(10^{-14})	10^{12}	$2.7345e-14$	ZNNHM(10^{-14})	10^{12}	$1.7446e-12$
ZNNHM(10^{-16})	10^{12}	$2.9818e-14$	ZNNHM(10^{-16})	10^{12}	$1.6834e-12$
ZNNNM	10^{13}	$2.6716e-14$	ZNNNM	10^{13}	$1.7769e-12$
ZNNCM	10^{13}	$4.5023e-14$	ZNNCM	10^{13}	$4.1402e-12$
ZNNHM(10^{-12})	10^{13}	$3.1053e-14$	ZNNHM(10^{-12})	10^{13}	$1.7335e-12$
ZNNHM(10^{-14})	10^{13}	$2.7970e-14$	**ZNNHM**(10^{-14})	10^{13}	$1.6732e-12$
ZNNHM(10^{-16})	10^{13}	$2.8306e-14$	ZNNHM(10^{-16})	10^{13}	$2.0054e-12$

$(10^{-7}, 20, ode15s)$, X_0 is arbitrary			$(10^{-7}, 30, ode15s)$, $X_0 = 0$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
ZNNNM	10^{14}	$2.7645e-14$	ZNNNM	10^{14}	$5.3491e-12$
ZNNCM	10^{14}	$4.3998e-14$	ZNNCM	10^{14}	$5.4503e-12$
ZNNHM(10^{-12})	10^{14}	$2.3267e-14$	**ZNNHM**(10^{-12})	10^{14}	$1.9956e-12$
ZNNHM(10^{-14})	10^{14}	$2.7345e-14$	ZNNHM(10^{-14})	10^{14}	$2.0058e-12$
ZNNHM(10^{-16})	10^{14}	$2.9818e-14$	ZNNHM(10^{-16})	10^{14}	$2.4750e-12$

Table 5.4. Comparison of models ZNNNM, ZNNCM and ZNNHM with Power Sigmoid activation function defined by the parameter $p = 3$.

$(10^{-10}, 30, ode15s)$, $X_0 = 0$			$(10^{-5}, 30, ode15s)$, $X_0 = 0$		
Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
ZNNNM	10^{12}	$1.2136e-12$	ZNNNM	10^{12}	$1.6829e-12$
ZNNCM	10^{12}	$1.8007e-09$	ZNNCM	10^{12}	$3.7154e-12$
ZNNHM(10^{-12})	10^{12}	$1.2410e-11$	ZNNHM(10^{-12})	10^{12}	$2.3492e-12$
ZNNHM(10^{-14})	10^{12}	$1.8332e-12$	**ZNNHM**(10^{-14})	10^{12}	$1.3554e-12$
ZNNHM(10^{-16})	10^{12}	$1.8565e-12$	ZNNHM(10^{-16})	10^{12}	$1.8953e-12$
ZNNNM	10^{13}	$1.6428e-12$	ZNNNM	10^{13}	$1.9011e-12$
ZNNCM	10^{13}	$4.4561e-11$	ZNNCM	10^{13}	$5.9375e-12$
ZNNHM(10^{-12})	10^{13}	$1.7544e-12$	ZNNHM(10^{-12})	10^{13}	$1.7805e-12$
ZNNHM(10^{-14})	10^{13}	$1.5256e-12$	**ZNNHM**(10^{-14})	10^{13}	$1.7413e-12$
ZNNHM(10^{-16})	10^{13}	$1.6547e-12$	ZNNHM(10^{-16})	10^{13}	$2.0168e-12$
ZNNNM	10^{14}	$1.7526e-12$	ZNNNM	10^{14}	$1.9145e-12$
ZNNCM	10^{14}	$4.4716e-12$	ZNNCM	10^{14}	$4.6704e-12$
ZNNHM(10^{-12})	10^{14}	$1.7646e-12$	ZNNHM(10^{-12})	10^{14}	$1.9448e-12$
ZNNHM(10^{-14})	10^{14}	$1.2410e-12$	**ZNNHM**(10^{-14})	10^{14}	$1.5412e-12$
ZNNHM(10^{-16})	10^{14}	$2.0978e-12$	ZNNHM(10^{-16})	10^{14}	$1.9367e-12$

The following conclusion arises from the results presented in Table 5.3 and Table 5.4:
(a) The ZNNHM method gives best results for appropriately selected intermediate time t_0. This value is, in most cases, equal to $t_0 = 10^{-14}$.

Example 5.5. The results produced by the Simulink and based on the power sigmoid activation function are arranged in Table 5.5.

Table 5.5. Comparison of models ZNNNM and ZNNCM with the power sigmoid activation function.

Method	γ	$\|X - A^{-1}\|_1$	Method	γ	$\|X - A^{-1}\|_1$
$(10^{-8}, 10, ode45)$, $X_0 = 0$			$(10^{-8}, 10, ode15s)$, $X_0 = 0$		
ZNNNM	10^6	0.99000742007452	ZNNNM	10^6	0.99000742007452
ZNNCM	10^6	11.796704121711	ZNNCM	10^6	11.796704121712
ZNNHM(10^{-10})	10^6	0.98921005121423	**ZNNHM**(10^{-10})	10^6	0.98921005121423
$(10^{-5}, 10, ode45)$, $X_0 = 0$			$(10^{-5}, 10, ode15s)$, $X_0 = 0$		
ZNNNM	10^6	$1.7721525005302e-09$	ZNNNM	10^6	$1.7721525005302e-09$
ZNNCM	10^6	$8.9282757020914e-09$	ZNNCM	10^6	$8.9286161242264e-09$
ZNNHM(10^{-8})	10^6	$1.5373390357755e-09$	ZNNHM(10^{-8})	10^6	$1.5373790308806e-09$
$(10^{-3}, 10, ode45)$, $X_0 = 0$			$(10^{-3}, 10, ode15s)$, $X_0 = 0$		
ZNNNM	10^6	$2.7877256580469e-15$	ZNNNM	10^6	$2.776852376599e-15^*$
ZNNCM	10^6	$9.048317650695e-15$	ZNNCM	10^6	$7.549516567451e-15$
ZNNHM(10^{-5})	10^6	$2.7947546454768e-15$	**ZNNHM**(10^{-5})	10^6	$2.140363747233e-15$

The sign star in Table 5.5 means that the ZNNM model stopped the computation with the message "Relative tolerance of 1.0E-15 is too small, setting relative tolerance to 2.8421709430404007E-14".

Example 5.6. As it was observed in Example 5.3, `ode15s` is more appropriate solver than `ode45` with respect to the ZNNCM method in the case when X_0 is a nonzero value. Comparison of the results produced by the ZNNCM and based on the linear the power sigmoid defined by $p = 3$ activation function and `ode45` and `ode15s` solvers are arranged in Table 5.6. Now, the initial point is $X_0 = 0$.

Table 5.6. Comparison of `ode45` and `ode15s` solvers in ZNNCM.

Solver	γ	$\|X - A^{-1}\|_1$	Solver	γ	$\|X - A^{-1}\|_1$
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-8}, 10, ode45)$	10^6	11.796704121711	$(10^{-8}, 10, ode45)$	10^6	11.796704121711
$(10^{-8}, 10, ode15s)$	10^6	11.796706187324	$(10^{-8}, 10, ode15s)$	10^6	11.796706181998
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-5}, 10, ode45)$	10^6	$8.9291933014213e-09$	$(10^{-5}, 10, ode45)$	10^6	$8.9291446458972e-09$
$(10^{-5}, 10, ode15s)$	10^6	$9.1885836162042e-09$	$(10^{-5}, 10, ode15s)$	10^6	$8.9691688520688e-09$
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-3}, 10, ode45)$	10^6	$8.0971731383772e-07$	$(10^{-3}, 10, ode45)$	10^6	$9.5967148786014e-07$
$(10^{-3}, 10, ode15s)$	10^6	$8.6042284408450e-15$	$(10^{-3}, 10, ode15s)$	10^6	$1.0158540675320e-14$
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-8}, 10, ode45)$	10^7	11.796704121711	$(10^{-8}, 10, ode45)$	10^7	11.796704121711
$(10^{-8}, 10, ode15s)$	10^7	11.796705368789	$(10^{-8}, 10, ode15s)$	10^7	11.796704581549
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-5}, 10, ode45)$	10^7	$8.9291800342561e-09$	$(10^{-5}, 10, ode45)$	10^7	$8.9287367777130e-09$
$(10^{-5}, 10, ode15s)$	10^7	$1.1175724984325e-08$	$(10^{-5}, 10, ode15s)$	10^7	$9.3463157213591e-09$
$\mathcal{H}$ is linear			$\mathcal{H}$ is power sigmoid		
$(10^{-4}, 10, ode45)$	10^7	$6.3143934525556e-15$	$(10^{-4}, 10, ode45)$	10^7	$6.1944325618920e-15$
$(10^{-4}, 10, ode15s)$	10^7	$7.5495165674511e-15$	$(10^{-4}, 10, ode15s)$	10^7	$6.9781294338186e-15$

According to results arranged in Table 5.6, the significant difference between the sovers `ode45` and `ode15s` is observable only in the configurations $(10^{-3}, 10, ode45)$ and $(10^{-3}, 10, ode15s)$. In this case, `ode15s` is significantly better choice. This further implies that the solver `ode15s` is more appropriate for smaller values of γ and longer periods of time. In general, this observation leads to the conclusion that `ode15s` leads to a faster convergence of the ZNNCM model, which ensures its better convergence for smaller values of γ. Also, there is no significant difference in numerical results caused by the choice between the linear and the power sigmoid activation functions.

6. Conclusion

An analogy between the scaled hyperpower family (SHPI family) of iterative methods for computing the matrix inverse and the discretization of Zhang Neural Network (ZNN) models is observed. On the basis of the discovered analogy, a family of ZNN models corresponding to the family of hyperpower iterative methods are defined. The ZNN model corresponding to the hyperpower method of the order 2 (resp. of the order 3) is denoted as ZNNNM (resp. ZNNCM). The implementation in Matlab Simulink of the introduced ZNN models is described in the case of the scaled hyperpower methods of the order 2 and 3.

Derived simulation results indicate that the results derived by the ZNNCM method are not favorable. But, the ZNNCM model becomes useful in the initialization of the ZNNNM method. For the time being, it is very difficult to determine or estimate the optimal value of the decisive time moment t_0. These investigations should be interesting topic for further research. In the current research, we recommend only

heuristics and verification. Additionally, it is observable that ZNNCM is most sensible to the choice of the initial approximation X_0.

Also, general conclusion is that an alternative to matrix iterations is found. The proposed alternative is based on the ZNN model an its Simulink implementation.

References

[1] A. Benchabane, A. Bennia, F. Charif, A. Taleb-Ahmed, Multi-dimensional Capon spectral estimation using discrete Zhang neural networks, Multidim. Syst. Sign. Process. 24 (2013), 583–598.

[2] F. Charif, A. Benchabane, N. Djedi, A. Taleb-Ahmed, Horn & Schunck meets a discrete Zhang neural networks for computing 2D optical flow, The International Conference on Electronics & Oil: From Theory to Applications, March 05-06, 2013, Ouargla, Algeria.

[3] J.-J. Climent, N. Thome, Y. Wei, A geometrical approach on generalized inverses by Neumann-type series, Linear Algebra Appl. 332-334 (2001), 533-540.

[4] D.S. Djordjević, P.S. Stanimirović, Y. Wei, The representation and approximation of outer generalized inverses, Acta Math. Hungar. 104 (2004), 1–26.

[5] S. Li, Y. Li, Nonlinearly activated neural network for solving time-varying complex Sylvester equation, IEEE Trans. Cybern. 44 (2013), 1397–1407.

[6] W. Li, L. Juan, T. Qiao, A family of iterative methods for computing Moore-Penrose inverse of a matrix, Linear Algebra Appl. 438 (2013), 47–56.

[7] W. Li, Z. Li, A family of iterative methods for computing the approximate inverse of a square matrix and inner inverse of a non-square matrix, Appl. Math. Comput. 215 (2010), 3433–3442.

[8] B. Liao, Y. Zhang, Different complex ZFs leading to different complex ZNN models for time-varying complex generalized inverse matrices, IEEE Trans. Neural Network Learn. Syst. 25 (2014), 1621–1631.

[9] X. Liu, H. Jin, Y. Yu, Higher-order convergent iterative method for computing the generalized inverse and its application to Toeplitz matrices, Linear Algebra Appl. 439 (2013), 1635-1650.

[10] S. Osowski, Neural networks in interpolation problems, Neurocomputing 5 (1993), 105–118.

[11] M.D. Petković, P.S. Stanimirović, Iterative method for computing Moore-Penrose inverse based on Penrose equations, J. Comput. Appl. Math. 235 (2011), 1604–1613.

[12] M.D. Petković, P.S. Stanimirović, Two improvements of the iterative method for computing Moore-Penrose inverse based on Penrose equations, J. Comput. Appl. Math. 267 (2014), 61–71.

[13] S. Srivastava, D.K. Gupta, A third order iterative method for $A^{\dagger}$, Int. J. Computing Science and Mathematics, 4 (2013), 140–151.

[14] X. Sheng, G. Chen, Full-rank representation of generalized inverse $A^{(2)}_{T,S}$ and its applications, Comput. Math. Appl. 54 (2007), 1422–1430.

[15] X. Sheng, G. Chen, Several representations of generalized inverse $A^{(2)}_{T,S}$ and their applications, Int. J. Comput. Math. 85 (2008), 1441–1453.

[16] P.S. Stanimirović, I. Živković, Y. Wei, Recurrent neural network approach based on the integral representation of the Drazin inverse, Neural Computation, 27 (10) (2015), 2107–2131.

[17] P.S. Stanimirović, I. Živković, Y. Wei, Recurrent neural network for computing the Drazin inverse, IEEE Transactions on Neural Networks and Learning Systems, 26 (2015), 2830–2843.

[18] G. Wang, Y. Wei, S. Qiao, Generalized Inverses: Theory and Computations, Science Press, Beijing/New York, 2004.

[19] J. Wang, A recurrent neural network for real-time matrix inversion, Appl. Math. Comput. 55 (1993), 89–100.

[20] H. Wang, M. Wei X. Liu, Several representations of {2}-inverses, Arab. J. Sci. Eng. 36 (6) (2011), 1161–1169.

[21] X.-Z. Wang, Y. Wei, P.S. Stanimirović, Complex neural network models for time-varying Drazin inverse, Neural Comput. 28 (2016), doi:10.1162/NECO a 00866.

[22] Y. Wei, H. Wu, The representation and approximation for the generalized inverse $A^{(2)}_{T,S}$, Appl. Math. Comput. 135 (2003), 263–276.

[23] Y. Wei, A characterization and representation of the generalized inverse $A^{(2)}_{T,S}$ and its applications, Linear Algebra Appl. 280 (1998), 79–86.

[24] Y. Zhang, Y. Yang, N. Tan, B. Cai, Zhang neural network solving for time-varying full-rank matrix Moore-Penrose inverse, Computing 92 (2011), 97–121.

[25] Y. Zhang, B. Qiu, L. Jin, D. Guo, Infinitely many Zhang functions resulting in various ZNN models for time-varying matrix inversion with link to Drazin inverse, Information Processing Letters 115 (2015), 703–706.

[26] Y. Zhang, C. Yi, W. Ma, Simulation and verification of Zhang neural network for online time–varying matrix inversion, Simulation Modelling Practice and Theory 17 (2009), 1603–1617.

[27] Y. Zhang, W. Ma, B. Cai, From Zhang neural network to Newton iteration for matrix inversion, IEEE Transactions on Circuits and Systems-I: Regular Papers 56(7) (2009), 1405–1415.

[28] Y. Zhang, S. Sam Ge, Design and analysis of a general recurrent neural network model for time-varying matrix inversion, IEEE Transactions on Neural Networks 16(6) (2005), 1477–1490.

[29] Y. Zhang, X. Guo, W. Ma, K. Chen, B. Cai, MATLAB Simulink modeling and simulation of Zhang neural network for online time-varying matrix inversion, IEEE International Conference on Networking, Sensing and Control, ICNSC 2008.

[30] I. Živković, P.S. Stanimirović, Y. Wei, Recurrent neural network for computing outer inverses, Neural Computation 28:5 (2016), 970–998.