AIMS Mathematics, 11(11): 36634–36659.
DOI: 10.3934/math.20261416
Received: 25 August 2026
Revised: 02 October 2026
Accepted: 10 October 2026
Published: 05 November 2026

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

Research article

Gradient neurodynamics (GN), zeroing neurodynamics (ZN), and combination application

Zhengyang Tang[1], **Yunong Zhang**[1,*] **and Yanli Ji**[1,2]

[1] School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, China

[2] Shenzhen Loop Area Institution, Shenzhen 518107, China

* **Correspondence:** Email: zhynong@mail.sysu.edu.cn.

Abstract: This paper addresses the matrix similarity equation problem by presenting the neurodynamic solution, and applies the solution to solve the output tracking control problem for time-invariant linear systems (TILSs). The feedback gain matrix in the control input is obtained by introducing the matrix similarity-based pole placement equation (MSBPPE) and transforming it into the form of a neurodynamic system. That is, a gradient neurodynamic (GN) model is constructed using an energy function to solve the MSBPPE, enabling the determination of the state feedback gain matrix. Additionally, a zeroing neurodynamic (ZN) model is employed to design the reference input and to ensure that the system output exponentially tracks the desired path. Theoretical analyses show that the proposed neurodynamic models can be asymptotically stable and exhibit exponential convergence. Numerical experiments are conducted under various pole configurations—including distinct real poles, repeated real poles, and complex conjugate poles—verifying the effectiveness and fast convergence of the models. The results show that the proposed models achieve precise output tracking under different dimensional and different types of desired path settings, confirming their theoretical applicability.

Keywords: time-invariant linear system (TILS); output tracking control; pole placement; gradient neurodynamic (GN); zeroing neurodynamic (ZN)
Mathematics Subject Classification: 93D15, 93D20, 93D23

1. Introduction

Tracking control, which aims to force the output of a dynamic system to follow a desired trajectory, constitutes a fundamental and widely encountered problem in control theory and engineering, and its importance spans diverse fields, including multi-agent systems [1, 2], UAVs [3, 4], robotic manipulators [5–7], and autonomous vehicles [8–10]. To solve the output tracking control problem,

there are various methods available such as sliding mode control [11], proportional-integral-derivative control [12], and neurodynamic methods [6,9]. For the time-invariant linear system (TILS), a prevalent and powerful control strategy is the state feedback control law [13]. This strategy involves designing a controller that stabilizes the closed-loop system and assigns its dynamic performance by placing the poles to desired locations, a process known as pole placement.

While the theory of pole placement is well established for control systems [14], traditional methods often have some limitations in solving the TILS in the form of $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t)$. For example, Ackermann's formula $\mathbf{k} = [0, \ldots, 0, 1]M_c^{-1}\Phi(S)$ is a classic pole placement method in modern control theory [14], where M_c denotes the controllability discriminant matrix $[B, AB, \ldots, A^{n-1}B]$ and $\Phi(S)$ denotes the characteristic equation for the closed-loop poles with $S = A$. It does not require complicated matrix decomposition and guarantees the existence of a unique feedback gain matrix as long as the system is completely controllable [14]. However, Ackermann's formula is not applicable to the multiple-input systems and not numerically reliable. It starts to break down rapidly for problems of dimension greater than 10 or for weakly controllable systems [15]. Currently, the most widely used pole placement method is to find a gain matrix F to maximize the robustness of the closed-loop system matrix $A + BF$ [16]. It performs QR decomposition on matrix $B = [U_0, U_1][Z^{\mathrm{T}}, \mathbf{0}]^{\mathrm{T}}$, where $[U_0, U_1]$ is an orthogonal matrix and Z is an upper triangular matrix. Then it uses singular value decomposition (SVD) decomposition on matrix $U_1^{\mathrm{T}}(A-\lambda I)$ (where λ denotes the desired poles) to find the orthonormal basis of its null space, which minimizes the condition number. This basis is chosen as the eigenvector matrix of $A + BF$, allowing the gain matrix F to be computed [16]. However, in the real world, when the multiplicity of the poles to be placed is greater than the rank of matrix B, this method often fails to find the matrix F that satisfies the optimal robustness (such as the routine "place" in MATLAB) [15]. To overcome this dilemma, the optimal control method, namely the linear quadratic regulator (LQR), has been developed to solve the pole placement problem [15]. It computes and finds the state feedback control law that minimizes the cost function by solving the Riccati equation [17]. However, the LQR method also has limitations. In the method, the input weight matrix R must be a positive definite matrix. The state weight matrix Q, input weight matrix R, and cross-term matrix N must satisfy that Q and $Q - NR^{-1}N^{\mathrm{T}}$ are positive semi-definite matrices. Besides, $(Q - NR^{-1}N^{\mathrm{T}}, A - BR^{-1}N^{\mathrm{T}})$ must have no unobservable modes on the imaginary axis (or the unit circle in discrete time) [15].

To overcome the limitations of the above traditional methods, some neurodynamic methods have been used to solve the pole placement problem. In 2002, Zhang et al. proposed a neurodynamic method to obtain the gain matrix by iteratively solving the Sylvester equation [13]. Since the solution of the Sylvester equation is unique under certain conditions, the method has exponential convergence for the gain matrix under arbitrary initial values and appropriate parameter design. However, the solutions of the Sylvester equation usually have a large dimension, and for the solution of the Sylvester equation to exist and be unique, it is necessary to satisfy that the desired poles cannot be the same as any eigenvalue of A [13]. The dilemmas of these methods result in the fact that the poles often cannot be arbitrarily placed in actual control. To address these challenges, this paper proposes a novel matrix similarity-based pole placement equation (MSBPPE). This equation elegantly transforms the pole placement problem into solving a system of matrix equations coupled with similarity constraints and circumventing the need for matrix inverse. To solve the MSBPPE efficiently, we transform the MSBPPE into a time-variant system and develop a neurodynamic model using a gradient neurodynamic (GN) method [18]. The model first operates by constructing a scalar-

valued energy function, and then updates the time-varying matrix along the gradient direction until convergence. The minimum value of the constructed energy function corresponds to the solution of the MSBPPE.

Furthermore, achieving precise asymptotic tracking requires not only stabilizing the system but also incorporating a suitable reference input to reduce steady-state errors. This necessitates an integrated design that seamlessly combines pole placement with a mechanism for reference compensation. Based on the feedback closed-loop control theory, the zeroing neurodynamics (ZN, also termed Zhang neurodynamics) method was first proposed by Zhang et al. in solving the time-variant Sylvester equation [19], then it was subsequently extended to address various time-variant problems such as the time-variant matrix inverse problem [20, 21], time-variant quadratic programming [22–24], and output tracking problem [25]. In 2018, Li et al. proposed a novel real-time tracking control method based on the forward-looking ZN for serial and parallel robotic manipulators [26]. In the same year, Chen et al. proposed a ZN-based Jacobian matrix adaptive method to solve the trajectory tracking control problem of robotic manipulators with an unknown model [5]. In 2023, Ding et al. first applied the ZN method to the liquid level tracking control of a nonlinear double-holding water tank system and systematically analyzed its anti-disturbance performance [27]. In 2024, Katsikis et al. proposed a new method based on ZN to solve the complex-valued time-varying linear matrix equation, and it was applied to robot motion tracking and angle-of-arrival localization [28]. In 2025, Chen et al. proposed a novel model-free ZN scheme without pseudoinverse computation to solve the key problems in the path tracking control of robotic manipulators [6].

For the tracking item in control input, in this paper, a ZN model is designed to generate the appropriate reference input in real time. By utilizing the time derivative information and error feedback information, the ZN model aims to ensure that the output tracking error converges to zero exponentially, with the convergence rate depending directly on the design parameter in the ZN design formula. Although GN and ZN are established techniques, the control problem considered here involves two distinct tasks: determining a feedback gain K that assigns the desired closed-loop poles, and generating a time-varying reference input $\boldsymbol{\mu}(t)$ for output tracking. The proposed framework assigns these tasks to GN and ZN, respectively. Specifically, GN solves the matrix equations associated with pole placement, while ZN uses the output-error dynamics to update the reference input as the desired trajectory changes. Their combination therefore provides a coordinated solution to the pole-assignment and tracking requirements; the contribution is the problem-specific formulation and integration. In this paper, we use GN and ZN models for the two parts in the control input, thereby presenting the neurodynamic solution for the tracking control problem in neurodynamic-solvable TILS. For different cases of the poles, we conduct simulation experiments to verify the effectiveness of the GN model, and also carry out tracking control experiments for different tracking trajectories to illustrate the validity of the ZN model.

The main contributions of this work are listed as follows:

- The proposal of the MSBPPE transforms the pole placement problem into a new matrix equation problem. Based on the GN method, a novel neurodynamic solution is provided for solving the MSBPPE.
- Based on the ZN method as an attempt, the reference input is provided concisely and efficiently. In addition, the theoretical analyses show the convergence and stability of the control input.
- Extensive numerical simulations under various situations of poles (distinct real, repeated real, and

complex conjugate) confirm the model's relative feasibility and effectiveness.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation. Section 3 details the derivation of the MSBPPE. The neurodynamic solutions for the MSBPPE and the reference input are developed in Sections 4 and 5, respectively. Section 6 provides the theoretical convergence analyses. Simulation results and comparative studies are discussed in Section 7, followed by the conclusion in Section 8.

2. Preliminary and problem formulation

In this section, we give the mathematical description of the TILS output tracking problem.

Consider the following TILS output tracking problem:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \tag{2.1a}$$

$$\mathbf{y}(t) = C\mathbf{x}(t), \tag{2.1b}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector, $\mathbf{u}(t) \in \mathbb{R}^m$ is the control input vector, and $\mathbf{y}(t) \in \mathbb{R}^l$ is the actual output vector. Besides, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and $C \in \mathbb{R}^{l\times n}$ are coefficient matrices [14].

The task of solving the output tracking problem is to design a control input $\mathbf{u}(t)$ in the form of

$$\mathbf{u}(t) = K\mathbf{x}(t) + \boldsymbol{\mu}(t), \tag{2.2}$$

where $K \in \mathbb{R}^{m\times n}$ is a state feedback gain matrix and $\boldsymbol{\mu}(t) \in \mathbb{R}^m$ is a reference input vector [13], such that the actual output $\mathbf{y}(t)$ tracks the desired trajectory $\mathbf{y}_\mathrm{d}(t)$. In other words, while the output tracking error of the system (2.1) is given as

$$\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{y}_\mathrm{d}(t), \tag{2.3}$$

the aim of the control input is to ensure the output tracking error is as small as possible or even asymptotically (preferably exponentially) convergent to zero.

Note that, in conventional state-feedback design, pole placement determines K, while the reference input is selected separately according to the tracking objective. For a constant reference $\mathbf{y}_\mathrm{d}(t) = \mathbf{r}$, a standard approach is to find steady-state values $\mathbf{x}_\mathrm{s}$ and $\mathbf{u}_\mathrm{s}$ satisfying

$$A\mathbf{x}_\mathrm{s} + B\mathbf{u}_\mathrm{s} = \mathbf{0}, \qquad C\mathbf{x}_\mathrm{s} = \mathbf{r},$$

and set $\boldsymbol{\mu} = \mathbf{u}_\mathrm{s} - K\mathbf{x}_\mathrm{s}$. If $A + BK$ is Hurwitz, then $\mathbf{x}(t) - \mathbf{x}_\mathrm{s}$ converges to zero. When these steady-state values are expressed as linear functions of $\mathbf{r}$, this construction gives a constant reference precompensator [29]. Integral augmentation provides another standard means of eliminating constant tracking offsets, subject to appropriate stabilization of the augmented system [30].

For time-varying references, a constant precompensator generally cannot compensate for the reference dynamics. One conventional approach uses a feasible state-input trajectory satisfying $\dot{\mathbf{x}}_\mathrm{r} = A\mathbf{x}_\mathrm{r} + B\mathbf{u}_\mathrm{r}$ and $C\mathbf{x}_\mathrm{r} = \mathbf{y}_\mathrm{d}$, with $\boldsymbol{\mu}(t) = \mathbf{u}_\mathrm{r}(t) - K\mathbf{x}_\mathrm{r}(t)$ [31]. Alternatively, output-regulation methods determine the required compensation through regulator equations associated with a reference-generating exosystem [32]; internal-model-based controllers are used when robust regulation is required [33]. In this paper, the ZN formulation aims to construct $\boldsymbol{\mu}(t)$ directly from the desired trajectory and its derivative by prescribing the output-error dynamics, as detailed in Section 5.

Since the system output is able to track the desired trajectory, the rank of matrix C should be equal to the rank of augmented matrix $[C, \mathbf{y}_{\mathrm{d}}(t)]$. If not, the equation $\mathbf{y}(t) - \mathbf{y}_{\mathrm{d}}(t) = C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t) = \mathbf{0}$ has no solution, i.e., there is no state that makes the output tracking error equal to zero. Without loss of generality, in this paper, we assume that each component of the desired trajectory is linearly independent, i.e., matrix C is a full row rank matrix. Similarly, the rank of the matrix B should be no less than the rank of matrix C. If not, the rank of control input term $B\mathbf{u}(t)$ is less than the rank of matrix C, which means that there are less input elements than output, i.e., the system is uncontrollable. Therefore, in this paper, we assume that the rank of matrix B is full.

3. Matrix similarity-based pole placement equation

In this section, we give the mathematical descriptions of the MSBPPE.

If the linear state feedback control law (2.2) is applied to (2.1a), the closed-loop system is in the following form:

$$\dot{\mathbf{x}}(t) = (A + BK)\mathbf{x}(t) + B\boldsymbol{\mu}(t). \tag{3.1}$$

According to linear system theory [14], there exists a matrix K such that $A + BK$ has any specified eigenvalue if system (2.1) is controllable. Next, we perform a similarity transformation on the matrix $A + BK$.

Case 1 – Diagonal form: With all desired poles real and distinct, the matrix $A + BK$ is similar to the matrix Λ, where $\Lambda \in \mathbb{R}^{n\times n}$ is a diagonal matrix and each desired pole corresponds to the diagonal element of Λ [34]. Moreover, there exists an invertible matrix $P \in \mathbb{R}^{n\times n}$ such that

$$A + BK = P\Lambda P^{-1},$$

where P consists of the eigenvectors corresponding to each eigenvalue of matrix $A + BK$.

Case 2 – Jordan form: When the desired poles are real but repeated, the matrix $A + BK$ is similar to a Jordan matrix Λ, which is in the form of

$$\Lambda = \begin{bmatrix} J_{n_1} & & & \\ & J_{n_2} & & \\ & & \ddots & \\ & & & J_{n_k} \end{bmatrix}_{n\times n},$$

where $n_1 + n_2 + \cdots + n_k = n$. That is, there exists a invertible matrix $P \in \mathbb{R}^{n\times n}$ such that

$$A + BK = P\Lambda P^{-1}.$$

The diagonal elements of the Jordan matrix Λ is in the form of

$$J_{n_k}(\lambda) = \begin{bmatrix} \lambda & 1 & & & \\ & \lambda & 1 & & \\ & & \ddots & \ddots & \\ & & & \ddots & 1 \\ & & & & \lambda \end{bmatrix}_{n_k\times n_k}.$$

According to linear system theory [14], when the system (2.1) is completely controllable, we assume that the matrix $A + BK$ is a cyclic matrix. Then, the geometric multiplicity of each eigenvalue of $A + BK$ equals to 1, which means that each distinct eigenvalue corresponds to single Jordan block, respectively.

Case 3 – Complex conjugate form: When some of the desired poles are complex conjugate pairs, the matrix $A + BK$ is similar to a real Jordan matrix Λ which is in the form of

$$\Lambda = \begin{bmatrix} J_{n_1} & & & & & & \\ & \ddots & & & & & \\ & & J_{n_p} & & & & \\ & & & C_{m_1} & & & \\ & & & & \ddots & & \\ & & & & & C_{m_r} \end{bmatrix}_{n\times n},$$

where $n_1 + \cdots + n_p + m_1 + \cdots + m_r = n$ [35]. The matrix $C_{m_k}(a_k, b_k)$ in diagonal elements of the Jordan matrix Λ is expressed as

$$\begin{bmatrix} C(a_k,b_k) & I_{2\times 2} & & & & \\ & C(a_k,b_k) & I_{2\times 2} & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & I_{2\times 2} & \\ & & & & C(a_k,b_k) \end{bmatrix}_{m_k\times m_k},$$

which corresponds to the complex conjugate poles $\lambda_k = a_k \pm b_k\mathrm{i}$ with multiplicity m_k, where i is the imaginary unit, a_k and b_k are real numbers, and $b_k > 0$. The matrix $C(a_k, b_k)$ is expressed as

$$\begin{bmatrix} a_k & b_k \\ -b_k & a_k \end{bmatrix}.$$

That is, there exists a invertible matrix $P \in \mathbb{R}^{n\times n}$ such that

$$A + BK = P\Lambda P^{-1}.$$

To avoid the inverse calculation of matrices, we replace the P^{-1} with a square matrix $Q \in \mathbb{R}^{n\times n}$, and then we obtain the MSBPPE:

$$\begin{cases} A + BK = P\Lambda Q, \\ PQ = I_{n\times n}, \\ QP = I_{n\times n}. \end{cases} \tag{3.2}$$

Introducing Q as an independent matrix and imposing $PQ = QP = I$ avoids explicitly inverting the evolving matrix P during the GN computation. This reformulation provides the matrix residuals, from which the energy function and the corresponding GN dynamics can be constructed for the pole-assignment stage.

4. Neurodynamic solution for MSBPPE

In this section, to find a solution of indeterminate system of Eq (3.2), we transform the MSBPPE into a time-variant indeterminate system of equations and introduce a neurodynamic method.

To construct a neurodynamic solution for MSBPPE (3.2), we treat the matrices K, P, and Q as time-variant matrices $K(t)$, $P(t)$, and $Q(t)$. Then, we construct an energy function as below:

$$E(t) = \frac{1}{2}\left(\|A + BK(t) - P(t)\Lambda Q(t)\|_{\mathrm{F}}^2 + \|P(t)Q(t) - I_{n\times n}\|_{\mathrm{F}}^2 + \|Q(t)P(t) - I_{n\times n}\|_{\mathrm{F}}^2\right). \tag{4.1}$$

Here, $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm [13]. Although $PQ = I$ and $QP = I$ are equivalent for square matrices at an exact solution, their residuals can have different magnitudes during numerical iterations. Both terms are retained, as an advantage, to penalize deviations from left- and right-inverse consistency simultaneously, without changing the exact solution set. By applying the GN formula [18] to the energy function (4.1), we obtain the following GN model:

$$\begin{cases} \dot{K}(t) = -\gamma B^{\mathrm{T}}\left(A + BK(t) - P(t)\Lambda Q(t)\right), \\ \dot{P}(t) = -\gamma\Big(-\left(A + BK(t) - P(t)\Lambda Q(t)\right)\left(\Lambda Q(t)\right)^{\mathrm{T}} \\ \qquad + \left(P(t)Q(t) - I_{n\times n}\right)Q^{\mathrm{T}}(t) + Q^{\mathrm{T}}(t)\left(Q(t)P(t) - I_{n\times n}\right)\Big), \\ \dot{Q}(t) = -\gamma\Big(-\left(P(t)\Lambda\right)^{\mathrm{T}}\left(A + BK(t) - P(t)\Lambda Q(t)\right) \\ \qquad + P^{\mathrm{T}}(t)\left(P(t)Q(t) - I_{n\times n}\right) + \left(Q(t)P(t) - I_{n\times n}\right)P^{\mathrm{T}}(t)\Big), \end{cases} \tag{4.2}$$

where $\gamma \in \mathbb{R}^+$ is the neurodynamic design parameter and $\dot{K}(t)$, $\dot{P}(t)$, and $\dot{Q}(t)$ are the derivatives of $K(t)$, $P(t)$, and $Q(t)$, respectively. By using the Euler forward differential formula, which provides a first-order numerical approximation, we convert the continuous model (4.2) into a discrete algorithm:

$$\begin{cases} K_{n+1} = K_n - \sigma\gamma B^{\mathrm{T}}\left(A + BK_n - P_n\Lambda Q_n\right), \\ P_{n+1} = P_n - \sigma\gamma\Big(-\left(A + BK_n - P_n\Lambda Q_n\right)\left(\Lambda Q_n\right)^{\mathrm{T}} \\ \qquad + \left(P_nQ_n - I_{n\times n}\right)Q_n^{\mathrm{T}} + Q_n^{\mathrm{T}}\left(Q_nP_n - I_{n\times n}\right)\Big), \\ Q_{n+1} = Q_n - \sigma\gamma\Big(-\left(P_n\Lambda\right)^{\mathrm{T}}\left(A + BK_n - P_n\Lambda Q_n\right) \\ \qquad + P_n^{\mathrm{T}}\left(P_nQ_n - I_{n\times n}\right) + \left(Q_nP_n - I_{n\times n}\right)P_n^{\mathrm{T}}\Big), \end{cases} \tag{4.3}$$

where σ is the sampling period, K_n, P_n, and Q_n represent the solution state at time instant $t = n\sigma$, $n \in \{0, \lfloor t_{\mathrm{f}}/\sigma \rfloor\}$, and t_{f} is the final time instant.

5. Neurodynamic solution for reference input

In this section, we introduce a neurodynamic solution as an attempt for the reference input $\boldsymbol{\mu}(t)$ in the linear state feedback control law (2.2).

By applying the ZN design formula $\dot{\mathbf{z}}(t) = -\lambda\mathbf{z}(t)$ [13] to the output tracking error (2.3), we obtain

$$\dot{\mathbf{e}}(t) = \dot{\mathbf{y}}(t) - \dot{\mathbf{y}}_{\mathrm{d}}(t) = -\lambda\left(\mathbf{y}(t) - \mathbf{y}_{\mathrm{d}}(t)\right), \tag{5.1}$$

where $\lambda \in \mathbb{R}^+$ is the neurodynamic design parameter. By solving Eqs (2.1) and (5.1) simultaneously, we obtain

$$CA\mathbf{x}(t) + CB\mathbf{u}(t) - \dot{\mathbf{y}}_{\mathrm{d}}(t) = -\lambda\left(C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t)\right). \tag{5.2}$$

By substituting Eq (2.2) into Eq (5.2), we obtain the ZN model for the reference input $\mu(t)$:

$$\boldsymbol{\mu}(t) = (CB)^{\dagger}[\dot{\mathbf{y}}_{\mathrm{d}}(t) - C(A + BK)\mathbf{x}(t) - \lambda(C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t))], \tag{5.3}$$

where $(CB)^{\dagger}$ denotes the pseudoinverse of the matrix CB and the gain feedback matrix K is computed as in Section 4. By using the Euler forward differential formula, we convert the continuous model into a discrete algorithm:

$$\begin{cases} \boldsymbol{\mu}_n = (CB)^{\dagger}\left[\dot{\mathbf{y}}_{\mathrm{d}}(n\sigma) - C\left(A + BK\right)\mathbf{x}_n - \lambda\left(C\mathbf{x}_n - \mathbf{y}_{\mathrm{d}}(n\sigma)\right)\right], \\ \mathbf{u}_n = K\mathbf{x}_n + \mu_n, \\ \mathbf{x}_{n+1} = \mathbf{x}_n + \sigma\left(A\mathbf{x}_n + B\mathbf{u}_n\right), \\ \mathbf{y}_{n+1} = C\mathbf{x}_{n+1}. \end{cases} \tag{5.4}$$

The structure diagram of the neurodynamic solution is shown in Figure 1, and the key steps of the neurodynamic algorithm are summarized in Algorithm 1.

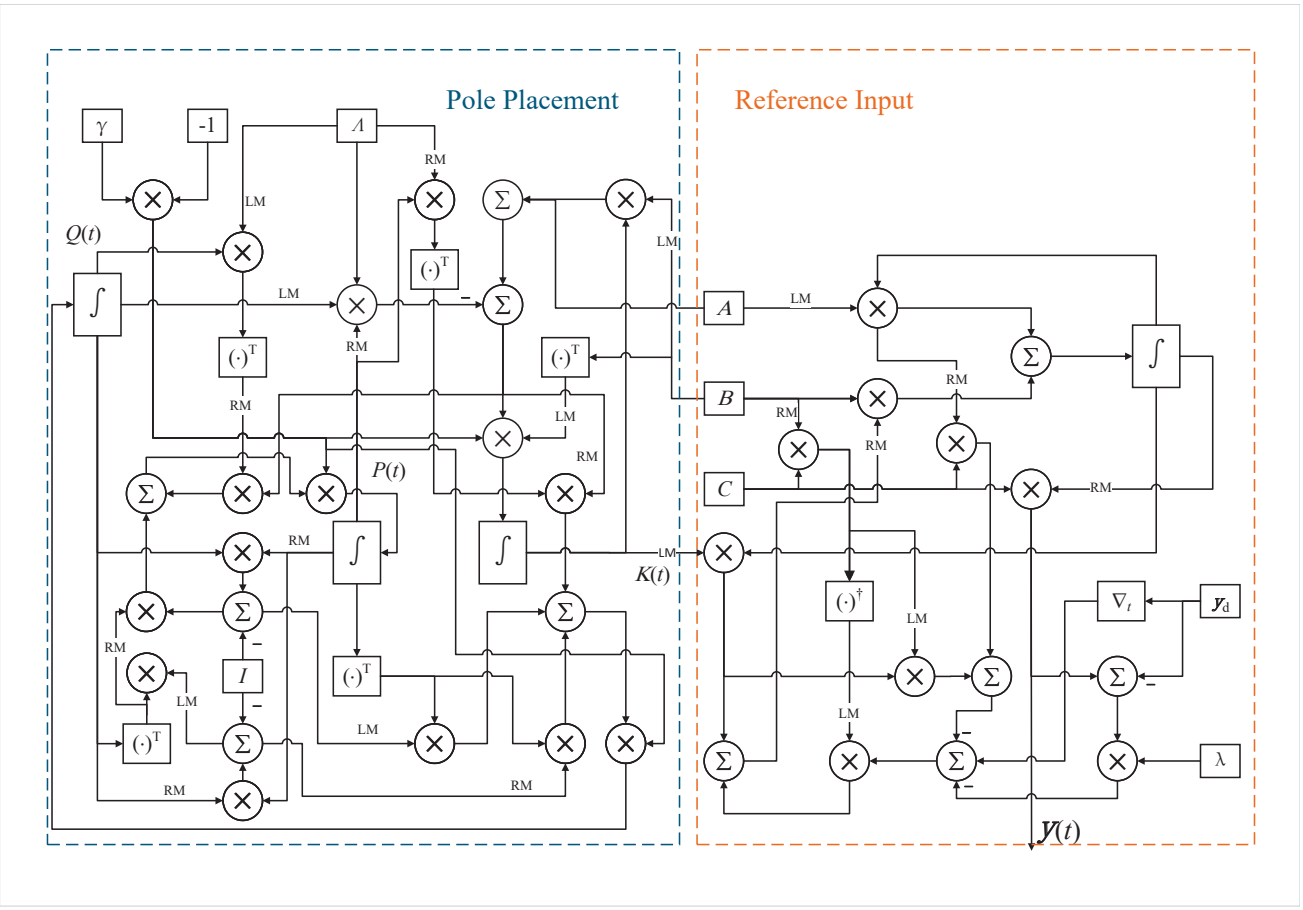


Figure 1. Structure diagram of neurodynamic solution for output tracking problem solving in TILS, where LM denotes left multiplication and RM denotes right multiplication.

Algorithm 1: Neurodynamic solution for TILS (2.1) output tracking problem solving.

Initialize: Gain feedback matrix $K_1 \in \mathbb{R}^{n\times n}$, nonsingular matrices P_1 and $Q_1 \in \mathbb{R}^{n\times n}$, state vector $\mathbf{x}_1 \in \mathbb{R}^n$;

Input: Constant matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{l\times n}$, desired pole $\Lambda \in \mathbb{R}^{n\times n}$, design parameters γ, $\lambda \in \mathbb{R}^+$, task durations T_1 and T_2, sampling gaps σ_1 and σ_2, desired trajectory $\mathbf{y}_{\mathrm{d}}(t) \in \mathbb{R}^l$

$i \leftarrow 1$;
while $i = 1 : \lfloor T_1/\sigma_1 \rfloor$ **do**
  $\mathbf{e}_{1i} \leftarrow A + BK_i - P_i\Lambda Q_i$;
  $\mathbf{e}_{2i} \leftarrow P_iQ_i - I$;
  $\mathbf{e}_{3i} \leftarrow Q_iP_i - I$;
  $\dot{K}_i \leftarrow -\gamma B^{\mathrm{T}}\mathbf{e}_{1i}$;
  $\dot{P}_i \leftarrow -\gamma(-\mathbf{e}_{1i}Q_i^{\mathrm{T}}\Lambda^{\mathrm{T}} + \mathbf{e}_{2i}Q_i^{\mathrm{T}} + Q_i^{\mathrm{T}}\mathbf{e}_{3i})$;
  $\dot{Q}_i \leftarrow -\gamma(-\Lambda^{\mathrm{T}}P_i^{\mathrm{T}}\mathbf{e}_{1i} + P_i^{\mathrm{T}}\mathbf{e}_{2i} + \mathbf{e}_{3i}P_i^{\mathrm{T}})$;
  $K_{i+1} \leftarrow K_i - \sigma_1\dot{K}_i$;
  $P_{i+1} \leftarrow P_i - \sigma_1\dot{P}_i$;
  $Q_{i+1} \leftarrow Q_i - \sigma_1\dot{Q}_i$;
end
$K \leftarrow K_{\lfloor T_1/\sigma_1 \rfloor}$;
$i \leftarrow 1$;
while $i = 1 : \lfloor T_2/\sigma_2 \rfloor$ **do**
  $\mu_i \leftarrow (CB)^{\dagger}[\dot{\mathbf{y}}_{\mathrm{d}}(i\sigma_2) - C(A + BK)\mathbf{x}_i - \lambda(C\mathbf{x}_i - \mathbf{y}_{\mathrm{d}}(i\sigma_2))]$;
  $\mathbf{u}_i \leftarrow K\mathbf{x}_i + \mu_i$;
  $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \sigma_2(A\mathbf{x}_i + B\mathbf{u}_i)$;
  $\mathbf{y}_{i+1} \leftarrow C\mathbf{x}_{i+1}$;
end

Remark 5.1. *The supporting roles of GN and ZN are determined by the temporal characteristics of the two tasks. The MSBPPE has constant problem data and is solved offline; GN provides an energy-based formulation that combines the matrix residuals in (4.1). The time dependence of $K(t)$, $P(t)$, and $Q(t)$ describes the solver evolution, rather than a moving target solution. By contrast, the output tracking requires the controller to follow a time-varying desired trajectory. ZN uses $\dot{\mathbf{y}}_{\mathrm{d}}(t)$ to account explicitly for this variation and, when rank$(CB) = l$, yields $\dot{\mathbf{e}}(t) = -\lambda\mathbf{e}(t)$. Consequently, $\mathbf{e}(t) = \mathbf{e}(0)\exp(-\lambda t)$ in the ideal continuous-time model. This assignment does not imply that GN is restricted to matrix equations or that ZN is restricted to vector equations. A ZN alternative for the MSBPPE would require solving the differentiated residual equations, with their solvability and possible constraint redundancy taken into account. Conversely, GN can be considered for tracking, but a conventional gradient formulation without compensation for target variation generally introduces tracking lag. Additional design and analysis would therefore be required to investigate the tracking guarantee obtained here. The present combination provides a convenient division between gain computation and trajectory tracking; it does not establish universal superiority over alternative assignments.*

6. Theoretical analyses

In this section, we analyze the convergence properties of the GN (4.2) and ZN (5.3) models.

Proposition 6.1. *For the MSBPPE (3.2), if matrices $P(t)$ and $Q(t)$ satisfy $det(P(t))det(Q(t)) > 0$, the GN model (4.2) is asymptotically stable, and time-variant matrices $K(t)$, $P(t)$, and $Q(t)$ converge to the solution of the MSBPPE (3.2).*

Proof. It is evident that the energy function $E(t)$ is positive define. Then, the derivative of the energy function is derived as follows:

$$\frac{\mathrm{d}E(t)}{\mathrm{d}t} = \mathrm{tr}\left\{\left(\frac{\partial E(t)}{\partial K(t)}\right)^{\mathrm{T}}\frac{\mathrm{d}K(t)}{\mathrm{d}t} + \left(\frac{\partial E(t)}{\partial P(t)}\right)^{\mathrm{T}}\frac{\mathrm{d}P(t)}{\mathrm{d}t} + \left(\frac{\partial E(t)}{\partial Q(t)}\right)^{\mathrm{T}}\frac{\mathrm{d}Q(t)}{\mathrm{d}t}\right\}. \tag{6.1}$$

From the GN model (4.2), we obtain that

$$\dot{K}(t) = -\gamma\frac{\partial E(t)}{\partial K(t)}, \dot{P}(t) = -\gamma\frac{\partial E(t)}{\partial P(t)}, \dot{Q}(t) = -\gamma\frac{\partial E(t)}{\partial Q(t)}. \tag{6.2}$$

Combine Eqs (6.1) and (6.2), and we have

$$\begin{aligned}\frac{\mathrm{d}E(t)}{\mathrm{d}t} =& \mathrm{tr}\left\{\left(\frac{\partial E(t)}{\partial K(t)}\right)^{\mathrm{T}}\left(-\gamma\frac{\partial E(t)}{\partial K(t)}\right)\right.\\ &+\left(\frac{\partial E(t)}{\partial P(t)}\right)^{\mathrm{T}}\left(-\gamma\frac{\partial E(t)}{\partial P(t)}\right)\\ &\left.+\left(\frac{\partial E(t)}{\partial Q(t)}\right)^{\mathrm{T}}\left(-\gamma\frac{\partial E(t)}{\partial Q(t)}\right)\right\}\\ =& -\gamma\left(\left\|\frac{\partial E(t)}{\partial K(t)}\right\|_{\mathrm{F}}^2 + \left\|\frac{\partial E(t)}{\partial P(t)}\right\|_{\mathrm{F}}^2 + \left\|\frac{\partial E(t)}{\partial Q(t)}\right\|_{\mathrm{F}}^2\right) \le 0.\end{aligned}$$

By Lyapunov stability theory [36, 37], the equilibrium point (6.3) is asymptotically stable. When the determinant of $P(t)$ multiplied by the determinant of $Q(t)$ is greater than 0, the equilibrium point equals to $E(t) = 0$, thereby satisfying

$$\begin{cases} A + BK(t) - P(t)\Lambda Q(t) = \mathbf{0}, \\ P(t)Q(t) - I = \mathbf{0}, \\ Q(t)P(t) - I = \mathbf{0}, \end{cases} \tag{6.3}$$

which means that the equilibrium state of time-variant matrices $K(t)$, $P(t)$, and $Q(t)$ is a solution of the MSBPPE (3.2). □

Proposition 6.2. *For the MSBPPE (3.2), the theoretical solutions are matrices K^*, P^*, and Q^*. Let $\hat{K}(t) = K(t) - K^*$, $\delta > 0$ as the minimal eigenvalue of B^TB, and $\epsilon(t) = P^*\Lambda Q^* - P(t)\Lambda Q(t)$. Starting from an initial value $K(0)$, when there exist constants $\zeta > 0$ and $\omega > 0$ such that the following inequalities hold:*

$$\|\epsilon(t)\|_F \le \zeta, \ \|B\|_F \le \omega, \tag{6.4}$$

then the steady-state error bound for solving the gain matrix $K(t)$ is obtained as

$$\lim_{t\to\infty}\|\hat{K}(t)\|_F \le \frac{\zeta\omega}{\delta}. \tag{6.5}$$

Proof. We define a Lyapunov candidate function $W(t) = \|\hat{K}(t)\|_{\mathrm{F}}^2/2$. It is apparent that function $W(t)$ is positive definite. The derivative of $W(t)$ is

$$\begin{aligned}\dot{W}(t) =& \mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)\dot{\hat{K}}(t)\right)\\ =& \mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)\dot{K}(t)\right)\\ =& \mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)\left(-\gamma B^{\mathrm{T}}\left(A + BK(t) - P(t)\Lambda Q(t)\right)\right)\right)\\ =& -\gamma\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}\left(B\hat{K}(t) + P^*\Lambda Q^* - P(t)\Lambda Q(t)\right)\right)\\ =& -\gamma\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}B\hat{K}(t)\right) - \gamma\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}\epsilon(t)\right).\end{aligned} \tag{6.6}$$

Before analyzing the time derivative of $W(t)$, we have to handle the two terms in the last line of (6.6) individually. For the first term, we have

$$\begin{aligned}-\gamma\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}B\hat{K}(t)\right) &\le -\gamma\delta\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)\hat{K}(t)\right)\\ &= -\gamma\delta\left\|\hat{K}(t)\right\|_{\mathrm{F}}^2.\end{aligned} \tag{6.7}$$

For the second term, we have

$$\begin{aligned}-\gamma\mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}\epsilon(t)\right) &\le \gamma\left\|-\hat{K}^{\mathrm{T}}(t)\right\|_{\mathrm{F}}\left\|B^{\mathrm{T}}\right\|_{\mathrm{F}}\|\epsilon(t)\|_{\mathrm{F}}\\ &\le \gamma\zeta\omega\left\|\hat{K}(t)\right\|_{\mathrm{F}}.\end{aligned} \tag{6.8}$$

Then, substituting (6.7) and (6.8) into (6.6) yields

$$\begin{aligned}\dot{W}(t) &\le -\gamma\delta\left\|\hat{K}(t)\right\|_{\mathrm{F}}^2 + \gamma\zeta\omega\left\|\hat{K}(t)\right\|_{\mathrm{F}}\\ &= -\gamma\left\|\hat{K}(t)\right\|_{\mathrm{F}}\left(\delta\left\|\hat{K}(t)\right\|_{\mathrm{F}} - \zeta\omega\right).\end{aligned} \tag{6.9}$$

As time increases, the inequality (6.9) falls into one of the following three situations: i) $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega > 0$, ii) $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega = 0$, and iii) $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega < 0$.

(i) When $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega > 0$ holds, inequality (6.9) satisfies $\dot{W}(t) \le -\|\hat{K}(t)\|_{\mathrm{F}}(\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega) < 0$. That is, $\|\hat{K}(t)\|_{\mathrm{F}}$ is decreasing toward $\zeta\omega/\delta$. If $\|\hat{K}(t)\|_{\mathrm{F}} = \zeta\omega/\delta$, it turns into situation (ii)).

(ii) When $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega = 0$ holds, we have $\dot{W}(t) \le 0$. If $\dot{W}(t) = 0$, it implies that $W(t)$ is stable at $\|\hat{K}(t)\|_{\mathrm{F}} = \zeta\omega/\delta$, i.e., $\|\hat{K}(t)\|_{\mathrm{F}}$ remains on the ball of radius $\zeta\omega/\delta$. If $\dot{W}(t) < 0$, $\|\hat{K}(t)\|_{\mathrm{F}}$ is decreasing, and it turns to the situation (i)).

(iii) When $\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega < 0$ holds, $-\|\hat{K}(t)\|_{\mathrm{F}}(\delta\|\hat{K}(t)\|_{\mathrm{F}} - \zeta\omega) > 0$. If $\dot{W}(t) \le 0$, $\|\hat{K}(t)\|_{\mathrm{F}}$ descends toward zero as time increases. If $\dot{W}(t) > 0$, $W(t)$ is an increasing function, which means that $\|\hat{K}(t)\|_{\mathrm{F}}$ is increasing toward $\zeta\omega/\delta$.

In summary of the above analysis, the steady state error $\lim_{t\to\infty}\|\hat{K}(t)\|_{\mathrm{F}}$ is upper bounded by $\zeta\omega/\delta$. □

Proposition 6.3. *In addition to the conclusion in Proposition 6.2, when there exists a constant $\alpha \in [0, \delta/\omega)$ such that the following inequality holds:*

$$\|\epsilon(t)\|_F \le \alpha\left\|\hat{K}(t)\right\|_F, \tag{6.10}$$

the gain matrix $K(t)$ converges to the theoretical solution K^* exponentially with convergence speed being at least $\gamma(\delta - \omega\alpha)$.

Proof. If the inequality (6.10) holds, (6.8) is written as

$$\begin{aligned} -\gamma \mathrm{tr}\left(\hat{K}^{\mathrm{T}}(t)B^{\mathrm{T}}\boldsymbol{\epsilon}(t)\right) &\leq \gamma \left\|-\hat{K}(t)\right\|_{\mathrm{F}} \|B\|_{\mathrm{F}} \|\boldsymbol{\epsilon}(t)\|_{\mathrm{F}} \\ &\leq \gamma\omega \left\|\hat{K}(t)\right\|_{\mathrm{F}} \alpha \left\|\hat{K}(t)\right\|_{\mathrm{F}} \\ &= \gamma\omega\alpha \left\|\hat{K}(t)\right\|_{\mathrm{F}}^2. \end{aligned} \tag{6.11}$$

Then, substituting (6.7) and (6.11) into (6.6) yields the following:

$$\begin{aligned} \dot{W}(t) &\leq -\gamma\delta \left\|\hat{K}(t)\right\|_{\mathrm{F}}^2 + \gamma\omega\alpha \left\|\hat{K}(t)\right\|_{\mathrm{F}}^2 \\ &= -\gamma\left(\delta - \omega\alpha\right) \left\|\hat{K}(t)\right\|_{\mathrm{F}}^2. \end{aligned} \tag{6.12}$$

Since $\gamma > 0$ and $\alpha < \delta/\omega$, we have $\dot{W}(t) \leq 0$. According to Lyapunov theory [36, 37], the matrix $\hat{K}(t)$ is asymptotically convergent to zero. By the inequality (6.12), $W(t)$ and its derivative also satisfy

$$\dot{W}(t) \leq -\gamma\left(\delta - \omega\alpha\right) \left\|\hat{K}(t)\right\|_{\mathrm{F}}^2 = -2\gamma\left(\delta - \omega\alpha\right) W(t). \tag{6.13}$$

Since the inequality (6.13) holds for all $t \in [0, \infty)$, we obtain

$$W(t) \leq W(0)\exp\left(-2\gamma\left(\delta - \omega\alpha\right) t\right). \tag{6.14}$$

Therefore, the matrix $\hat{K}(t)$ satisfies

$$\left\|\hat{K}(t)\right\|_{\mathrm{F}} \leq \left\|\hat{K}(0)\right\|_{\mathrm{F}} \exp\left(-\gamma\left(\delta - \omega\alpha\right) t\right), \tag{6.15}$$

which implies that $\|\hat{K}(t)\|_{\mathrm{F}}$ exponentially converges to zero with convergence speed being at least $\gamma(\delta - \omega\alpha)$, i.e., the gain matrix $K(t)$ converges to the theoretical solution K^* exponentially with convergence speed being at least $\gamma(\delta - \omega\alpha)$. □

Remark 6.1. *The above two propositions concern the convergence of the gain matrix K and require B to have full column rank, equivalently $\delta = \lambda_{\min}(B^{\mathbf{T}}B) > 0$. This requirement is stronger than the rank-deficient case and is imposed specifically for the bounds on $\|K - K^*\|_F$. Note that, if B is allowed to be rank-deficient, $\delta = 0$ and these bounds do not apply: The GN update cannot change the component of K in the null space of B. In that case, the convergence of the effective gain K, rather than the convergence to a particular K^*, would require a separate analysis.*

Remark 6.2. *The inequality in Proposition 3 is an additional sufficient condition for the stated exponential-convergence bound, rather than a necessary condition for exponential convergence. It is not implied by the general assumptions of the problem. Its verification depends on a specified reference solution (K^*, P^*, Q^*) and the corresponding trajectory. Since admissible pole-placement solutions need not be unique, a failure of the inequality for one reference solution does not imply the failure for all reference solutions. When a reference solution is available, the ratio $\|\epsilon(t)\|_F / \|K(t) - K^*\|_F$ may be monitored wherever the denominator is nonzero; checking it at discrete sample points provides the numerical evidence over the sampled interval, not a proof for all t.*

Proposition 6.4. *For the neurodynamic-solvable TILS output tracking problem (2.1), starting from an initial state $\mathbf{x}(0) = \mathbf{x}_0 \in \mathbb{R}^n$, the control input $\mathbf{u}(t)$ obtained by state feedback control law (2.2) and ZN model (5.3) makes the output tracking error (2.3) converge to zero exponentially with the convergence speed being λ.*

Proof. We define a Lyapunov candidate function $V(t) = \|\mathbf{e}(t)\|_2^2/2$. It is apparent that function $V(t)$ is positive definite. Then, the derivative of the Lyapunov candidate function is derived as

$$\begin{aligned}\dot{V}(t) =& \mathbf{e}^{\mathrm{T}}(t)\dot{\mathbf{e}}(t) = \mathbf{e}^{\mathrm{T}}(t)\left(C\dot{\mathbf{x}}(t) - \dot{\mathbf{y}}_{\mathrm{d}}(t)\right)\\ =& \mathbf{e}^{\mathrm{T}}(t)\left(C\left(A + BK\right)\mathbf{x}(t) + CB\boldsymbol{\mu}(t) - \dot{\mathbf{y}}_{\mathrm{d}}(t)\right)\\ =& \mathbf{e}^{\mathrm{T}}(t)\Big(C\left(A + BK\right)\mathbf{x}(t) + (CB)\left(CB\right)^{\dagger}\left(\dot{\mathbf{y}}_{\mathrm{d}}(t)\right.\\ & \left. -C\left(A + BK\right)\mathbf{x}(t) - \lambda\left(C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t)\right)\right) - \dot{\mathbf{y}}_{\mathrm{d}}(t)\Big).\end{aligned} \tag{6.16}$$

For matrices B and C of different dimensions, the matrix $CB \in \mathbb{R}^{l\times m}$ falls into one of the following three situations: i) $m > l$, ii) $m = l$, and iii) $m < l$.

i) As known in Section 2, $\mathrm{rank}(B) \geq \mathrm{rank}(C) = l$. Thus, when $m > l$, the rank of matrix CB equals to l, which means that the pseudoinverse matrix $(CB)^{\dagger}$ is a right inverse matrix of CB. Then matrix $(CB)^{\dagger}$ is written as $(CB)^{\mathrm{T}}[CB(CB)^{\mathrm{T}}]^{-1}$, and the matrix $(CB)(CB)^{\dagger}$ is an l-dimensional identity matrix.

ii) When $m = l$, the matrices B and C are square matrices. As known in Section 2, $m = \mathrm{rank}(B) \geq l = m = \mathrm{rank}(C)$. Therefore, the matrix $(CB)^{\dagger}$ is written as $(CB)^{-1}$. It is evident that $(CB)(CB)^{\dagger}$ is an l-dimensional identity matrix.

iii) When $m < l$, the pseudoinverse matrix $(CB)^{\dagger}$ is a left inverse matrix of CB. However, in this situation, the number of element of output vector $\mathbf{y}(t)$ is more than the number of element of control input vector $\mathbf{u}(t)$, which means that the system is uncontrollable. Therefore, this situation contradicts the assumption.

From the above three situations, we find that the matrix $(CB)(CB)^{\dagger}$ is an l-dimensional identity matrix. Then the Lyapunov candidate function (6.16) is derived as

$$\begin{aligned}\dot{V}(t) =& \mathbf{e}^{\mathrm{T}}(t)\left(C\left(A + BK\right)\mathbf{x}(t) + I_{l\times l}\left(\dot{\mathbf{y}}_{\mathrm{d}}(t)\right.\right.\\ & \left.\left. -C\left(A + BK\right)\mathbf{x}(t) - \lambda\left(C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t)\right)\right) - \dot{\mathbf{y}}_{\mathrm{d}}(t)\right)\\ =& \mathbf{e}^{\mathrm{T}}(t)\left(-\lambda\left(C\mathbf{x}(t) - \mathbf{y}_{\mathrm{d}}(t)\right)\right)\\ =& -\lambda\mathbf{e}^{\mathrm{T}}(t)\mathbf{e}(t)\\ =& -\lambda\left\|\mathbf{e}(t)\right\|_2^2.\end{aligned} \tag{6.17}$$

Since $\lambda > 0$, we have $\dot{V}(t) \leq 0$. According to Lyapunov theory [36,37], the output tracking error $\mathbf{e}(t)$ is asymptotically convergent to zero. Moreover, the Lyapunov candidate function $V(t)$ and its derivative satisfy

$$\dot{V}(t) = -\lambda\left\|\mathbf{e}(t)\right\|_2^2 = -2\lambda V(t). \tag{6.18}$$

By solving the ordinary differential equation (6.18), we obtain

$$V(t) = V(0)\exp\left(-2\lambda t\right). \tag{6.19}$$

Therefore, the output tracking error (2.3) satisfies

$$\left\|\mathbf{e}(t)\right\|_2 = \left\|\mathbf{e}(0)\right\|_2\exp\left(-\lambda t\right). \tag{6.20}$$

It is evident that, as the time t increases, $\|\mathbf{e}(t)\|_2$ exponentially converges to zero with convergence speed being λ, i.e., the output tracking error (2.3) converges to zero exponentially with the convergence speed being λ. □

7. Simulation experiments and results

In this section, we conduct numerical experiments to show the validity of the neurodynamic models proposed in Sections 4 and 5 and to verify the correctness of theoretical results discussed in Section 6. We design three numerical experiments to verify the effectiveness of the GN model (4.2). Besides, two numerical experiments show the efficiency of the ZN model (5.3). Two comparative experiments are designed for neurodynamic parameters in models. The numerical experiments illustrate the behavior of the proposed models under different pole configurations and tracking tasks. In particular, the GN experiments report the evolution of the energy function and the closed-loop poles; they do not by themselves verify the additional sufficient condition in Proposition 3. We set the initial value P_1 and Q_1 as identity matrices $I_{n\times n}$, and set each element of $\mathbf{x}_1$ and K_1 as a random value between $[-1, 1]$. The numerical results are rounded to two decimal digits. We set $P_1 = Q_1 = I$ because these matrices are nonsingular and satisfy both inverse-consistency equations at initialization. The entries of K_1 and $\mathbf{x}_1$ were chosen in $[-1, 1]$ as test initial states; no optimality or convergence claim is based on this choice. To enable exact reproduction, the randomly selected initial values are to be given below. All differential equations were discretized by the fixed-step forward Euler method, so adaptive-solver integration tolerances are not applicable. The sampling step corresponds to σ_1 and σ_2 in the equations. The simulations were run using Matlab R2024a platform.

The gains and integration steps should be selected jointly because both algorithms use forward Euler discretization. For the GN solver, the settings are $(\gamma, \sigma) = (10^3, 10^{-5}\text{ s})$ in Example 1 and $(10^4, 10^{-6}\text{ s})$ in Examples 2 and 3. These settings retain the same effective update factor $\sigma\gamma = 10^{-2}$, with the larger gain accompanied by a proportionally smaller integration step. For the tracking experiments in Example 4, we use $\lambda = 10^3$ and $\sigma = 10^{-3}$ s.

In practice, the gains should be increased only together with a compatible integration step. Larger gains shorten the continuous-time transient, whereas smaller steps improve temporal resolution at the cost of more numerical updates. At a fixed step, excessive gains may cause oscillation or instability. For a new problem, a practical procedure is to monitor the equation residuals, tracking errors, and state trajectories, and reduce the step when necessary. A step-refinement comparison can also be used to assess whether the numerical accuracy is adequate.

7.1. Example 1: Distinct real desired poles

Consider a 4-dimension TILS with coefficient matrices:

$$A = \begin{bmatrix} -5 & -2 & -6 & -5 \\ 2 & -4 & 1 & -3 \\ -1 & 1 & 0 & 7 \\ 1 & 1 & 0 & -7 \end{bmatrix}, B = \begin{bmatrix} 0 & 2 & 2 \\ 5 & -4 & 3 \\ 3 & 4 & 3 \\ 5 & 5 & -1 \end{bmatrix}.$$

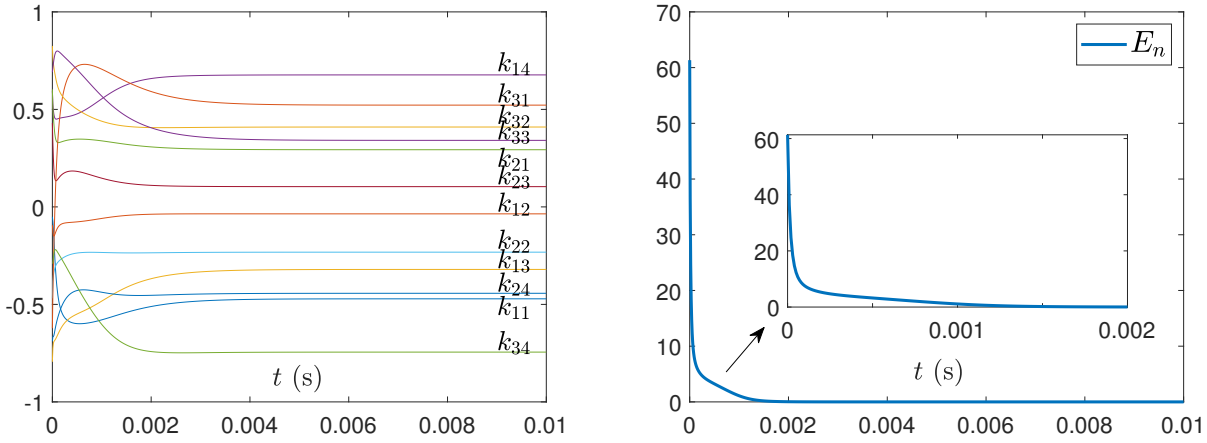
The initial poles of the system are $\{-0.36, -4.25, -5.70 \pm 1.24\mathbf{i}\}$. Suppose that the desired poles are $\{-1, -2, -3, -4\}$, then the diagonal matrix Λ is set as $\text{diag}(-1, -2, -3, -4)$. The random initial value K_1 is set as

$$\begin{bmatrix} 0.08 & -0.11 & -0.99 & 0.74 \\ 0.99 & -0.79 & 0.55 & -0.83 \\ -0.84 & 0.92 & 0.63 & -0.20 \end{bmatrix}.$$

The sampling period σ_1 is set as 10^{-5} s, and design parameter γ is set as 10^3. The numerical results are shown in Figure 2. The matrix $K(0.01)$ obtained by discrete GN algorithm (4.3) is

$$\begin{bmatrix} -0.47 & -0.036 & -0.32 & 0.68 \\ 0.29 & -0.23 & 0.10 & -0.44 \\ 0.52 & 0.41 & 0.34 & -0.74 \end{bmatrix}.$$

As known in Table 1, discrete GN algorithm (4.3) makes the real parts of the eigenvalues of $A+BK_n$ converge to the desired poles, and the imaginary parts converge to zero rapidly and precisely. As shown in Figure 2a, each element of matrix K_n converges to a stable value in a short time, which is consistent with the convergence stability in Propositions 6.2 and 6.3. Besides, Figure 2b shows that the discrete energy function $E_n = (\|A + BK_n - P_n\Lambda Q_n\|_F^2 + \|P_nQ_n - I\|_F^2 + \|Q_nP_n - I\|_F^2)/2$ converges to zero rapidly as proven in Proposition 6.1. Moreover, the convergence time for the discrete energy function to achieve $E_n \leq 0.001$ is 0.003 s. When the time t reaches to 0.01 s, the discrete energy function achieves $E_n \leq 1.25 \times 10^{-11}$. The convergence behavior indicates strong robustness against numerical discretization errors inherent in discrete time implementations.



(a) (b)

Figure 2. Numerical results of Example 1. (a) Elements trajectories $k_{11}, k_{12}, \ldots, k_{34}$ of matrix K_n. (b) Discrete energy function trajectory E_n.

Table 1. Real parts and imaginary parts of eigenvalues of matrix $A + BK_n$ in Example 1.

eig$(A + BK_n)$	0 s	0.002 s	0.004 s	0.006 s	0.008 s	0.01 s
Re$(\sigma_1(A + BK_n))$	-4.19	-4.12	-4.01	**-4.00**	**-4.00**	**-4.00**
Re$(\sigma_2(A + BK_n))$	-4.19	-1.01	-2.98	**-3.00**	**-3.00**	**-3.00**
Re$(\sigma_3(A + BK_n))$	-2.52	-2.78	**-1.00**	**-1.00**	**-1.00**	**-1.00**
Re$(\sigma_4(A + BK_n))$	-0.03	**-2.00**	**-2.00**	**-2.00**	**-2.00**	**-2.00**
Im$(\sigma_1(A + BK_n))$	5.24	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Im$(\sigma_2(A + BK_n))$	-5.24	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Im$(\sigma_3(A + BK_n))$	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Im$(\sigma_4(A + BK_n))$	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**

7.2. Example 2: Repeated real desired poles

Consider a 5-dimension TILS with coefficient matrices:

$$A = \begin{bmatrix} 0 & -2 & 0 & 1 & 1 \\ 5 & -2 & 4 & -1 & 1 \\ -1 & 1 & 0 & -3 & 1 \\ -3 & -2 & 0 & -1 & 4 \\ 0 & 0 & 1 & -3 & 0 \end{bmatrix}, B = \begin{bmatrix} -4 & 0 & -1 \\ 2 & -1 & 1 \\ -1 & -2 & 0 \\ -1 & 1 & -4 \\ -3 & -3 & 5 \end{bmatrix}.$$

The initial poles of the system are $\{-1.11, -1.46 \pm 3.85\mathbf{i}, 0.52 \pm 1.93\mathbf{i}\}$. Suppose that the desired poles are $\{-3, -3, -3, -3, -3\}$. According to Section 3, the Jordan matrix Λ is set as

$$\begin{bmatrix} -3 & 1 & 0 & 0 & 0 \\ 0 & -3 & 1 & 0 & 0 \\ 0 & 0 & -3 & 1 & 0 \\ 0 & 0 & 0 & -3 & 1 \\ 0 & 0 & 0 & 0 & -3 \end{bmatrix}.$$

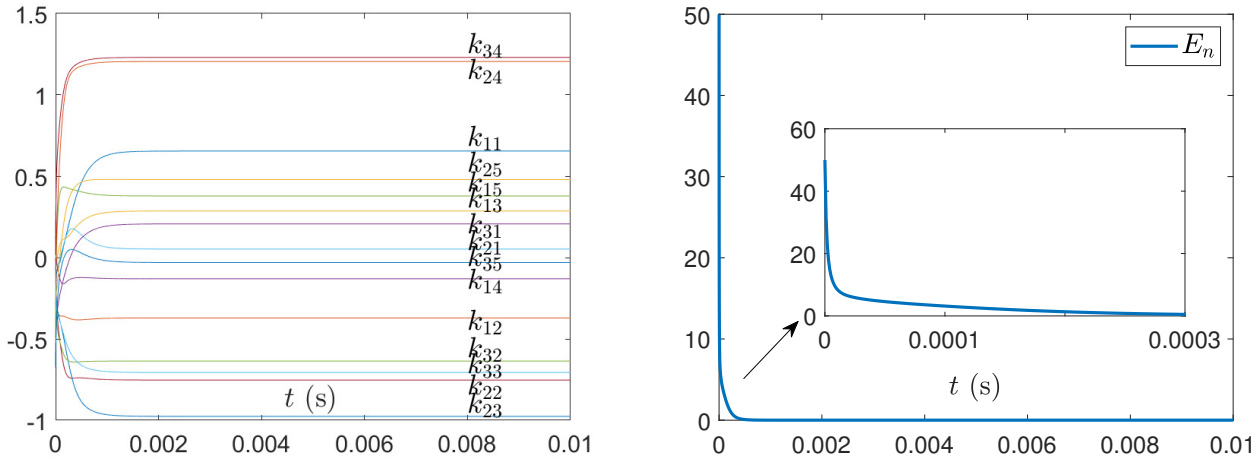
The random initial value of K_1 is set as

$$\begin{bmatrix} -0.49 & -0.17 & -0.03 & 0.21 & -0.78 \\ 0.01 & 0.03 & -0.04 & -0.19 & -0.28 \\ -0.79 & -0.24 & -0.12 & -0.26 & -0.88 \end{bmatrix}.$$

The sampling period σ_1 is set as 10^{-6} s, and design parameter γ is set as 10^4. The experimental results are shown in Figure 3. The matrix $K(0.01)$ is

$$\begin{bmatrix} 0.65 & -0.37 & 0.29 & -0.13 & 0.38 \\ 0.05 & -0.75 & -0.98 & 1.21 & 0.48 \\ 0.21 & -0.64 & -0.71 & 1.23 & -0.03 \end{bmatrix}.$$

To further illustrate the efficiency of the designed model, Table 2 shows that the GN algorithm (4.3) also has excellent performance in the situation of repeated real desired poles. As shown in Figure 3a and 3b, each element of K_n reaches stable state in a short time, and the discrete energy function descends to zero rapidly. The numerical results illustrate strong consistency with the conclusions in Propositions 6.1 and 6.2. The observed decrease in the discrete energy function and the approach of the closed-loop poles to their desired values illustrate the numerical performance of the GN algorithm in this example. Moreover, the convergence time for the discrete energy function to achieve $E_n \leq 0.001$ is 0.001 s, and $E_n \leq 4.46 \times 10^{-28}$ when t reaches 0.01 s.



(a) (b)

Figure 3. Numerical results of Example 2. (a) Elements trajectories $k_{11}, k_{12}, \ldots, k_{35}$ of matrix K_n. (b) Discrete energy function E_n trajectory.

Table 2. Real parts and imaginary parts of eigenvalues of matrix $A + BK_n$ in Example 2.

eig$(A + BK_n)$	0 s	0.002 s	0.004 s	0.006 s	0.008 s	0.01 s
Re$(\sigma_1(A + BK_n))$	-0.46	-3.09	-3.01	-3.01	**-3.00**	**-3.00**
Re$(\sigma_2(A + BK_n))$	-0.46	-3.09	-3.01	**-3.00**	**-3.00**	**-3.00**
Re$(\sigma_3(A + BK_n))$	1.09	-3.10	-3.03	**-3.00**	**-3.00**	**-3.00**
Re$(\sigma_4(A + BK_n))$	1.09	-2.85	-2.97	**-3.00**	**-3.00**	**-3.00**
Re$(\sigma_5(A + BK_n))$	-4.22	-2.85	-2.97	**-3.00**	**-3.00**	**-3.00**
Im$(\sigma_1(A + BK_n))$	5.00	0.13	0.03	**0.00**	**0.00**	**0.00**
Im$(\sigma_2(A + BK_n))$	-5.00	-0.13	-0.03	0.01	**0.00**	**0.00**
Im$(\sigma_3(A + BK_n))$	0.81	0.00	0.00	-0.01	**-0.00**	**-0.00**
Im$(\sigma_4(A + BK_n))$	-0.81	0.13	0.02	**0.00**	**0.00**	**0.00**
Im$(\sigma_5(A + BK_n))$	0.00	-0.13	-0.02	**-0.00**	**-0.00**	**-0.00**

7.3. Example 3: Complex conjugate pairs

Consider a 4-dimension TILS with coefficient matrices:

$$A = \begin{bmatrix} 3 & -4 & 1 & -2 \\ -3 & 0 & 2 & 1 \\ -3 & 3 & -3 & 0 \\ -3 & 3 & 5 & -1 \end{bmatrix}, B = \begin{bmatrix} -1 & 1 & 2 \\ 1 & 3 & 1 \\ 5 & -1 & 1 \\ 2 & 1 & 2 \end{bmatrix}.$$

The initial poles are $\{7.13, -2.81, -2.66 \pm 1.72\mathbf{i}\}$. Suppose that the desired poles are $\{-3 \pm \mathbf{i}, -2 \pm 2\mathbf{i}\}$.

According to Section 3, the real Jordan matrix Λ is set as

$$\begin{bmatrix} -3 & 1 & 0 & 0 \\ -1 & -3 & 0 & 0 \\ 0 & 0 & -2 & 2 \\ 0 & 0 & -2 & -2 \end{bmatrix}.$$

We set the random initial value of K_1 as

$$\begin{bmatrix} 0.074 & 0.11 & -0.028 & -0.30 \\ -0.17 & -0.36 & -0.54 & -0.18 \\ 0.00 & 0.47 & 0.48 & -0.38 \end{bmatrix}.$$

The sampling period σ_1 and design parameter γ are set as 10^{-6} s and 10^4. The experimental results are shown in Figure 4. The matrix $K(0.01)$ is

$$\begin{bmatrix} 1.64 & -1.41 & -0.05 & -0.55 \\ 0.51 & -1.00 & -0.19 & -0.56 \\ -1.52 & 1.49 & -1.39 & 0.59 \end{bmatrix}.$$

Specifically, Figure 4a and 4b are trajectories of the eigenvalue of matrix $A + BK_n$. It is shown that the poles of the closed-loop system converge to the desired poles precisely in a short time. As seen from Figure 4c and 4d, each element of K_n reaches equilibrium state quickly, and the discrete energy function converges to zero rapidly. The numerical results once again confirm the conclusions of Propositions 6.1 and 6.2. The observed discrete energy function reduction and the convergence of closed-loop poles to their desired locations effectively illustrate the numerical performance of the GN algorithm in this example. Moreover, the convergence time for the discrete energy function to achieve $E_n \leq 0.001$ is 0.0005 s, and $E_n \leq 3.35 \times 10^{-28}$ when time t reaches 0.01s.

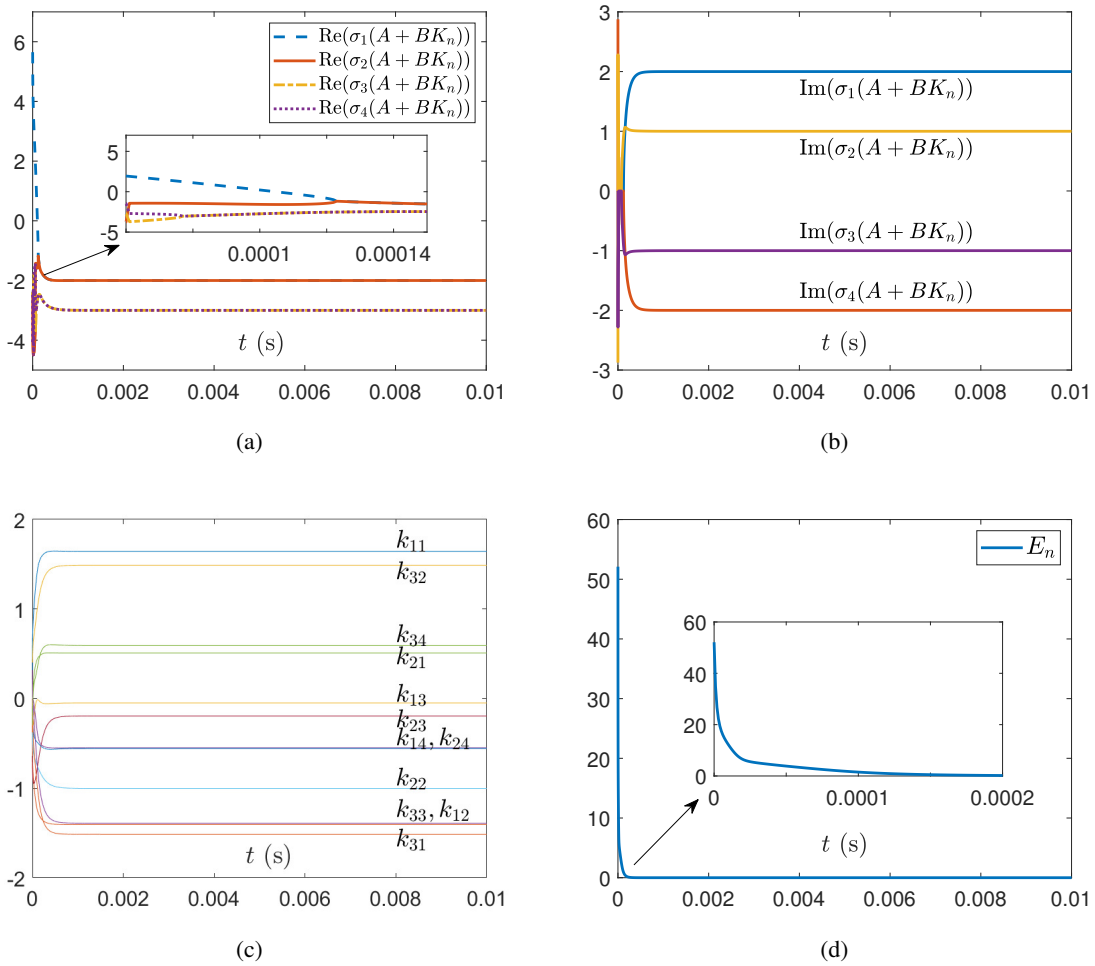


Figure 4. Numerical results of Example 3. (a) Real parts trajectories of eigenvalues of matrix $A + BK_n$. (b) Imaginary parts trajectories of eigenvalues of matrix $A + BK_n$. (c) Elements trajectories $k_{11}, k_{12}, \ldots, k_{35}$ of matrix K_n. (d) Discrete energy function trajectory E_n.

7.4. Example 4: Output tracking control

In this section, we perform output tracking control on the systems in Examples 2 and 3. Based on the system in Example 2, we set the coefficient matrix C as

$$\begin{bmatrix} 0 & 1 & -2 & 2 & -1 \\ 1 & 0 & 3 & 0 & 0 \\ 0 & 1 & 0 & 2 & 4 \end{bmatrix}.$$

The random initial value of $\mathbf{x}_1$ is set as $[0.02, 0.55, 0.04, -0.31, -0.16]^{\mathrm{T}}$. The design parameter and sampling period are set as $\lambda = 10^3$ and $\sigma_2 = 10^{-3}$ s. The desired trajectories are set as $\mathbf{y}_{\mathrm{d}}(t) = [\cos(100/(t+1)), \exp(\sin(t)), 1/(2+\sin(t))]^{\mathrm{T}}$. By using the discrete ZN algorithm (5.4), the experimental results are shown in Figure 5. Based on the system in Example 3, we set the coefficient matrix C as:

$$\begin{bmatrix} -3 & 2 & 1 & -1 \\ -3 & 2 & 0 & 3 \end{bmatrix}.$$

The random initial value of $\mathbf{x}_1$ is set as $[0.37, 0.11, -0.10, 0.56]^{\mathrm{T}}$. The design parameter and sampling period are set as $\lambda = 10^3$ and $\sigma_2 = 10^{-3}$ s. The desired trajectories are set as $\mathbf{y}_{\mathrm{d}}(t) = [\sin(t)/(t+1), \sin(t)\exp(-t/20)]^{\mathrm{T}}$. The experimental results using the discrete ZN algorithm (5.4) are shown in Figure 6. The actual output and desired trajectories of the two experiments are shown in Figures 5a and 6a. The state $\mathbf{x}_n$ of the system and the input vector $\mathbf{u}_n$ are shown in Figure 5b and 5c, Figure 6b and 6c. The output tracking errors are presented in Figures 5d and 6d. From Figures 5d and 6d, we find that each component of the output tracking error $\mathbf{e}_n$ is less than 1×10^{-3} as time increases. From Figures 5 and 6, we find that the discrete ZN algorithm (5.4) has excellent performance in solving the output tracking problem in TILS. Moreover, the ZN algorithm (5.4) shows its effectiveness in both scenarios when CB is a full rank square matrix and CB is a full row rank matrix, which corresponds to the conclusion in Proposition 6.4.

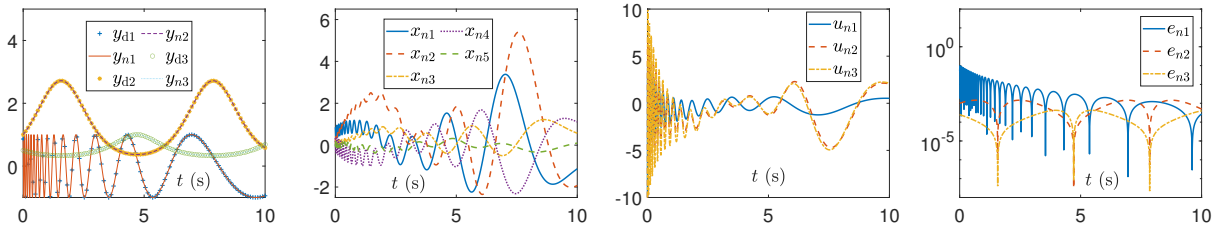


Figure 5. Numerical result of Example 4 based on Example 2. (a) Actual outputs y_{n1}, y_{n2}, y_{n3} and desired trajectories $y_{\mathrm{d}1}$, $y_{\mathrm{d}2}$, $y_{\mathrm{d}3}$. (b) States of system x_{n1}, x_{n2}, x_{n3}, x_{n4}, x_{n5}. (c) Control inputs u_{n1}, u_{n2}, u_{n3}. (d) Semi-logarithmic plot of output tracking errors e_{n1}, e_{n2}, e_{n3}.

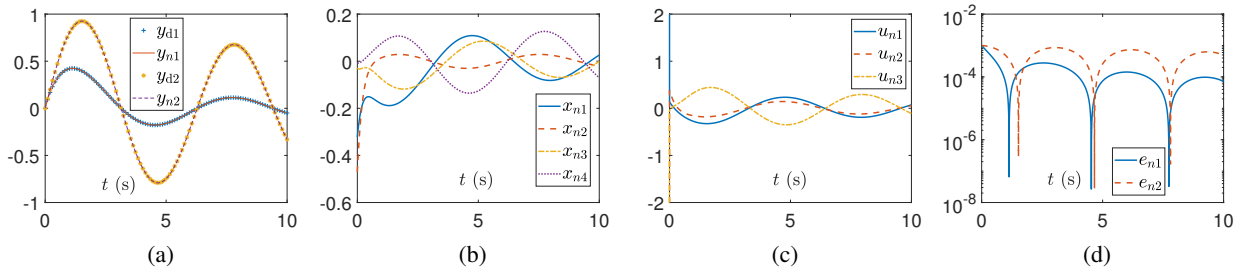


Figure 6. Numerical results of Example 4 based on Example 3. (a) Actual outputs y_{n1}, y_{n2} and desired trajectories $y_{\mathrm{d}1}$, $y_{\mathrm{d}2}$. (b) States of system x_{n1}, x_{n2}, x_{n3}. (c) Control inputs u_{n1}, u_{n2}, u_{n3}. (d) Semi-logarithmic plot of output tracking errors e_{n1}, e_{n2}.

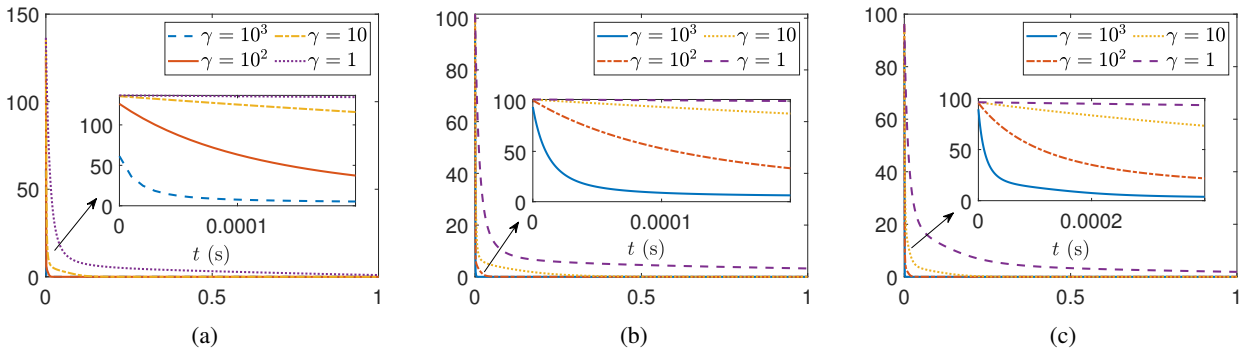


Figure 7. Numerical results of Example 5 based on Examples 1–3. (a) Discrete energy function trajectories on the system in Example 1 with $\gamma = \{10^3, 10^2, 10, 1\}$. (b) Discrete energy function trajectories with $\gamma = \{10^3, 10^2, 10, 1\}$. (c) Discrete energy function trajectories with $\gamma = \{10^3, 10^2, 10, 1\}$.

7.5. Ablation study

In this section, we perform two comparative experiments of the design parameters γ and λ in the GN model (4.2) and ZN model (5.1) on the Examples 1 through 4. Based on the Examples 1–3, γ is set as $\{1, 10, 10^2, 10^3\}$. In Example 4, we set the λ as $\{1, 10, 10^2, 10^3\}$. The comparative results are shown in Figures 7 and 8. From Figure 7, we find that for all the situations of desired poles, the larger design parameter γ leads to the faster convergence speed for the discrete energy function E_n. This is consistent with the conclusion in Proposition 6.1. From Figures 8a and 8b, we find that the larger design parameter μ makes the faster convergence rate for output tracking errors. It is highly consistent with the conclusion in Proposition 6.4.

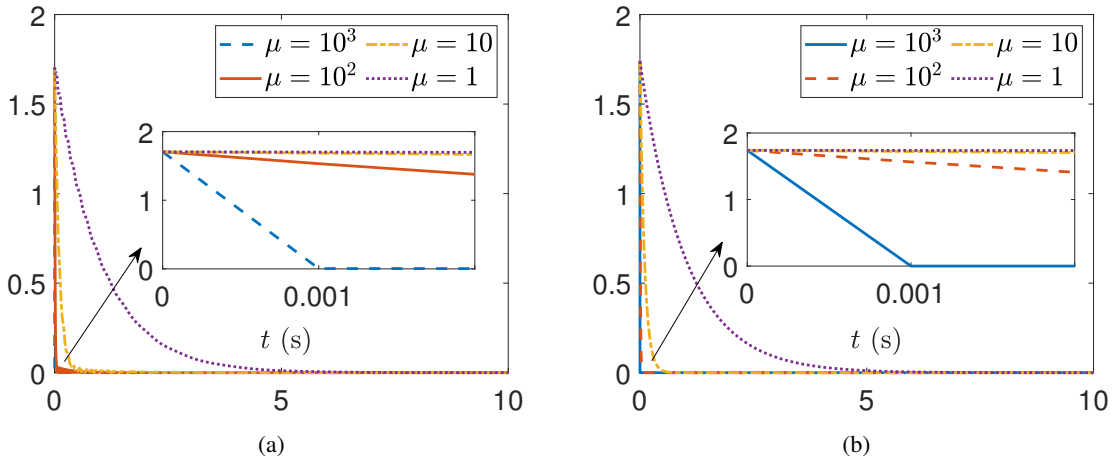


Figure 8. Numerical results of Example 5 based on Example 4. (a) $\|\mathbf{e}_n\|_2$ trajectories of Example 2 on system 1 in Example 4 with $\mu = \{10^3, 10^2, 10, 1\}$. (b) $\|\mathbf{e}_n\|_2$ trajectories of Example 3 on system 2 in Example 4 with $\mu = \{10^3, 10^2, 10, 1\}$.

7.6. Computational cost

To assess the computational overhead, we measure the execution time of the GN pole-placement stage and the ZN tracking stage separately in MATLAB R2024a on a computer with an Intel Core i9-14900HX processor, 32 GB RAM, and 64-bit Windows 11. Each experiment is repeated 100 times after a warm-up run, and the average value is reported. Initial states are restored before every run. Timing includes the numerical updates but excludes the plotting and file output. Table 3 reports the data of the system dimension, the number of updates, and the CPU time.

Table 3. Measured computational cost of GN and ZN stages.

Case	Stage	n	m	N	CPU time ($\times 10^{-2}$ s)
Example 1	GN	4	3	1000	0.258
Example 2	GN	5	3	10000	3.703
Example 3	GN	4	3	10000	2.820
Example 4 ($n = 5$)	ZN	5	3	10000	1.984
Example 4 ($n = 4$)	ZN	4	3	10000	1.844

Note: N denotes the number of numerical updates; times are reported as average value over repeated runs; plotting and file output are excluded from timing.

8. Conclusions

This paper has addressed the output tracking control problem of neurodynamic-solvable TILS and proposed the MSBPPE. The feedback gain matrix has been obtained offline through the GN model, and the reference input has been generated using the ZN model as an attempt, thereby achieving the desired path precise tracking. Theoretical analyses have shown the asymptotic stability and exponential convergence of the proposed neurodynamics models. Numerical experiments under various situations of poles (including distinct real poles, repeated real poles, and complex conjugate poles) have verified the relative feasibility and effectiveness of the proposed method in terms of convergence speed and tracking accuracy. Comparative experiments have revealed that, within the hardware permitted, the larger the parameters are, the better the tracking effect is.

Note that the applicability of the proposed framework is subject to several limitations. The GN stage requires the chosen MSBPPE to admit a feasible solution. Its energy function is generally nonconvex, and numerical performance can depend on initialization, matrix conditioning, and the integration step. Small energy gradients alone do not certify that all equation residuals are sufficiently small. The ZN tracking construction additionally requires $\text{rank}(CB) = l$ and an available desired-trajectory derivative.

Future work will investigate some improved initialization, regularization, and safeguards against ill-conditioned iterates, together with adaptive gain and step-size strategies for the GN solver. For systems with rank-deficient CB, extensions based on higher-order output dynamics or alternative controller structures will be explored where the corresponding tracking problem is feasible. Further developments will also consider noisy measurements, model uncertainty, and actuator constraints to broaden the practical applicability of the framework.

Author contributions

Zhengyang Tang: Data curation, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review and editing; Yunong Zhang: Conceptualization, data curation, formal analysis, funding acquisition, investigation, methodology, project administration, resources, supervision, validation, visualization, writing—review and editing; Yanli Ji: Formal analysis, funding acquisition, investigation, methodology, resources, supervision, validation, writing—review and editing. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

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

Acknowledgments

This work is supported by the National Natural Science Foundation of China with number 61976230, the Shenzhen Science and Technology Program with number JCYJ20250604175409013, and the Science and Technology Innovation Committee of Shenzhen Municipality Foundation with number JCYJ20230807120000001.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. F. Zhu, P. Li, Adaptive fuzzy consensus tracking control of multi-agent systems with predefined time, *AIMS Mathematics*, **10** (2025), 5307–5331. https://doi.org/10.3934/math.2025245
2. J. Li, Y. Zhang, H. J. Lee, Y. Wang, Fuzzy tracking control of singular multi-agent systems under switching topology, *AIMS Mathematics*, **9** (2024), 29718–29735. https://doi.org/10.3934/math.20241440
3. M. Aly, B. Ali, F. Y. Mekonnen, M. Elhesasy, M. Wang, M. M. Kamra, et al., Real-flight-path tracking control design for quadrotor UAVs: A precision-guided approach, *Automation*, **6** (2025), 93. https://doi.org/10.3390/automation6040093
4. Y. Xu, J. Liu, H. Yan, A. Wang, H. Xu, Y. Ma, et al., Target tracking-based online calibration of UAV electro-optical pod installation errors, *Automation*, **7** (2026), 59. https://doi.org/10.3390/automation7020059
5. D. Chen, Y. Zhang, S. Li, Tracking control of robot manipulators with unknown models: A Jacobian-matrix-adaption method, *IEEE Trans. Ind. Inform.*, **14** (2018), 3044–3053. https://doi.org/10.1109/TII.2017.2766455

6. J. Chen, Y. Pan, Y. Zhang, Model-free and pseudoinverse-free Zhang neurodynamics scheme for robotic arms' path tracking control, *IEEE Trans. Neural Netw. Learn. Syst.*, **36** (2025), 10201–10212. https://doi.org/10.1109/TNNLS.2025.3540589
7. H. Zhong, W. Wen, J. Fan, W. Yang, Reinforcement learning-based adaptive tracking control for flexible-joint robotic manipulators, *AIMS Mathematics*, **9** (2024), 27330–27360. https://doi.org/10.3934/math.20241328
8. L. Jin, L. Liu, X. Wang, M. Shang, F. Wang, Physical-informed neural network for MPC-based trajectory tracking of vehicles with noise considered, *IEEE Trans. Intell. Veh.*, **9** (2024), 4493–4503. https://doi.org/10.1109/TIV.2024.3358229
9. Y. Liufu, L. Jin, M. Shang, X. Wang, F. Wang, ACP-incorporated perturbation-resistant neural dynamics controller for autonomous vehicles, *IEEE Trans. Intell. Veh.*, **9** (2024), 4675–4686. https://doi.org/10.1109/TIV.2023.3348632
10. T. Belkebir, H. Belkebir, A. Mansouri, Real-time embedded NMPC for autonomous vehicle path tracking with curvature-aware speed adaptation and sensitivity analysis, *Automation*, **7** (2026), 44. https://doi.org/10.3390/automation7020044
11. D. Xu, Y. Shi, Z. Ji, Model-free adaptive discrete-time integral sliding-mode-constrained-control for autonomous 4WMV parking systems, *IEEE Trans. Ind. Electron.*, **65** (2018), 834–843. https://doi.org/10.1109/TIE.2017.2739680
12. S. Chaturvedi, N. Kumar, R. Kumar, A PSO-optimized novel PID neural network model for temperature control of jacketed CSTR: Design, simulation, and a comparative study, *Soft Comput.*, **28** (2024), 4759–4773. https://doi.org/10.1007/s00500-023-09138-0
13. Y. Zhang, J. Wang, Global exponential stability of recurrent neural networks for synthesizing linear feedback control systems via pole assignment, *IEEE Trans. Neural Netw.*, **13** (2002), 633–644. https://doi.org/10.1109/TNN.2002.1000129
14. F. M. Callier, C. A. Desoer, *Linear system theory*, New York: Springer, 1991. https://doi.org/10.1007/978-1-4612-0957-7
15. J. H. Mathews, K. D. Fink, *Numerical methods using MATLAB*, Prentice Hall, 2004.
16. J. Kautsky, N. K. Nichols, P. V. Dooren, Robust pole assignment in linear state feedback, *Int. J. Control*, **41** (1985), 1129–1155. https://doi.org/10.1080/0020718508961188
17. P. Blanchard, *Differential equations*, Brooks/Cole Pub. Co., 1998.
18. Y. Zhang, Y. Shi, K. Chen, C. Wang, Global exponential convergence and stability of gradient-based neural network for online matrix inversion, *Appl. Math. Comput.*, **215** (2009), 1301–1306. https://doi.org/10.1016/j.amc.2009.06.048
19. Y. Zhang, D. Jiang, J. Wang, A recurrent neural network for solving Sylvester equation with time-varying coefficients, *IEEE Trans. Neural Netw.*, **13** (2002), 1053–1063. https://doi.org/10.1109/TNN.2002.1031938
20. Y. Zhang, S. Ge, Design and analysis of a general recurrent neural network model for time-varying matrix inversion, *IEEE Trans. Neural Netw.*, **16** (2005), 1477–1490. https://doi.org/10.1109/TNN.2005.857946

21. B. Liao, L. Han, X. Cao, S. Li, J. Li, Double integral-enhanced Zeroing neural network with linear noise rejection for time-varying matrix inverse, *CAAI Trans. Intell. Technol.*, **9** (2024), 197–210. https://doi.org/10.1049/cit2.12161

22. Y. Yang, Y. Zhang, Superior robustness of power-sum activation functions in Zhang neural networks for time-varying quadratic programs perturbed with large implementation errors, *Neural Comput. Appl.*, **22** (2013), 175–185. https://doi.org/10.1007/s00521-011-0692-5

23. Y. Zhang, G. Ruan, K. Li, Y. Yang, Robustness analysis of the Zhang neural network for online time-varying quadratic optimization, *J. Phys. A: Math. Theor.*, **43** (2010), 245202. https://doi.org/10.1088/1751-8113/43/24/245202

24. B. Liao, Y. Zhang, L. Jin, Taylor $O(h^3)$ discretization of ZNN models for dynamic equality-constrained quadratic programming with application to manipulators, *IEEE Trans. Neural Netw. Learn. Syst.*, **27** (2016), 225–237. https://doi.org/10.1109/TNNLS.2015.2435014

25. Z. Zhang, Y. Zhang, Acceleration-level cyclic-motion generation of constrained redundant robots tracking different paths, *IEEE Trans. Syst. Man Cybern., Part B, Cybern.*, **42** (2012), 1257–1269. https://doi.org/10.1109/TSMCB.2012.2189003

26. J. Li, Y. Zhang, S. Li, M. Mao, New discretization-formula-based Zeroing dynamics for real-time tracking control of serial and parallel manipulators, *IEEE Trans. Ind. Inform.*, **14** (2018), 3416–3425. https://doi.org/10.1109/TII.2017.2780892

27. Y. Ding, H. Jia, Y. Zhang, B. Qiu, High-order modeling, Zeroing dynamics control, and perturbations rejection for non-linear double-holding water tank, *Mathematics*, **11** (2023), 2989. https://doi.org/10.3390/math11132989

28. V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, Y. Zhang, 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.*, **33** (2022), 3415–3424. https://doi.org/10.1109/TNNLS.2021.3052896

29. A. Winursito, G. N. P. Pratama, LQR state feedback controller with precompensator for magnetic levitation system, *J. Phys. Conf. Ser.*, **2111** (2021), 012004. https://doi.org/10.1088/1742-6596/2111/1/012004

30. E. J. Davison, H. W. Smith, A note on the design of industrial regulators: Integral feedback and feedforward controllers, *Automatica*, **10** (1974), 329–332. https://doi.org/10.1016/0005-1098(74)90044-2

31. E. J. Davison, The feedforward control of linear multivariable time-invariant systems, *Automatica*, **9** (1973), 561–573. https://doi.org/10.1016/0005-1098(73)90042-3

32. B. A. Francis, The linear multivariable regulator problem, *SIAM J. Control Optim.*, **15** (1977), 486–505. https://doi.org/10.1137/0315033

33. B. A. Francis, W. M. Wonham, The internal model principle of control theory, *Automatica*, **12** (1976), 457–465. https://doi.org/10.1016/0005-1098(76)90006-6

34. R. A. Horn, C. R. Johnson, *Matrix analysis*, Cambridge University Press, 2012. https://doi.org/10.1017/CBO9780511810817

35. W. Greub, *Linear algebra*, New York: Springer, 1975. https://doi.org/10.1007/978-1-4684-9446-4

36. Y. Zhang, K. Chen, H. Tan, Performance analysis of gradient neural network exploited for online time-varying matrix inversion, *IEEE Trans. Automat. Control*, **54** (2009), 1940–1945. https://doi.org/10.1109/TAC.2009.2023779

37. P. Guo, Y. Zhang, S. Li, Reciprocal-kind Zhang neurodynamics method for temporal-dependent Sylvester equation and robot manipulator motion planning, *IEEE Trans. Neural Netw. Learn. Syst.*, **36** (2025), 15110–15124. https://doi.org/10.1109/TNNLS.2025.3525747



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0)