mathematics MDPI

Article

Zeroing Neural Network for Pseudoinversion of an Arbitrary Time-Varying Matrix Based on Singular Value Decomposition

Mariya Kornilova [1], Vladislav Kovalnogov [1], Ruslan Fedorov [1], Mansur Zamaleev [1], Vasilios N. Katsikis [2], Spyridon D. Mourtas [2] and Theodore E. Simos [1,3,4,5,6,*]

[1] Laboratory of Inter-Disciplinary Problems in Clean Energy Production, Ulyanovsk State Technical University, 32 Severny Venetz Street, 432027 Ulyanovsk, Russia; vesk@ulstu.ru (M.K.); kvn@ulstu.ru (V.K.); r.fedorov@ulstu.ru (R.F.); mansur_zamaleev@mail.ru (M.Z.)
[2] Department of Economics, Division of Mathematics and Informatics, National and Kapodistrian University of Athens, Sofokleous 1 Street, 10559 Athens, Greece; vaskatsikis@econ.uoa.gr (V.N.K.); spirosmourtas@gmail.com (S.D.M.)
[3] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 40402, Taiwan
[4] Data Recovery Key Laboratory of Sichun Province, Neijing Normal University, Neijiang 641100, China
[5] Section of Mathematics, Department of Civil Engineering, Democritus University of Thrace, 67100 Xanthi, Greece
[6] Department of Mathematics, University of Western Macedonia, 50100 Kozani, Greece
[*] Correspondence: tsimos.conf@gmail.com

Abstract: Many researchers have investigated the time-varying (TV) matrix pseudoinverse problem in recent years, for its importance in addressing TV problems in science and engineering. In this paper, the problem of calculating the inverse or pseudoinverse of an arbitrary TV real matrix is considered and addressed using the singular value decomposition (SVD) and the zeroing neural network (ZNN) approaches. Since SVD is frequently used to compute the inverse or pseudoinverse of a matrix, this research proposes a new ZNN model based on the SVD method as well as the technique of Tikhonov regularization, for solving the problem in continuous time. Numerical experiments, involving the pseudoinversion of square, rectangular, singular, and nonsingular input matrices, indicate that the proposed models are effective for solving the problem of the inversion or pseudoinversion of time varying matrices.

Keywords: singular value decomposition (SVD); zeroing neural network (ZNN); Moore–Penrose inverse; Tikhonov regularization; dynamical system

MSC: 15A09; 65F20; 68T05

check for updates

Citation: Kornilova, M.; Kovalnogov, V.; Fedorov, R.; Zamaleev, M.; Katsikis, V.N.; Mourtas, S.D.; Simos, T.E. Zeroing Neural Network for Pseudoinversion of an Arbitrary Time-Varying Matrix Based on Singular Value Decomposition. *Mathematics* **2022**, *10*, 1208. https://doi.org/10.3390/math10081208

Academic Editor: Viktor N. Orlov

Received: 11 March 2022
Accepted: 6 April 2022
Published: 7 April 2022

1. Introduction and Preliminaries

In this paper, the zeroing neural network (ZNN) approach is used to address the problem of calculating the inverse or pseudoinverse of an arbitrary time-varying (TV) real matrix. On the one hand, the pseudoinverse, or Moore–Penrose inverse, of $A \in \mathbb{R}^{m\times n}$ is the unique matrix $A^{\dagger}$, such that the system of Penrose equations holds for $X := A^{\dagger}$ [1–3]:

$$A = AXA,\ X = XAX,\ (XA)^{\mathrm{T}} = XA,\ (AX)^{\mathrm{T}} = AX, \tag{1}$$

where A^{T} denotes the transpose of A. Note that, if A is a nonsingular square matrix, $A^{\dagger}$ becomes the usual inverse A^{-1}. On the other hand, the singular value decomposition (SVD) of $A \in \mathbb{R}^{m\times n}$ is a factorization of the form [4]:

$$A = USV^{\mathrm{T}}, \tag{2}$$

where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices, i.e., $U^{\mathrm{T}} = U^{-1}$ and $V^{\mathrm{T}} = V^{-1}$, while $S \in \mathbb{R}^{m\times n}$ is a rectangular (or square, in the case $m = n$) diagonal matrix with the singular values of A on its main diagonal. SVD is frequently used to compute the inverse or pseudoinverse of a matrix, while commonly existing in fields of scientific research, such as medical treatment and industrial applications, lattice computing [5], automatic classification of electromyograms [6], and face recognition [7]. In a recent work [8], the authors provided a zeroing neural network for computing the singular value decomposition of an arbitrary matrix. This work move things one step further, by designing a new ZNN model for calculating the inverse or pseudoinverse of an arbitrary TV matrix based on the singular value decomposition. For comparison purposes, we build another model based on direct pseudoinversion in accordance with the paper [9], and the experiments section demonstrates the efficacy of the proposed SVD model.

Zhang et al. in [10], developed a ZNN design for generating online solutions to TV problems. It is worth noting that most ZNN based dynamical systems fall under the category of recurrent neural networks (RNN) that are designed to find equation zeros. As a consequence, numerous valuable research findings have been presented in the literature. Addressing generalized inversion problems [11,12], tensor and matrix inversion problems [13], systems of linear equations [14,15], systems of matrix equations [14,16], quadratic optimization problems [17], and diverse matrix functions approximation [18,19] are the main applications of ZNNs. The first stage in developing ZNN dynamics is to design an error function $E(t)$ that is tailored to the underlying problem, commonly known as the Zhang function [20]. The second stage takes advantage of the proper dynamical evolution that follows:

$$\dot{E}(t) = \frac{\mathrm{d}E(t)}{\mathrm{d}t} = -\lambda\mathcal{F}(E(t)), \tag{3}$$

where $\dot{E}(t) \in \mathbb{R}^{m\times n}$ is the time derivative of $E(t) \in \mathbb{R}^{m\times n}$, $\lambda > 0$ is the design parameter that is used for scaling the convergence, while $\mathcal{F}(\cdot) : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ means elementwise utilization of an odd and increasing activation function on $E(t)$. In our research, we will consider the ZNN evolution (3) under the linear activation function. That is,

$$\dot{E}(t) = \frac{\mathrm{d}E(t)}{\mathrm{d}t} = -\lambda E(t). \tag{4}$$

This work's key points may be summarized as below:

- A novel ZNN approach, which is based on SVD, is employed for solving the problem of calculating the pseudoinverse of an arbitrary TV real matrix.
- Two ZNN models for calculating the pseudoinverse of an arbitrary TV matrix are offered: one called ZNNSVDP, which is based on SVD, and the other called ZNNP, which is based on a more direct approach to the problem and is offered for comparison purposes.
- Four numerical experiments, involving the pseudoinversion of square, rectangular, singular, and nonsingular input matrices, indicate that both models are effective for solving the problem and that the ZNNSVDP model converges to the problem's solution faster than the ZNNP model.

Additionally, it is worth mentioning some of the paper's general notations: the symbols $\mathbf{1}_n, \mathbf{0}_n$ denote a vector in $\mathbb{R}^n$ consisting of ones and zeros, respectively; $\mathbf{O}_{n\times n} \in \mathbb{R}^{n\times n}$ denotes a zero matrix of $n \times n$ dimensions; $I_n \in \mathbb{R}^{n\times n}$ denotes the identity $n \times n$ matrix; $\otimes$ denotes Kronecker product; $\mathrm{vec}(\cdot)$ denotes the vectorization technique; $\odot$ denotes the Hadamard (or element wise) product; and $\|\cdot\|_F$ denotes the matrix Frobenius norm.

The paper is constituted as follows. Sections 2 and 3, respectively, define and analyse the ZNNSVDP and ZNNP models. Section 4 presents and discusses the results of four numerical experiments employing the pseudoinversion of square, rectangular, singular, and nonsingular input matrices. Lastly, the final remarks and conclusions are offered in Section 5.

2. Time-Varying Pseudoinverse Computation Based on SVD

This section presents and analyses the ZNNSVDP model for calculating the pseudoinverse of an arbitrary TV real matrix. Considering a smooth TV matrix $A(t) \in \mathbb{R}^{m\times n}$, the inverse or pseudoinverse of $A(t)$ based on SVD (2) is the following:

$$\begin{cases} A^{-1}(t) = V(t)S^{-1}(t)U^{\mathrm{T}}(t), \ m = n = \operatorname{rank}(A(t)) \\ A^{\dagger}(t) = V(t)S^{\dagger}(t)U^{\mathrm{T}}(t), \quad \text{otherwise,} \end{cases} \tag{5}$$

where $U(t) \in \mathbb{R}^{m\times m}$ and $V(t) \in \mathbb{R}^{n\times n}$ are TV orthogonal matrices, and $S(t) \in \mathbb{R}^{m\times n}$ is a rectangular (or square, in the case $m = n$) diagonal matrix with the singular values of $A(t)$ on its main diagonal. Here, we consider decomposition so that the singular values of $A(t)$ are in descending order on the main diagonal of $S(t)$. Based on (2) and (5), the ZNNSVDP model considers the following group of error functions for calculating the inverse or pseudoinverse of $A(t)$:

$$\begin{cases} E_1(t) = A(t)V(t) - U(t)S(t) \\ E_2(t) = U^{\mathrm{T}}(t)U(t) - I_m \\ E_3(t) = V^{\mathrm{T}}(t)V(t) - I_n \\ E_4(t) = X(t) - V(t)Y(t)U^{\mathrm{T}}(t), \end{cases} \tag{6}$$

where $X(t)$ is the desired solution of the problem, i.e., the inverse or pseudoinverse of $A(t)$, and

$$Y(t) = \begin{cases} S^{-1}(t), \ m = n = \operatorname{rank}(A(t)) \\ S^{\dagger}(t), \quad \text{otherwise.} \end{cases} \tag{7}$$

The following proposition about the structure and construction of the pseudoinverse of a diagonal matrix is offered, whereas [21] provides a full examination of this proposition.

Proposition 1. *For a rectangular (or a square singular) diagonal matrix $B \in \mathbb{R}^{m\times n}$, let $b_1, b_2 \ldots, b_w$ with $w = \operatorname{rank}(B)$ signify the elements of the main diagonal of B. Then, the pseudoinverse matrix of B is the following:*

$$B^{\dagger} = \begin{bmatrix} \bar{B}^{-1} & \mathbf{O}_{w\times(n-w)} \\ \mathbf{O}_{(m-w)\times w} & \mathbf{O}_{(m-w)\times(n-w)} \end{bmatrix}, \quad \text{with} \quad \bar{B}^{-1} = \begin{bmatrix} \frac{1}{b_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{b_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{b_w} \end{bmatrix}. \tag{8}$$

In addition, the first time derivative of (6) is the following:

$$\begin{cases} \dot{E}_1(t) = \dot{A}(t)V(t) + A(t)\dot{V}(t) - \dot{U}(t)S(t) - U(t)\dot{S}(t) \\ \dot{E}_2(t) = \dot{U}^{\mathrm{T}}(t)U(t) + U^{\mathrm{T}}(t)\dot{U}(t) \\ \dot{E}_3(t) = \dot{V}^{\mathrm{T}}(t)V(t) + V^{\mathrm{T}}(t)\dot{V}(t) \\ \dot{E}_4(t) = \dot{X}(t) - \dot{V}(t)Y(t)U^{\mathrm{T}}(t) - V(t)\dot{Y}(t)U^{\mathrm{T}}(t) - V(t)Y(t)\dot{U}^{\mathrm{T}}(t), \end{cases} \tag{9}$$

where the first time derivative of $Y(t)$ is the following [22]:

$$\dot{Y}(t) = \begin{cases} \dot{S}^{-1}(t) = -S^{-1}(t)\dot{S}(t)S^{-1}(t), \quad m = n = \operatorname{rank}(A(t)) \\ \dot{S}^{\dagger}(t) = \ -S^{\dagger}(t)\dot{S}(t)S^{\dagger}(t) + (S^{\dagger}(t)(S^{\dagger}(t))^{\mathrm{T}})\dot{S}^{\mathrm{T}}(t)(I_m - S(t)S^{\dagger}(t)) \\ \qquad\qquad +(I_n - S^{\dagger}(t)S(t))\dot{S}^{\mathrm{T}}(t)((S^{\dagger}(t))^{\mathrm{T}}S^{\dagger}(t)), \quad \text{otherwise,} \end{cases} \tag{10}$$

or equivalent

$$\begin{aligned}\dot{Y}(t) = &- Y(t)\dot{S}(t)Y(t) + (Y(t)Y^{\mathrm{T}}(t))\dot{S}^{\mathrm{T}}(t)(I_m - S(t)Y(t)) \\ &+ (I_n - Y(t)S(t))\dot{S}^{\mathrm{T}}(t)(Y^{\mathrm{T}}(t)Y(t)).\end{aligned} \tag{11}$$

Notice that (11) is simplified to $\dot{Y}(t) = -Y(t)\dot{S}(t)Y(t)$ for $m = n = \mathrm{rank}(A(t))$. Then, combining (6), (9) and (11) with the ZNN design under the linear activation function (4), the following may be acquired:

$$\begin{cases}\dot{A}(t)V(t) + A(t)\dot{V}(t) - \dot{U}(t)S(t) - U(t)\dot{S}(t) = -\lambda E_1(t) \\ \dot{U}^{\mathrm{T}}(t)\ U(t) + U^{\mathrm{T}}(t)\dot{U}(t) = -\lambda E_2(t) \\ \dot{V}^{\mathrm{T}}(t)\ V(t) + V^{\mathrm{T}}(t)\dot{V}(t) = -\lambda E_3(t) \\ \dot{X}(t) - \dot{V}(t)Y(t)U^{\mathrm{T}}(t) - V(t)\big(-Y(t)\dot{S}(t)Y(t) + (Y(t)Y^{\mathrm{T}}(t))\dot{S}^{\mathrm{T}}(t)(I_m - S(t)Y(t)) \\ \qquad + (I_n - Y(t)S(t))\dot{S}^{\mathrm{T}}(t)(Y^{\mathrm{T}}(t)Y(t))\big)U^{\mathrm{T}}(t) - V(t)Y(t)\dot{U}^{\mathrm{T}}(t) = -\lambda E_4(t).\end{cases} \tag{12}$$

Using vectorization and the Kronecker product, the dynamics of (12) are modified as follows:

$$\begin{cases}-(S^{\mathrm{T}}(t)\otimes I_m)\mathrm{vec}(\dot{U}(t)) + (I_n\otimes A(t))\mathrm{vec}(\dot{V}(t)) - (I_n\otimes U(t))\mathrm{vec}(\dot{S}(t)) \\ \qquad = \mathrm{vec}(-\lambda E_1(t) - \dot{A}(t)V(t)) \\ (U^{\mathrm{T}}(t)\otimes I_m)\mathrm{vec}(\dot{U}^{\mathrm{T}}(t)) + (I_m\otimes U^{\mathrm{T}}(t))\mathrm{vec}(\dot{U}(t)) = \mathrm{vec}(-\lambda E_2(t)) \\ (V^{\mathrm{T}}(t)\otimes I_n)\mathrm{vec}(\dot{V}^{\mathrm{T}}(t)) + (I_n\otimes V^{\mathrm{T}}(t))\mathrm{vec}(\dot{V}(t)) = \mathrm{vec}(-\lambda E_3(t)) \\ -(I_m\otimes V(t)Y(t))\mathrm{vec}(\dot{U}^{\mathrm{T}}(t)) - (U(t)Y^{\mathrm{T}}(t)\otimes I_n)\mathrm{vec}(\dot{V}(t)) - \mathbf{K}_1(t) + I_{mn}\mathrm{vec}(\dot{X}(t)) \\ \qquad = \mathrm{vec}(-\lambda E_4(t)),\end{cases} \tag{13}$$

where

$$\begin{aligned}\mathbf{K}_1(t) = &-\big(U(t)Y^{\mathrm{T}}(t)\otimes V(t)Y(t)\big)\mathrm{vec}(\dot{S}(t)) \\ &+ \big(U(t)(I_m - S(t)Y(t))^{\mathrm{T}}\otimes V(t)Y(t)Y^{\mathrm{T}}(t)\big)\mathrm{vec}(\dot{S}^{\mathrm{T}}(t)) \\ &+ \big(U(t)Y^{\mathrm{T}}(t)Y(t)\otimes V(t)(I_n - Y(t)S(t))\big)\mathrm{vec}(\dot{S}^{\mathrm{T}}(t))\end{aligned} \tag{14}$$

Note that (13) must be simplified in order to produce a simple and explicit dynamical model that may easily calculate $U(t)$, $V(t)$, $S(t)$, and $X(t)$. As a result, the following lemmas about vectorization and the Kronecker product are offered, whereas [23] provides a full examination of Lemmas' 1 and 2 content.

Lemma 1.

$$(L^{\mathrm{T}}\otimes A)\mathrm{vec}(X) = \mathrm{vec}(Y).$$

Lemma 2. *For $B \in \mathbb{R}^{m\times m}$, let $\mathrm{vec}(B) \in \mathbb{R}^{m^2}$ signify the matrix B vectorization. The following occurs:*

$$\mathrm{vec}(B^{\mathrm{T}}) = Q_m\,\mathrm{vec}(B), \tag{15}$$

where $Q_m \in \mathbb{R}^{m^2\times m^2}$ is a constant permutation matrix defined exclusively by m.

The following, Algorithm 1, presents an algorithmic process for obtaining the permutation matrix Q_m in (15), which corresponds to a matrix of $m\times m$ dimensions. Note that the notations eye(.) and reshape(.) in Algorithm 1 have the typical notion of the related MATLAB functions [24].

Algorithm 1 Permutation matrix calculation

```
Require: The rows or columns number $m$ of a square matrix $B \in \mathbb{R}^{m\times m}$.
 1: procedure PERMUTATION_MATRIX($m$)
 2:     Set $a =$eye$(m^2)$ and $b =$reshape$(1 : m^2, m, m)$
 3:     return $Q = a(:,$reshape$(b', 1, m^2))$
 4: end procedure
Ensure: $Q_m$, i.e., the permutation matrix.
```

Furthermore, because $S(t)$ and $S^{\mathrm{T}}(t)$ are rectangular (or square, in the case $m = n$) diagonal matrices, just the nonzero elements of $\dot{S}(t)$ and $\dot{S}^{\mathrm{T}}(t)$ that are placed in their main diagonal must be obtained. By doing so, we may confine $S(t)$ to being a diagonal matrix, while also reducing the dimensions of (13). Hence, employing the nonzero elements on the main diagonal of $S(t)$ and $S^{\mathrm{T}}(t)$, whose number is $w = \mathrm{rank}(A(t))$, we utilize the equations $\mathrm{vec}(\dot{S}(t)) = G_1\dot{\mathbf{s}}(t)$ and $\mathrm{vec}(\dot{S}^{\mathrm{T}}(t)) = G_2\dot{\mathbf{s}}(t)$, respectively, to replace $\dot{S}(t)$ and $\dot{S}^{\mathrm{T}}(t)$ in (14), where the matrices $G_1, G_2 \in \mathbb{R}^{mn\times w}$ are operational matrices that can be calculated using the algorithmic procedure presented Algorithm 2. Additionally, the notation sum(.), min(.), zeros(.), mod(.) and floor(.) in Algorithm 2 have the typical notion of the related MATLAB functions [24].

Algorithm 2 Operational matrix calculation

```
Require: The number of the rows and columns, respectively, $m$ and $n$ of a matrix $B \in \mathbb{R}^{m\times n}$,
    and $w = \mathrm{rank}(B)$.
 1: procedure OPERATIONAL_MATRIX($m, n, w$)
 2:     if $w < \min(m, n)$ then
 3:         Set $h = w$
 4:     else
 5:         Set $h = m$
 6:     end if
 7:     Set $G =$zeros$(mn, w)$
 8:     for $k = 1 : mn$ do
 9:         Set $c =$mod$(k - 1, h) + 1$ and $d =$floor$(\frac{k-1}{h}) + 1$
10:         if $d == c$ then
11:             Set $G(k, c) = 1$
12:         end if
13:     end for
14:     return $G$
15: end procedure
Ensure: The operational matrix $G$.
```

Based on the aforementioned discussion, (13) can be reformulated as follows:

$$\begin{cases} -(S^{\mathrm{T}}(t)\otimes I_m)\mathrm{vec}(\dot{U}(t)) + (I_n\otimes A(t))\mathrm{vec}(\dot{V}(t)) - (I_n\otimes U(t))G_1\dot{\mathbf{s}}(t) \\ \qquad = \mathrm{vec}(-\lambda E_1(t) - \dot{A}(t)V(t)) \\ (U^{\mathrm{T}}(t)\otimes I_m)Q_m\mathrm{vec}(\dot{U}(t)) + (I_m\otimes U^{\mathrm{T}}(t))\mathrm{vec}(\dot{U}(t)) = \mathrm{vec}(-\lambda E_2(t)) \\ (V^{\mathrm{T}}(t)\otimes I_n)Q_n\mathrm{vec}(\dot{V}(t)) + (I_n\otimes V^{\mathrm{T}}(t))\mathrm{vec}(\dot{V}(t)) = \mathrm{vec}(-\lambda E_3(t)) \\ -(I_m\otimes V(t)Y(t))\mathrm{vec}(\dot{U}^{\mathrm{T}}(t)) - (U(t)Y^{\mathrm{T}}(t)\otimes I_n)\mathrm{vec}(\dot{V}(t)) - \mathbf{K}_2(t)\dot{\mathbf{s}}(t) \\ \qquad + I_{mn}\mathrm{vec}(\dot{X}(t)) = \mathrm{vec}(-\lambda E_4(t)), \end{cases} \tag{16}$$

where

$$\begin{aligned} \mathbf{K}_2(t) = &-\big(U(t)Y^{\mathrm{T}}(t)\otimes V(t)Y(t)\big)G_1 \\ &+\big(U(t)(I_m - S(t)Y(t))^{\mathrm{T}}\otimes V(t)Y(t)Y^{\mathrm{T}}(t)\big)G_2 \\ &+\big(U(t)Y^{\mathrm{T}}(t)Y(t)\otimes V(t)(I_n - Y(t)S(t))\big)G_2 \end{aligned} \tag{17}$$

As a result, setting

$$
\begin{aligned}
&Z_1(t)=\begin{bmatrix} -(S^{\mathrm{T}}(t)\otimes I_m) \\ (U^{\mathrm{T}}(t)\otimes I_m)Q_m+(I_m\otimes U^{\mathrm{T}}(t)) \\ \mathbf{0}_{n^2\times m^2} \\ -(I_m\otimes V(t)Y(t)) \end{bmatrix}, \quad Z_3(t)=\begin{bmatrix} -(I_n\otimes U(t))G_1 \\ \mathbf{0}_{m^2\times w} \\ \mathbf{0}_{n^2\times w} \\ -\mathbf{K}_2(t) \end{bmatrix}, \\
&Z_2(t)=\begin{bmatrix} (I_n\otimes A(t)) \\ \mathbf{0}_{m^2\times n^2} \\ (V^{\mathrm{T}}(t)\otimes I_n)Q_n+(I_n\otimes V^{\mathrm{T}}(t)) \\ -(U(t)Y^{\mathrm{T}}(t)\otimes I_n) \end{bmatrix}, \quad Z_4(t)=\begin{bmatrix} \mathbf{0}_{mn\times mn} \\ \mathbf{0}_{m^2\times mn} \\ \mathbf{0}_{n^2\times mn} \\ I_{mn} \end{bmatrix}, \\
&Z(t)=\begin{bmatrix} Z_1(t) & Z_2(t) & Z_3(t) & Z_4(t) \end{bmatrix},
\end{aligned}
\tag{18}
$$

$$
q(t)=\begin{bmatrix} \mathrm{vec}(-\lambda E_1(t)-\dot{A}(t)V(t)) \\ \mathrm{vec}(-\lambda E_2(t)) \\ \mathrm{vec}(-\lambda E_3(t)) \\ \mathrm{vec}(-\lambda E_4(t)) \end{bmatrix}^{\mathrm{T}}, \quad \dot{\mathbf{x}}=\begin{bmatrix} \mathrm{vec}(\dot{U}(t)) \\ \mathrm{vec}(\dot{V}(t)) \\ \mathrm{vec}(\dot{\mathbf{s}}(t)) \\ \mathrm{vec}(\dot{X}(t)) \end{bmatrix}, \quad \mathbf{x}=\begin{bmatrix} \mathrm{vec}(U(t)) \\ \mathrm{vec}(V(t)) \\ \mathrm{vec}(\mathbf{s}(t)) \\ \mathrm{vec}(X(t)) \end{bmatrix}, \tag{19}
$$

we propose the following ZNN model:

$$
Z^{\mathrm{T}}(t)Z(t)\dot{\mathbf{x}}(t) = Z^{\mathrm{T}}(t)q(t), \tag{20}
$$

where $Z^{\mathrm{T}}(t)Z(t)$ is a singular mass matrix. To solve the singularity problem, the Tikhonov regularization is used and (20) is converted into:

$$
(Z^{\mathrm{T}}(t)Z(t) + \beta I_{m^2+n^2+w+mn})\dot{\mathbf{x}}(t) = Z^{\mathrm{T}}(t)q(t), \tag{21}
$$

where $\beta \geq 0$ signifies the regularization parameter. The ZNN model (21) is termed as the ZNNSVDP model and can be solved efficiently with an appropriate `ode` Matlab solver. The exponential convergence of the ZNNSVDP model (21) to the theoretical TV inverse or pseudoinverse of the input matrix $A(t)$ is proven in Theorem 1.

Remark 1. *According to MATLAB's* `ode` *solvers syntax [24], the mass matrix is a symmetric matrix M that expresses the connection between the time derivative $\dot{x}$ of the generalized coordinate vector x of a system, by the equation:*

$$
M\dot{x} = x.
$$

Theorem 1. *Let $U(t) \in \mathbb{R}^{m\times m}$, $V(t), \in \mathbb{R}^{n\times n}$, $S(t) \in \mathbb{R}^{m\times n}$ be differentiable and $S(t)$ be a rectangular diagonal matrix. The ZNNSVDP model* (21)*, starting from any initial value $\mathbf{x}(0)$, converges exponentially to the theoretical TV inverse or pseudoinverse of the input matrix $A(t)$.*

Proof. In order to obtain the solution $\mathbf{x}(t)$, which corresponds to the TV inverse or pseudoinverse of the input matrix $A(t)$, the error matrix equation group is defined as in (9), inline with the ZNN design. Following that, by adopting the linear design formula for zeroing (9), the model (12) is obtained. From [Theorem 1] [10], each error matrix equation in the error matrix equation group (12) converges to the theoretical solution when $t \to \infty$. As a result, the solution of (12) converges to the theoretical TV inverse or pseudoinverse of the input matrix $A(t)$ when $t \to \infty$. Furthermore, from the derivation procedure of (21) from (12), the proof is completed. □

3. Alternative Time-Varying Pseudoinverse Computation

This section presents and analyzes a ZNN model, namely, ZNNP, for calculating the pseudoinverse of any TV real matrix, based on a recent work [9] on ZNN pseudoinverse computation, and this new model will serve as a strong and fair competitor to the proposed ZNNSVDP model. Considering a smooth TV matrix $A(t) \in \mathbb{R}^{m\times n}$, then, if $\mathrm{rank}(A(t)) = n < m$, the MP inverse $A^{\dagger}(t)$ becomes the left inverse $A^{\dagger}(t) = A_L^{-1}(t) \equiv (A^{\mathrm{T}}(t)A(t))^{-1}A^{\mathrm{T}}(t)$ of $A(t)$, where $A^{\mathrm{T}}(t)A(t)A^{\dagger}(t) = A^{\mathrm{T}}(t)$. Otherwise, if $\mathrm{rank}(A(t)) = n > m$, the MP inverse $A^{\dagger}(t)$ becomes the right inverse $A^{\dagger}(t) = A_R^{-1}(t) \equiv A^{\mathrm{T}}(t)(A(t)A^{\mathrm{T}}(t))^{-1}$, where $A^{\dagger}(t)A(t)A^{\mathrm{T}}(t) = A^{\mathrm{T}}(t)$. Therefore, we can design a ZNN model according to the following equations:

$$\begin{cases} A^{\mathrm{T}}(t)A(t)A^{-1}(t) = A^{\mathrm{T}}(t), \ m = n = \mathrm{rank}(A(t)) \\ A^{\mathrm{T}}(t)A(t)A^{\dagger}(t) = A^{\mathrm{T}}(t), \quad \mathrm{rank}(A(t)) \leq n < m \\ A^{\dagger}(t)A(t)A^{\mathrm{T}}(t) = A^{\mathrm{T}}(t), \quad \mathrm{rank}(A(t)) \leq m < n. \end{cases} \tag{22}$$

Based on (22), the ZNNP model considers the following error function for calculating the inverse or pseudoinverse of $A(t)$:

$$E_D(t) = \begin{cases} A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t), \ \mathrm{rank}(A(t)) \leq n \leq m \\ X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t), \ \mathrm{rank}(A(t)) \leq m < n. \end{cases} \tag{23}$$

where $X(t)$ is the desired solution of the problem, i.e., the inverse or pseudoinverse of $A(t)$. Furthermore, the first time derivative of (23) is the following:

$$\dot{E}_D(t) = \begin{cases} \dot{A}^{\mathrm{T}}(t)A(t)X(t) + A^{\mathrm{T}}(t)\dot{A}(t)X(t) + A^{\mathrm{T}}(t)A(t)\dot{X}(t) - \dot{A}^{\mathrm{T}}(t), \ \mathrm{rank}(A(t)) \leq n \leq m \\ \dot{X}(t)A(t)A^{\mathrm{T}}(t) + X(t)\dot{A}(t)A^{\mathrm{T}}(t) + X(t)A(t)\dot{A}^{\mathrm{T}}(t) - \dot{A}^{\mathrm{T}}(t), \ \mathrm{rank}(A(t)) \leq m < n. \end{cases} \tag{24}$$

Then, combining (23) and (24) with the ZNN design (4), under the linear activation function, the following can be obtained:

$$\begin{cases} \dot{A}^{\mathrm{T}}(t)A(t)X(t) + A^{\mathrm{T}}(t)\dot{A}(t)X(t) + A^{\mathrm{T}}(t)A(t)\dot{X}(t) - \dot{A}^{\mathrm{T}}(t) \\ \qquad = -\lambda(A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t)), \quad \mathrm{rank}(A(t)) \leq n \leq m \\ \dot{X}(t)A(t)A^{\mathrm{T}}(t) + X(t)\dot{A}(t)A^{\mathrm{T}}(t) + X(t)A(t)\dot{A}^{\mathrm{T}}(t) - \dot{A}^{\mathrm{T}}(t) \\ \qquad = -\lambda(X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t)), \quad \mathrm{rank}(A(t)) \leq m < n. \end{cases} \tag{25}$$

Using vectorization and the Kronecker product, the dynamics of (25) are modified as follows:

$$\begin{cases} (I_m \otimes A^{\mathrm{T}}(t)A(t))\mathrm{vec}(\dot{X}(t)) = \mathrm{vec}\big(-\lambda(A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t)) - \dot{A}^{\mathrm{T}}(t)A(t)X(t) \\ \qquad - A^{\mathrm{T}}(t)\dot{A}(t)X(t) + \dot{A}^{\mathrm{T}}(t)\big), \quad \mathrm{rank}(A(t)) \leq n \leq m \\ (A(t)A^{\mathrm{T}}(t) \otimes I_n)\mathrm{vec}(\dot{X})(t) = \mathrm{vec}\big(-\lambda(X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t)) - X(t)\dot{A}(t)A^{\mathrm{T}}(t) \\ \qquad - X(t)A(t)\dot{A}^{\mathrm{T}}(t) + \dot{A}^{\mathrm{T}}(t)\big), \quad \mathrm{rank}(A(t)) \leq m < n. \end{cases} \tag{26}$$

As a result, setting

$$\begin{aligned} L(t) &= \begin{cases} I_m \otimes A^{\mathrm{T}}(t)A(t) + \beta I_{mn}, & \mathrm{rank}(A(t)) < n \leq m \\ I_m \otimes A^{\mathrm{T}}(t)A(t), & \mathrm{rank}(A(t)) = n \leq m \\ A(t)A^{\mathrm{T}}(t) \otimes I_n + \beta I_{mn}, & \mathrm{rank}(A(t)) < m < n \\ A(t)A^{\mathrm{T}}(t) \otimes I_n, & \mathrm{rank}(A(t)) = m < n, \end{cases} \\ r(t) &= \begin{cases} \mathrm{vec}\big(-\lambda\,(A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t)) - \dot{A}^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t)\dot{A}(t)X(t) \\ \qquad + \dot{A}^{\mathrm{T}}(t)\big), \quad \mathrm{rank}(A(t)) \leq n \leq m \\ \mathrm{vec}\big(-\lambda\,(X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t)) - X(t)\dot{A}(t)A^{\mathrm{T}}(t) - X(t)A(t)\dot{A}^{\mathrm{T}}(t) \\ \qquad + \dot{A}^{\mathrm{T}}(t)\big), \quad \mathrm{rank}(A(t)) \leq m < n. \end{cases} \\ \dot{\mathbf{x}}(t) &= \mathrm{vec}(\dot{X}(t)), \quad \mathbf{x}(t) = \mathrm{vec}(X(t)), \end{aligned} \tag{27}$$

where $\beta \geq 0$ signifies the Tikhonov regularization parameter, we have the next ZNN model:

$$L(t)\dot{\mathbf{x}}(t) = r(t), \tag{28}$$

where $L(t)$ is a mass matrix. Note that the Tikhonov regularization is used in $L(t)$ to solve the singularity problem of the cases $\text{rank}(A(t)) < n \leq m$ and $\text{rank}(A(t)) < m < n$, respectively, because the products $A(t)A^{\mathrm{T}}(t)$ and $A^{\mathrm{T}}(t)A(t)$ result to singular matrices. The ZNN model (28) is termed as the ZNNP model and can be solved efficiently with an `ode` Matlab solver, while its exponential convergence to the theoretical TV inverse or pseudoinverse of the input matrix $A(t)$ is proven in Theorem 2.

Theorem 2. *The ZNNP model* (28) *starting form any initial value* $\mathbf{x}(0)$*, converges exponentially to the theoretical TV inverse or pseudoinverse of the input matrix* $A(t)$*.*

Proof. In order to obtain the solution $\mathbf{x}(t)$, which corresponds to the TV inverse or pseudoinverse of the input matrix $A(t)$, the error matrix equation group is defined as in (23), inline with the ZNN design. Following that, by adopting the linear design formula for zeroing (23), the model (25) is obtained. From [Theorem 1] [10], each error matrix equation in the error matrix equation group (25) converges to the theoretical solution when $t \to \infty$. As a consequence, the solution of (25) converges to the theoretical TV inverse or pseudoinverse of the input matrix $A(t)$ when $t \to \infty$. Moreover, from the derivation procedure of (28), we know it is (25) in a different form. The proof is, thus, completed. □

4. Numerical Experiments

This section compares and contrasts the performances of the ZNNSVDP model (21) with the ZNNP model (28) on four numerical experiments (NE), involving the pseudoinversion of square, rectangular, singular, and nonsingular input matrices. In all NE, the time interval is restricted to $[0, 10]$ during the computation, which indicates that the starting time is $t_0 = 0$ and the ending time is $t_f = 10$, while the ZNN design parameter has been set to $\lambda = 10$ and the Tikhonov regularization parameter has been set to $\beta = 1e - 8$. It is worth mentioning that the notation ZNNSVDP and ZNNP in the legends of Figure 1, respectively, denote the solutions produced by the ZNNSVDP and ZNNP models. Lastly, the MATLAB solver `ode45` has been used, while the initial value for both models has been set to $\mathbf{x}(0) = \text{sign}(\mathbf{x}^*(0))$, where $\mathbf{x}^*(0)$ is the theoretical solution at $t = 0$ and sign is the signum function.

4.1. Experiment 1

This NE deals with the inversion of the following square matrix:

$$A(t) = \begin{bmatrix} 4/(t+18) & \sin(t)+2 \\ (t+18)/(2t+2) & \cos(t)-20 \end{bmatrix}.$$

Note that $A(t)$ is a full rank matrix with dimensions 2×2.

4.2. Experiment 2

This NE deals with the pseudoinversion of the following rectangular matrix:

$$A(t) = \begin{bmatrix} 3\sin(t)+7 & 3+\cos(t) & \sin(2t)+4 \\ 5-\sin(t) & 4/(t+8) & 7-\sin(t) \\ 7/2+\sin(3t) & 4+\cos(t) & 6+\cos(3t) \\ 3\sin(t) & \cos(t)-20 & 7/2+\sin(5t) \\ 5-\sin(t) & \sin(t)+1 & 3+\cos(t) \end{bmatrix}.$$

Notice that $A(t)$ is a full column rank matrix with dimensions 5×3.

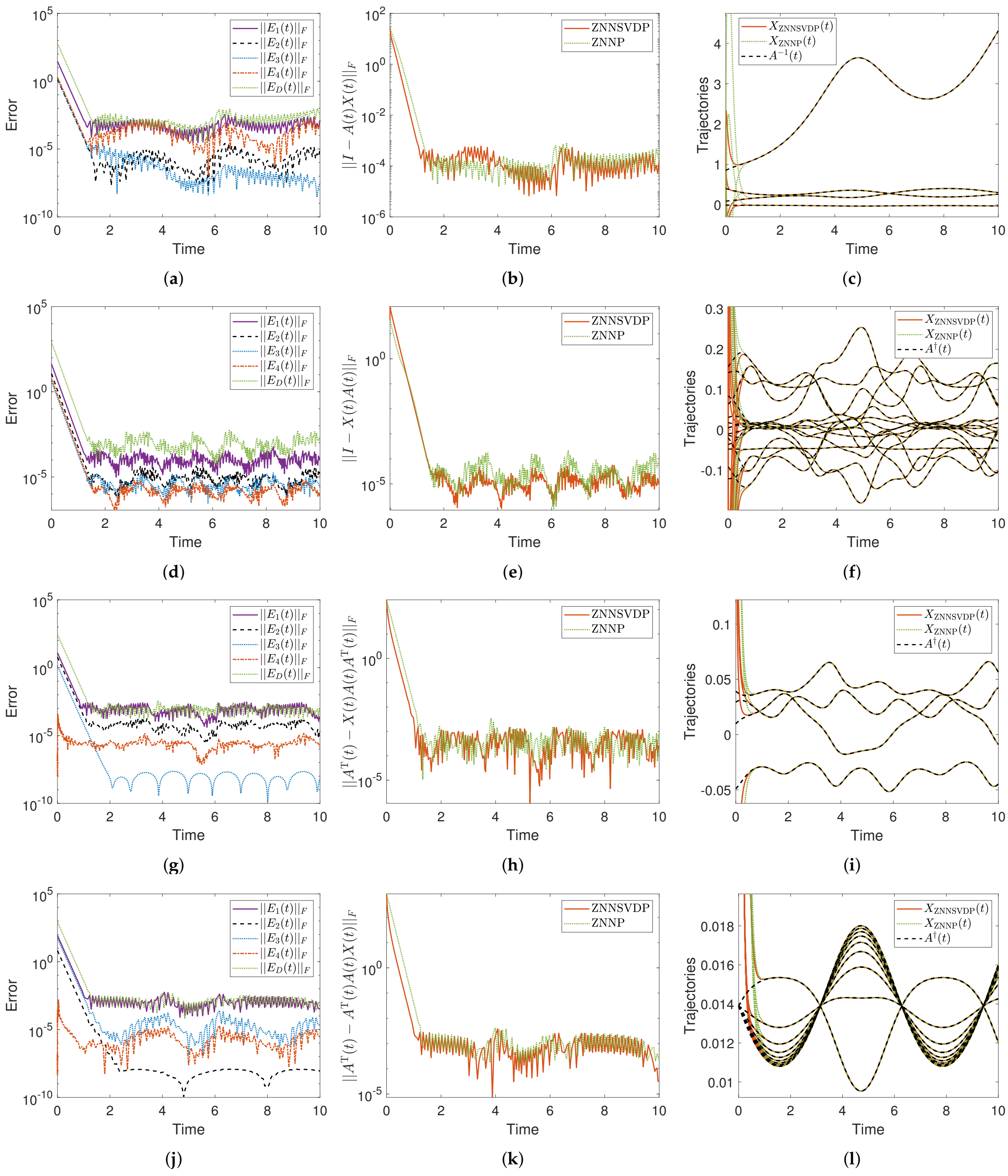

Figure 1. The convergence of ZFs and the solutions' convergence and trajectories in NEs Sections 4.1–4.4. (**a**) NE Section 4.1: Convergence of ZFs. (**b**) NE Section 4.1: Solutions convergence. (**c**) NE Section 4.1: Solutions trajectories. (**d**) NE Section 4.2: Convergence of ZFs. (**e**) NE Section 4.2: Solutions convergence. (**f**) NE Section 4.2: Solutions trajectories. (**g**) NE Section 4.3: Convergence of ZFs. (**h**) NE Section 4.3: Solutions convergence. (**i**) NE Section 4.3: Solutions trajectories. (**j**) NE Section 4.4: Convergence of ZFs. (**k**) NE Section 4.4: Solutions convergence. (**l**) NE Section 4.4: Solutions trajectories.

4.3. Experiment 3

The pseudoinversion of the following rectangular matrix is the subject of this NE:

$$A(t) = \begin{bmatrix} 5-\cos(\pi t) & 3+\sin(\pi t) & -4-\cos(t) & 1+3\sin(t) \end{bmatrix}^{\mathrm{T}} \odot \mathbf{1}_{4\times 2}.$$

The matrix $A(t)$ is rank deficient, with $\operatorname{rank}(A(t)) = 1$, and its dimensions are 4×2.

4.4. Experiment 4

This NE is related to the pseudoinversion of the rectangular matrix given below:

$$A(t) = \begin{bmatrix} 2+\sin(t) & 2+1/2\sin(t) & \dots & 2+1/n\sin(t) \end{bmatrix} \odot \mathbf{1}_{m\times n}.$$

With $\operatorname{rank}(A(t)) = 1$, the matrix $A(t)$ is rank deficient, and its dimensions are $m \times n$, where $m = 4$ and $n = 9$.

4.5. Analysis of Numerical Experiments—Results and Comparison

The performance of the ZNNSVDP and ZNNP models for calculating the inverse or pseudoinverse of an arbitrary matrix $A(t)$ is investigated through the four NE defined in Sections 4.1–4.4. For all the experiments, the results produced by the ZNNSVDP and ZNNP models are depicted in Figure 1. It is worth noting that Figure 1 has the following layout: the first column figures show the convergence of the error function, i.e., $\|E_i(t)\|, i = 1, \dots, 4$, of the ZNNSVDP model and $\|E_D(t)\|$ of the ZNNP model; the second column figures show the convergence of the models according to the appropriate error function, i.e., residual errors; the third column figures show the trajectories of the solutions generated by the models.

The following can be deduced from the NE of this section. Overall, the error functions of the ZNNSVDP model, i.e., $\|E_i(t)\|, i = 1, \dots, 4$, receive lower values than the error function of the ZNNP model, i.e., $\|E_D(t)\|$, in all NE, as depicted in Figure 1a,d,g,j. When $X(t)$ corresponds to the solution of the ZNNSVDP model rather than the solution of the ZNNP model, the convergence in Figure 1b,h,k is faster, while the convergence in Figure 1e is almost identical. It is worth noting that Figure 1b depicts the residual error $\|I - A(t)X(t)\|_{\mathrm{F}}$ in the case of NE Section 4.1, Figure 1e depicts the residual error $\|I - X(t)A(t)\|_{\mathrm{F}}$ in the case of NE Section 4.2, Figure 1h depicts the residual error $\|A^{\mathrm{T}}(t) - X(t)A(t)A^{\mathrm{T}}(t)\|_{\mathrm{F}}$ in the case of NE Section 4.3, and Figure 1b depicts the residual error $\|A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t)A(t)X(t)\|_{\mathrm{F}}$ in the case of NE Section 4.4. Finally, Figure 1c,f,i,l show that both models' solutions match the theoretical inverse in the case of NE Section 4.1, and the theoretical pseudoinverse in the cases of NE Sections 4.2–4.4.

According to the presented NE, the following are some general generalizations that can be drawn. The ZNNSVDP model presented in this paper, which is based on the SVD method, shows better performances than the ZNNP model, which is based on a more direct approach for calculating the inverse or pseudoinverse. In addition, the ZNNSVDP model generates the minimum amount for the Frobenius norm of both the error functions and the residual errors. It is also important to note that, for both models, the larger the value of the design parameter λ, the higher the degree of convergence.

5. Conclusions

The problem of calculating the inverse or pseudoinverse of an arbitrary TV real matrix is addressed using the ZNN approach in this paper. Two ZNN models for calculating the inverse or pseudoinverse of an arbitrary TV matrix, one called ZNNSVDP, which is based on SVD, and the other called ZNNP, which is based on a more direct approach to the problem, are defined, analysed and compared. Four numerical experiments, involving the pseudoinversion of square, rectangular, singular, and nonsingular input matrices, indicate that both models are effective for solving the problem and that the ZNNSVDP model converges to the problem's solution faster than the ZNNP model.

Some potential study areas can be identified:

1. It is possible to explore the streams of the ZNNSVDP and ZNNP models that are accelerated by a nonlinear activation function, as well as nonlinear ZNNSVDP and ZNNP model flows, with a terminal convergence in this direction.
2. Another option is to use carefully chosen fuzzy parameters to define future ZNN dynamics upgrades.
3. The presented ZNNSVDP and ZNNP models have the drawback of not being noise tolerant, because all types of noise have a substantial impact on the accuracy of the proposed ZNN approaches. As a consequence, future research could focus on adapting the ZNNSVDP and ZNNP models to an integration enhanced and noise-handling ZNN class of dynamical systems.

Author Contributions: M.K.: conceptualization, methodology. V.K.: validation, investigation. R.F.: formal analysis, investigation. M.Z.: methodology, investigation. V.N.K.: conceptualization, methodology, validation, formal analysis, investigation, writing—original draft. S.D.M.: conceptualization, methodology, validation, formal analysis, investigation, writing—original draft. T.E.S.: methodology, formal analysis, investigation. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by a Mega Grant from the Government of the Russian Federation within the framework of federal project No. 075-15-2021-584.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Penrose, R. A generalized inverse for matrices. *Proc. Cambridge Philos. Soc.* **1955**, *51*, 406–413. [CrossRef]
2. Sayevand, K.; Pourdarvish, A.; Machado, J.A.T.; Erfanifar, R. On the Calculation of the Moore-Penrose and Drazin Inverses: Application to Fractional Calculus. *Mathematics* **2021**, *9*, 2501. [CrossRef]
3. Chien, M.T. Numerical Range of Moore-Penrose Inverse Matrices. *Mathematics* **2020**, *8*, 830. [CrossRef]
4. Crane, D.K.; Gockenbach, M.S. The Singular Value Expansion for Arbitrary Bounded Linear Operators. *Mathematics* **2020**, *8*, 1346. [CrossRef]
5. Valverde-Albacete, F.J.; Peláez-Moreno, C. The Singular Value Decomposition over Completed Idempotent Semifields. *Mathematics* **2020**, *8*, 1577. [CrossRef]
6. Hazarika, A.; Barthakur, M.; Dutta, L.; Bhuyan, M. F-SVD based algorithm for variability and stability measurement of bio-signals, feature extraction and fusion for pattern recognition. *Biomed. Signal Process. Control* **2019**, *47*, 26–40. [CrossRef]
7. Wang, J.; Le, N.T.; Lee, J.; Wang, C. Illumination compensation for face recognition using adaptive singular value decomposition in the wavelet domain. *Inf. Sci.* **2018**, *435*, 69–93. [CrossRef]
8. Chen, J.; Zhang, Y. Online singular value decomposition of time-varying matrix via zeroing neural dynamics. *Neurocomputing* **2020**, *383*, 314–323. [CrossRef]
9. Katsikis, V.N.; Stanimirović, P.S.; Mourtas, S.D.; Xiao, L.; Karabasević, D.; Stanujkić, D. Zeroing Neural Network with Fuzzy Parameter for Computing Pseudoinverse of Arbitrary Matrix. *IEEE Trans. Fuzzy Syst.* 2021, *Early Access*. [CrossRef]
10. Zhang, Y.; Ge, S.S. Design and analysis of a general recurrent neural network model for time-varying matrix inversion. *IEEE Trans. Neural Netw.* **2005**, *16*, 1477–1490. [CrossRef] [PubMed]
11. Wang, X.; Che, M.; Wei, Y. Recurrent neural network for computation of generalized eigenvalue problem with real diagonalizable matrix pair and its applications. *Neurocomputing* **2016**, *216*, 230–241. [CrossRef]
12. Stanimirović, P.S.; Katsikis, V.N.; Zhang, Z.; Li, S.; Chen, J.; Zhou, M. Varying-parameter Zhang neural network for approximating some expressions involving outer inverses. *Optim. Methods Softw.* **2020**, *35*, 1304–1330. [CrossRef]
13. Ma, H.; Li, N.; Stanimirović, P.S.; Katsikis, V.N. Perturbation theory for Moore–Penrose inverse of tensor via Einstein product. *Comput. Appl. Math.* **2019**, *38*, 111. [CrossRef]
14. Katsikis, V.N.; Mourtas, S.D.; Stanimirović, P.S.; Zhang, Y. Solving Complex-Valued Time-Varying Linear Matrix Equations via QR Decomposition With Applications to Robotic Motion Tracking and on Angle-of-Arrival Localization. *IEEE Trans. Neural Netw. Learn. Syst.* 2021, *Early Access*. [CrossRef]
15. Stanimirović, P.S.; Katsikis, V.N.; Li, S. Hybrid GNN-ZNN models for solving linear matrix equations. *Neurocomputing* **2018**, *316*, 124–134. [CrossRef]

16. Stanimirović, P.S.; Katsikis, V.N.; Li, S. Integration enhanced and noise tolerant ZNN for computing various expressions involving outer inverses. *Neurocomputing* **2019**, *329*, 129–143. [CrossRef]
17. Zhang, Z.; Yang, S.; Zheng, L. A Penalty Strategy Combined Varying-Parameter Recurrent Neural Network for Solving Time-Varying Multi-Type Constrained Quadratic Programming Problems. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 2993–3004. [CrossRef] [PubMed]
18. Katsikis, V.N.; Stanimirović, P.S.; Mourtas, S.D.; Li, S.; Cao, X. Chapter Towards Higher Order Dynamical Systems. In *Generalized Inverses: Algorithms and Applications*; Mathematics Research Developments, Nova Science Publishers, Inc.: Hauppauge, NY, USA, 2021; pp. 207–239.
19. Katsikis, V.N.; Mourtas, S.D.; Stanimirović, P.S.; Zhang, Y. Continuous-Time Varying Complex QR Decomposition via Zeroing Neural Dynamics. *Neural Process. Lett.* **2021**, *53*, 3573–3590 [CrossRef]
20. Zhang, Y.; Guo, D. *Zhang Functions and Various Models*; Springer: Berlin/Heidelberg, Germany, 2015. [CrossRef]
21. Ben-Israel, A.; Greville, T.N.E. *Generalized Inverses: Theory and Applications*, 2nd ed.; CMS Books in Mathematics; Springer: New York, NY, USA, 2003. [CrossRef]
22. Golub, G.H.; Pereyra, V. The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate. *SIAM J. Numer. Anal.* **1973**, *10*, 413–432. [CrossRef]
23. Graham, A. *Kronecker Products and Matrix Calculus with Applications*; Courier Dover Publications: Mineola, NY, USA, 2018.
24. Gupta, A.K. *Numerical Methods Using MATLAB*; MATLAB Solutions Series; Springer: New York, NY, USA, 2014.