AIMS Mathematics, 8(10): 22875–22895.
DOI: 10.3934/math.20231164
Received: 19 May 2023
Revised: 28 June 2023
Accepted: 11 July 2023
Published: 19 July 2023

http://www.aimspress.com/journal/Math

Research article

Computing quaternion matrix pseudoinverse with zeroing neural networks

Vladislav N. Kovalnogov[1], Ruslan V. Fedorov[1], Denis A. Demidov[1], Malyoshina A. Malyoshina[1], Theodore E. Simos[1,2,3,4,5*], Spyridon D. Mourtas[6,7] and Vasilios N. Katsikis[6]

[1] Laboratory of Interdisciplinary Problems in Energy Production, Ulyanovsk State Technical University, 32 Severny Venetz Street, 432027 Ulyanovsk, Russia
[2] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung City 40402, Taiwan
[3] Center for Applied Mathematics and Bioinformatics, Gulf University for Science and Technology, West Mishref, 32093 Kuwait
[4] Data Recovery Key Laboratory of Sichun Province, Neijing Normal Univ., Neijiang 641100, China
[5] Section of Mathematics, Dept. of Civil Engineering, Democritus Univ. of Thrace, Xanthi 67100, Greece
[6] Department of Economics, Mathematics-Informatics and Statistics-Econometrics, National and Kapodistrian University of Athens, Sofokleous 1 Street, 10559 Athens, Greece
[7] Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, Prosp. Svobodny 79, 660041 Krasnoyarsk, Russia

* **Correspondence:** Email: simos@ulstu.ru.

Abstract: In recent years, it has become essential to compute the time-varying quaternion (TVQ) matrix Moore-Penrose inverse (MP-inverse or pseudoinverse) to solve time-varying issues in a range of disciplines, including engineering, physics and computer science. This study examines the problem of computing the TVQ matrix MP-inverse using the zeroing neural network (ZNN) approach, which is nowadays considered a cutting edge technique. As a consequence, three new ZNN models are introduced for computing the TVQ matrix MP-inverse in the literature for the first time. Particularly, one model directly employs the TVQ input matrix in the quaternion domain, while the other two models, respectively, use its complex and real representations. In four numerical simulations and a real-world application involving robotic motion tracking, the models exhibit excellent performance.

Keywords: quaternion; Moore-Penrose inverse; zeroing neural network; dynamical system; mobile object localization
Mathematics Subject Classification: 15A24, 65F20, 68T05

1. Introduction

The real-time solution to the Moore-Penrose inverse (MP-inverse or pseudoinverse) [1, 2], that frequently arises in robotics [3–5], game theory [6], nonlinear systems [7] and other technical and scientific disciplines [8–10], has attracted a lot of interest in recent times. Quaternions, on the other hand, are crucial in a wide range of domains, such as computer graphics [11–13], robotics [14, 15], navigation [16], quantum mechanics [17], electromagnetism [18] and mathematical physics [19, 20]. Let $\mathbb{H}^{m\times n}$ present the set of all $m \times n$ matrices on the quaternion skew field $\mathbb{H} = \{\delta_1 + \delta_2\imath + \delta_3\jmath + \delta_4 k \mid \imath^2 = \jmath^2 = k^2 = \imath\jmath k = -1, \delta_1, \delta_2, \delta_3, \delta_4 \in \mathbb{R}\}$. Considering that $\tilde{A} \in \mathbb{H}^{m\times n}$, its conjugate transpose is denoted by $\tilde{A}^*$, and its rank by rank($\tilde{A}$). The generalization of the inverse matrix $\tilde{A}^{-1}$ is the MP-inverse $\tilde{A}^\dagger$, whereas $\tilde{A}^\dagger$ is just one solution $\tilde{X}$ that satisfies the next Penrose equations [21, 22]:

$$(i)\ \tilde{A}\tilde{X}\tilde{A} = \tilde{A},\quad (ii)\ \tilde{X}\tilde{A}\tilde{X} = \tilde{X},\quad (iii)\ (\tilde{A}\tilde{X})^* = \tilde{A}\tilde{X},\quad (iv)\ (\tilde{X}\tilde{A})^* = \tilde{X}\tilde{A}. \tag{1.1}$$

Recently, research has begun to focus on time-varying quaternion (TVQ) problems involving matrices, such as inversion of TVQ matrices [23], solving the dynamic TVQ Sylvester matrix equation [24], addressing the TVQ constrained matrix least-squares problem [25] and solving the TVQ linear matrix equation for square matrices [26]. Furthermore, real-world applications involving TVQ matrices are employed in kinematically redundant manipulator of robotic joints [15, 27], chaotic systems synchronization [25], mobile manipulator control [23,28] and image restoration [26,29]. All these studies have one thing in common: they all use the zeroing neural network (ZNN) approach to derive the solution.

ZNNs are a subset of recurrent neural networks that are especially good at parallel processing and are used to address time-varying issues. They were initially developed by Zhang *et al.* [30] to handle the problem of time-varying matrix inversion, but their subsequent iterations were dynamic models for computing the time-varying MP-inverse of full-row/column rank matrices [31–34] in the real and complex domain. Today, their use has expanded to include the resolution of generalized inversion issues [35–40], linear and quadratic programming tasks [41–43], certain types of matrix equation [44,45], systems of nonlinear equations [46,47], systems of linear equations [48–50] and systems of equations with noise [51]. The TVQ MP-inverse (TVQ-MPI) problem for any TVQ matrix will be addressed in this paper using the ZNN approach. Of greater significance, we will determine whether a direct solution in the quaternion domain or an indirect solution through representation in the complex and real domains is more efficient. To do this, we will create three ZNN models, one for each domain, and rigorously validate them on four numerical simulations and a real-world application involving robotic motion tracking. By doing theoretical analysis and analyzing the computational complexity of all presented models, this research strengthens the existing body of literature.

The rest of the article is divided into the following sections. Section 2 presents introductory information and the TVQ-MPI problem formulation. Section 3 introduces the three ZNN models, while their theoretical analysis is presented in Section 4. Numerical simulations and applications are explored in Section 5 and, finally, Section 6 provides concluding thoughts and comments.

2. Introductory information and problem formulation

This part outlines some introductory information about TVQ matrices, the TVQ-MPI problem, ZNNs and the notation that will be used throughout the remainder of the study as well as the primary findings that will be covered.

A division algebra or skew-field over the field of real numbers makes up a quaternion. As a result, the set of quaternions $\mathbb{H}$ is not commutative under the operation of multiplication, which causes complexity to rise quickly in real-world applications [52]. On the other hand, a real 4×4 matrix or a complex 2×2 matrix can both be used to easily express a scalar quaternion [53]. The dimensions of the representation matrices scale suitably, and this fact also applies to matrices of quaternions. In order to tackle quaternion-based problems, it is now customary to first solve an analogous issue in the real or complex domain before converting the result back to quaternion form. This method, which is undeniably effective even in a static setting, excels at solving issues with time-varying characteristics.

Let $\tilde{A}(t) = A_1(t) + A_2(t)\imath + A_3(t)\jmath + A_4(t)k \in \mathbb{H}^{m\times n}$, with $A_i(t) \in \mathbb{R}^{m\times n}$ for $i = 1, 2, 3, 4$, be a TVQ matrix and $t \in [0, t_f) \subseteq [0, +\infty)$ be the time. The conjugate transpose of a TVQ matrix $\tilde{A}(t)$ is the following [52, 53]:

$$\tilde{A}^*(t) = A_1^{\mathrm{T}}(t) - A_2^{\mathrm{T}}(t)\imath - A_3^{\mathrm{T}}(t)\jmath - A_4^{\mathrm{T}}(t)k, \tag{2.1}$$

where the operator $()^{\mathrm{T}}$ denotes transposition. The product of the two TVQ matrices $\tilde{A}(t)$ and $\tilde{B}(t) = B_1(t) + B_2(t)\imath + B_3(t)\jmath + B_4(t)k \in \mathbb{H}^{n\times g}$, with $B_i(t) \in \mathbb{R}^{n\times g}$ for $i = 1, \cdots, 4$, is:

$$\tilde{A}(t)\tilde{B}(t) = \tilde{Y}(t) = Y_1(t) + Y_2(t)\imath + Y_3(t)\jmath + Y_4(t)k \in \mathbb{H}^{m\times g} \tag{2.2}$$

where

$$\begin{aligned}
Y_1(t)&=A_1(t)B_1(t)-A_2(t)B_2(t)-A_3(t)B_3(t)-A_4(t)B_4(t),\\
Y_2(t)&=A_1(t)B_2(t)+A_2(t)B_1(t)+A_3(t)B_4(t)-A_4(t)B_3(t),\\
Y_3(t)&=A_1(t)B_3(t)+A_3(t)B_1(t)+A_4(t)B_2(t)-A_2(t)B_4(t),\\
Y_4(t)&=A_1(t)B_4(t)+A_4(t)B_1(t)+A_2(t)B_3(t)-A_3(t)B_2(t),
\end{aligned} \tag{2.3}$$

with $Y_i(t) \in \mathbb{R}^{m\times g}$ for $i = 1, \cdots, 4$. Additionally, one complex representation of the TVQ matrix $\tilde{A}(t)$ is the following [24, 54]:

$$\hat{A}(t) = \begin{bmatrix} A_1(t) - A_4(t)\imath & -A_3(t) - A_2(t)\imath \\ A_3(t) - A_2(t)\imath & A_1(t) + A_4(t)\imath \end{bmatrix} \in \mathbb{C}^{2m\times 2n}, \tag{2.4}$$

and one real representation of the TVQ matrix $\tilde{A}(t)$ is the following [26]:

$$A(t) = \begin{bmatrix} A_1(t) & A_4(t) & -A_3(t) & A_2(t) \\ -A_4(t) & A_1(t) & -A_2(t) & -A_3(t) \\ A_3(t) & A_2(t) & A_1(t) & -A_4(t) \\ -A_2(t) & A_3(t) & A_4(t) & A_1(t) \end{bmatrix} \in \mathbb{R}^{4m\times 4n}. \tag{2.5}$$

In this paper, the following TVQ matrix equations problem is taken into consideration for computing the TVQ-MPI of any $\tilde{A}(t) \in \mathbb{H}^{m\times n}$ [21, 22]:

$$\begin{cases} \tilde{A}^*(t)\tilde{A}(t)\tilde{X}(t) - \tilde{A}^*(t) = \mathbf{0}_{n\times m}, & m \geq n \\ \tilde{X}(t)\tilde{A}(t)\tilde{A}^*(t) - \tilde{A}^*(t) = \mathbf{0}_{n\times m}, & m < n \end{cases}, \tag{2.6}$$

where the TVQ matrix $\tilde{X}(t) = X_1(t) + X_2(t)\imath + X_3(t)\jmath + X_4(t)k \in \mathbb{H}^{n\times m}$, with $X_i(t) \in \mathbb{R}^{n\times m}$ for $i = 1, 2, 3, 4$, is the TVQ matrix of interest. Additionally, we consider that $\tilde{A}(t)$ is a smoothly time-varying matrix and its time derivative is either given or can be accurately estimated. It is important to note that (2.6) is the TVQ-MPI problem and it is satisfied only for $\tilde{X}(t) = \tilde{A}^{\dagger}(t)$.

On the one hand, taking into account that the complex representation of the TVQ matrix acquired by multiplying two TVQ matrices is similar to the TVQ matrix acquired by multiplying the complex representations of two TVQ matrices [24, Theorem 1], solving (2.6) is equivalent to solving the real matrix equation:

$$\begin{cases} \hat{A}^*(t)\hat{A}(t)\hat{X}(t) - \hat{A}^*(t) = \mathbf{0}_{2n\times 2m}, & m \geq n \\ \hat{X}(t)\hat{A}(t)\hat{A}^*(t) - \hat{A}^*(t) = \mathbf{0}_{2n\times 2m}, & m < n \end{cases}, \tag{2.7}$$

where $\hat{X}(t) \in \mathbb{C}^{2n\times 2m}$. On the other hand, taking into account that the real representation of the TVQ matrix acquired by multiplying two TVQ matrices is similar to the TVQ matrix acquired by multiplying the real representations of two TVQ matrices [26, Corollary 1], solving (2.6) is equivalent to solving the real matrix equation:

$$\begin{cases} A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t) = \mathbf{0}_{4n\times 4m}, & m \geq n \\ X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t) = \mathbf{0}_{4n\times 4m}, & m < n \end{cases}, \tag{2.8}$$

where $X(t) \in \mathbb{R}^{4n\times 4m}$.

Three novel ZNN models are introduced for solving the TVQ-MPI problem of (2.6) in this research. One model, dubbed as ZNNQ, is created for directly resolving the TVQ-MPI problem of (2.6). The two additional models, dubbed as ZNNQC and ZNNQR, are created to indirectly solve the TVQ-MPI problem of (2.6) through (2.7) in the complex domain and (2.8) in the real domain, respectively. The creation of a ZNN model typically involves two fundamental steps. First, one defines an error matrix equation (ERME) function $E(t)$. Second, the next ZNN dynamical system under the linear activation function must be used:

$$\dot{E}(t) = -\lambda E(t), \tag{2.9}$$

where the operator $\dot{(\cdot)}$ denotes the time derivative. Additionally, the design parameter $\lambda > 0$ is a positive real number, though one may adjust the convergence rate of the model. For instance, a higher value for λ will result in the model converging even faster [55–57]. It is important to point out that continual learning is defined as learning continually from non-stationary data, while transferring and preserving prior knowledge. It is true that as time evolves, the ZNN's architecture relies around driving each entry of the error function $E(t)$ to 0. The continuous-time learning rule, which is the consequence of the definition of the ERME function (2.9), is used to do this. Therefore, it is possible to think of the error function as a tool for tracking the learning of ZNN models.

The key conclusions of the paper are listed next:

(1) For the first time, the TVQ-MPI problem is addressed through the ZNN approach.
(2) With the purpose of addressing the TVQ-MPI problem, three novel ZNN models are provided.
(3) Matrices of any dimension can be used with the proposed ZNN models.
(4) The models are subjected to a theoretical analysis that validates them.
(5) Numerical simulations and applications are carried out to complement the theoretical concepts.

The following notations are employed in the remainder of this article: I_u refers to the identity $u \times u$ matrix; $\mathbf{0}_u$ and $\mathbf{0}_{m\times n}$, respectively, refer to the zero $u \times u$ and $m \times n$ matrices; $\|\cdot\|_{\mathrm{F}}$ denotes the matrix Frobenius norm; vec$(\cdot)$ denotes the vectorization process; $\odot$ denotes the elementwise multiplication; $\otimes$ denotes the Kronecker product.

3. ZNN models in solving the TVQ-MPI

Three ZNN models, each working in a distinct domain, will be developed in this section. Further, we assume that $\tilde{A}(t) \in \mathbb{H}^{m\times n}$ is a differentiable TVQ matrix, and $\tilde{X}(t) \in \mathbb{H}^{n\times m}$ is the unknown MP-inverse matrix of $\tilde{A}(t)$ to be found.

3.1. The ZNNQ model

To develop the ZNNQ model, the TVQ-MPI of (2.6) is considered. According to (2.1) and (2.2), we set $\tilde{A}^*(t)\tilde{A}(t) = \tilde{U}(t) = U_1(t) + U_2(t)\imath + U_3(t)\jmath + U_4(t)k$, where

$$\begin{aligned} U_1(t) =& A_1^{\mathrm{T}}(t)A_1(t) + A_2^{\mathrm{T}}(t)A_2(t) + A_3^{\mathrm{T}}(t)A_3(t) + A_4^{\mathrm{T}}(t)A_4(t),\\ U_2(t) =& A_1^{\mathrm{T}}(t)A_2(t) - A_2^{\mathrm{T}}(t)A_1(t) - A_3^{\mathrm{T}}(t)A_4(t) + A_4^{\mathrm{T}}(t)A_3(t),\\ U_3(t) =& A_1^{\mathrm{T}}(t)A_3(t) - A_3^{\mathrm{T}}(t)A_1(t) - A_4^{\mathrm{T}}(t)A_2(t) + A_2^{\mathrm{T}}(t)A_4(t),\\ U_4(t) =& A_1^{\mathrm{T}}(t)A_4(t) - A_4^{\mathrm{T}}(t)A_1(t) - A_2^{\mathrm{T}}(t)A_3(t) + A_3^{\mathrm{T}}(t)A_2(t), \end{aligned} \tag{3.1}$$

with $U_i(t) \in \mathbb{R}^{n\times n}$ for $i = 1, \cdots, 4$, and $\tilde{A}(t)\tilde{A}^*(t) = \tilde{V}(t) = V_1(t) + V_2(t)\imath + V_3(t)\jmath + V_4(t)k$, where

$$\begin{aligned} V_1(t) =& A_1(t)A_1^{\mathrm{T}}(t) + A_2(t)A_2^{\mathrm{T}}(t) + A_3(t)A_3^{\mathrm{T}}(t) + A_4(t)A_4^{\mathrm{T}}(t),\\ V_2(t) =& - A_1(t)A_2^{\mathrm{T}}(t) + A_2(t)A_1^{\mathrm{T}}(t) - A_3(t)A_4^{\mathrm{T}}(t) + A_4(t)A_3^{\mathrm{T}}(t),\\ V_3(t) =& - A_1(t)A_3^{\mathrm{T}}(t) + A_3(t)A_1^{\mathrm{T}}(t) - A_4(t)A_2^{\mathrm{T}}(t) + A_2(t)A_4^{\mathrm{T}}(t),\\ V_4(t) =& - A_1(t)A_4^{\mathrm{T}}(t) + A_4(t)A_1^{\mathrm{T}}(t) - A_2(t)A_3^{\mathrm{T}}(t) + A_3(t)A_2^{\mathrm{T}}(t), \end{aligned} \tag{3.2}$$

with $V_i(t) \in \mathbb{R}^{m\times m}$ for $i = 1, \cdots, 4$. Taking into account (3.1) and (3.2), the TVQ-MPI (2.6) can be rewritten as below:

$$\begin{cases} \tilde{U}(t)\tilde{X}(t) - \tilde{A}^*(t) = \mathbf{0}_{n\times m}, & m \geq n\\ \tilde{X}(t)\tilde{V}(t) - \tilde{A}^*(t) = \mathbf{0}_{n\times m}, & m < n \end{cases}, \tag{3.3}$$

or equivalent,

$$\begin{cases} C_1(t) - A_1^{\mathrm{T}}(t) + (C_2(t) + A_2^{\mathrm{T}}(t))\imath + (C_3(t) + A_3^{\mathrm{T}}(t))\jmath + (C_4(t) + A_4^{\mathrm{T}}(t))k = \mathbf{0}_{n\times m}, & m \geq n\\ D_1(t) - A_1^{\mathrm{T}}(t) + (D_2(t) + A_2^{\mathrm{T}}(t))\imath + (D_3(t) + A_3^{\mathrm{T}}(t))\jmath + (D_4(t) + A_4^{\mathrm{T}}(t))k = \mathbf{0}_{n\times m}, & m < n \end{cases}, \tag{3.4}$$

where

$$\begin{aligned} C_1(t) =& U_1(t)X_1(t) - U_2(t)X_2(t) - U_3(t)X_3(t) - U_4(t)X_4(t),\\ C_2(t) =& U_1(t)X_2(t) + U_2(t)X_1(t) + U_3(t)X_4(t) - U_4(t)X_3(t),\\ C_3(t) =& U_1(t)X_3(t) + U_3(t)X_1(t) + U_4(t)X_2(t) - U_2(t)X_4(t),\\ C_4(t) =& U_1(t)X_4(t) + U_4(t)X_1(t) + U_2(t)X_3(t) - U_3(t)X_2(t), \end{aligned} \tag{3.5}$$

with $C_i(t) \in \mathbb{R}^{n\times m}$ for $i = 1, \cdots, 4$, and

$$\begin{aligned} D_1(t) =& X_1(t)V_1(t) - X_2(t)V_2(t) - X_3(t)V_3(t) - X_4(t)V_4(t),\\ D_2(t) =& X_1(t)V_2(t) + X_2(t)V_1(t) + X_3(t)V_4(t) - X_4(t)V_3(t),\\ D_3(t) =& X_1(t)V_3(t) + X_3(t)V_1(t) + X_4(t)V_2(t) - X_2(t)V_4(t),\\ D_4(t) =& X_1(t)V_4(t) + X_4(t)V_1(t) + X_2(t)V_3(t) - X_3(t)V_2(t), \end{aligned} \tag{3.6}$$

with $C_i(t) \in \mathbb{R}^{n\times m}$ for $i = 1, \cdots, 4$. Then, setting

$$Z_1(t) = \begin{bmatrix} U_1(t) & -U_2(t) & -U_3(t) & -U_4(t) \\ U_2(t) & U_1(t) & -U_4(t) & U_3(t) \\ U_3(t) & U_4(t) & U_1(t) & -U_2(t) \\ U_4(t) & -U_3(t) & U_2(t) & U_1(t) \end{bmatrix}, \quad Z_2(t) = \begin{bmatrix} V_1(t) & V_2(t) & V_3(t) & V_4(t) \\ -V_2(t) & V_1(t) & -V_4(t) & V_3(t) \\ -V_3(t) & V_4(t) & V_1(t) & -V_2(t) \\ -V_4(t) & -V_3(t) & V_2(t) & V_1(t) \end{bmatrix} \in \mathbb{R}^{4n\times 4m},$$
$$Y_1(t) = [X_1^{\mathrm{T}}(t), X_2^{\mathrm{T}}(t), X_3^{\mathrm{T}}(t), X_4^{\mathrm{T}}(t)]^{\mathrm{T}} \in \mathbb{R}^{4n\times m}, \quad Y_2(t) = [X_1(t), X_2(t), X_3(t), X_4(t)] \in \mathbb{R}^{n\times 4m},$$
$$W_1(t) = [A_1(t), -A_2(t), -A_3(t), -A_4(t)]^{\mathrm{T}} \in \mathbb{R}^{4n\times m}, \quad W_2(t) = [A_1^{\mathrm{T}}(t), -A_2^{\mathrm{T}}(t), -A_3^{\mathrm{T}}(t), -A_4^{\mathrm{T}}(t)] \in \mathbb{R}^{n\times 4m}, \tag{3.7}$$

where $Z_1(t) \in \mathbb{R}^{4m\times 4n}$, $Z_2(t) \in \mathbb{R}^{4n\times 4m}$, $Y_1(t), W_1(t) \in \mathbb{R}^{4n\times m}$ and $Y_2(t), W_2(t) \in \mathbb{R}^{n\times 4m}$, the following ERME is considered:

$$E_Q(t) = \begin{cases} E_1(t) = Z_1(t)Y_1(t) - W_1(t), & m \geq n \\ E_2(t) = Y_2(t)Z_2(t) - W_2(t), & m < n \end{cases}, \tag{3.8}$$

where $E_1(t) \in \mathbb{R}^{4n\times m}$ and $E_2(t) \in \mathbb{R}^{n\times 4m}$. The first time derivative of (3.8) is as follows:

$$\dot{E}_Q(t) = \begin{cases} \dot{E}_1(t) = \dot{Z}_1(t)Y_1(t) + Z_1(t)\dot{Y}_1(t) - \dot{W}_1(t), & m \geq n \\ \dot{E}_2(t) = \dot{Y}_2(t)Z_2(t) + Y_2(t)\dot{Z}_2(t) - \dot{W}_2(t), & m < n \end{cases}. \tag{3.9}$$

When $E_Q(t)$ of (3.8) and $\dot{E}_Q(t)$ of (3.9) are replaced in (2.9) and solving in terms of $\dot{Y}_1(t)$ and $\dot{Y}_2(t)$, we have the next result:

$$\begin{cases} Z_1(t)\dot{Y}_1(t) = -\lambda E_1(t) - \dot{Z}_1(t)Y_1(t) + \dot{W}_1(t), & m \geq n \\ \dot{Y}_2(t)Z_2(t) = -\lambda E_2(t) - Y_2(t)\dot{Z}_2(t) + \dot{W}_2(t), & m < n \end{cases}. \tag{3.10}$$

Then, with the aid of the Kronecker product and the vectorization process, the dynamic model of (3.10) may be simplified:

$$\begin{cases} (I_m \otimes Z_1(t))\mathrm{vec}(\dot{Y}_1(t)) = \mathrm{vec}(-\lambda E_1(t) - \dot{Z}_1(t)Y_1(t) + \dot{W}_1(t)), & m \geq n \\ (Z_2(t) \otimes I_n)\mathrm{vec}(\dot{Y}_2(t)) = \mathrm{vec}(-\lambda E_2(t) - Y_2(t)\dot{Z}_2(t) + \dot{W}_2(t)), & m < n \end{cases}, \tag{3.11}$$

and after setting:

$$K(t) = \begin{cases} I_m \otimes Z_1(t), & m \geq n \ \& \ \mathrm{rank}(\tilde{A}(t)) = n \\ I_m \otimes Z_1(t) + \gamma I_{4mn}, & m \geq n \ \& \ \mathrm{rank}(\tilde{A}(t)) < n \\ Z_2(t) \otimes I_n, & m < n \ \& \ \mathrm{rank}(\tilde{A}(t)) = m \\ Z_2(t) \otimes I_n + \gamma I_{4mn}, & m < n \ \& \ \mathrm{rank}(\tilde{A}(t)) < m \end{cases}, \quad \dot{\mathbf{y}}(t) = \begin{cases} \mathrm{vec}(\dot{Y}_1(t)), & m \geq n \\ \mathrm{vec}(\dot{Y}_2(t)), & m < n \end{cases},$$
$$L(t) = \begin{cases} \mathrm{vec}(-\lambda E_1(t) - \dot{Z}_1(t)Y_1(t) + \dot{W}_1(t)), & m \geq n \\ \mathrm{vec}(-\lambda E_2(t) - Y_2(t)\dot{Z}_2(t) + \dot{W}_2(t)), & m < n \end{cases}, \quad \mathbf{y}(t) = \begin{cases} \mathrm{vec}(Y_1(t)), & m \geq n \\ \mathrm{vec}(Y_2(t)), & m < n \end{cases}, \tag{3.12}$$

the next ZNNQ model is derived for solving the TVQ-MPI of (2.6):

$$K(t)\dot{\mathbf{y}}(t) = L(t) \tag{3.13}$$

where $\dot{\mathbf{y}}(t), \mathbf{y}(t), L(t) \in \mathbb{R}^{4mn}$, $K(t) \in \mathbb{R}^{4mn\times 4mn}$ is a nonsingular mass matrix and $\gamma \geq 0$ is the regularization parameter.

Given that we perform $4mn$ additions/subtractions and $(4mn)^2$ multiplications in each iteration of (3.13), the complexity of solving (3.13) is $O((4mn)^2)$ operations. In addition, the complexity of solving (3.13) through use of an implicit `ode` MATLAB solver is $O((4mn)^3$ as it involves a $(4mn) \times (4mn)$ matrix. As a consequence, the computational complexity of the ZNNQ model of (3.13) is $O((4mn)^3)$.

3.2. The ZNNQC model

To develop the ZNNQC model, the TVQ-MPI of (2.7) is considered. Let $\hat{A}(t) \in \mathbb{C}^{2m\times 2n}$ and $\hat{X}(t) \in \mathbb{C}^{2n\times 2m}$, we set the following ERME:

$$E_C(t) = \begin{cases} E_1(t) = \hat{A}^*(t)\hat{A}(t)\hat{X}(t) - \hat{A}^*(t), & m \geq n \\ E_2(t) = \hat{X}(t)\hat{A}(t)\hat{A}^*(t) - \hat{A}^*(t), & m < n \end{cases}, \tag{3.14}$$

where $E_1(t), E_2(t) \in \mathbb{C}^{2n\times 2m}$. The first time derivative of (3.14) is as follows:

$$\dot{E}_C(t) = \begin{cases} \dot{E}_1(t) = (\dot{\hat{A}}^*(t)\hat{A}(t) + \hat{A}^*(t)\dot{\hat{A}}(t))\hat{X}(t) + \hat{A}^*(t)\hat{A}(t)\dot{\hat{X}}(t) - \dot{\hat{A}}^*(t), & m \geq n \\ \dot{E}_2(t) = \dot{\hat{X}}(t)\hat{A}(t)\hat{A}^*(t) + \hat{X}(t)(\dot{\hat{A}}(t)\hat{A}^*(t) + \hat{A}(t)\dot{\hat{A}}^*(t)) - \dot{\hat{A}}^*(t), & m < n \end{cases}. \tag{3.15}$$

When $E_C(t)$ of (3.14) and $\dot{E}_C(t)$ of (3.15) are replaced in (2.9) and solving in terms of $\dot{\hat{X}}(t)$, we have the next result:

$$\begin{cases} \hat{A}^*(t)\hat{A}(t)\dot{\hat{X}}(t) = -\lambda E_1(t) - (\dot{\hat{A}}^*(t)\hat{A}(t) + \hat{A}^*(t)\dot{\hat{A}}(t))\hat{X}(t) + \dot{\hat{A}}^*(t), & m \geq n \\ \dot{\hat{X}}(t)\hat{A}(t)\hat{A}^*(t) = -\lambda E_2(t) - \hat{X}(t)(\dot{\hat{A}}(t)\hat{A}^*(t) + \hat{A}(t)\dot{\hat{A}}^*(t)) + \dot{\hat{A}}^*(t), & m < n \end{cases}. \tag{3.16}$$

Then, with the aid of the Kronecker product and the vectorization process, the dynamic model of (3.16) may be simplified:

$$\begin{cases} (I_{2m} \otimes \hat{A}^*(t)\hat{A}(t))\mathrm{vec}(\dot{\hat{X}}(t)) = \mathrm{vec}(-\lambda E_1(t) - (\dot{\hat{A}}^*(t)\hat{A}(t) + \hat{A}^*(t)\dot{\hat{A}}(t))\hat{X}(t) + \dot{\hat{A}}^*(t)), & m \geq n \\ (\hat{A}(t)\hat{A}^*(t) \otimes I_{2n})\mathrm{vec}(\dot{\hat{X}}(t)) = \mathrm{vec}(-\lambda E_2(t) - \hat{X}(t)(\dot{\hat{A}}(t)\hat{A}^*(t) + \hat{A}(t)\dot{\hat{A}}^*(t)) + \dot{\hat{A}}^*(t)), & m < n \end{cases}, \tag{3.17}$$

and after setting:

$$\begin{aligned} W(t) &= \begin{cases} I_{2m} \otimes \hat{A}^*(t)\hat{A}(t), & m \geq n \ \&\ \mathrm{rank}(\hat{A}(t)) = 2n \\ I_{2m} \otimes \hat{A}^*(t)\hat{A}(t) + \gamma I_{4mn}, & m \geq n \ \&\ \mathrm{rank}(\hat{A}(t)) < 2n \\ \hat{A}(t)\hat{A}^*(t) \otimes I_{2n}, & m < n \ \&\ \mathrm{rank}(\hat{A}(t)) = 2m \\ \hat{A}(t)\hat{A}^*(t) \otimes I_{2n} + \gamma I_{4mn}, & m < n \ \&\ \mathrm{rank}(\hat{A}(t)) < 2m \end{cases}, \\ H(t) &= \begin{cases} \mathrm{vec}(-\lambda E_1(t) - (\dot{\hat{A}}^*(t)\hat{A}(t) + \hat{A}^*(t)\dot{\hat{A}}(t))\hat{X}(t) + \dot{\hat{A}}^*(t)), & m \geq n \\ \mathrm{vec}(-\lambda E_2(t) - \hat{X}(t)(\dot{\hat{A}}(t)\hat{A}^*(t) + \hat{A}(t)\dot{\hat{A}}^*(t)) + \dot{\hat{A}}^*(t)), & m < n \end{cases}, \\ \dot{\hat{\mathbf{x}}}(t) &= \mathrm{vec}(\dot{\hat{X}}(t)), \quad \hat{\mathbf{x}}(t) = \mathrm{vec}(\hat{X}(t)), \end{aligned} \tag{3.18}$$

the ZNNQC model is derived for solving the TVQ-MPI of (2.6):

$$W(t)\dot{\hat{\mathbf{x}}}(t) = H(t) \tag{3.19}$$

where $\dot{\hat{\mathbf{x}}}(t), \hat{\mathbf{x}}(t), H(t) \in \mathbb{C}^{4mn}$, $W(t) \in \mathbb{C}^{4mn\times 4mn}$ is a nonsingular mass matrix and $\gamma \geq 0$ is the regularization parameter.

In terms of computational complexity, it is important to note that multiplying two complex numbers results in the calculation $(c+d\imath)(k+h\imath) = ck - dh + ch\imath + dk\imath$, which calls for a total of 4 multiplications and 2 addition/subtraction operations. Taking this into account, the complexity of computing (3.19) is $O(4(4mn)^2) = O((8mn)^2)$ as each iteration of (3.19) has $4(4mn)^2$ multiplication and $2(4mn)$ addition/subtraction operations. In addition, the complexity of solving (3.19) through use of an implicit ode MATLAB solver is $O((8mn)^3)$ as it involves a $(4mn) \times (4mn)$ matrix in the complex domain. As a consequence, the computational complexity of the ZNNQC model of (3.19) is $O((8mn)^3)$.

3.3. The ZNNQR model

To develop the ZNNQR model, the TVQ-MPI of (2.8) is considered. Let $A(t) \in \mathbb{C}^{4m\times 4n}$ and $X(t) \in \mathbb{R}^{4n\times 4m}$, we set the following ERME:

$$E_R(t) = \begin{cases} E_1(t) = A^{\mathrm{T}}(t)A(t)X(t) - A^{\mathrm{T}}(t), & m \geq n \\ E_2(t) = X(t)A(t)A^{\mathrm{T}}(t) - A^{\mathrm{T}}(t), & m < n \end{cases}, \tag{3.20}$$

where $E_1(t), E_2(t) \in \mathbb{R}^{4n\times 4m}$. The first time derivative of (3.20) is as follows:

$$\dot{E}_R(t) = \begin{cases} \dot{E}_1(t) = (\dot{A}^{\mathrm{T}}(t)A(t) + A^{\mathrm{T}}(t)\dot{A}(t))X(t) + A^{\mathrm{T}}(t)A(t)\dot{X}(t) - \dot{A}^{\mathrm{T}}(t), & m \geq n \\ \dot{E}_2(t) = \dot{X}(t)A(t)A^{\mathrm{T}}(t) + X(t)(\dot{A}(t)A^{\mathrm{T}}(t) + A(t)\dot{A}^{\mathrm{T}}(t)) - \dot{A}^{\mathrm{T}}(t), & m < n \end{cases}. \tag{3.21}$$

When $E_R(t)$ of (3.20) and $\dot{E}_R(t)$ of (3.21) are replaced in (2.9) and solving in terms of $X(t)$, we have the next result:

$$\begin{cases} A^{\mathrm{T}}(t)A(t)\dot{X}(t) = -\lambda E_1(t) - (\dot{A}^{\mathrm{T}}(t)A(t) + A^{\mathrm{T}}(t)\dot{A}(t))X(t) + \dot{A}^{\mathrm{T}}(t), & m \geq n \\ \dot{X}(t)A(t)A^{\mathrm{T}}(t) = -\lambda E_2(t) - X(t)(\dot{A}(t)A^{\mathrm{T}}(t) + A(t)\dot{A}^{\mathrm{T}}(t)) + \dot{A}^{\mathrm{T}}(t), & m < n \end{cases}. \tag{3.22}$$

Then, with the aid of the Kronecker product and the vectorization process, the dynamic model of (3.22) may be simplified:

$$\begin{cases} (I_{4m} \otimes A^{\mathrm{T}}(t)A(t))\mathrm{vec}(\dot{X}(t)) = \mathrm{vec}(-\lambda E_1(t) - (\dot{A}^{\mathrm{T}}(t)A(t) + A^{\mathrm{T}}(t)\dot{A}(t))X(t) + \dot{A}^{\mathrm{T}}(t)), & m \geq n \\ (A(t)A^{\mathrm{T}}(t) \otimes I_{4n})\mathrm{vec}(\dot{X}(t)) = \mathrm{vec}(-\lambda E_2(t) - X(t)(\dot{A}(t)A^{\mathrm{T}}(t) + A(t)\dot{A}^{\mathrm{T}}(t)) + \dot{A}^{\mathrm{T}}(t)), & m < n \end{cases}, \tag{3.23}$$

and after setting:

$$\begin{aligned} M(t) &= \begin{cases} I_{4m} \otimes A^{\mathrm{T}}(t)A(t), & m \geq n \ \&\ \mathrm{rank}(A(t)) = 4n \\ I_{4m} \otimes A^{\mathrm{T}}(t)A(t) + \gamma I_{16mn}, & m \geq n \ \&\ \mathrm{rank}(A(t)) < 4n \\ A(t)A^{\mathrm{T}}(t) \otimes I_{4n}, & m < n \ \&\ \mathrm{rank}(A(t)) = 4m \\ A(t)A^{\mathrm{T}}(t) \otimes I_{4n} + \gamma I_{16mn}, & m < n \ \&\ \mathrm{rank}(A(t)) < 4m \end{cases}, \\ P(t) &= \begin{cases} \mathrm{vec}(-\lambda E_1(t) - (\dot{A}^{\mathrm{T}}(t)A(t) + A^{\mathrm{T}}(t)\dot{A}(t))X(t) + \dot{A}^{\mathrm{T}}(t)), & m \geq n \\ \mathrm{vec}(-\lambda E_2(t) - X(t)(\dot{A}(t)A^{\mathrm{T}}(t) + A(t)\dot{A}^{\mathrm{T}}(t)) + \dot{A}^{\mathrm{T}}(t)), & m < n \end{cases}, \\ \dot{\mathbf{x}}(t) &= \mathrm{vec}(\dot{X}(t)), \quad \mathbf{x}(t) = \mathrm{vec}(X(t)), \end{aligned} \tag{3.24}$$

the ZNNQR model is derived for solving the TVQ-MPI of (2.6):

$$M(t)\dot{\mathbf{x}}(t) = P(t) \tag{3.25}$$

where $\dot{\mathbf{x}}(t), \mathbf{x}(t), P(t) \in \mathbb{R}^{16mn}$, $M(t) \in \mathbb{R}^{16mn\times 16mn}$ is a nonsingular mass matrix and $\gamma \geq 0$ is the regularization parameter.

Given that we perform $16mn$ additions/subtractions and $(16mn)^2$ multiplications in each iteration of (3.25), the complexity of solving (3.25) is $O((16mn)^2)$ operations. In addition, the complexity of solving (3.25) through use of an implicit `ode` MATLAB solver is $O((16mn)^3$ as it involves a $(16mn) \times (16mn)$ matrix. As a consequence, the computational complexity of the ZNNQR model of (3.25) is $O((16mn)^3)$.

4. Stability and convergence analysis

This section examines the convergence and stability of the ZNNQ (3.13), ZNNQC (3.19), and ZNNQR (3.25) models.

Theorem 4.1. *Assuming that* $Z_1(t) \in \mathbb{R}^{4m\times 4n}$, $Z_2(t) \in \mathbb{R}^{4n\times 4m}$, $Y_1(t), W_1(t) \in \mathbb{R}^{4n\times m}$ *and* $Y_2(t), W_2(t) \in \mathbb{R}^{n\times 4m}$, *and* $Z_1(t), Z_2(t), W_1(t)$ *and* $W_2(t)$ *are differentiable, the dynamical system* (3.10) *converges to* $\tilde{A}^{\dagger}(t)$, *which is the theoretical solution (THSO) of the TVQ-MPI* (2.6). *The solution is then stable, based on Lyapunov.*

Proof. The substitution $\bar{Y}_i(t) := \check{Y}_i(t) - Y_i(t), i = 1, 2$, implies $Y_i(t) = \check{Y}_i(t) - \bar{Y}_i(t)$, where $\check{Y}_i(t)$ is a THSO. The time derivative of $Y_i(t), i = 1, 2$, is $\dot{Y}_i(t) = \dot{\check{Y}}_i(t) - \dot{\bar{Y}}_i(t)$. Notice that

$$\begin{cases} Z_1(t)\check{Y}_1(t) - W_1(t) = \mathbf{0}_{4n\times m}, & m \geq n \\ \check{Y}_2(t)Z_2(t) - W_2(t) = \mathbf{0}_{n\times 4m}, & m < n \end{cases}, \tag{4.1}$$

and its first derivative

$$\begin{cases} \dot{Z}_1(t)\check{Y}_1(t) + Z_1(t)\dot{\check{Y}}_1(t) - \dot{W}_1(t) = \mathbf{0}_{4n\times m}, & m \geq n \\ \dot{\check{Y}}_2(t)Z_2(t) + \check{Y}_2(t)\dot{Z}_2(t) - \dot{W}_2(t) = \mathbf{0}_{n\times 4m}, & m < n \end{cases}. \tag{4.2}$$

As a result, following the substitution of $Y_i(t) = \check{Y}_i(t) - \bar{Y}_i(t), i = 1, 2$, into (3.8), one can verify

$$\bar{E}_Q(t) = \begin{cases} Z_1(t)(\check{Y}_1(t) - \bar{Y}_1(t)) - W_1(t), & m \geq n \\ (\check{Y}_2(t) - \bar{Y}_2(t))Z_2(t) - W_2(t), & m < n \end{cases}. \tag{4.3}$$

Further, the implicit dynamics (2.9) imply

$$\begin{aligned} \dot{\bar{E}}_Q(t) &= \begin{cases} \dot{Z}_1(t)(\check{Y}_1(t) - \bar{Y}_1(t)) + Z_1(t)(\dot{\check{Y}}_1(t) - \dot{\bar{Y}}_1(t)) - \dot{W}_1(t), & m \geq n \\ (\dot{\check{Y}}_2(t) - \dot{\bar{Y}}_2(t))Z_2(t) + (\check{Y}_2(t) - \bar{Y}_2(t))\dot{Z}_2(t) - \dot{W}_2(t), & m < n \end{cases} \\ &= -\lambda\bar{E}_Q(t). \end{aligned} \tag{4.4}$$

We then determine the candidate Lyapunov function so as to confirm convergence:

$$\mathcal{L}(t) = \frac{1}{2}\left\|\bar{E}_Q(t)\right\|_{\mathrm{F}}^2 = \frac{1}{2}\mathrm{Tr}\left(\bar{E}_Q(t)\left(\bar{E}_Q(t)\right)^{\mathrm{T}}\right). \tag{4.5}$$

Then, the next identities can be verified:

$$\dot{\mathcal{L}}(t) = \frac{2\mathrm{Tr}\left(\left(\bar{E}_Q(t)\right)^{\mathrm{T}}\dot{\bar{E}}_Q(t)\right)}{2} = \mathrm{Tr}\left(\left(\bar{E}_Q(t)\right)^{\mathrm{T}}\dot{\bar{E}}_Q(t)\right) = -\lambda\mathrm{Tr}\left(\left(\bar{E}_Q(t)\right)^{\mathrm{T}}\bar{E}_Q(t)\right). \tag{4.6}$$

Consequently, it holds

$$\begin{aligned}
&\frac{\mathrm{d}\mathcal{L}(t)}{\mathrm{d}t}\begin{cases}<0, & \bar{E}_Q(t)\neq 0\\ =0, & \bar{E}_Q(t)=0,\end{cases}\\
&\Leftrightarrow \dot{\mathcal{L}}(t)\begin{cases}<0, & \begin{cases}Z_1(t)(\check{Y}_1(t)-\bar{Y}_1(t))-W_1(t)\neq 0, m\geq n\\ (\check{Y}_2(t)-\bar{Y}_2(t))Z_2(t)-W_2(t)\neq 0, m<n\end{cases}\\ =0, & \begin{cases}Z_1(t)(\check{Y}_1(t)-\bar{Y}_1(t))-W_1(t)=0, m\geq n\\ (\check{Y}_2(t)-\bar{Y}_2(t))Z_2(t)-W_2(t)=0, m<n\end{cases}\end{cases},\\
&\Leftrightarrow \dot{\mathcal{L}}(t)\begin{cases}<0, & \begin{cases}\bar{Y}_1(t)\neq 0, m\geq n\\ \bar{Y}_2(t)\neq 0, m<n\end{cases}\\ =0, & \begin{cases}\bar{Y}_1(t)=0, m\geq n\\ \bar{Y}_2(t)=0, m<n\end{cases}\end{cases},
\end{aligned} \tag{4.7}$$

With $\bar{Y}(t)=\begin{cases}\bar{Y}_1(t), & m\geq n\\ \bar{Y}_2(t), & m<n\end{cases}$ being the equilibrium point of the system (4.4) and $E_Q(0)=0$, we have that:

$$\frac{\mathrm{d}\mathcal{L}(t)}{\mathrm{d}t}\leq 0,\quad \forall\ \bar{Y}(t)\neq 0. \tag{4.8}$$

By the Lyapunov stability theory, we infer that the equilibrium state:

$$\begin{cases}\bar{Y}_1(t)=\check{Y}_1(t)-Y_1(t)=0, & m\geq n\\ \bar{Y}_2(t)=\check{Y}_2(t)-Y_2(t)=0, & m<n\end{cases}, \tag{4.9}$$

is stable. Thus, $Y_i(t)\to\check{Y}_i(t), i=1,2$, as $t\to\infty$. □

Theorem 4.2. *Let $\tilde{A}(t)\in\mathbb{H}^{m\times n}$ be differentiable. For any initial value* $\mathbf{y}(0)$ *that one may consider, the ZNNQ model* (3.13) *converges exponentially to the THSO* $\check{\mathbf{y}}(t)$ *at each time t.*

Proof. First, the ERME of (3.8) is declared so as to determine the THSO of the TVQ-MPI. Second, the model (3.10) is developed utilizing the ZNN's architecture (2.9) for zeroing (3.8). So, when $t\to\infty$, $Y(t)\to\check{Y}(t)$ for any initial value, according to Theorem 4.1. Third, with the aid of the Kronecker product and the vectorization process, the dynamic model of (3.10) is simplified into the ZNNQ model (3.13). Therefore, the ZNNQ model (3.13) converges to the THSO $\check{\mathbf{y}}(t)$ for any initial value $\mathbf{y}(0)$ when $t\to\infty$, as it is simply an alternative version of (3.10). The proof is thus completed. □

Theorem 4.3. *Assuming that $\hat{A}(t)\in\mathbb{C}^{2m\times 2n}$ is differentiable, the dynamical system* (3.16) *converges to $\hat{A}^{\dagger}(t)$, which is the THSO of the TVQ-MPI* (2.7)*. The solution is then stable, based on Lyapunov.*

Proof. Given that the proof mirrors the proof of Theorem 4.1, it is omitted. □

Theorem 4.4. *Let $\hat{A}(t)\in\mathbb{C}^{2m\times 2n}$ be differentiable. For any initial value* $\hat{\mathbf{x}}(0)$ *that one may consider, the ZNNQC model* (3.19) *converges exponentially to the THSO* $\check{\hat{\mathbf{x}}}(t)$ *at each time t.*

Proof. Given that the proof mirrors the proof of Theorem 4.2 once we replace Theorem 4.1 with Theorem 4.3, it is omitted. □

Theorem 4.5. *Assuming that $A(t) \in \mathbb{R}^{4m\times 4n}$ is differentiable, the dynamical system* (3.22) *converges to $A^{\dagger}(t)$, which is the THSO of the TVQ-MPI* (2.8)*. The solution is then stable, based on Lyapunov.*

Proof. Given that the proof mirrors the proof of Theorem 4.1, it is omitted. □

Theorem 4.6. *Let $A(t) \in \mathbb{R}^{4m\times 4n}$ be differentiable. For any initial value* $\mathbf{x}(0)$ *that one may consider, the ZNNQR model* (3.25) *converges exponentially to the THSO* $\check{\mathbf{x}}(t)$ *at each time t.*

Proof. Given that the proof mirrors the proof of Theorem 4.2 once we replace Theorem 4.1 with Theorem 4.5, it is omitted. □

5. Experiments

In this section, four numerical simulations (NSs) and a real-world application involving robotic motion tracking are presented. The essential clarifications that have been used across all NSs and application are shown below. The ZNN design parameter λ is used with values of 10 and 100 in NSs and 10 in application, while the initial values of the ZNNQ, ZNNQC and ZNNQR models have been set to $\mathbf{y}(0) = \mathbf{0}_{4mn}$, $\hat{\mathbf{x}}(0) = \mathbf{0}_{4mn}$ and $\mathbf{x}(0) = \mathbf{0}_{16mn}$, respectively. Additionally, we have set $\alpha(t) = \sin(t)$ and $\beta(t) = \cos(t)$ and we will refer to the four Penrose equations in (1.1) as (P-i), (P-ii), (P-iii) and (P-iv) for convenience. The notation QMP (i.e. quartenion MP) in the figures legend refers to the MP-inverse of the input TVQ matrix $\tilde{A}(t)$, i.e. $\tilde{A}^{\dagger}(t)$. Finally, the NSs have used the MATLAB `ode15s` solver in the time interval $[0, 10]$ using the default double precision arithmetic ($eps = 2.22 \cdot 10^{-16}$), whereas the application has used the solver in the time interval $[0, 20]$. As a consequence, the minimum value in all of the figures of this section are mostly of order 10^{-5}.

5.1. Numerical simulations

Example 5.1. Considering the next coefficients:

$$A_1(t) = \begin{bmatrix} 2\alpha(t)+7 & 1 & 1 \\ -2 & 1 & 1 \\ 4 & 1 & 1 \end{bmatrix}, A_2(t) = \begin{bmatrix} 5 & 1 & 1 \\ 2\alpha(t)+1 & 1 & 1 \\ 3 & 1 & 1 \end{bmatrix}, A_3(t) = \begin{bmatrix} 3\alpha(t)+2 & 1 & 1 \\ 12 & 1 & 1 \\ 5 & 1 & 1 \end{bmatrix}, A_4(t) = \begin{bmatrix} -2 & 1 & 1 \\ 2\alpha(t)+1 & 1 & 1 \\ 7 & 1 & 1 \end{bmatrix},$$

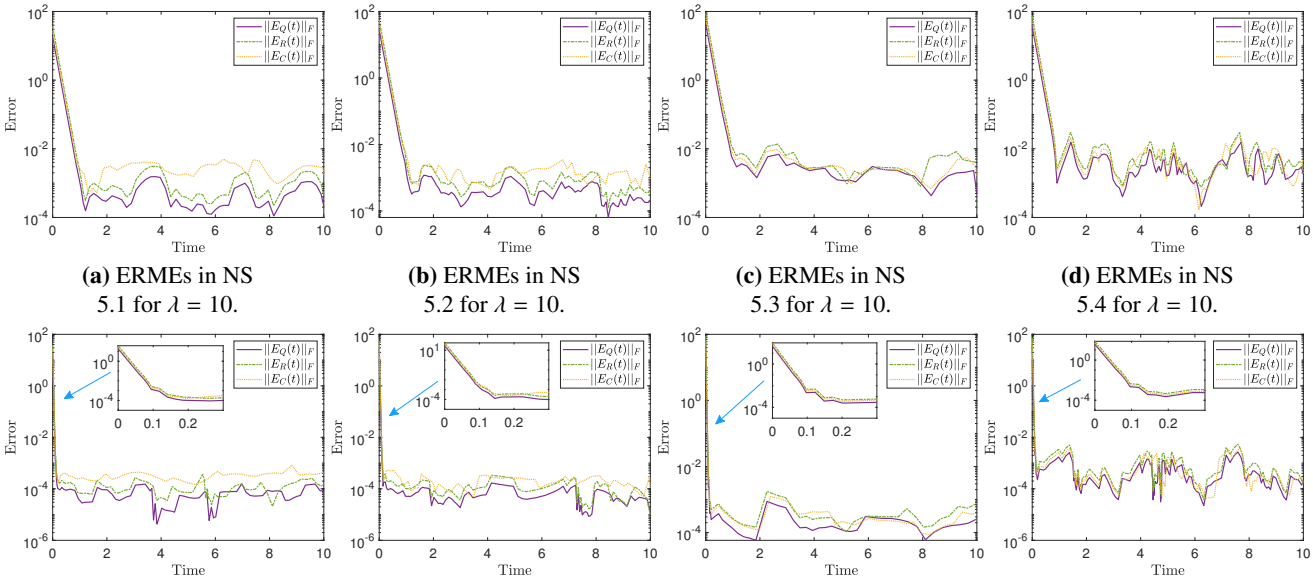
the input matrix $\tilde{A}(t) \in \mathbb{H}^{3\times 3}$ is a singular TVQ matrix with $\text{rank}(\tilde{A}(t)) = 2$. As a consequence, we set $\gamma = 10^{-8}$ in the ZNNQ, ZNNQC and ZNNQR models. The performance of the ZNN models is shown in Figures 1, 2 and 3.

Example 5.2. Utilizing the next coefficients:

$$A_1(t) = \begin{bmatrix} 2\alpha(t)+1 & 7 \\ 3\beta(t)+2 & 7 \\ 1 & \beta(t)+2 \\ 2\beta(t)+4 & 7 \end{bmatrix}, \quad A_2(t) = \begin{bmatrix} 2 & 2\alpha(t)+1 \\ 2\alpha(t)-3 & 7 \\ -2\beta(t)+4 & \beta(t)+6 \\ 7 & 2\alpha(t)+1 \end{bmatrix},$$

$$A_3(t) = \begin{bmatrix} 3\alpha(t)+2 & 6 \\ 2\alpha(t)+1 & 7 \\ -\beta(t)+2 & 3 \\ -\beta(t)+5 & 7 \end{bmatrix}, \quad A_4(t) = \begin{bmatrix} 3 & 2\alpha(t)+1 \\ 7 & 7 \\ 3\alpha(t)+2 & 5 \\ 7 & 2\alpha(t)+1 \end{bmatrix},$$

the input matrix $\tilde{A}(t) \in \mathbb{H}^{4\times 2}$ is a full rank TVQ matrix with $\text{rank}(\tilde{A}(t)) = 2$. As a consequence, we set $\gamma = 0$ in the ZNNQ, ZNNQC and ZNNQR models. The performance of the ZNN models is shown in Figures 1, 2 and 3.



(a) ERMEs in NS 5.1 for $\lambda = 10$. **(b)** ERMEs in NS 5.2 for $\lambda = 10$. **(c)** ERMEs in NS 5.3 for $\lambda = 10$. **(d)** ERMEs in NS 5.4 for $\lambda = 10$.

(e) ERMEs in NS 5.1 for $\lambda = 100$. **(f)** ERMEs in NS 5.2 for $\lambda = 100$. **(g)** ERMEs in NS 5.3 for $\lambda = 100$. **(h)** ERMEs in NS 5.4 for $\lambda = 100$.

Figure 1. ERMEs in NSs 5.1–5.4 for λ with values 10 and 100.

Example 5.3. Using the following coefficients:

$$A_1(t) = \begin{bmatrix} 2\alpha(t)+7 & -2 & 4 & \beta(t) & 8 & 5 \\ 2\alpha(t)+7 & -2 & 4 & \beta(t) & 9 & 1 \\ 2\alpha(t)+7 & -2 & 4 & \beta(t) & 9 & 1 \end{bmatrix}, \quad A_2(t) = \begin{bmatrix} 5 & 2\alpha(t)+1 & 3 & \beta(t) & 8 & 5 \\ 5 & 2\alpha(t)+1 & 3 & \beta(t) & 9 & 1 \\ 5 & 2\alpha(t)+1 & 3 & \beta(t) & 9 & 1 \end{bmatrix},$$

$$A_3(t) = \begin{bmatrix} 3\alpha(t)+2 & 12 & 5 & \beta(t) & 8 & 5 \\ 3\alpha(t)+2 & 12 & 5 & \beta(t) & 9 & 1 \\ 3\alpha(t)+2 & 12 & 5 & \beta(t) & 9 & 1 \end{bmatrix}, \quad A_4(t) = \begin{bmatrix} -2 & 2\alpha(t)+1 & 7 & \beta(t) & 8 & 5 \\ -2 & 2\alpha(t)+1 & 7 & \beta(t) & 9 & 1 \\ -2 & 2\alpha(t)+1 & 7 & \beta(t) & 9 & 1 \end{bmatrix},$$

the input matrix $\tilde{A}(t) \in \mathbb{H}^{3\times 6}$ is a rank deficient TVQ matrix with $\text{rank}(\tilde{A}(t)) = 2$. As a consequence, we set $\gamma = 10^{-8}$ in the ZNNQ, ZNNQC and ZNNQR models. The performance of the ZNN models is shown in Figures 1, 2 and 3.

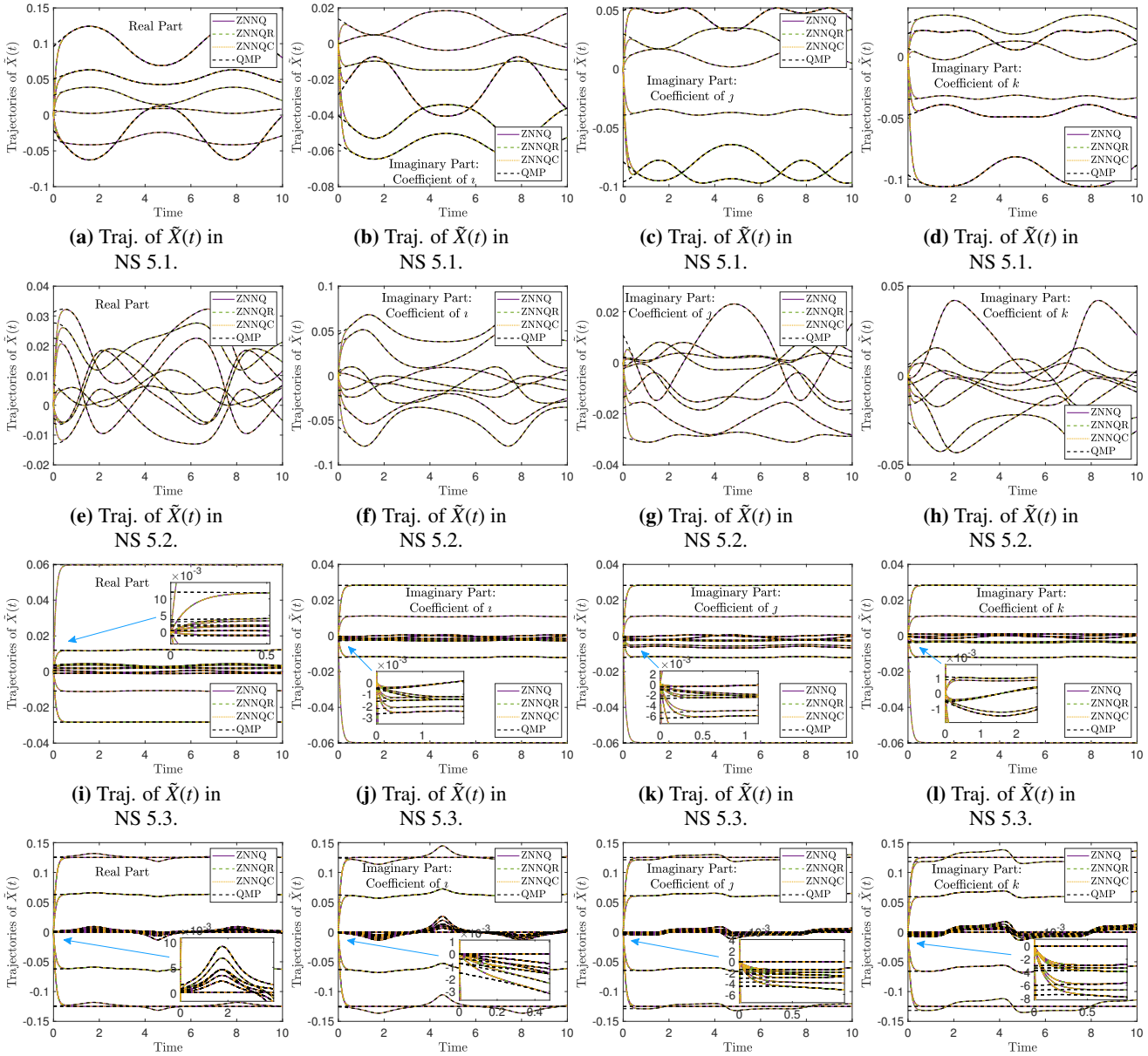
Example 5.4. Considering the following matrix

$$K = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix},$$

the coefficients of the input matrix $\tilde{A}(t)$ have been set to

$$A_1(t) = K^{\mathrm{T}} \odot (1 + \alpha(t)), \quad A_2(t) = K^{\mathrm{T}} \odot (1 + 2\alpha(t)),$$
$$A_3(t) = K^{\mathrm{T}} \odot (1 + 2\beta(t)), \quad A_4(t) = K^{\mathrm{T}} \odot (1 + 4\beta(t)).$$

As a consequence, $\tilde{A}(t) \in \mathbb{H}^{12\times 4}$ is a rank deficient TVQ matrix with $\mathrm{rank}(\tilde{A}(t)) = 3$ and, thus, we set $\gamma = 10^{-8}$ in the ZNNQ, ZNNQC and ZNNQR models. The performance of the ZNN models is shown in Figures 1, 2 and 3.



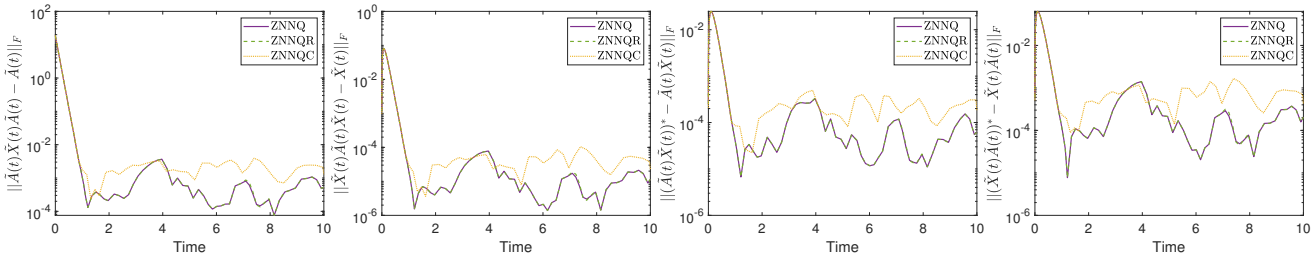
(a) Traj. of $\tilde{X}(t)$ in NS 5.1. **(b)** Traj. of $\tilde{X}(t)$ in NS 5.1. **(c)** Traj. of $\tilde{X}(t)$ in NS 5.1. **(d)** Traj. of $\tilde{X}(t)$ in NS 5.1.

(e) Traj. of $\tilde{X}(t)$ in NS 5.2. **(f)** Traj. of $\tilde{X}(t)$ in NS 5.2. **(g)** Traj. of $\tilde{X}(t)$ in NS 5.2. **(h)** Traj. of $\tilde{X}(t)$ in NS 5.2.

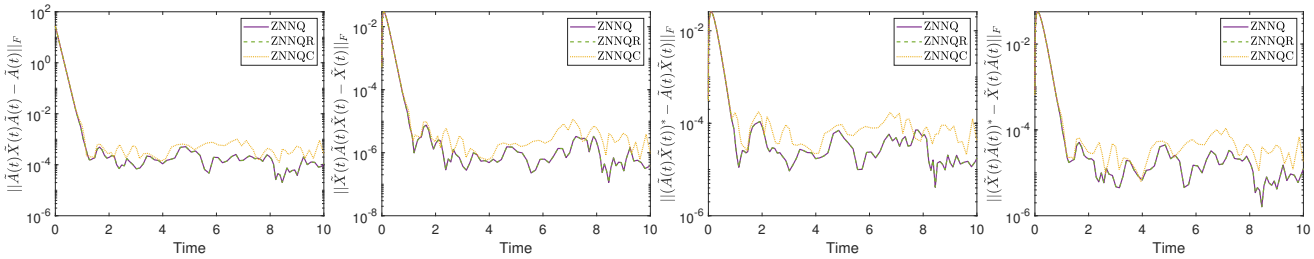
(i) Traj. of $\tilde{X}(t)$ in NS 5.3. **(j)** Traj. of $\tilde{X}(t)$ in NS 5.3. **(k)** Traj. of $\tilde{X}(t)$ in NS 5.3. **(l)** Traj. of $\tilde{X}(t)$ in NS 5.3.

(m) Traj. of $\tilde{X}(t)$ in NS 5.4. **(n)** Traj. of $\tilde{X}(t)$ in NS 5.4. **(o)** Traj. of $\tilde{X}(t)$ in NS 5.4. **(p)** Traj. of $\tilde{X}(t)$ in NS 5.4.

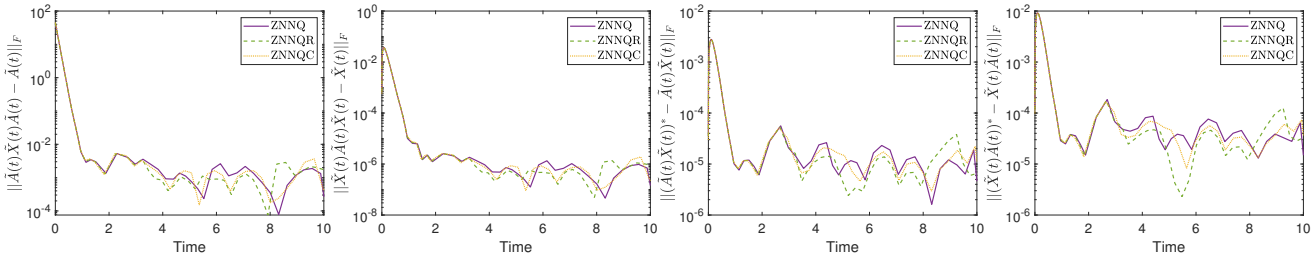
Figure 2. Real and imaginary parts of the trajectories of $\tilde{X}(t)$ in NSs 5.1–5.4 for $\lambda = 10$.



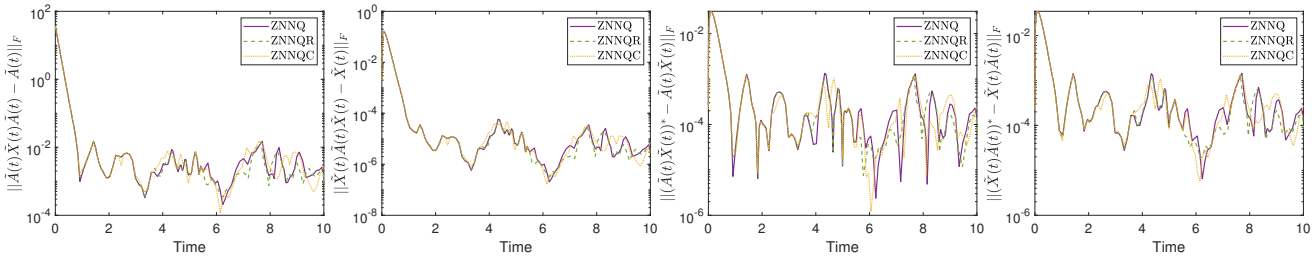
(a) Error of (P-i) in NS 5.1. **(b)** Error of (P-ii) in NS 5.1. **(c)** Error of (P-iii) in NS 5.1. **(d)** Error of (P-iv) in NS 5.1.



(e) Error of (P-i) in NS 5.2. **(f)** Error of (P-ii) in NS 5.2. **(g)** Error of (P-iii) in NS 5.2. **(h)** Error of (P-iv) in NS 5.2.



(i) Error of (P-i) in NS 5.3. **(j)** Error of (P-ii) in NS 5.3. **(k)** Error of (P-iii) in NS 5.3. **(l)** Error of (P-iv) in NS 5.3.



(m) Error of (P-i) in NS 5.4. **(n)** Error of (P-ii) in NS 5.4. **(o)** Error of (P-iii) in NS 5.4. **(p)** Error of (P-iv) in NS 5.4.

Figure 3. Error of Penrose equations (1.1) in NSs 5.1–5.4 for $\lambda = 10$.

5.2. Application to robotic motion tracking

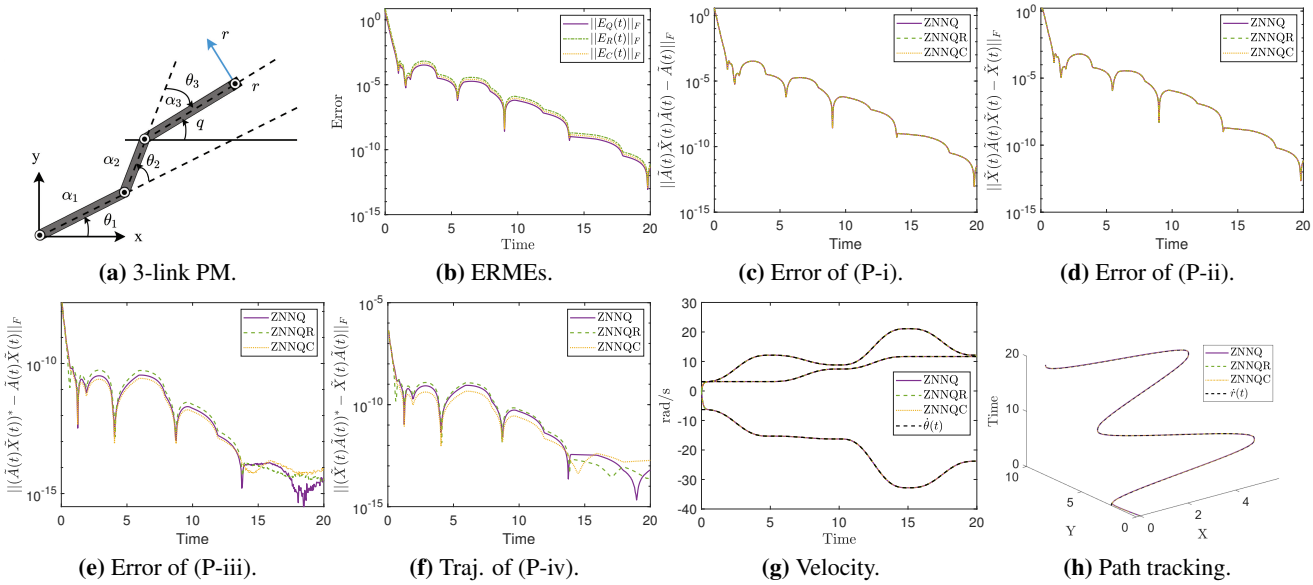
The applicability of the ZNNQ, ZNNQC and ZNNQR models is validated in this experiment using a 3-link planar manipulator (PM), as shown in Figure 4a. It is important to mention that the 3-link PM's kinematics equations at the position level $r(t) \in \mathbb{R}^m$ and the velocity level $\dot{r}(t) \in \mathbb{R}^m$ are expressed as follows:

$$r(t) = f(\theta(t)), \quad \dot{r}(t) = J(\theta)\dot{\theta}(t),$$

where $\theta \in \mathbb{R}^n$ is the angle of the 3-link PM, $f(\cdot)$ is a smooth nonlinear mapping function, $r(t)$ is the end-effector's position, and $J(\theta) = \partial f(\theta)/\partial\theta \in \mathbb{R}^{m\times n}$.

To comprehend how this 3-link PM tracked motion, the inverse kinematic equation is solved. The equation of velocity can be thought of as a system of linear equations when the end-effector motion tracking task is assigned with $\dot{r}(t)$ known and $\dot{\theta}(t)$ unknown. To put it another way, by setting $\tilde{A}(t) = J(\theta)$, we find $\tilde{X}(t) = A^{\dagger}(t)$ to solve $\dot{\theta}(t) = \tilde{X}(t)\dot{r}(t)$. Therefore, we may track control of the 3-link PM by using the ZNN models to resolve the underlying linear equation system.

The 3-link PM's end-effector is anticipated to follow a "M"-shaped path in the simulation experiment; [58] contains the X and Y-axis velocity functions of this path along with the specifications of 3-link PM. The task duration $4T$ is 20 seconds (i.e., $T = 5$ seconds) in these functions, and the design parameter is $s = 6$ cm. Additionally, the link length is $\alpha = [1, 2/3, 5/4]^{\mathrm{T}}$ and the initial value of the joints is $\theta(0) = [\pi/4, \pi/4, \pi/4]^{\mathrm{T}}$. The performance of the ZNN models is shown in Figure 4.



(a) 3-link PM. **(b)** ERMEs. **(c)** Error of (P-i). **(d)** Error of (P-ii).

(e) Error of (P-iii). **(f)** Traj. of (P-iv). **(g)** Velocity. **(h)** Path tracking.

Figure 4. Robotic motion tracking application results in Section 5.2.

5.3. Numerical simulations discussion

The performance of the ZNNQ (3.13), ZNNQC (3.19) and ZNNQR (3.10) models for solving the TVQ-MPI (2.6) is investigated throughout the NSs 5.1–5.4. While the input TVQ matrices $\tilde{A}(t)$ used have varied dimensions and rank conditions, each NS solves the TVQ-MPI problem. Particularly, NS 5.1 has a singular TVQ matrix of dimensions 3×3, NS 5.2 has a full rank TVQ matrix of dimensions 4×2, NS 5.3 has a rank deficient TVQ matrix of dimensions 3×6, while NS 5.4 has a rank deficient TVQ matrix of dimensions 12×4, which is much larger than the other NSs.

Figsure 1a–1d and Figure 1e–1h show the ERME's Frobenius norms of the ZNNQ, ZNNQC and ZNNQR models for λ values of 10 and 100 in NSs 5.1–5.4, respectively. In the case of $\lambda = 10$ in Figure 1a–1d, it can be observed that the error values in all NSs start from a high error value at $t = 0$ and, by the time-mark of $t \approx 1$, they experience a steep decline that brings them to the range $[10^{-4}, 10^{-2}]$.

Notice that a higher value for λ will typically cause the ZNN models to converge even more quickly. This is demonstrated in the case of $\lambda = 100$ in Figures 1e–1h, where we observe that the error values in all NSs start from a high error value at $t = 0$ and, by the time-mark of $t \approx 0.1$, they experience a steep decline that brings them to the range $[10^{-4}, 10^{-3}]$. Also, all ZNN models exhibit the same convergence speed, but the ZNNQC has the highest overall error in the region $[0, 10]$ while the ZNNQ has the lowest. In other words, the ZNNQ model shows better performance than the ZNNQC and ZNNQR models.

The fact that all three models successfully converged is further highlighted in Figure 2, which contrasts the THSO's real and imaginary parts trajectories with the corresponding $\tilde{X}(t)$ trajectories produced by the three models. Particularly, Figure 2a depicts the real part and Figure 2b–2d depict the imaginary parts in NS 5.1, Figure 2e depicts the real part and Figure 2f–2h depict the imaginary parts in NS 5.2, Figure 2i depicts the real part and Figure 2j–2l depict the imaginary parts in NS 5.3, and Figure 2m depicts the real part and Figure 2n–2p depict the imaginary parts in NS 5.4. In these figures, it can be observed that the three models' $\tilde{X}(t)$ trajectories coincide with the corresponding THSO's trajectories, whereas their convergence speed follows the convergence tendency of the ZNNQ, ZNNQC and ZNNQR models' ERMEs shown in Figure 1. It is important to note that the convergence and stability theorems of Section 4 are validated in all of the figures in this section by the convergence tendency of the ERME's Frobenius norms alongside the solution trajectories that match the THSO's trajectories. That is, the ZNNQ, ZNNQC and ZNNQR models of Section 3 converge exponentially to the QMP, i.e. $\tilde{A}^{\dagger}(t)$, for any initial value when $t \to \infty$.

By assessing the error of Penrose equations (1.1), the three models' performance is examined in order to further validate them. Particularly, Figure 3a, 3e, 3i and 3m depict the error of (P-i) in NSs 5.1–5.4, respectively, Figure 3b, 3f, 3j and 3n depict the error of (P-ii) in NSs 5.1–5.4, respectively, Figure 3c, 3g, 3k and 3o depict the error of (P-iii) in NSs 5.1–5.4, respectively, and Figure 3d, 3h, 3l and 3p depict the error of (P-iv) in NSs 5.1–5.4, respectively. In these figures, it can be observed that convergence speed of the error produced by the three models follows the convergence tendency of the ZNNQ, ZNNQC and ZNNQR models' ERMEs shown in Figure 1. In NSs 5.1 and 5.2, the ZNNQ and ZNNQR have identical performance. Additionally, the ZNNQC has the highest overall error in the region $[0, 10]$ while the ZNNQ and ZNNQR have has the lowest. In NSs 5.3 and 5.4, all models produces almost identical overall error values.

Additionally, the applicability of the ZNNQ, ZNNQC and ZNNQR models is validated in the experiment of Section 5.2 using a 3-link PM. Particularly, Figure 4b shows the ERME's Frobenius norms of the ZNN models. It can be observed that the error values start from a high error value at $t = 0$ and, by the time-mark of $t \approx 1$, they experience a steep decline that brings them to the range $[10^{-5}, 10^{-4}]$. The error values fall more gradually after $t \approx 1$ until $t = 20$, when they drop to a range of $[10^{-13}, 10^{-11}]$. All ZNN models exhibit the same convergence speed and very similar overall error in the region $[0, 20]$, but the ZNNQR has the highest overall error while the ZNNQ has the lowest. Figsure 4c–4f depict the error of (P-i)-(P-iv), respectively. In these figures, it can be observed that convergence speed of the error produced by the three models follows the convergence tendency of the ZNNQ, ZNNQC and ZNNQR models' ERMEs shown in Figure 4b. In other words, the ZNNQ model shows better performance than the ZNNQC and ZNNQR models. Figure 4g and 4h depict the trajectories of the velocity and the "M"-shaped path tracking. As seen in these figures, all ZNN model solutions match the actual velocity $\dot{\theta}(t)$, and the 3-link PM successfully completes the "M"-shaped path tracking task, where $\dot{r}(t)$

is the actual "M"-shaped path.

Not to mention, once we take into account the complexity of each model, the results above can be placed into better context. Because the dimensions of the associated real valued matrix $A(t)$ are 2 times larger than those of the complex valued matrix $\hat{A}(t)$ and 4 times larger than those of the quaternion valued matrix $\tilde{A}(t)$, the ZNNQR is, by far, the most complex model. Because of this, choosing to solve the TVQ-MPI problem in the real domain has a significant memory penalty, with RAM fast being a limiting factor as $\tilde{A}(t)$ grows in size. When everything is taken into account, all three ZNN models can solve the TVQ-MPI problem, although the ZNNQ appears to have the most potential.

6. Conclusions

Three models, namely ZNNQ, ZNNQC and ZNNQR, have been presented in order to address the TVQ-MPI problem for TVQ matrices of arbitrary dimension. The creation of those models has been aided by theoretical research and an examination of their computing complexity, in addition to simulated examples and the real-world application involving robotic motion tracking. The TVQ-MPI problem has been successfully solved both directly in the quaternion domain and indirectly, through representation in the complex and real domains. Of the three methods, the direct method, implemented by the ZNNQ model, has been suggested as the most effective and efficient. In light of this, the established findings pave the path for more engaging research projects. The following considerations ought to be taken into account:

- Using the predefined-time ZNN architecture to TVQ-based problems is something that can be looked into.
- Solving nonlinear TVQ-based matrix equations is another task that could be taken into consideration.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by a Mega Grant from the Government of the Russian Federation within the framework of federal project No. 075-15-2021-584.

Conflict of interest

Vasilios N. Katsikis is an editorial board member for AIMS Mathematics and was not involved in the editorial review or the decision to publish this article. All authors declare that there are no competing interests.

References

1. A. Ben-Israel, T. N. E. Greville, *Generalized Inverses: Theory and Applications*, 2nd edition, CMS Books in Mathematics, Springer, New York, NY, 2003.

2. G. Wang, Y. Wei, S. Qiao, P. Lin, Y. Chen, *Generalized inverses: Theory and computations*, vol. 53, Springer: Singapore, 2018.
3. S. Zhang, Y. Dong, Y. Ouyang, Z. Yin, K. Peng, Adaptive neural control for robotic manipulators with output constraints and uncertainties, *IEEE T. Neur. Net. Lear.* , **29** (2018), 5554–5564.
4. Y. Shi, W. Zhao, S. Li, B. Li, X. Sun, Novel discrete-time recurrent neural network for robot manipulator: A direct discretization technical route, *IEEE T. Neur. Net. Lear.*, **34** (2023), 2781–2790.
5. Y. Shi, J. Wang, S. Li, B. Li, X. Sun, Tracking control of cable-driven planar robot based on discrete-time recurrent neural network with immediate discretization method, *IEEE T. Ind. Inform.*, **19** (2023), 7414–7423.
6. Y. Yuan, Z. Wang, L. Guo, Event-triggered strategy design for discrete-time nonlinear quadratic games with disturbance compensations: The noncooperative case, *IEEE T. Syst. Man, Cy-S.*, **48** (2018), 1885–1896.
7. S. D. Mourtas, V. N. Katsikis, C. Kasimis, Feedback control systems stabilization using a bio-inspired neural network, *EAI Endorsed Trans. AI Robotics*, **1** (2022), 1–13.
8. X. Yang, H. He, Self-learning robust optimal control for continuous-time nonlinear systems with mismatched disturbances, *Neural Network*, **99** (2018), 19–30. https://doi.org/10.1016/j.neunet.2017.11.022
9. S. D. Mourtas, A weights direct determination neuronet for time-series with applications in the industrial indices of the federal reserve bank of St. Louis, *J. Forecasting*, **14** (2022), 1512–1524.
10. S. Li, J. He, Y. Li, M. U. Rafique, Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective, *IEEE T. Neur. Net. Lear.*, **28** (2017), 415–426. https://doi.org/10.1177/0959683617729447
11. M. Joldeş, J. M. Muller, Algorithms for manipulating quaternions in floating-point arithmetic, In: *2020 IEEE 27th Symposium on Computer Arithmetic (ARITH)*, IEEE, 2020, 48–55.
12. A. Szynal-Liana, I. Włoch, Generalized commutative quaternions of the Fibonacci type, *Boletín de la Sociedad Matemática Mexicana*, **28** (2022), 1.
13. D. Pavllo, C. Feichtenhofer, M. Auli, D. Grangier, Modeling human motion with quaternion-based neural networks, *Int. J. Comput. Vision*, **128** (2020), 855–872. https://doi.org/10.1007/s11263-019-01207-y
14. E. Özgür, Y. Mezouar, Kinematic modeling and control of a robot arm using unit dual quaternions, *Robot. Auton. Syst.*, **77** (2016), 66–73. https://doi.org/10.1016/j.robot.2015.12.005
15. G. Du, Y. Liang, B. Gao, S. A. Otaibi, D. Li, A cognitive joint angle compensation system based on self-feedback fuzzy neural network with incremental learning, *IEEE T. Ind. Inform.*, **17** (2021), 2928–2937.
16. A. M. S. Goodyear, P. Singla, D. B. Spencer, Analytical state transition matrix for dual-quaternions for spacecraft pose estimation, In: *AAS/AIAA Astrodynamics Specialist Conference, 2019*, Univelt Inc., 2020, 393–411.
17. S. Giardino, Quaternionic quantum mechanics in real Hilbert space, *J. Geom. Phys.*, **158** (2020), 103956. https://doi.org/10.1016/j.geomphys.2020.103956

18. M. E. Kansu, Quaternionic representation of electromagnetism for material media, *Int. J. Geom. Methods M.*, **16** (2019), 1950105. https://doi.org/10.1142/S0219887819501056

19. Z. H. Weng, Field equations in the complex quaternion spaces, *Adv. Math. Phys.*, 2014.

20. R. Ghiloni, V. Moretti, A. Perotti, Continuous slice functional calculus in quaternionic Hilbert spaces, *Rev. Math. Phys.*, **25** (2013), 1350006. https://doi.org/10.1142/S0129055X13500062

21. I. I. Kyrchei, D. Mosić, P. S. Stanimirović, MPCEP-∗CEPMP-solutions of some restricted quaternion matrix equations, *Adv. Appl. Clifford Al.*, **32** (2022), 22, Id/No 16.

22. L. Huang, Q. W. Wang, Y. Zhang, The Moore-Penrose inverses of matrices over quaternion polynomial rings, *Linear Algebra Appl.*, **475** (2015), 45–61. https://doi.org/10.1016/j.laa.2015.02.033

23. L. Xiao, S. Liu, X. Wang, Y. He, L. Jia, Y. Xu, Zeroing neural networks for dynamic quaternion-valued matrix inversion, *IEEE T. Ind. Inform.*, **18** (2022), 1562–1571.

24. L. Xiao, W. Huang, X. Li, F. Sun, Q. Liao, L. Jia, et al., ZNNs with a varying-parameter design formula for dynamic Sylvester quaternion matrix equation, *IEEE T. Neur. Net. Lear.* , 1–11.

25. L. Xiao, P. Cao, W. Song, L. Luo, W. Tang, A fixed-time noise-tolerance ZNN model for time-variant inequality-constrained quaternion matrix least-squares problem, *IEEE T. Neur. Net. Lear.*, 1–10.

26. L. Xiao, Y. Zhang, W. Huang, L. Jia, X. Gao, A dynamic parameter noise-tolerant zeroing neural network for time-varying quaternion matrix equation with applications, *IEEE T. Neur. Net. Lear.*, 1–10.

27. N. Tan, P. Yu, F. Ni, New varying-parameter recursive neural networks for model-free kinematic control of redundant manipulators with limited measurements, *IEEE T. Instrum. Meas.*, **71** (2022), 1–14.

28. R. Abbassi, H. Jerbi, M. Kchaou, T. E. Simos, S. D. Mourtas, V. N. Katsikis, Towards higher-order zeroing neural networks for calculating quaternion matrix inverse with application to robotic motion tracking, *Mathematics*, **11** (2023), 2756.

29. V. N. Kovalnogov, R. V. Fedorov, D. A. Demidov, M. A. Malyoshina, T. E. Simos, V. N. Katsikis, et al., Zeroing neural networks for computing quaternion linear matrix equation with application to color restoration of images, *AIMS Math.*, **8** (2023), 14321–14339. https://doi.org/10.3934/math.2023733

30. Y. Zhang, S. S. Ge, Design and analysis of a general recurrent neural network model for time-varying matrix inversion, *IEEE T. Neural Networ.*, **16** (2005), 1477–1490.

31. Y. Chai, H. Li, D. Qiao, S. Qin, J. Feng, A neural network for Moore-Penrose inverse of time-varying complex-valued matrices, *Int. J. Comput. Intell. Syst.*, **13** (2020), 663–671.

32. Z. Sun, F. Li, L. Jin, T. Shi, K. Liu, Noise-tolerant neural algorithm for online solving time-varying full-rank matrix Moore-Penrose inverse problems: A control-theoretic approach, *Neurocomputing*, **413** (2020), 158–172. https://doi.org/10.1016/j.neucom.2020.06.050

33. W. Wu, B. Zheng, Improved recurrent neural networks for solving Moore-Penrose inverse of real-time full-rank matrix, *Neurocomputing*, **418** (2020), 221–231. https://doi.org/10.1016/j.neucom.2020.08.026

34. 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. https://doi.org/10.1007/s00607-010-0133-9

35. S. Qiao, X. Z. Wang, Y. Wei, Two finite-time convergent Zhang neural network models for time-varying complex matrix Drazin inverse, *Linear Algebra Appl.*, **542** (2018), 101–117.

36. S. Qiao, Y. Wei, X. Zhang, Computing time-varying ML-weighted pseudoinverse by the Zhang neural networks, *Numer. Func. Anal. Opt.*, **41** (2020), 1672–1693.

37. X. Wang, P. S. Stanimirovic, Y. Wei, Complex ZFs for computing time-varying complex outer inverses, *Neurocomputing*, **275** (2018), 983–1001. https://doi.org/10.1016/j.neucom.2017.09.034

38. T. E. Simos, V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, D. Gerontitis, A higher-order zeroing neural network for pseudoinversion of an arbitrary time-varying matrix with applications to mobile object localization, *Inform. Sciences*, **600** (2022), 226–238. https://doi.org/10.1016/j.ins.2022.03.094

39. M. Zhou, J. Chen, P. S. Stanimirovic, V. N. Katsikis, H. Ma, Complex varying-parameter Zhang neural networks for computing core and core-EP inverse, *Neural Process. Lett.*, **51** (2020), 1299–1329.

40. J. Liu, H. Cai, C. Jiang, X. Han, Z. Zhang, An interval inverse method based on high dimensional model representation and affine arithmetic, *Appl. Math. Model.*, **63** (2018), 732–743. https://doi.org/10.1016/j.apm.2018.07.009

41. S. D. Mourtas, V. N. Katsikis, Exploiting the Black-Litterman framework through error-correction neural networks, *Neurocomputing*, **498** (2022), 43–58. https://doi.org/10.1016/j.neucom.2022.05.036

42. V. N. Kovalnogov, R. V. Fedorov, D. A. Generalov, A. V. Chukalin, V. N. Katsikis, S. D. Mourtas, et al., Portfolio insurance through error-correction neural networks, *Mathematics*, **10** (2022), 3335.

43. S. D. Mourtas, C. Kasimis, Exploiting mean-variance portfolio optimization problems through zeroing neural networks, *Mathematics*, **10** (2022), 3079. https://doi.org/10.3390/math10173079

44. Y. Shi, L. Jin, S. Li, J. Li, J. Qiang, D. Gerontitis, Novel discrete-time recurrent neural networks handling discrete-form time-variant multi-augmented Sylvester matrix problems and manipulator application, *IEEE T. Neur. Net. Lear.*, **33** (2022), 587–599.

45. L. Xiao, B. Liao, S. Li, Z. Zhang, L. Ding, L. Jin, Design and analysis of FTZNN applied to the real-time solution of a nonstationary Lyapunov equation and tracking control of a wheeled mobile manipulator, *IEEE T. Ind. Inform.*, **14** (2018), 98–105.

46. W. Jiang, C. L. Lin, V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, T. E. Simos, Zeroing neural network approaches based on direct and indirect methods for solving the Yang–Baxter-like matrix equation, *Mathematics*, **10** (2022), 1950.

47. V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, Y. Zhang, Continuous-time varying complex QR decomposition via zeroing neural dynamics, *Neural Processing Letters*.

48. P. S. Stanimirović, V. N. Katsikis, S. Li, Higher-order ZNN dynamics, *Neural Process. Lett.*, 1–25.

49. V. N. Katsikis, P. S. Stanimirović, S. D. Mourtas, L. Xiao, D. Karabasević, D. Stanujkić, Zeroing neural network with fuzzy parameter for computing pseudoinverse of arbitrary matrix, *IEEE T. Fuzzy Syst.*, **30** (2022), 3426–3435.
50. M. Kornilova, V. Kovalnogov, R. Fedorov, M. Zamaleev, V. N. Katsikis, S. D. Mourtas, et al., Zeroing neural network for pseudoinversion of an arbitrary time-varying matrix based on singular value decomposition, *Mathematics*, **10** (2022), 1208. `https://www.mdpi.com/2227-7390/10/8/1208`.
51. L. Jin, S. Li, L. Xiao, R. Lu, B. Liao, Cooperative motion generation in a distributed network of redundant robot manipulators with noises, *IEEE T. Syst. Man Cy-S.*, **48** (2018), 1715–1724.
52. F. Zhang, Quaternions and matrices of quaternions, *Linear Algebra Appl.*, **251** (1997), 21–57.
53. J. Groß, G. Trenkler, S. O. Troschke, Quaternions: Further contributions to a matrix oriented approach, *Linear Algebra Appl.*, **326** (2001), 205–213.
54. R. W. Farebrother, J. Groß, S. O. Troschke, Matrix representation of quaternions, *Linear Algebra Appl.*, **362** (2003), 251–255.
55. J. Dai, P. Tan, X. Yang, L. Xiao, L. Jia, Y. He, A fuzzy adaptive zeroing neural network with superior finite-time convergence for solving time-variant linear matrix equations, *Knowl-Based Syst.*, **242** (2022), 108405. https://doi.org/10.1016/j.knosys.2022.108405
56. L. Xiao, H. Tan, J. Dai, L. Jia, W. Tang, High-order error function designs to compute time-varying linear matrix equations, *Inform. Sciences*, **576** (2021), 173–186. https://doi.org/10.1016/j.ins.2021.06.038
57. N. Zhong, Q. Huang, S. Yang, F. Ouyang, Z. Zhang, A varying-parameter recurrent neural network combined with penalty function for solving constrained multi-criteria optimization scheme for redundant robot manipulators, *IEEE Access*, **9** (2021), 50810–50818. https://doi.org/10.1109/ACCESS.2021.3068731
58. Y. Zhang, L. Jin, *Robot Manipulator Redundancy Resolution*, John Wiley & Sons: Hoboken, NJ, USA, 2017.

AIMS Press