Power-Type Varying-Parameter RNN for Solving TVQP Problems: Design, Analysis, and Applications

Zhijun Zhang, *Member, IEEE*, Ling-Dong Kong, *Student Member, IEEE*, and Lunan Zheng, *Student Member, IEEE*

***Abstract*— Many practical problems can be solved by being formulated as time-varying quadratic programing (TVQP) problems. In this paper, a novel power-type varying-parameter recurrent neural network (VPNN) is proposed and analyzed to effectively solve the resulting TVQP problems, as well as the original practical problems. For a clear understanding, we introduce this model from three aspects: *design*, *analysis*, and *applications*. Specifically, the reason why and the method we use to design this neural network model for solving online TVQP problems subject to time-varying linear equality/inequality are described in detail. The theoretical analysis confirms that when activated by six commonly used activation functions, VPNN achieves a superexponential convergence rate. In contrast to the traditional zeroing neural network with fixed design parameters, the proposed VPNN has better convergence performance. Comparative simulations with state-of-the-art methods confirm the advantages of VPNN. Furthermore, the application of VPNN to a robot motion planning problem verifies the feasibility, applicability, and efficiency of the proposed method.**

***Index Terms*— Convergence, dynamic programing, quadratic programing (QP), recurrent neural network (RNN), time-varying problem.**

I. Introduction

TIME-VARYING quadratic programing (TVQP) problems have been widely studied for decades. In this paper, three *Subtopics* related to why and how to exploit a new method to handle TVQP problems are proposed and discussed. Specifically, starting from the TVQP problem itself, the train-of-thought behind different methods for solving this problem will be discussed in Section I-A. Furthermore, to compensate for the shortcomings of the traditional methods, a novel model is proposed, and the design idea is analyzed in Section I-B. Moreover, to implicitly consider the proposed model for practical applications, Section I-C provides an overview of the related works and offers a method for the proposed neural model to solve kinematic problems of robots.

Manuscript received June 20, 2017; revised May 28, 2018 and November 14, 2018; accepted December 2, 2018. This work was supported in part by the National Natural Science Foundation under Grant 61603142 and Grant 61633010, in part by the Guangdong Foundation for Distinguished Young Scholars under Grant 2017A030306009, in part by the Guangdong Youth Talent Support Program of Scientific and Technological Innovation under Grant 2017TQ04X475, in part by the Science and Technology Program of Guangzhou under Grant 201707010225, in part by the Fundamental Research Funds for Central Universities under Grant x2zdD2182410, in part by the Scientific Research Starting Foundation of the South China University of Technology, in part by the National Key R&D Program of China under Grant 2017YFB1002505, in part by the National Key Basic Research Program of China (973 Program) under Grant 2015CB351703, and in part by the Guangdong Natural Science Foundation under Grant 2014A030312005. *(Corresponding authors: Zhijun Zhang; Ling-Dong Kong.)*

The authors are with the School of Automation Science and Engineering, South China University of Technology (SCUT), Guangzhou 510641, China, and also with the Human-Robot Intelligence Lab, Center for Brain Computer Interfaces and Brain Information Processing, SCUT, Guangzhou 510641, China (e-mail: auzjzhang@scut.edu.cn; ldkong@ieee.org; aulnzheng@sina.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2885042

A. Subtopic 1: TVQP Problems Solving

The origin of quadratic programing (QP) can be traced to the 1950s when H. W. Kuhn and A. W. Tucker redesigned the optimal conditions of nonlinear problems. In recent years, as computers have become more efficient, the global optimization of heuristic algorithms and large-scale problems has become increasingly popular; thus, numerous practical optimization problems in industrial scenarios have been described as QP problems [1]–[3]. For some problems which emphasize high accuracy and dynamic impact [4], for example, robot motion planning [5]–[10], optimal controller design [11], [12], and electric power dispatching [13], [14], the impact of time factors should be earnestly considered [15], [16]. Therefore, we must find ways to describe and solve such TVQP problems.

TVQP problems can be handled in several ways. The traditional approaches include the interior point method, the conjugate gradient method, and an active set method. Schochetman *et al.* [17] first proved the existence of a solution for TVQP problems, i.e., if the eigenvalues of the cost matrices of the TVQP are bounded from zero, then there must exist a unique solution. On the basis of this property, some numerical algorithms [18], in which the minimal arithmetic operations are typically proportional to the dimension of the cube of the Hessian matrix $[O(n^3)]$, were proposed to solve TVQP problems. A numerical method named E47, which is based on the linear variational equation, was proposed in [19] to solve TVQP problems. Although feasible, however, when subject to large-scale online problems, these numerical algorithms may not be adequately efficient for computation.

$O(n^2)$ operation algorithms have been proposed to overcome the computational deficiency. Jakovetic *et al.* [20] proposed an augmented Lagrangian gossiping method based

on Lagrangian dual functions to solve cooperative convex TVQP problems. Zhang *et al.* [21] proposed a Toeplitz scheme for time-series Gaussian process regression of large data sets. An $O(n^2)$ active set method was proposed by Gomez [22] to solve related box-constrained TVQP problems. Nevertheless, despite being computable problems, the computational time is expensive. An $O(n^2)$-operation algorithm requires more than one and a half hours to invert a 100 000-dimensional matrix [21]. Clearly, such performance is not satisfactory.

The recent resurgence of neural networks led to the proposal of several neural-network-based approaches. Wang *et al.* proposed a primal-dual neural network (PDNN) for solving convex TVQP problems [23]–[26] and applied the network to the kinematics resolution of robots [27]. Theoretical analyses proved the global exponential convergence stability of PDNN for tracking time-varying solutions of TVQP problems. However, a complex projection function must be found when designing and exploiting PDNN [27], which may increase the difficulty in practical applications. A gradient neural network (GNN) based on gradient descent was proposed to solve TVQP problems [9]. GNN offers a simpler method for designing the neural model, which is attached to a scalar-valued norm energy function. Researchers have utilized GNN to solve practical problems based on a convex TVQP framework [28]. However, when the task is time-varying, the traditional GNN has difficulty tracking the theoretical solutions [29]. In other words, residual errors always exist when the GNN is used to solve time-varying problems [30]. Therefore, the GNN model is more often applied to solving constant problems [31] rather than time-varying problems.

B. Subtopic 2: Neural Model Design

Why choose a recurrent neural network (RNN) to handle TVQP problems? An RNN [32]–[35] is a special type of neural network that is good at processing sequence data, and its units connect to form a directed loop. Generally, an artificial neural network consists of multilayer neurons. A typical connection model in a feedforward neural network has interconnection only between layers and not between neurons in the same layer. On this basis, an RNN combines the circular connection of hidden layers to learn features and their dependencies from sequential to temporal data [36]–[38]. Each individual computing unit in an RNN hidden layer corresponds to the state of a certain time node in the data, which could be a simple neuron or a neuron layer of myriad gated control systems. Moreover, each unit is sequentially connected through layers that share parameters and is then propagated along with the data sequence. This feature enables the current state of each unit in an RNN to depend on its past state and thus has a function similar to "memory," i.e., the storage and processing of long-term data signals [38]. Therefore, an RNN is suitable for handling variable-length sequences and can theoretically model any dynamic system, such as TVQP problems [39].

One of the classic RNNs, i.e., zeroing neural network (ZNN), was proposed by Leung *et al.* [28] to improve the performance of GNN for solving time-varying problems [40]–[42]. ZNN tracks the optimal solutions of time-varying problems through the utilization of derivative information [43]. Many researchers have used the ZNN model to handle time-varying problems. Xiao and Zhang [44] utilized ZNN to address time-varying linear inequalities, and they compared the convergence performance with that of GNN. Comparisons of ZNN and GNN for solving time-varying matrix equations were performed by Chen *et al.* [31]. The global exponential convergence of ZNN for solving TVQP problems was proved by Zhang and Li [45], and the robustness performance of ZNN was theoretically proved and verified through simulations by Zhang *et al.* [46]. The aforementioned researchers and their research achievements have made considerable contributions to the application of neural networks to time-varying problems. However, when facing computationally large-scale situations, substantial time is required to calculate the results. Zhang *et al.* [47] theoretically proved that the residual errors of ZNN are upper bounded when solving large-scale computations but cannot converge to zero. The design parameter of ZNN must be set as large as possible theoretical solutions must be tracked, which is clearly not practical in real situations.

Under the aforementioned background, we aim to exploit an effective computational method to solve TVQP problems. A novel power-type varying-parameter RNN (VPNN) is designed and proposed in this paper to satisfy this high demand requirement. A graphical representation of the development of the VPNN model for solving TVQP problems constrained by equality (TVQP-E) is shown in Fig. 1. First, the TVQP-E problem is constructed and a series of optimizations are performed. Specifically, the Lagrangian method [48] is implemented to combine the objective function and constraints of TVQP-E problems into a Lagrange function, which can be rewritten in standard matrix form. A nonnegative slack variable is introduced for TVQP problems constrained by inequality (TVQP-I). On the basis of the Karush–Kuhn–Tucker (KKT) conditions [49], the same matrix form as that of TVQP-E can be obtained, and an important error function is defined according to this matrix equation. An implicit neural dynamics that contains a varying parameter is designed based on our previous research [47] and inspired by GNN and ZNN. In terms of the neural dynamics function, the block diagram and the topological graph are thoroughly discussed for the physical implementation of the proposed neural network. In contrast to the existing ZNN model, which achieves only an exponential convergence rate [44], [45], VPNN achieves rapid convergence. Theoretical proofs demonstrate that the convergence rate of VPNN is superexponential due to the inclusion of the varying parameter, which will be discussed in detail in Sections IV and V.

C. Subtopic 3: Applications to Robots

The application of neural networks in the control filed has been studied for decades. With the prosperity of robotics in recent years, increasingly advanced control algorithms are needed for precise and flexible robot operations. RNN, which possesses powerful computational capabilities, has become one of the most important control methods [50].

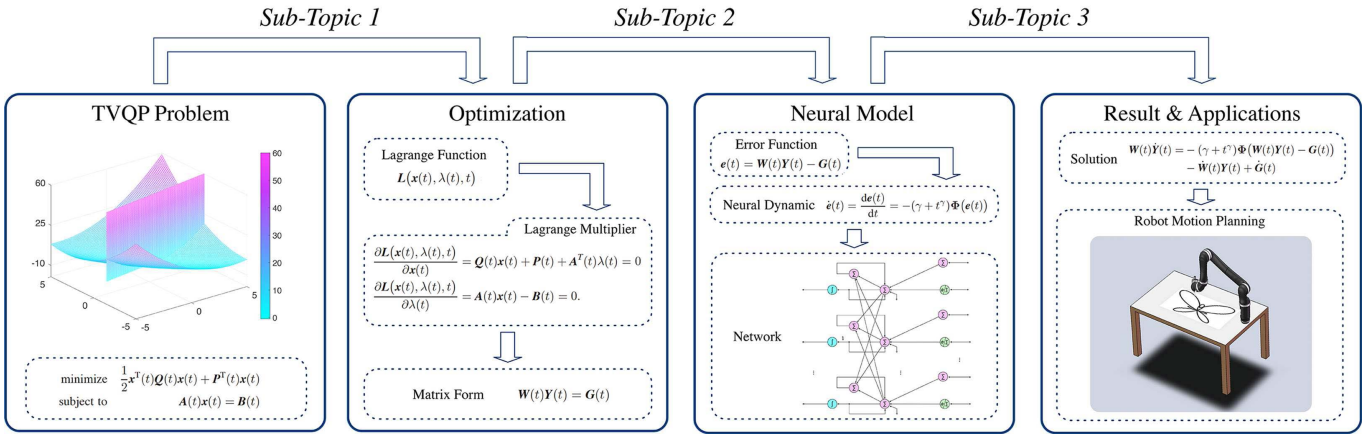

Fig. 1. Graphical representation of the development of the proposed VPNN model for solving TVQP-E.

Tang and Wang [51] utilized the PDNN method to solve the kinematic control problem of redundant manipulators. Zhang *et al.* [52] designed a repetitive motion planning (RMP) scheme based on linear variational inequalities attached to PDNN and applied the scheme to control a PUMA560 manipulator. Furthermore, in terms of the RMP scheme, three RNNs and three numerical methods were applied to a Kinova manipulator in [6], proving the superiority of neural dynamics methods. Li *et al.* [53] proposed a distributed RNN method based on the Nash equilibrium for cooperative control of robot manipulators. Jin *et al.* [54] studied the optimization of the manipulability of redundant robot manipulators by using neural dynamics, and they carefully compared the advantages of feedforward neural networks, dual neural networks, and other RNNs for solving robot control problems [55]. A fully connected RNN model using multiobjective continuous ant colony optimization was proposed in [56] for gait generation of an NAO robot. For the control planning of parallel robots, Chen and Zhang proposed an approach based on ZNN dynamics to protect against the superposition of noise [16].

In view of the related work, a practical application of VPNN for robot control is considered by referring to [43], [52]–[54], and [58]. Specifically, given the position and orientation of a robot manipulator, the possible joint angles and joint velocities that could be used to obtain the given position and orientation are calculated [57]. This problem is called kinematics, and the task of motion planning of robot manipulators is constructed under a TVQP framework; hence, the joint angles and the joint velocities of the end-effector can be precisely solved by the proposed VPNN. In this paper, an RMP scheme based on the VPNN is proposed for the kinematics resolution of a six degrees-of-freedom (DOF) Kinova JACO^2 robot manipulator. On the basis of the TVQP framework, the applications verify the feasibility and effectiveness of the VPNN model, and a practical experiment is conducted to help the readers thoroughly understand the proposed method.

The remainder of this paper is organized as follows. Section II presents the problem formulation of TVQP problems. In Section III, the VPNN model is proposed and analyzed, and the ZNN model is also presented for comparison. The convergence performance of VPNN which is activated by six commonly used activation functions is analyzed in Section IV in detail. Computer simulations of VPNN for solving TVQP problems are presented in Section V. In Section VI, we analyze the practical application of a robot tracking task by using the VPNN model. Section VII presents the conclusion.

The main contributions of this paper are listed as follows.

1) A power-type VPNN is proposed to solve TVQP problems in real time.
2) To the best of our knowledge, this paper is the first time that this type of neural model has been proposed to solve TVQP-E and TVQP-I.
3) The exponential convergence performance of VPNN with six commonly used activation functions is discussed and analyzed in detail. Mathematically, the residual errors of VPNN are proved converge to zero in a superexponential manner.
4) Detailed comparative simulations of VPNN and the state-of-the-art method confirm the superiority of the proposed model.
5) A practical example of a robot tracking problem illustrates the effectiveness, accuracy, and practicability of the proposed VPNN method.

II. Problem Formulation

A. TVQP-E Problem

The standard form of the TVQP-E problem is described as

$$\begin{aligned}&\min\ \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\boldsymbol{x}(t)+\boldsymbol{P}^{\mathrm{T}}(t)\boldsymbol{x}(t)\\&\text{s.t. } \boldsymbol{A}(t)\boldsymbol{x}(t)=\boldsymbol{B}(t)\end{aligned}\tag{1}$$

where vector $\boldsymbol{x}(t)\in\mathbb{R}^n$ at time instant $t\in[0,+\infty)$ is unknown and to be solved in real time, $\boldsymbol{Q}(t)\in\mathbb{R}^{n\times n}$ denotes the positive-definite Hessian matrix, $\boldsymbol{P}(t)\in\mathbb{R}^n$ denotes the coefficient vector, $\boldsymbol{A}(t)\in\mathbb{R}^{m\times n}$ denotes full-rank coefficient matrix, and $\boldsymbol{B}(t)\in\mathbb{R}^n$ denotes a coefficient vector. Moreover, coefficient matrices $\boldsymbol{Q}(t)$ and $\boldsymbol{A}(t)$ and vectors $\boldsymbol{P}(t)$ and $\boldsymbol{B}(t)$, together with their time derivatives $\dot{\boldsymbol{Q}}(t)$, $\dot{\boldsymbol{A}}(t)$, $\dot{\boldsymbol{P}}(t)$, and $\dot{\boldsymbol{B}}(t)$, are assumed to be known and smoothly time varying or can be estimated accurately. To guarantee the uniqueness of the solution, such a TVQP-E problem (1) should be strictly

convex with positive-definite $\boldsymbol{Q}(t) \in \mathbb{R}^{n\times n}$ at any time instant $t \in [0, +\infty)$ [49]. To obtain the solution to the TVQP-E problem (1), a solution algorithm whose errors between the optimal solution and the state solution converge to zero as t approaches ∞ must be designed.

To solve the TVQP-E problem (1), a Lagrange form of this problem is constructed as follows:

$$\boldsymbol{L}(\boldsymbol{x}(t), \lambda(t), t) = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\boldsymbol{x}(t) + \boldsymbol{P}^{\mathrm{T}}(t)\boldsymbol{x}(t) + \lambda^{\mathrm{T}}(t)(\boldsymbol{A}(t)\boldsymbol{x}(t) - \boldsymbol{B}(t)), \quad t \in [0, +\infty) \tag{2}$$

where $\lambda(t) \in \mathbb{R}^m$ denotes the vector of the Lagrange multiplier.

Lemma 1 (Lagrangian Multiplier): For the TVQP-E problem (1), in the case of the existence and continuity of $\partial \boldsymbol{L}(\boldsymbol{x}(t), \lambda(t), t)/\partial \boldsymbol{x}(t)$ and $\partial \boldsymbol{L}(\boldsymbol{x}(t), \lambda(t), t)/\partial \lambda(t)$, the optimum solutions will be obtained when the following two equations are established:

$$\frac{\partial \boldsymbol{L}(\boldsymbol{x}(t), \lambda(t), t)}{\partial \boldsymbol{x}(t)} = \boldsymbol{Q}(t)\boldsymbol{x}(t) + \boldsymbol{P}(t) + \boldsymbol{A}^{\mathrm{T}}(t)\lambda(t) = 0 \tag{3}$$

$$\frac{\partial \boldsymbol{L}(\boldsymbol{x}(t), \lambda(t), t)}{\partial \lambda(t)} = \boldsymbol{A}(t)\boldsymbol{x}(t) - \boldsymbol{B}(t) = 0. \tag{4}$$

Proof: See [48]. ∎

The above two equations can be further rewritten into a matrix form as follows:

$$\boldsymbol{W}(t)\boldsymbol{Y}(t) = \boldsymbol{G}(t) \tag{5}$$

where

$$\begin{aligned} \boldsymbol{W}(t) &:= \begin{bmatrix} \boldsymbol{Q}(t) & \boldsymbol{A}^{\mathrm{T}}(t) \\ \boldsymbol{A}(t) & \boldsymbol{0}_{m\times m} \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)} \\ \boldsymbol{Y}(t) &:= \begin{bmatrix} \boldsymbol{x}(t) \\ \lambda(t) \end{bmatrix} \in \mathbb{R}^{n+m} \\ \boldsymbol{G}(t) &:= \begin{bmatrix} -\boldsymbol{P}(t) & \boldsymbol{B}(t) \end{bmatrix} \in \mathbb{R}^{n+m}. \end{aligned} \tag{6}$$

$\boldsymbol{W}(t)$ and $\boldsymbol{G}(t)$ are a smoothly time-varying coefficient matrix and vector due to the smoothness and continuation of time-varying coefficient matrices $\boldsymbol{Q}(t)$ and $\boldsymbol{A}(t)$ and vector $\boldsymbol{B}(t)$. $\boldsymbol{Y}(t) \in \mathbb{R}^{(n+m)}$ denotes an unknown vector, and it needs to be solved at any time instant t. According to Lemma 1, solving the TVQP-E problem (1) is equivalent to solve the matrix equation (5). Since this problem (1) is time varying (i.e., the coefficient vectors and matrices are changing as time t passes), the theoretical solutions will continuously change. Moreover, to obtain a better understanding and comparison of the proposed algorithm, the time-varying theoretical solution can be written as

$$\boldsymbol{Y}^*(t) = [\boldsymbol{x}^{*\mathrm{T}}(t), \lambda^{*\mathrm{T}}(t)]^{\mathrm{T}} = \boldsymbol{W}^{-1}(t)\boldsymbol{G}(t) \in \mathbb{R}^{n+m}. \tag{7}$$

B. TVQP-I Problem

Considering that the inequality constraint is involved in the TVQP-E problem (1), i.e.,

$$\begin{aligned} \min\ & \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\boldsymbol{x}(t) + \boldsymbol{P}^{\mathrm{T}}(t)\boldsymbol{x}(t) \\ \text{s.t. } & \boldsymbol{A}(t)\boldsymbol{x}(t) = \boldsymbol{B}(t) \\ & \boldsymbol{K}(t)\boldsymbol{x}(t) \le \boldsymbol{D}(t) \end{aligned} \tag{8}$$

where $\boldsymbol{K}(t) \in \mathbb{R}^{m\times n}$ is a full-rank coefficient matrix, $\boldsymbol{D}(t) \in \mathbb{R}^n$ denotes a coefficient vector, and other variable is consistent with before. The above inequality-constrained problem is called the TVQP-I problem.

On the basis of convex optimization [49], a nonnegative slack term is introduced into the inequality in the constraint of TVQP-I (8), i.e.,

$$\boldsymbol{K}(t)\boldsymbol{x}(t) + \boldsymbol{l}^2(t) - \boldsymbol{D}(t) = 0 \tag{9}$$

where vector $\boldsymbol{l}^2(t) \in \mathbb{R}^p$ is defined as $\boldsymbol{l}^2(t) = \boldsymbol{l}(t) \star \boldsymbol{l}(t)$ and operator $\star$ denotes $\boldsymbol{m} \star \boldsymbol{n} = [m_1 n_1, m_2 n_2, \ldots, m_t n_t] \in \mathbb{R}^t$.

On the basis of [48], a Lagrange function is defined as follows:

$$\begin{aligned} &\boldsymbol{L}(\boldsymbol{x}(t), \jmath_a(t), \jmath_b(t), t) \\ &= \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\boldsymbol{x}(t) \\ &\quad + \boldsymbol{P}^{\mathrm{T}}(t)\boldsymbol{x}(t) + \jmath_a^{\mathrm{T}}(t)(\boldsymbol{A}(t)\boldsymbol{x}(t) - \boldsymbol{B}(t)) \\ &\quad + \jmath_b^{\mathrm{T}}(t)(\boldsymbol{K}(t)\boldsymbol{x}(t) + \boldsymbol{l}^2(t) - \boldsymbol{D}(t)) \end{aligned} \tag{10}$$

where $\jmath_a(t)$ and $\jmath_b(t)$ denote the vectors of the Lagrange multiplier.

The KKT conditions are introduced for (10), i.e.,

$$\boldsymbol{Q}(t)\boldsymbol{x}(t) + \boldsymbol{P}(t) + \jmath_a(t)\boldsymbol{A}^{\mathrm{T}}(t) + \jmath_b(t)\boldsymbol{K}^{\mathrm{T}}(t) = 0 \tag{11}$$

$$\boldsymbol{A}(t)\boldsymbol{x}(t) - \boldsymbol{B}(t) = 0 \tag{12}$$

$$\boldsymbol{K}(t)\boldsymbol{x}(t) + \boldsymbol{l}^2(t) - \boldsymbol{D}(t) = 0 \tag{13}$$

$$\jmath_b(t) \star \boldsymbol{l}(t) = 0. \tag{14}$$

The above equations can be formulated as a matrix form as in (5), i.e.,

$$\boldsymbol{W}(t)\boldsymbol{Y}(t) = \boldsymbol{G}(t) \tag{15}$$

where

$$\begin{aligned} \boldsymbol{W}(t) &:= \begin{bmatrix} \boldsymbol{Q}(t) & \boldsymbol{A}^{\mathrm{T}}(t) & \boldsymbol{K}^{\mathrm{T}}(t) & \boldsymbol{0} \\ \boldsymbol{A}(t) & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{K}(t) & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{l}(t) \\ \boldsymbol{0} & \boldsymbol{0} & 2\tilde{\boldsymbol{l}}(t) & \tilde{\jmath}_b(t) \end{bmatrix} \\ \boldsymbol{Y}(t) &:= [\boldsymbol{x}(t) \quad \jmath_a(t) \quad \jmath_b(t) \quad \boldsymbol{l}(t)]^{\mathrm{T}} \\ \boldsymbol{G}(t) &:= [-\boldsymbol{P}(t) \quad \boldsymbol{B}(t) \quad \boldsymbol{D}(t) \quad \boldsymbol{0}]^{\mathrm{T}}. \end{aligned} \tag{16}$$

Diagonal matrices $\tilde{\boldsymbol{l}}(t)$ and $\tilde{\jmath}_b(t)$ are defined as follows:

$$\tilde{\boldsymbol{l}}(t) = \begin{bmatrix} l_1(t) & 0 & \cdots & 0 \\ 0 & l_2(t) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & l_n(t) \end{bmatrix} \tag{17}$$

$$\tilde{\jmath}_b(t) = \begin{bmatrix} \jmath_{b1}(t) & 0 & \cdots & 0 \\ 0 & \jmath_{b2}(t) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \jmath_{bn}(t) \end{bmatrix}. \tag{18}$$

Following the same procedure, the time-varying theoretical solution of the TVQP-I problem (8) is:

$$\boldsymbol{Y}^*(t) = [\boldsymbol{x}^{*\mathrm{T}}(t), \lambda^{*\mathrm{T}}(t)]^{\mathrm{T}} = \boldsymbol{W}^{-1}(t)\boldsymbol{G}(t) \in \mathbb{R}^{n+m}. \tag{19}$$

Therefore, the design process of the neural model for solving TVQP-I problems follows the same procedure as that used for TVQP-E problems.

III. Neural Models

The design process of the proposed VPNN for solving TVQP problems is discussed and analyzed in detail in this section. For comparisons and illustration, the traditional ZNN model is also presented.

A. VPNN Model

To obtain the optimum of matrix equations (5) and (15), a vector-type error function is defined as

$$\boldsymbol{e}(t) = \boldsymbol{W}(t)\boldsymbol{Y}(t) - \boldsymbol{G}(t). \tag{20}$$

According to our previous neural dynamic design experience [47], the negative time derivative of error function $\boldsymbol{e}(t)$ is necessary to make this error function $\boldsymbol{e}(t)$ approach zero. For this reason, a power-type varying-parameter neural dynamic design formula is described as

$$\dot{\boldsymbol{e}}(t) = \frac{\mathrm{d}\boldsymbol{e}(t)}{\mathrm{d}t} = -(\gamma + t^{\gamma})\Phi(\boldsymbol{e}(t)) \tag{21}$$

where $\gamma > 0$ denotes the constant scalar-valued design parameter, and the design parameter $\gamma + t^{\gamma}$ is used to scale the convergence rate of the formula. In contrast to the traditional neural dynamic design approach, the design parameter $\gamma + t^{\gamma}$ is a function of time t, and the approach is thus termed as a varying-parameter neural dynamic design method. $\Phi(\cdot)$ is the activation-function processing array. In addition, each scalar-valued processing unit $\phi(\cdot)$ of activation-function processing array $\Phi(\cdot)$ should be a monotonically increasing odd activation function. In this paper, six commonly used activation functions are applied and analyzed, which are described as follows.

1) *Linear-Type Activation Function (Abbreviated AFlinear)*:

$$\phi_1(\boldsymbol{u}) = \boldsymbol{u}.$$

2) *Power-Type Activation Function (Abbreviated AFpower)*:

$$\phi_2(\boldsymbol{u}) = \boldsymbol{u}^{\omega} \quad \text{with} \omega > 1.$$

3) *Bipolar-Sigmoid-Type Activation Function (Abbreviated AFb-Sigmoid)*:

$$\phi_3(\boldsymbol{u}) = \frac{(1+\exp(-\mu))(1-\exp(-\mu\boldsymbol{u}))}{(1-\exp(-\mu))(1+\exp(-\mu\boldsymbol{u}))} \quad \text{with} \mu \geqslant 2.$$

4) *Sinh-Type Activation Function (Abbreviated AFsinh)*:

$$\phi_4(\boldsymbol{u}) = \frac{\exp(\boldsymbol{u}) - \exp(-\boldsymbol{u})}{2}.$$

5) *Tanh-Type Activation Function (Abbreviated AFtanh)*:

$$\phi_5(\boldsymbol{u}) = \frac{\exp(\boldsymbol{u}) - \exp(-\boldsymbol{u})}{\exp(\boldsymbol{u}) + \exp(-\boldsymbol{u})}.$$

6) *Tunable-Type Activation Function (Abbreviated AFtunable)*:

$$\phi_6(\boldsymbol{u}) = \mathrm{sig}^{r}(\boldsymbol{u}) + \mathrm{sig}(\boldsymbol{u}) + \mathrm{sig}^{\frac{1}{r}}(\boldsymbol{u})$$

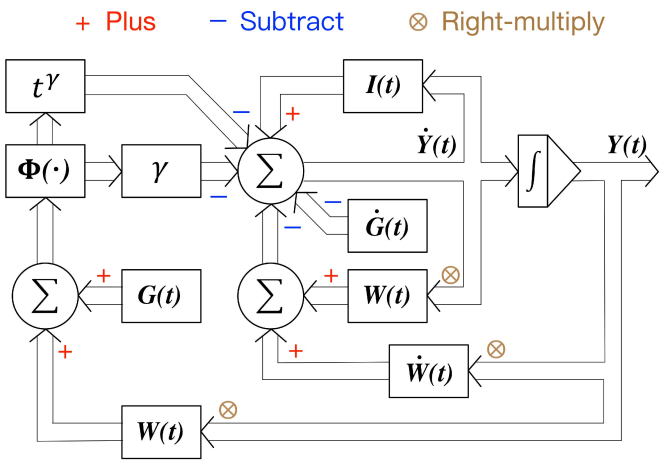

Fig. 2. Block diagram of VPNN.

with $r > 0$ and $r \neq 1$. Function $\mathrm{sig}^{r}(\boldsymbol{u})$ is defined as

$$\mathrm{sig}^{r}(\boldsymbol{u}) = \begin{cases} |\boldsymbol{u}|^{r}, & \text{if} \boldsymbol{u} > 0 \\ 0, & \text{if and only if} \boldsymbol{u} = 0 \\ -|\boldsymbol{u}|^{r}, & \text{if} \boldsymbol{u} < 0 \end{cases} \tag{22}$$

where $|\boldsymbol{u}|$ denotes the absolute value of $\boldsymbol{u} \in \mathbb{R}$.

Since the constant scalar-valued designed parameter $\gamma > 0$, parameter $(\gamma + t^{\gamma}) \to +\infty$ when time $t \to +\infty$. The detailed convergence proof is presented in Theorem 1.

Substituting (20) into (21), the implicit dynamic equation of VPNN is shown as

$$\begin{aligned} \boldsymbol{W}(t)\dot{\boldsymbol{Y}}(t) = &-(\gamma + t^{\gamma})\Phi(\boldsymbol{W}(t)\boldsymbol{Y}(t) - \boldsymbol{G}(t)) \\ &-\dot{\boldsymbol{W}}(t)\boldsymbol{Y}(t) + \dot{\boldsymbol{G}}(t) \end{aligned} \tag{23}$$

where $\dot{\boldsymbol{W}}(t) = \mathrm{d}\boldsymbol{W}(t)/\mathrm{d}t$, $\dot{\boldsymbol{Y}}(t) = \mathrm{d}\boldsymbol{Y}(t)/\mathrm{d}t$, and $\dot{\boldsymbol{G}}(t) = \mathrm{d}\boldsymbol{G}(t)/\mathrm{d}t$. According to the previously mentioned definition of $\boldsymbol{Y}(t) \in \mathbb{R}^{n+m}$, we have

$$\begin{aligned} \boldsymbol{Y}(t) &:= [\boldsymbol{x}^{\mathrm{T}}(t), \lambda^{\mathrm{T}}(t)]^{\mathrm{T}} \\ &= [x_1(t), x_2(t), \ldots, x_n(t), \lambda_1(t), \lambda_2(t), \ldots, \lambda_m(t)]^{\mathrm{T}} \end{aligned} \tag{24}$$

which is an unknown matrix and needs to be solved at any time instant t. Moreover, the initial state of matrix $\boldsymbol{Y}(t)$ is set as $\boldsymbol{Y}(0) \in \mathbb{R}^{n+m}$.

Since the designed parameter $\gamma + t^{\gamma}$ is changing with time t, the model based on (23) is called a power-type VPNN. The block diagram of VPNN, which can be physically implemented for the neural network, is shown in Fig. 2.

Rewriting (23) leads to the following result:

$$\begin{aligned} \dot{\boldsymbol{Y}}(t) = &(\boldsymbol{I}(t) - \boldsymbol{W}(t))\dot{\boldsymbol{Y}}(t) - \dot{\boldsymbol{W}}(t)\boldsymbol{Y}(t) \\ &-(\gamma + t^{\gamma}) \cdot \Phi(\boldsymbol{W}(t)\boldsymbol{Y}(t) - \boldsymbol{G}(t)) + \dot{\boldsymbol{G}}(t) \end{aligned} \tag{25}$$

where $\boldsymbol{I}(t)$ denotes the identity matrix.

The ith neural dynamics of VPNN (23) is

$$\begin{aligned} \dot{Y}_i = &\sum_{j=1}^{n+m}(I_{ij} - W_{ij})\dot{Y}_i - \sum_{j=1}^{n+m}\dot{W}_{ij}Y_j - (\gamma + t^{\gamma}) \\ &\cdot\phi\left(\sum_{j=1}^{n+m}W_{ij}Y_j - G_i\right) + \dot{G}_i \end{aligned} \tag{26}$$

where $\phi(\cdot)$ is the scalar-valued processing unit of $\Phi(\cdot)$.

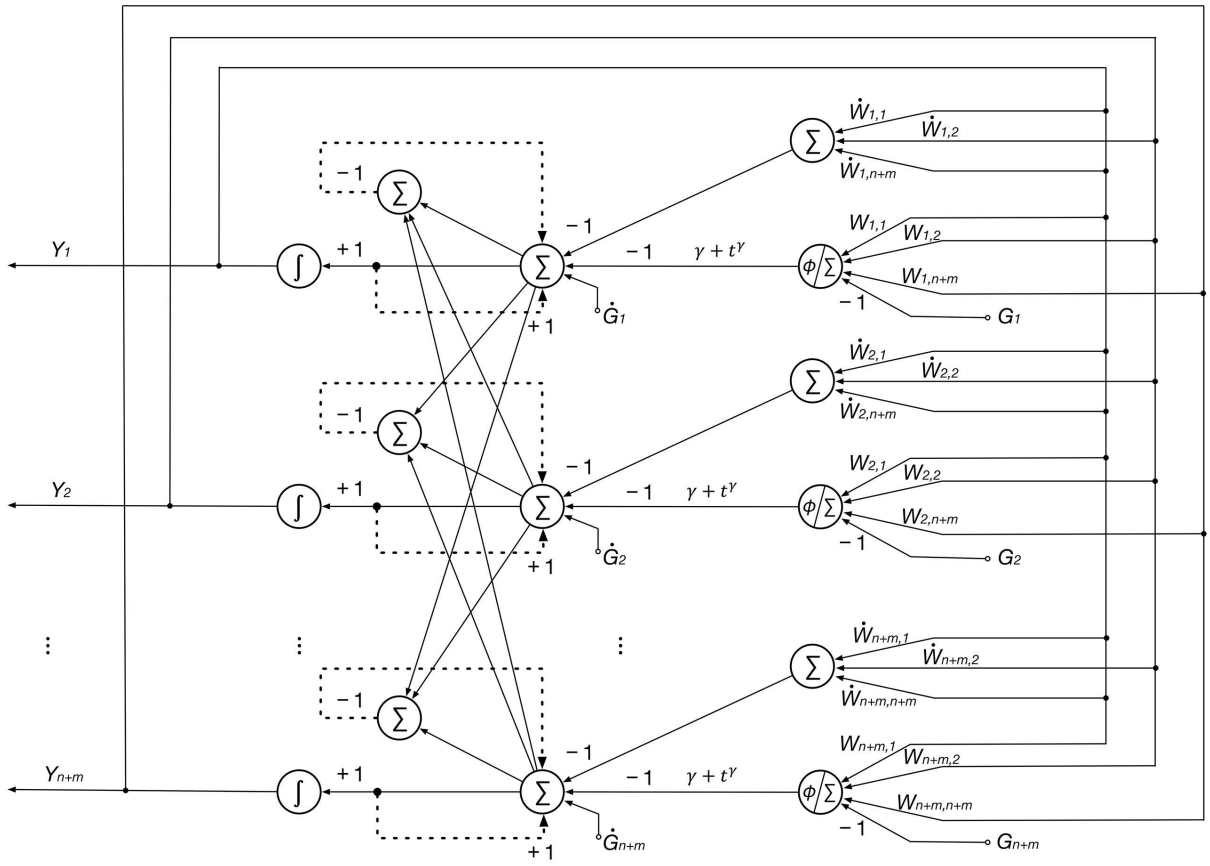

Fig. 3. Neural topological graph of VPNN.

Based on (26), the topological graph of VPNN (23) is shown in Fig. 3.

B. ZNN Model

In the traditional neural dynamic design method, the design parameter is set as a constant, i.e., it is fixed; thus, it is called the fixed-parameter neural dynamic design method. By omitting time term t^{γ}, the VPNN (23) degrades into the ZNN model, i.e.,

$$\boldsymbol{W}(t)\dot{\boldsymbol{Y}}(t) = -\gamma\,\Phi(\boldsymbol{W}(t)\boldsymbol{Y}(t)-\boldsymbol{G}(t))-\dot{\boldsymbol{W}}(t)\boldsymbol{Y}(t)+\dot{\boldsymbol{G}}(t) \quad (27)$$

where the parameters are defined the same as those of VPNN.

Remark 1 (Difference Between VPNN and ZNN): Mathematically, from (23) and (27), we state that the VPNN would degrade into the ZNN when omitting time-varying parameter t^{γ}. The VPNN can be considered the general form of ZNN, and ZNN can be considered as a particular case of VPNN with $t = 0$. Such a difference leads to the following distinctions.

1) The design parameter of VPNN takes time variable t into consideration, whereas ZNN does not. In other words, the convergent parameter of the ZNN is fixed, while the proposed VPNN is time varying.
2) Due to the influence of time-varying parameters, the design parameter γ only needs to be set to a small value, and the practical performance of the VPNN would be good.
3) The theoretical analysis and mathematical proof show that the robustness of the VPNN is much better than that of ZNN. Specifically, when solving optimization problems, the convergence rate of ZNN is exponential, whereas the rate of the proposed VPNN is superexponential.

IV. Theoretical Analysis

In this section, the convergence performance of the VPNN model is mathematically discussed and analyzed. In addition, comparison results of convergence speed between the traditional ZNN and the proposed VPNN are illustrated. Note that solving convex TVQP-E (1) and TVQP-I (8) problems is equivalent to solving the matrix forms (5) and (15), i.e., $\boldsymbol{W}(t)\boldsymbol{Y}(t) = \boldsymbol{G}(t)$. Discussing the solution to TVQP problems is equivalent to discussing the solutions to (5) and (15).

Theorem 1 (Global Convergence Theorem): Considering the TVQP-E (1) and TVQP-I (8) problems, if a monotonically increasing odd activation function array $\Phi(\cdot)$ is used, then the state variable $\boldsymbol{Y}(t) = [\boldsymbol{x}^{\mathrm{T}}(t), \lambda^{\mathrm{T}}(t)]^{\mathrm{T}}$ of the VPNN model (23), starting from any initial state $\boldsymbol{Y}(0) \in \mathbb{R}^{n+m}$, globally converges to the unique theoretical solution (19), i.e., $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^{*\mathrm{T}}(t), \lambda^{*\mathrm{T}}(t)]^{\mathrm{T}}$, as well as $\lim_{t\to+\infty}(\boldsymbol{Y}(t) - \boldsymbol{Y}^*(t)) = 0$. Furthermore, the theoretical solution $\boldsymbol{x}^*(t)$ to TVQP problems is the first n elements of $\boldsymbol{Y}^*(t)$.

Proof: A Lyapunov function candidate is defined as

$$\boldsymbol{V}(t) = \frac{\|\boldsymbol{e}(t)\|_2^2}{2} = \frac{\boldsymbol{e}^{\mathrm{T}}(t)\boldsymbol{e}(t)}{2} \geq 0 \quad (28)$$

where $\boldsymbol{e}(t)$ is defined as $\boldsymbol{e}(t) = \boldsymbol{W}(t)\boldsymbol{Y}(t) - \boldsymbol{G}(t)$ and $\|\cdot\|_2$ denotes the two norm of a vector. The time derivative of

Lyapunov function $\boldsymbol{V}(t)$ is

$$\dot{\boldsymbol{V}}(t) = \frac{\mathrm{d}\boldsymbol{V}(t)}{\mathrm{d}t} = \boldsymbol{e}^{\mathrm{T}}(t)\frac{\mathrm{d}\boldsymbol{e}(t)}{\mathrm{d}t} = \boldsymbol{e}^{\mathrm{T}}(t)\dot{\boldsymbol{e}}(t). \tag{29}$$

Substituting (21) into (29), we have

$$\begin{aligned}\dot{\boldsymbol{V}}_{\mathrm{VP}}(t) &= -(\gamma + t^{\gamma})\boldsymbol{e}^{\mathrm{T}}(t)\Phi(\boldsymbol{e}(t)) \\ &= -(\gamma + t^{\gamma})\sum_{i=1}^{n+m} e_i(t)\phi(e_i(t))\end{aligned} \tag{30}$$

where $\phi(\cdot)$ denotes the element of activation function vector $\Phi(\cdot)$. Since $\gamma > 0$ and $t > 0$, the monotonic singular odd activation function $\phi(\cdot)$ guarantees the following conditions.

1) If $e_i(t) > 0$ or $e_i(t) < 0$, then $e_i(t)\phi(e_i(t)) > 0$ and $\dot{\boldsymbol{V}}_{\mathrm{VP}}(t) < 0$.
2) If and only if $e_i(t) = 0$, then $e_i(t)\phi(e_i(t)) = 0$ and $\dot{\boldsymbol{V}}_{\mathrm{VP}}(t) = 0$.

According to [58], since Lyapunov candidate $\boldsymbol{V}(t)$ is positive semidefinite and its derivative $\dot{\boldsymbol{V}}(t)$ is negative definite, we conclude that $\boldsymbol{Y}(t) - \boldsymbol{Y}^*(t)$ converges to zero globally, i.e., $\boldsymbol{x}(t) - \boldsymbol{x}^*(t)$ converges to zero. The proof is thus completed. ■

Remark 2: For comparison, the differential form $\dot{\boldsymbol{V}}_{\mathrm{Z}}(t)$ of Lyapunov candidate $\boldsymbol{V}(t)$ of the ZNN model (27) is as follows:

$$\dot{\boldsymbol{V}}_{\mathrm{Z}}(t) = -\gamma\,\boldsymbol{e}^{\mathrm{T}}(t)\Phi(\boldsymbol{e}(t)) = -\gamma\sum_{i=1}^{n+m} e_i(t)\phi(e_i(t)). \tag{31}$$

It can be easily proven that $\dot{\boldsymbol{V}}_{\mathrm{Z}}(t) \leqslant 0$ and that the ZNN model (27) is also convergent.

Theorem 2 (Convergence Time Theorem): Considering two Lyapunov functions $\mathbf{V}_{\mathcal{A}}(t)$ and $\boldsymbol{V}_{\mathcal{B}}(t)$, if the time derivatives of $\boldsymbol{V}_{\mathcal{A}}(t)$ and $\boldsymbol{V}_{\mathcal{B}}(t)$ (i.e., $\dot{\boldsymbol{V}}_{\mathcal{A}}(t)$ and $\dot{\boldsymbol{V}}_{\mathcal{B}}(t)$) satisfy $\dot{\boldsymbol{V}}_{\mathcal{A}}(t) - \dot{\boldsymbol{V}}_{\mathcal{B}}(t) > 0$, then the convergence time from the same initial value $e_{\mathcal{A}}(0) = e_{\mathcal{B}}(0)$ satisfies $\mathrm{T}_{\mathcal{A}} > \mathrm{T}_{\mathcal{B}}$. Moreover, the residual errors satisfy $\|e_{\mathcal{A}}(t)\|_2 > \|e_{\mathcal{B}}(t)\|_2$ for $t \in (0, +\infty)$.

Proof: According to the definition of the Lyapunov function in Theorem 1, computing the residual error $\boldsymbol{e}(t)$ is equivalent to computing the Lyapunov function $\boldsymbol{V}(t)$. To facilitate the following discussion, a scalar-type Lyapunov function of $e_i(t)$ is defined as $v_i(t) = e_i^2(t)/2$, where $e_i(t)$ is the ith element of $\boldsymbol{e}(t)$.

Considering the same $e_i(t)$ of Lyapunov functions $v_{\mathcal{A}}(t)$ and $v_{\mathcal{B}}(t)$ and the residual errors $e_{\mathcal{A}}(t)$ and $e_{\mathcal{B}}(t)$ with the same starting point $e_{\mathcal{A}}(0) = e_{\mathcal{B}}(0)$, there always exists $e_{\mathcal{A}}(t_{a1}) = e_{\mathcal{B}}(t_{b1}) = \tilde{\chi}$ and $v_{\mathcal{A}}(t_{a1}) = v_{\mathcal{B}}(t_{b1}) = \tilde{\chi}^2/2$. In other words, considering $e_{\mathcal{B}}$ at time instant t_{b1} is equivalent to considering $e_{\mathcal{A}}$ at time instant t_{a1}. Under this condition, we have

$$\begin{aligned}&\dot{v}_{\mathcal{A}}(t) - \dot{v}_{\mathcal{B}}(t) \\ &\quad= -\gamma\, e_{\mathcal{A}}(t_{a1})\phi(e_{\mathcal{A}}(t_{a1})) + \left(\gamma + t_{b1}^{\gamma}\right)e_{\mathcal{B}}(t_{b1})\phi(e_{\mathcal{B}}(t_{b1})) \\ &\quad= t_{b1}^{\gamma}\tilde{\chi}\phi(\tilde{\chi}) > 0.\end{aligned} \tag{32}$$

It means that $\dot{v}_A(t_{a1}) > \dot{v}_{\mathcal{B}}(t_{b1})$. In the next moment, we have $e_{\mathcal{A}}(t_{a2}) = e_{\mathcal{B}}(t_{b2}) = \tilde{\chi}'$ and $v_{\mathcal{A}}(t_{a2}) = v_{\mathcal{B}}(t_{b2}) = (\tilde{\chi}')^2/2 = \tilde{\chi}^2/2 + \Delta\tilde{\chi}$, where $\Delta\tilde{\chi} \to 0$ is the step length from $\tilde{\chi}^2/2$ to $(\tilde{\chi}')^2/2$. Due to $\dot{v}_{\mathcal{B}}(t) - \dot{v}_{\mathcal{A}}(t) > 0$, the time cost from $\tilde{\chi}^2/2$ to $(\tilde{\chi}')^2/2$ satisfies $t_{a2} - t_{a1} = \Delta t_{\mathcal{A}} > t_{b2} - t_{b1} = \Delta t_{\mathcal{B}}$.

The convergence time from the same initial value $v(0)$ to $v(t)$ possesses the following relationship:

$$\int_{v(0)}^{v(t)} \Delta t_{\mathcal{A}} = \mathrm{T}_{\mathcal{A}} > \mathrm{T}_{\mathcal{B}} = \int_{v(0)}^{v(t)} \Delta t_{\mathcal{B}}. \tag{33}$$

That is, when $e_{\mathcal{A}}(t) = e_{\mathcal{B}}(t)$, the convergence time $\mathrm{T}_{\mathcal{A}} > \mathrm{T}_{\mathcal{B}}$. In other words, the residual errors satisfy $e_{\mathcal{B}}(t) > e_{\mathcal{A}}(t)$ when they work with the same length of time. With $e_{\mathcal{A}}(0) = e_{\mathcal{B}}(0) = 0$, we can conclude that

$$\|e_{\mathcal{A}}(t)\|_2 > \|e_{\mathcal{B}}(t)\|_2 \forall t > 0. \tag{34}$$

The proof is thus completed. ■

Theorem 3 (Residual Error Theorem): Considering the TVQP-E (1) and TVQP-I (8) problems, assuming that ZNN (27) and VPNN (23) have the same initial state (i.e., the same parameter and starting point), the residual error of VPNN is always smaller than that of ZNN when they are applied to solving (5) and (15), as well as the TVQP-E (1) and TVQP-I (8) problems.

Proof: Based on Theorems 1 and 2, considering the same $e_i(t)$ of Lyapunov functions $v_{\mathrm{Z}}(t)$ and $v_{\mathrm{VP}}(t)$ that are starting from the same initial state $e_i(0)$, there exists $e_{\mathrm{Z}}(t_{c1}) = e_{\mathrm{VP}}(t_{d1}) = \chi$ and $v_{\mathrm{Z}}(t_{c1}) = v_{\mathrm{VP}}(t_{d1}) = \chi^2/2$, where $e_{\mathrm{Z}}(t_{c1})$ and $e_{\mathrm{VP}}(t_{d1})$ denote the residual errors $e_i(t)$ solved by ZNN and VPNN, respectively. Then, we have $\dot{v}_{\mathrm{Z}}(t) - \dot{v}_{\mathrm{VP}}(t) = t_{d1}^{\gamma}\chi\phi(\chi) > 0$, which means that $\dot{v}_{\mathrm{Z}}(t_{c1}) > \dot{v}_{\mathrm{VP}}(t_{d1})$, and the time cost from $\chi^2/2$ to $(\chi')^2/2$ satisfies $t_{c2} - t_{c1} = \Delta t_{\mathrm{Z}} > t_{d2} - t_{d1} = \Delta t_{\mathrm{VP}}$. After integration, the convergence time from the same initial value $v_i(0)$ to $v_i(t)$ satisfies $\mathrm{T}_{\mathrm{Z}} > \mathrm{T}_{\mathrm{VP}}$, $\forall i \in \{1, 2, \ldots, m+n\}$.

That is, when $e_{\mathrm{Z}}(t) = e_{\mathrm{VP}}(t)$, the convergence time of VPNN is shorter than that of ZNN. In other words, $e_{\mathrm{Z}}(t)$ is always greater than $e_{\mathrm{VP}}(t)$ when they work with the same length of time. With $e_{\mathrm{Z}}(0) = e_{\mathrm{VP}}(0)$, we can conclude that $\|e_{\mathrm{Z}}(t)\|_2 > \|e_{\mathrm{VP}}(t)\|_2$ for $t \in (0, +\infty)$, which indicates that the residual error $e_i(t)$ of VPNN is always smaller than that of ZNN. The proof is thus completed. ■

Theorem 4 (Convergence Theorem With AFlinear): Considering (5) and (15), by using VPNN (23) with a linear activation function $\phi_1(\boldsymbol{u}) = \boldsymbol{u}$, the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^*(t), \lambda^*(t)]^T$ with a superexponential convergence rate.

Proof: First, for comparison, the deviation vector $\tilde{\boldsymbol{Y}}(t) = \boldsymbol{Y}(t) - \boldsymbol{Y}^*(t)$ is defined, and the relationship between $\|\tilde{\boldsymbol{Y}}(t)\|_2$ and $\|\boldsymbol{e}(t)\|_2$ is presented. According to the matrix theory [59], we have

$$\sqrt{\lambda_{\min}((\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}\|\tilde{\boldsymbol{Y}}(t)\|_2 \leqslant \|\boldsymbol{W}(t)\tilde{\boldsymbol{Y}}(t)\|_2 \tag{35}$$

where $\lambda_{\min}((\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))$ is positive and denotes the minimum eigenvalue of $(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))$ if $\boldsymbol{W}(t)$ is a real matrix. The relationship between $\|\tilde{\boldsymbol{Y}}(t)\|_2$ and $\|\boldsymbol{e}(t)\|_2$ is

$$\begin{aligned}\|\tilde{\boldsymbol{Y}}(t)\|_2 &\leqslant \frac{\|\boldsymbol{W}(t)\tilde{\boldsymbol{Y}}(t)\|_2}{\sqrt{\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}} \\ &= \frac{\|\boldsymbol{W}(t)(\boldsymbol{Y}(t) - \boldsymbol{Y}^*(t))\|_2}{\sqrt{\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}} \\ &= \frac{\|\boldsymbol{e}(t)\|_2}{\sqrt{\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}}.\end{aligned} \tag{36}$$

The ith scalar-type formula of VPNN (23) with AFlinear ϕ_1 is

$$\dot{e}_i(t) = \frac{\mathrm{d}e_i(t)}{\mathrm{d}t} = -(\gamma + t^{\gamma})e_i(t). \tag{37}$$

Based on the differential equation theory in [60], the solution to (37) is

$$e_i(t) = e_i(0)\exp\left(-\left(\gamma t + \frac{1}{\gamma+1}t^{\gamma+1}\right)\right) \tag{38}$$

where $i = 1, 2, 3, \ldots, n+m$.

The vector-type solution is defined as

$$\boldsymbol{e}(t) = \boldsymbol{e}(0)\exp\left(-\left(\gamma t + \frac{1}{\gamma+1}t^{\gamma+1}\right)\right). \tag{39}$$

According to (36) and (39), the following inequality can be obtained:

$$\begin{aligned}\|\tilde{\boldsymbol{Y}}(t)\|_2 &\leqslant \frac{\|\boldsymbol{e}(t)\|_2}{\sqrt{\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}} \\ &= \sqrt{\sum_{i=1}^{n+m}\frac{e_i^2(0)\exp\left(-2\left(\gamma t + \frac{1}{\gamma+1}t^{\gamma+1}\right)\right)}{\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t))}}.\end{aligned} \tag{40}$$

Therefore, we have

$$\lim_{t\to+\infty}\|\tilde{\boldsymbol{Y}}(t)\|_2 = 0. \tag{41}$$

That is, $\tilde{\boldsymbol{Y}}(t)$ globally converges to zero with a superexponential convergence rate, and the state solution $\boldsymbol{Y}(t)$ converges to the unique theoretical solution $\boldsymbol{Y}^*(t)$ when $t \to +\infty$. The proof is thus completed. ∎

Theorem 5 (Convergence Theorem With AFpower): Considering (5) and (15), by using VPNN (23) with a power activation function $\phi_2(\boldsymbol{u}) = \boldsymbol{u}^{\omega}$ (where ω is odd and $\omega > 1$), the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^*(t), \lambda^*(t)]^{\mathrm{T}}$ with a superexponential convergence rate when $|\boldsymbol{e}(t)| \geq 1$ (where $|\cdot|$ denotes the absolute value).

Proof: Substituting AFpower $\phi_2 = u^{\omega}$ into (21) and integrating with time t, the solution to VPNN with AFpower ϕ_2 is formulated as

$$\boldsymbol{e}(t) = \left((1-\omega)\left(-\gamma t - \frac{1}{\gamma+1}t^{\gamma+1}\right) + \boldsymbol{e}^{1-\omega}(0)\right)^{\frac{1}{1-\omega}}. \tag{42}$$

Based on (36), we have $\|\tilde{\boldsymbol{Y}}(t)\|_2 \leqslant \|\boldsymbol{e}(t)\|_2/(\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t)))^{1/2}$, which means that $\|\tilde{\boldsymbol{Y}}(t)\|_2$ has an upper bound via using VPNN (23). Since $\omega > 1$, we have $(1-\omega) < 0$ and $(1/(1-\omega)) < 0$. When time $t \to +\infty$, the first term of (42), i.e., $(1-\omega)(-\gamma t - (1/\gamma+1)t^{\gamma+1})$, also approaches $+\infty$. Hence, with $(1/1-\omega) < 0$, each element of $\boldsymbol{e}(t)$ will converge to zero, which makes $\|\boldsymbol{e}(t)\|_2$ and $\|\tilde{\boldsymbol{Y}}(t)\|_2$ approach zero.

To prove the superexponential convergence, a Lyapunov function candidate is defined as

$$\boldsymbol{V}(t) = \frac{\|\boldsymbol{e}(t)\|_2^2}{2} = \frac{\boldsymbol{e}^{\mathrm{T}}(t)\boldsymbol{e}(t)}{2} \geqslant 0. \tag{43}$$

The time derivative of $\boldsymbol{V}(t)$ is

$$\dot{\boldsymbol{V}}(t) = \frac{\mathrm{d}\boldsymbol{V}(t)}{\mathrm{d}t} = \boldsymbol{e}^{\mathrm{T}}(t)\frac{\mathrm{d}\boldsymbol{e}(t)}{\mathrm{d}t} = \boldsymbol{e}^{\mathrm{T}}(t)\dot{\boldsymbol{e}}(t). \tag{44}$$

Substituting (21) into (44), with AFpower $\phi_2(e_i(t)) = (e_i(t))^{\omega}$, we have

$$\begin{aligned}\dot{\boldsymbol{V}}_{\mathrm{pow}}(t) &= -(\gamma + t^{\gamma})\sum_{i=1}^{n+m}e_i(t)\phi(e_i(t)) \\ &= -(\gamma + t^{\gamma})\sum_{i=1}^{n+m}e_i^{\omega+1}(t).\end{aligned} \tag{45}$$

According to (30), with linear activation $\phi_1(e_i(t)) = e_i(t)$, we can obtain the time derivative of $\boldsymbol{V}(t)$ with AFlinear, i.e.,

$$\dot{\boldsymbol{V}}_{\mathrm{lin}}(t) = -(\gamma + t^{\gamma})\sum_{i=1}^{n+m}e_i^2(t). \tag{46}$$

In the case of $|\boldsymbol{e}(t)| \geq 1$ with $\omega + 1 \geq 2$, we have $\dot{\boldsymbol{V}}_{\mathrm{pow}}(t) \leq \dot{\boldsymbol{V}}_{\mathrm{lin}}(t) \leq 0$ with the same $e_i(t)$ of using AFpower and AFlinear.

According to Theorem 2, the VPNN with AFpower possesses better convergence performance than that with AFlinear. Since the superexponential convergence of VPNN with AFlinear has been proved, VPNN with AFpower $\phi_2(e_i(t)) = (e_i(t))^{\omega}$ (where ω is odd and $\omega > 1$) also achieves superexponential convergence performance. The proof is thus completed. ∎

Theorem 6 (Convergence Theorem With AFb-Sigmoid): Considering (5) and (15), by using VPNN (23) with a bipolar-sigmoid activation function $\phi_3(u) = (1 + \exp(-\mu))(1 - \exp(-\mu\boldsymbol{u})/(1-\exp(-\mu))(1+\exp(-\mu\boldsymbol{u})$, the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^*(t), \lambda^*(t)]^{\mathrm{T}}$ with a superexponential convergence rate when $\mu \geq 2$ and $1 \geq |\boldsymbol{e}(t)| \geq 0$.

Proof: First, according to (36), we have $\|\tilde{\boldsymbol{Y}}(t)\|_2 \leq \|\boldsymbol{e}(t)\|_2/(\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t)))^{1/2}$, which means that $\|\tilde{\boldsymbol{Y}}(t)\|_2$ has an upper bound via using VPNN (23).

Second, with the b-sigmoid activation function, we have

$$\begin{aligned}\dot{e}_i(t) &= \frac{\mathrm{d}e_i(t)}{\mathrm{d}t} = -(\gamma + t^{\gamma})\phi_3(e_i(t)) \\ &= -(\gamma + t^{\gamma})\frac{(1+\exp(-\mu))(1-\exp(-\mu e_i(t)))}{(1-\exp(-\mu))(1+\exp(-\mu e_i(t)))}.\end{aligned} \tag{47}$$

Based on [60], the vector form of solutions to (47) is

$$\begin{aligned}\boldsymbol{e}(t) = \frac{1}{\mu}\ln\Bigg(&1 + \frac{1}{2\mathcal{C}_1}\exp(-\mu\Upsilon\varpi) \\ &\pm \frac{1}{2}\sqrt{\left(\frac{1}{\mathcal{C}_1}\exp(-\mu\Upsilon\varpi) + 2\right)^2 - 4}\Bigg)\end{aligned} \tag{48}$$

where $\Upsilon = (1 + \exp(-\mu))/(1 - \exp(-\mu))$, $\varpi = t^{\gamma+1}/(\gamma + 1) + \gamma t$, and $\mathcal{C}_1 = 1/(\exp(\mu\varepsilon(0)) + \exp(-\mu\varepsilon(0)) - 2)$ is a constant term associated with initial value $\boldsymbol{e}(0)$.

From (36) and (48), we find that the upper bound of $\|\tilde{\boldsymbol{Y}}(t)\|_2$ is determined by ϖ.

Since $\gamma > 1$, when $t \to +\infty$, ϖ will approach $+\infty$. Therefore, each element of $\boldsymbol{e}(t)$ in (48) will converge to zero, which makes $\|\boldsymbol{e}(t)\|_2$ and $\|\tilde{\boldsymbol{Y}}(t)\|_2$ in (36) approach zero.

Third, to prove the superexponential convergence rate with the same $e_i(t)$ when using AFlinear and AFb-sigmoid,

the error between the time derivatives of the Lyapunov candidates is obtained as

$$\dot{V}_{\text{bps}}(t) - \dot{V}_{\text{lin}}(t) = -(\gamma + t^{\gamma}) \sum_{i=1}^{n+m} e_i(t) \left(\Upsilon \frac{1-\exp(-\mu e_i(t))}{1+\exp(-\mu e_i(t))} - e_i(t) \right). \quad (49)$$

For simplicity and further discussion, the last term of (49) is defined as

$$\mathcal{H}_{\mathcal{A}}(e_i(t)) = \Upsilon \frac{1-\exp(-\mu e_i(t))}{1+\exp(-\mu e_i(t))} - e_i(t). \quad (50)$$

Considering $1 \geq |e_i(t)| \geq 0$, the derivative of function $\mathcal{H}_{\mathcal{A}}(e_i(t))$ is described as

$$\frac{\mathrm{d}\mathcal{H}_{\mathcal{A}}(e_i(t))}{\mathrm{d}e_i(t)} = \frac{\mathrm{d}\phi(e_{\text{bps}i}(t))}{\mathrm{d}e_i(t)} - \frac{\mathrm{d}\phi(e_{\text{lin}i}(t))}{\mathrm{d}e_i(t)} = \frac{2\Upsilon\mu \exp(-\mu e_i(t))}{(1+\exp(-\mu e_i(t)))^2} - 1 \quad (51)$$

where $\Upsilon = (1+\exp(-\mu))/(1-\exp(-\mu))$, $\phi(e_{\text{pos}i}(t))$ and $\phi(e_{\text{lin}i}(t))$ denote AFb-sigmoid and AFlinear activation units, respectively. Since $1/\exp(\mu e_i(t)) + \exp(\mu e_i(t)) \geq 2$, as long as $\mu > 2$ and $\Upsilon > 1$, the derivative of function $\mathcal{H}_{\mathcal{A}}(e_i(t))$ is nonnegative when $e_i(t) \in [0, \vartheta]$ and negative when $e_i(t) \in (\vartheta, 1]$, where $\vartheta = \ln(2(\Upsilon\mu - 1) \pm (4\Upsilon\mu(\Upsilon\mu - 2))^{1/2})/2$ is the point when $\dot{\mathcal{H}}_{\mathcal{A}}(e_i(t)) = 0$. $\mathcal{H}_{\mathcal{A}}(e_i(t))$ will approach the minimum value when $e_i(t) = 0$ and $e_i(t) = 1$, i.e., $\mathcal{H}_{\mathcal{A}}(0) = \mathcal{H}_{\mathcal{A}}(1) = 0$, and in other cases $\mathcal{H}_{\mathcal{A}}(e_i(t)) > 0$. Therefore, $\mathcal{H}_{\mathcal{A}}(e_i(t)) \geq 0$, $\forall t > 0$. Since $1 \geq |e_i(t)| \geq 0$, we have $e_i(t)\mathcal{H}_{\mathcal{A}}(e_i(t)) \geq 0$.

Similarly, according to the definitions of $e_i(t)$ and $\phi(e_i(t))$ in Theorem IV, we can draw the same conclusion when $e_i(t) < 0$. Hence, the following holds.

1) $\dot{V}_{\text{bps}}(t) - \dot{V}_{\text{lin}}(t) = 0$, if $|e_i(t)| = 0, \forall i = 1, 2, \ldots, n+m$.
2) $\dot{V}_{\text{bps}}(t) - \dot{V}_{\text{lin}}(t) < 0$, if $|e_i(t)| \neq 0, \exists i = 1, 2, \ldots, n+m$.

According to Theorem 2, the convergence performance of VPNN with AFb-sigmoid is better than that with AFlinear. In addition, from (36), $\|\tilde{\boldsymbol{Y}}(t)\|_2$ converges to zero with superexponential convergence speed when $1 \geq |\boldsymbol{e}(t)| \geq 0$.The proof is thus completed. ∎

Theorem 7 (Convergence Theorem With AFsinh): Considering (5) and (15), by using VPNN (23) with a sinh activation function array $\phi_4(\boldsymbol{u}) = (\exp(\boldsymbol{u}) - \exp(\boldsymbol{u}))/2$, the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^*(t), \lambda^*(t)]^T$ with a superexponential convergence rate.

Proof: First, the ith scalar-type formula of VPNN dynamic designed formula (23) with AFsinh is written as

$$\dot{e}_i(t) = \frac{\mathrm{d}e_i(t)}{\mathrm{d}t} = -(\gamma + t^{\gamma})\phi_4(e_i(t)) = -(\gamma + t^{\gamma}) \frac{\exp(\boldsymbol{e}(t)) - \exp(-\boldsymbol{e}(t))}{2}. \quad (52)$$

Based on the differential theory in [60], the solution to (52) is

$$\boldsymbol{e}(t) = \ln\left(-\frac{\mathcal{C}_2 + \exp(-\varpi)}{\mathcal{C}_2 - \exp(-\varpi)} \right) \quad (53)$$

where $\varpi = t^{\gamma+1}/(\gamma+1) + \gamma t$ and $\mathcal{C}_2 = (\exp(\boldsymbol{e}(0)) + 1)/(\exp(\boldsymbol{e}(0)) - 1)$ is a constant term associated with initial value $\boldsymbol{e}(0)$. According to (36), we have $\|\tilde{\boldsymbol{Y}}(t)\|_2 \leq \|\boldsymbol{e}(t)\|_2/(\lambda_{\min}(\boldsymbol{W}^{\mathrm{T}}(t)\boldsymbol{W}(t)))^{1/2}$, which means that $\|\tilde{\boldsymbol{Y}}(t)\|_2$ has an upper bound via using VPNN dynamic design formula (23). When time $t \to +\infty$, since $\gamma > 1$, $\varpi = t^{\gamma+1}/(\gamma+1) + \gamma t$ also approaches $+\infty$. Then, each element of $\boldsymbol{e}(t)$ will converge to zero, which makes $\|\boldsymbol{e}(t)\|_2$ and $\|\tilde{\boldsymbol{Y}}(t)\|_2$ approach zero.

Second, to prove the superexponential convergence of VPNN, the error between the time derivative of the Lyapunov candidate with AFlinear and AFsinh is obtained as

$$\dot{V}_{\text{sin}}(t) - \dot{V}_{\text{lin}}(t) = -(\gamma + t^{\gamma}) \sum_{i=1}^{n+m} e_i(t) \left(\frac{\exp(e_i(t)) - \exp(-e_i(t))}{2} - e_i(t) \right). \quad (54)$$

For simplicity and further discussion, the last term of (54) is defined as

$$\mathcal{H}_{\mathcal{B}}(e_i(t)) = \frac{\exp(e_i(t)) - \exp(-e_i(t))}{2} - e_i(t). \quad (55)$$

Considering $e_i(t) > 0$, the derivative of function $\mathcal{H}_{\mathcal{B}}(e_i(t))$ is

$$\frac{\mathrm{d}\mathcal{H}_{\mathcal{B}}(e_i(t))}{\mathrm{d}e_i(t)} = \frac{\mathrm{d}\phi(e_{\text{sin}i}(t))}{\mathrm{d}e_i(t)} - \frac{\mathrm{d}\phi(e_{\text{lin}i}(t))}{\mathrm{d}e_i(t)} = \frac{1}{2}\exp(e_i(t)) + \exp(-e_i(t))) - 1 = \frac{1}{2}\Big(\exp(e_i(t)) + \exp(-e_i(t)) - 2\exp\left(\frac{e_i(t)}{2}\right)\exp\left(-\frac{e_i(t)}{2}\right)\Big) \quad (56)$$

where $\phi(\varepsilon_{\text{sin}i}(t))$ and $\phi(\varepsilon_{\text{lin}i}(t))$, respectively, denote AFsinh and AFlinear activation units. According to inequality $x^2 + y^2 \geq 2xy$, the derivative of function $\mathcal{H}_{\mathcal{B}}(\varepsilon_i(t))$ is positive, i.e.,

$$\frac{\mathrm{d}\mathcal{H}_{\mathcal{B}}(e_i(t))}{\mathrm{d}e_i(t)} = \frac{1}{2}(\exp(e_i(t)) + \exp(-e_i(t))) - 1 \geq 0. \quad (57)$$

$\mathcal{H}_{\mathcal{B}}(e_i(t))$ will approach the minimum value when $e_i(t) = 0$, i.e., $\mathcal{H}_{\mathcal{B}}(0) = 0$, which means that $\mathcal{H}_{\mathcal{B}}(e_i(t)) \geq 0$, $\forall t > 0$. Since $e_i(t) > 0$, we have $e_i(t)\mathcal{H}_{\mathcal{B}}(e_i(t)) \geq 0$.

Similarly, according to the definitions of $e_i(t)$ and $\phi(e_i(t))$ in Theorem IV, we can draw the same conclusion when $e_i(t) < 0$, and the similar process is omitted.

Therefore, the following holds.

1) $\dot{V}_{\text{sin}}(t) - \dot{V}_{\text{lin}}(t) = 0$, if $|e_i(t)| = 0, \forall\, i = 1, 2, \ldots, n+m$.
2) $\dot{V}_{\text{sin}}(t) - \dot{V}_{\text{lin}}(t) < 0$, if $|e_i(t)| \neq 0, \exists\, i = 1, 2, \ldots, n+m$.

According to Theorem IV, the convergence performance of VPNN with AFsinh is better than that with AFlinear. From (36), we state that $\|\tilde{\boldsymbol{Y}}(t)\|_2$ converges to zero with superexponential convergence. The proof is thus completed. ∎

Theorem 8 (Convergence Theorem With AFtunable): Considering (5) and (15), by using VPNN (23) with tunable activation function $\phi_6(\boldsymbol{u}) = \text{sig}^r(\boldsymbol{u}) + \text{sig}(\boldsymbol{u}) + \text{sig}^{\frac{1}{r}}(\boldsymbol{u})$, the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [x^*(t), \lambda^*(t)]^{\mathrm{T}}$ with finite-time convergence property.

Proof: If the maximum element value $e_i(t)$ of the residual error converges to zero, then the residual error $\boldsymbol{e}(t)$ will converge to zero, i.e., the state vector $\boldsymbol{Y}(t)$ converges to the theoretical solution $\boldsymbol{Y}^*(t) = [\boldsymbol{x}^*(t), \lambda^*(t)]^{\mathrm{T}}$. Therefore, a Lyapunov function of the maximum value of $\boldsymbol{e}(t)$ with AFtunable ϕ_6 is defined as $\boldsymbol{V}_{\mathcal{T}} = \sigma^2/2$, where $\sigma = \max_{1\leqslant i\leqslant n+m}\{e_i(t)\}$. The derivative of $\boldsymbol{V}_{\mathcal{T}}$ is described as

$$\begin{aligned}\dot{\boldsymbol{V}}_{\mathcal{T}} &= \sigma\dot{\sigma}\\ &= -(\gamma + t^{\gamma})\sigma(\mathrm{sig}^{r}(\sigma) + \mathrm{sig}(\sigma) + \mathrm{sig}^{\frac{1}{r}}(\sigma))\\ &\leqslant -3(\gamma + t^{\gamma})\sigma\min\{\mathrm{sig}^{r}(\sigma), \mathrm{sig}(\sigma), \mathrm{sig}^{\frac{1}{r}}(\sigma)\}\\ &= -3(\gamma + t^{\gamma})\sigma\min\{\sigma^{r}, \sigma, \sigma^{\frac{1}{r}}\}\\ &= -3(\gamma + t^{\gamma})(2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2}}\min\{(2\boldsymbol{V}_{\mathcal{T}})^{\frac{r}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2r}}\}.\end{aligned} \tag{58}$$

According to the definition of the tunable action function, the following two cases should be discussed with $r > 0$ and $r \neq 1$.

1) When $0 < r < 1$, i.e., $\min\{(2\boldsymbol{V}_{\mathcal{T}})^{\frac{r}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2r}}\} = (2\boldsymbol{V}_{\mathcal{T}})^{\frac{r}{2}}$, then (58) is

$$\dot{\boldsymbol{V}}_{\mathcal{T}} \leqslant -3(\gamma + t^{\gamma})(2\boldsymbol{V}_{\mathcal{T}})^{\frac{r+1}{2}}. \tag{59}$$

$\boldsymbol{V}_{\mathcal{T}} \geqslant 0$ and $\dot{\boldsymbol{V}}_{\mathcal{T}} \leqslant 0$ (if and only if $\boldsymbol{V}_{\mathcal{T}} = 0$, $\dot{\boldsymbol{V}}_{\mathcal{T}} = 0$). According to [58], the state vector $\boldsymbol{Y}(t)$ of VPNN (23) globally converges to the theoretical solution $\boldsymbol{Y}^*(t)$ when $0 < r < 1$. The finite time can be obtained. By integrating (59) with time t, the following result is obtained:

$$\boldsymbol{V}_{\mathcal{T}} = \begin{cases} \leqslant \dfrac{1}{2}\big(\ell_a(t)\big)^{\frac{2}{1-r}} & \text{if} 0 \leqslant t \leqslant t_a \\ = 0 & \text{if} t > t_a \end{cases} \tag{60}$$

where $\ell_a(t) = (1-r)(-3t^{\gamma+1}/2(\gamma+1) - (3\gamma t/2)) + \boldsymbol{e}^{1-r}(0)$ and t_a is defined as the time at which convergence is achieved. The solution to $\ell_a(t) = 0$, i.e., $\ell_a(t_a) = (1-r)(-3t_a^{\gamma+1}/2(\gamma+1) - (3\gamma t_a/2)) + \boldsymbol{e}^{1-r}(0) = 0$.

2) When $r > 1$, then $\min\{(2\boldsymbol{V}_{\mathcal{T}})^{\frac{r}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2}}, (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2r}}\} = (2\boldsymbol{V}_{\mathcal{T}})^{\frac{1}{2r}}$, and (58) is

$$\dot{\boldsymbol{V}}_{\mathcal{T}} \leqslant -3(\gamma + t^{\gamma})(2\boldsymbol{V}_{\mathcal{T}})^{\frac{r+1}{2r}}. \tag{61}$$

Similarly, we have $\boldsymbol{V}_{\mathcal{T}} \geqslant 0$ and $\dot{\boldsymbol{V}}_{\mathcal{T}} \leqslant 0$ (if and only if $\boldsymbol{V}_{\mathcal{T}} = 0$, $\dot{\boldsymbol{V}}_{\mathcal{T}} = 0$). According to [58], the state vector $\boldsymbol{Y}(t)$ of VPNN (23) globally converges to the theoretical solution $\boldsymbol{Y}^*(t)$ when $r > 1$. The finite time can be obtained as follows. By integrating (61) with time t, the following result is obtained:

$$\boldsymbol{V}_{\mathcal{T}} = \begin{cases} \leqslant \dfrac{1}{2}(\ell_b(t))^{\frac{2r}{r-1}} & \text{if} 0 \leqslant t \leqslant t_b \\ = 0 & \text{if} t > t_b \end{cases} \tag{62}$$

where $\ell_b(t) = (r-1)(-3t^{\gamma+1}/2(\gamma+1) - (3\gamma t/2))/2r + \boldsymbol{e}^{1-r}(0)$, and t_b is defined as the time at which convergence is achieved, as well as the solution to $\ell_b(t) = 0$, i.e., $\ell_b(t_b) = (r-1)(-3t_b^{\gamma+1}/2(\gamma+1) - (3\gamma t_b/2))/2r + \boldsymbol{e}^{1-r}(0) = 0$.

In conclusion, when using AFtunable $\phi_6(\boldsymbol{e}(t))$, the upper bound of the convergence time of VPNN is

$$t_{\mathrm{VP}} = \begin{cases} t_a & \text{for} 0 < r < 1 \\ t_b & \text{for} r > 1. \end{cases} \tag{63}$$

Note that if $r = 1$, the AFtunable $\phi_6(\boldsymbol{e}(t))$ will degrade into the AFlinear $\phi_7(\boldsymbol{e}(t)) = 3\boldsymbol{e}(t)$.

On the basis of [61], the upper bound of the convergence time t_{Z} with ZNN is

$$t_{\mathrm{Z}} = \begin{cases} \dfrac{(\boldsymbol{e}(0))^{1-r}}{\gamma(1-r)} & \text{for} 0 < r < 1 \\ \dfrac{r(\boldsymbol{e}(0))^{\frac{r-1}{1}}}{\gamma(1-r)} & \text{for} r > 1. \end{cases} \tag{64}$$

Substituting t_{Z} into $\ell_a(t)$ and $\ell_b(t)$, the following inequalities are obtained:

$$\begin{aligned}\ell_a(t_{\mathrm{Z}}) &= -\frac{3}{2}(1-r)\left(\gamma t_{\mathrm{Z}} + \frac{t_{\mathrm{Z}}^{\gamma+1}}{\gamma+1}\right) + \boldsymbol{e}^{1-r}(0)\\ &= -\frac{3}{2}(1-r)\frac{t_{\mathrm{Z}}^{\gamma+1}}{\gamma+1} < 0\\ \ell_b(t_{\mathrm{Z}}) &= -\frac{3}{2}(r-1)\left(\gamma t_{\mathrm{Z}} + \frac{t_{\mathrm{Z}}^{\gamma+1}}{\gamma+1}\right) + \boldsymbol{e}^{1-r}(0)\\ &= -\frac{3}{2}(1-r)\frac{t_{\mathrm{Z}}^{\gamma+1}}{\gamma+1} < 0.\end{aligned} \tag{65}$$

Evidently, $\dot{\boldsymbol{V}}_{\mathcal{T}} \geqslant 0$ cannot be achieved with $\ell_a(t) < 0$ and $\ell_b(t) < 0$, which means that if $t = t_{\mathrm{Z}}$, then $\ell_a(t) = 0$, $\ell_b(t) = 0$, and $\dot{\boldsymbol{V}}_{\mathcal{T}} = 0$, i.e., $t_a < t_{\mathrm{Z}}$ and $t_b < t_{\mathrm{Z}}$. Since t_{Z} is finite-length time, the state vector $\boldsymbol{Y}(t)$ of VPNN (23) converges to the theoretical solution with finite-time convergence property. The proof is thus completed. ■

V. Simulation Verifications

In this section, comparative simulations are conducted to verify the effectiveness of the proposed VPNN for solving TVQP problems. For comparison, the simulation results of the same TVQP problems solved by ZNN are also presented.

The simulations are performed with MATLAB R2017b on a MacBook Pro (2017) with an Intel Core i7 CPU at 2.8 GHz with 16 GB of 2133-MHz LPDDR3 RAM.

Taking the following TVQP-E problem as an example, i.e.,

$$\begin{aligned}\min\ & \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\boldsymbol{x}(t) + \boldsymbol{P}^{\mathrm{T}}(t)x(t)\\ \text{s.t.}\ & \boldsymbol{A}(t)\boldsymbol{x}(t) = \boldsymbol{B}(t)\end{aligned} \tag{66}$$

where

$$\begin{aligned}\boldsymbol{Q}(t) &:= \begin{bmatrix}\sin 2t + 2 & \cos 2t\\ \cos 2t & \sin 2t + 2\end{bmatrix}\\ \boldsymbol{P}(t) &:= \begin{bmatrix}\sin 3t\\ \cos 3t\end{bmatrix}\\ \boldsymbol{A}(t) &:= [\sin 4t, \cos 4t]\\ \boldsymbol{B}(t) &:= \cos 5t\\ \boldsymbol{x}(t) &:= [x_1(t), x_2(t)]^{\mathrm{T}}.\end{aligned} \tag{67}$$

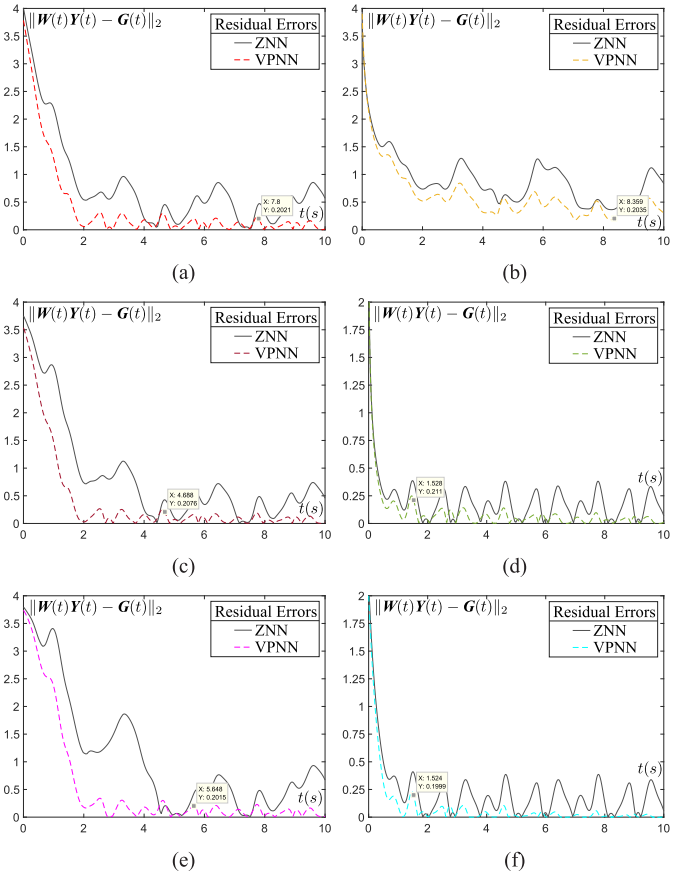

Fig. 4. Residual errors $\|\boldsymbol{W}(t)\boldsymbol{Y}(t)-\boldsymbol{G}(t)\|_2$ when VPNN (dashed line) and ZNN (solid line) are used to solve the TVQP-E problem (66) with different activation functions during $t \in [0, 10]$. (a) AFlinear. (b) AFpower. (c) AFb-sigmoid. (d) AFsinh. (e) AFtanh. (f) AFtunable.

Based on (5), the simplified form is rewritten as

$$\boldsymbol{W}(t)\boldsymbol{Y}(t) = \boldsymbol{G}(t) \tag{68}$$

where

$$\begin{aligned} \boldsymbol{W}(t) &:= \begin{bmatrix} \sin 2t+2 & \cos 2t & \sin 4t \\ \cos 2t & \sin 2t+2 & \cos 4t \\ \sin 4t & \cos 4t & 0 \end{bmatrix} \\ \boldsymbol{Y}(t) &:= [x_1(t), x_2(t), \lambda(t)]^{\mathrm{T}} \\ \boldsymbol{G}(t) &:= [-\sin 3t, -\cos 3t, \cos 5t]^{\mathrm{T}}. \end{aligned} \tag{69}$$

A. Experiment 1: Residual Errors With Activation Functions

To verify the generality of applications with different types of activation functions, the simulations of the TVQP problem (66) solved by VPNN and ZNN with six activation functions are illustrated in Fig. 4.

First, from the state curves of VPNN (i.e., dashed lines) in Fig. 4(a)–(f), we find that during $t \in [0, 10]$, all the state variables $\boldsymbol{Y}(t)$ converge to the theoretical solution, i.e., $\boldsymbol{Y}(t) - \boldsymbol{Y}^*(t)$ converges to zero. This result verifies the proposed global convergence Theorem 1. Second, by comparing the dashed and solid lines, we find that the dashed lines approach the t-axis earlier than do the solid lines. That is, comparison of the convergence time of VPNN with that of ZNN when using

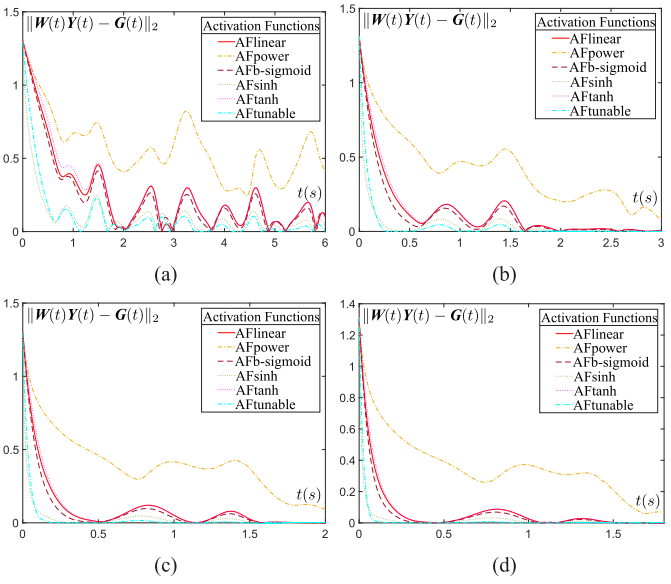

Fig. 5. Residual errors $\|\boldsymbol{W}(t)\boldsymbol{Y}(t)-\boldsymbol{G}(t)\|_2$ when VPNN with six different activation functions are used to solve the TVQP-E problem (66) with different design parameters γ. AFlinear (red solid line), AFpower (orange dashed-dotted line), AFbipolar-sigmoid (maroon dashed line), AFsinh (olive dotted line), AFtanh (magenta dashed line), and AFtunable (cyan dashed-dotted line). (a) $\gamma = 1$. (b) $\gamma = 5$. (c) $\gamma = 10$. (d) $\gamma = 15$.

TABLE I

TIME COST WHEN ERRORS REACH 0.2 WITH DIFFERENT γ

Time $t(s)$	$t_{\gamma=1}$	$t_{\gamma=5}$	$t_{\gamma=10}$	$t_{\gamma=15}$
AFlinear	7.8040	1.4690	0.2094	0.1341
AFpower	Cannot	2.8080	1.6540	1.5110
AFb-sigmoid	4.6880	0.3085	0.1862	0.1146
AFsinh	1.5280	0.1127	0.0521	0.0323
AFtanh	5.6480	0.4078	0.2094	0.1441
AFtunable	1.5240	0.1246	0.0647	0.0395

the six activation functions shows that the convergence time of the residual errors of VPNN is shorter than that of ZNN. This result validates Theorem 3, as well as the effectiveness, accuracy, and fast convergence performance of the proposed VPNN for solving TVQP problems.

B. Experiment 2: Residual Error With Different Parameter γ

In actual applications, the convergence performance can be further improved by increasing the design parameter γ. To illustrate this point, we monitor the residual errors $\|\boldsymbol{W}(t)\boldsymbol{Y}(t)-\boldsymbol{G}(t)\|_2$ during the process of solving the TVQP-E problem (66) with the VPNN model with $\gamma = 1, 5, 10, 15$. As shown in Fig. 5, all the residual errors synthesized by VPNN with different activation functions converge rapidly, and the convergence time of the residual errors decreases as γ increases. Table I quantitatively shows the specific convergence time with different γ values. Note that AFsinh and AFtunable have excellent performance, and the time cost is less than 0.13 s when $\gamma > 5$. The simulation results further verify Theorems 4–8.

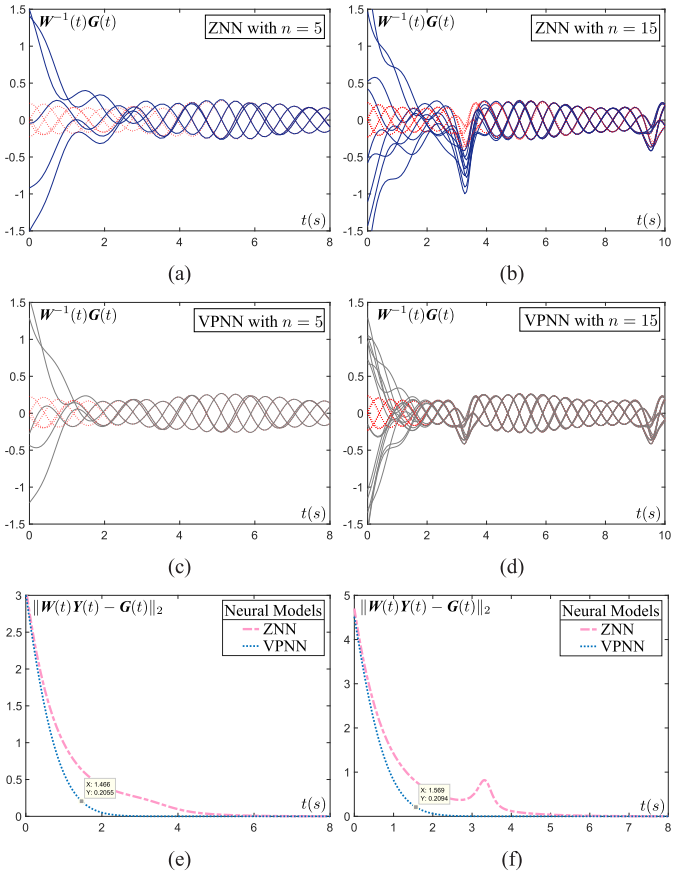

Fig. 6. Simulation results of VPNN and ZNN for solving the high-dimensional TVQP-E problem. Red dotted curves denote the entry trajectories of the theoretical solution. Blue solid curves denote the solution computed when using ZNN. Black solid curves denote the solution computed when using VPNN. (a) ZNN with five dimensions, $n = 5$. (b) ZNN with 15 dimensions, $n = 15$. (c) VPNN with five dimensions, $n = 5$. (d) VPNN with 15 dimensions, $n = 15$. (e) Residual errors for five dimensions, $n = 5$. (f) Residual errors for 15 dimensions, $n = 15$.

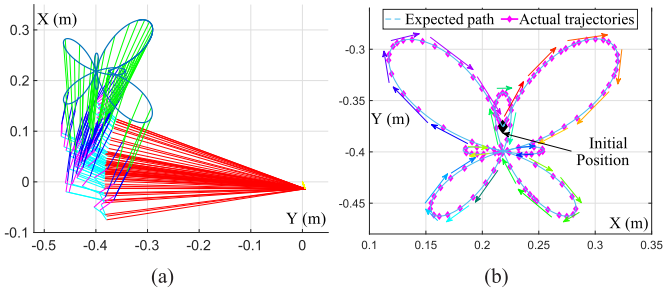

Fig. 7. Simulation results of a Kinova JACO^2 manipulator for tracking a butterfly path via VPNN. (a) Tracking trajectories of the links. (b) Expected path and actual trajectories.

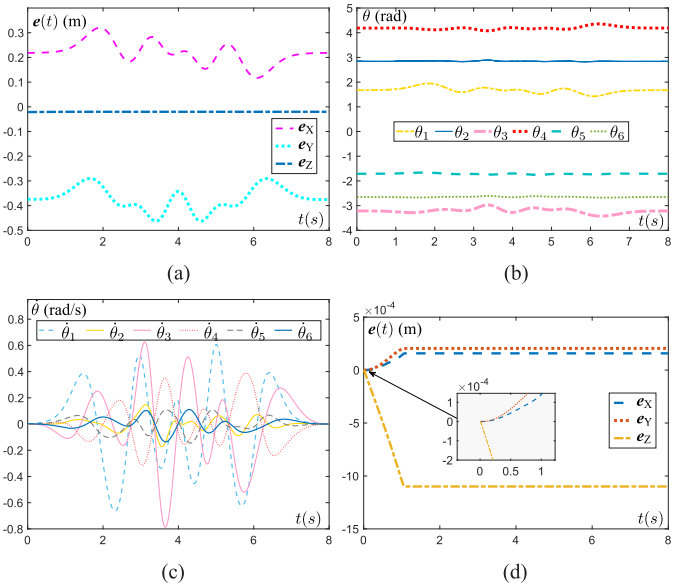

Fig. 8. Important variables during the task tracking period when the Kinova manipulator tracks a butterfly path via VPNN. (a) End-effector position. (b) Joint angles θ. (c) Joint-angle velocity $\dot{\theta}$. (d) Position error.

C. Experiment 3: High-Dimensional Cases

High-dimensional situations are common in actual applications. Therefore, the analysis of different dimensions n of time-varying coefficients $\boldsymbol{W}(t)$ and $\boldsymbol{G}(t)$ is necessary.

Suppose that the time-varying Toeplitz matrix, which is a diagonal-constant matrix, is obtained from the TVQP-E problem (66). The specific matrix $\boldsymbol{W}(t)$ is

$$\begin{bmatrix} W_1(t) & W_2(t) & W_3(t) & \cdots & W_n(t) \\ W_2(t) & W_1(t) & W_2(t) & \cdots & W_{n-1}(t) \\ W_3(t) & W_2(t) & W_1(t) & \cdots & W_{n-2}(t) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_n(t) & W_{n-1}(t) & W_{n-2}(t) & \cdots & W_1(t) \end{bmatrix} \tag{70}$$

where vector $\boldsymbol{W}_1(t) = [W_1(t), W_2(t), W_3(t), \ldots, W_n(t)]^{\mathrm{T}} \in \mathbb{R}^{n\times 1}$ denotes the first column of matrix $\boldsymbol{W}(t)$ (70).

Let the initial $W_1(t) = \sin t + 5$ and $\mathcal{W}_\epsilon(t) = \cos t/(\epsilon - 1)$ $(\epsilon = 2, 3, \ldots, n)$. Vector $\boldsymbol{G}(t) \in \mathbb{R}^{n\times 1}$ is

$$[\sin 4t \;\; \sin(4t + \pi/2) \;\; \sin(4t + [(n-1)\pi]/2)]^{\mathrm{T}}. \tag{71}$$

The entry trajectories of the theoretical solution $\boldsymbol{W}^{-1}(t)\boldsymbol{G}(t)$ with dimensions $n = 5$ and $n = 15$ are shown in Fig. 6(a)–(d), respectively, show the convergence of the computed solutions for solving TVQP problems with five and 15 dimensions. We can clearly see from Fig. 6(a)–(d) that the convergence performance of VPNN remains excellent when facing such large-scale cases. However, the results of ZNN are not that satisfactory. The convergence curves of ZNN are "confused," which means that the residual errors of ZNN are larger than those of VPNN during convergence. Fig. 6(e) and (f) presents the residual errors which quantitatively illustrate the convergence times of ZNN and VPNN and verify the effectiveness of VPNN for solving large-scale TVQP problems.

VI. Applications to Robots

In this section, the proposed VPNN model is applied to the inverse kinematics motion planning problem of a Kinova JACO^2 manipulator with six DOF. For such a robot, the joint vector is generally written as $\theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]^{\mathrm{T}} \in \mathbb{R}^6$. The desired path of the end-effector is a complex butterfly. Due to the nonlinearity of redundant robot manipulators, the position and orientation of the end-effector are difficult to obtain through forward kinematics [16], [53], [55]. According to the

Fig. 9. Experimental process of a Kinova JACO^2 manipulator for tracking a butterfly path controlled by VPNN.

inverse kinematics theory, which is a fundamental problem in the practical use of robot manipulators [57], the relationship between the end-effector velocity $\dot{\boldsymbol{r}}(t)$ and the joint velocity $\dot{\theta}(t)$ can be described as

$$\boldsymbol{J}(\theta(t))\dot{\theta}(t) = \dot{\boldsymbol{r}}(t) \tag{72}$$

where $\boldsymbol{J}(\theta) \in \mathbb{R}^{2\times 6}$ is the Jacobian matrix defined as $\boldsymbol{J}(\theta) = \partial\mathcal{F}(\theta)/\partial\theta$ and $\mathcal{F}(\theta)$ is the forward kinematics mapping. $m = 3$ denotes the spatial dimension of the end-effector and $n = 6$ denotes the six links. Considering RMP with the joint velocity limit of the robot manipulator, a TVQP problem can be introduced on the basis of [26] and [52], i.e.,

$$\begin{aligned} &\min \ \frac{1}{2}\|\boldsymbol{R}(t) + \dot{\theta}(t)\|_2^2 \\ &\text{s.t. } \boldsymbol{J}(\theta(t))\dot{\theta}(t) = \dot{\boldsymbol{r}}(t) + \Im(\boldsymbol{r}(t) - \mathcal{F}(\theta)) \end{aligned} \tag{73}$$

where $\boldsymbol{R}(t) = \varrho(\theta(t) - \theta(0))$ with parameter $\varrho > 0$, which denotes the criterion of RMP, is the magnitude of joint drift between joint $\theta(t)$ and initial joint $\theta(0)$. $\Im(t)$ denotes the feedback control matrix. The above QP problem can be rewritten as a matrix equation as in (5) and (15). Therefore, motion planning can be achieved by using the proposed VPNN model.

The simulation results of the Kinova JACO^2 manipulator for tracking a butterfly path via VPNN are shown in Figs. 7 and 8. Specifically, Fig. 7(a) illustrates the motion trajectories of the links of the manipulator. Fig. 7(b) illustrates the expected path and actual trajectories, which demonstrates that the actual trajectories are well matched with the expected path, and the end-effector task is completed satisfactorily. Snapshots of the physical experiment with a practical Kinova JACO^2 manipulator are shown in Fig. 9, which shows that the end-effector is well finished. Fig. 8(a) and (b) shows the end-effector position, joint angles, joint-angle velocities, and end-effector position errors when the manipulator is tracking a butterfly task. As shown in Fig. 8, the joint angles and velocities are all smooth, and the end-effector position errors are within $[-1.5 \times 10^{-3}, 0.5 \times 10^{-3}]$.

In summary, this application to the kinematics control of six-DOF robot manipulators demonstrates the effectiveness of the proposed VPNN model (23) in solving TVQP problems and in an application to a robot motion planning problem.

VII. Conclusion

In this paper, a power-type VPNN is proposed for solving TVQP problems. The state solutions with the VPNN model can converge to the theoretical solutions efficiently and exactly with a superexponential convergence rate. Theoretical analysis and simulation comparisons both prove that the proposed VPNN possesses better convergence performance than the traditional ZNN model. The illustrative example results demonstrate the advantages of the VPNN. Moreover, the application to a robot motion planning problem further verifies the practicability, effectiveness, efficiency, and accuracy of the VPNN.

References

[1] B. Gu and V. S. Sheng, "A solution path algorithm for general parametric quadratic programming problem," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4462–4472, Sep. 2018.

[2] F. A. Andaló, G. Taubin, and S. Goldenstein, "PSQP: Puzzle solving by quadratic programming," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 2, pp. 385–396, Feb. 2017.

[3] V. A. Papaspiliotopoulos, G. N. Korres, and N. G. Maratos, "A novel quadratically constrained quadratic programming method for optimal coordination of directional overcurrent relays," *IEEE Trans. Power Del.*, vol. 32, no. 1, pp. 3–10, Feb. 2015.

[4] Z. Wang, L. Liu, Q.-H. Shan, and H. Zhang, "Stability criteria for recurrent neural networks with time-varying delay based on secondary delay partitioning method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2589–2595, Oct. 2015.

[5] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C.-Y. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 6, pp. 740–749, Jun. 2016.

[6] Z. Zhang, L. Zheng, J. Yu, Y. Li, and Z. Yu, "Three recurrent neural networks and three numerical methods for solving a repetitive motion planning scheme of redundant robot manipulators," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 3, pp. 1423–1434, Jun. 2017.

[7] Z. Zhu, E. Schmerling, and M. Pavone, "A convex optimization approach to smooth trajectories for motion planning with car-like robots," in *Proc. IEEE Conf. Decision Control*, Dec. 2015, pp. 835–842.

[8] M. A. Mousavi, B. Moshiri, and Z. Heshmati, "Cooperative control of networked autonomous vehicles using convex optimization," in *Proc. RSI Int. Conf. Robot. Mechatronics*, Oct. 2015, pp. 681–687.

[9] A. Nazemi, "A neural network model for solving convex quadratic programming problems with some applications," *Eng. Appl. Artif. Intell.*, vol. 32, no. 32, pp. 54–62, 2014.

[10] Y. Zhang, X. Yan, D. Chen, D. Guo, and W. Li, "QP-based refined manipulability-maximizing scheme for coordinated motion planning and control of physically constrained wheeled mobile redundant manipulators," *Nonlinear Dyn.*, vol. 85, no. 1, pp. 245–261, Jul. 2016.

[11] J. V. Frasch, S. Sager, and M. Diehl, "A parallel quadratic programming method for dynamic optimization problems," *Math. Program. Comput.*, vol. 7, no. 3, pp. 289–329, 2015.

[12] A. Bemporad, "A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control," *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 1111–1116, Apr. 2016.

[13] K. Peng, F. Gao, and X. Guan, "Stochastic predictive control of battery energy storage for wind farm dispatching: Using probabilistic wind power forecasts," *Renew. Energy*, vol. 80, pp. 286–300, Aug. 2015.

[14] J. Xu, X. Wang, and X. Li, "Successive quadratic programming method for voltage/reactive power optimization in power systems," *Autom. Electr. Power Syst.*, vol. 25, no. 23, pp. 4–8, 2001.

[15] C. Ge, C. Hua, and X. Guan, "New delay-dependent stability criteria for neural networks with time-varying delay using delay-decomposition approach," *IEEE Trans. Neural Netw.*, vol. 25, no. 7, pp. 1378–1383, Jul. 2014.

[16] D. Chen and Y. Zhang, "Robust zeroing neural-dynamics and its time-varying disturbances suppression model applied to mobile robot manipulators," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4385–4397, Sep. 2018.

[17] I. E. Schochetman, S. K. Tsui, and R. L. Smith, "Solution existence for time-varying infinite horizon quadratic programming," *J. Math. Anal. Appl.*, vol. 195, no. 1, pp. 135–147, 1995.

[18] P. Tsiotras, M. Corless, and M. Rotea, "Optimal control of rigid body angular velocity with quadratic cost," in *Proc. IEEE Int. Conf. Decision Control*, vol. 2, Dec. 1996, pp. 1630–1635.

[19] Y. Zhang, S. Fu, Z. Zhang, L. Xiao, and X. Li, "On the LVI-based numerical method (E47 algorithm) for solving quadratic programming problems," in *Proc. IEEE Int. Conf. Autom. Logistics*, Aug. 2011, pp. 125–130.

[20] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication," *IEEE Trans. Signal Process.*, vol. 59, no. 8, pp. 3889–3902, Aug. 2011.

[21] Y. Zhang, W. E. Leithead, and D. J. Leith, "Time-series Gaussian process regression based on Toeplitz computation of $O(N^2)$ operations and O(N)-level storage," in *Proc. IEEE Conf. Decision Control*, Dec. 2005, pp. 3711–3716.

[22] M. A. Gómez, "An $O(n^2)$ active set algorithm for solving two related box constrained parametric quadratic programs," *Numer. Algorithms*, vol. 27, no. 4, pp. 367–375, 2001.

[23] J. Wang and Y. Xia, "A dual neural network solving quadratic programming problems," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 1, Jul. 1999, pp. 588–593.

[24] Y. Xia and J. Wang, "Primal neural networks for solving convex quadratic programs," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 1, Jul. 1999, pp. 582–587.

[25] Y. Zhang, D. Jiang, and J. Wang, "A recurrent neural network for solving Sylvester equation with time-varying coefficients," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1053–1063, Sep. 2002.

[26] Y. Zhang, J. Wang, and Y. Xu, "A dual neural network for bi-criteria kinematic control of redundant manipulators," *IEEE Trans. Robot. Autom.*, vol. 18, no. 6, pp. 923–931, Dec. 2002.

[27] Y. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 1, pp. 752–759, Feb. 2004.

[28] Y. Leung, K.-Z. Chen, Y.-C. Jiao, X.-B. Gao, and K. S. Leung, "A new gradient-based neural network for solving linear and quadratic programming problems," *IEEE Trans. Neural Netw.*, vol. 12, no. 5, pp. 1074–1083, Sep. 2001.

[29] D. Guo, C. Yi, and Y. Zhang, "Zhang neural network versus gradient-based neural network for time-varying linear matrix equation solving," *Neurocomputing*, vol. 74, no. 17, pp. 3708–3712, 2011.

[30] Y. Zhang, C. Yi, and W. Ma, "Simulation and verification of zhang neural network for online time-varying matrix inversion," *Simul. Model. Pract. Theory*, vol. 17, no. 10, pp. 1603–1617, 2009.

[31] K. Chen, S. Yue, and Y. Zhang, "MATLAB simulation and comparison of Zhang neural network and gradient neural network for online solution of linear time-varying matrix equation $AXB-C=0$," in *Proc. Int. Conf. Intell. Comput.*, 2008, pp. 68–75.

[32] X. Huang, X. Lou, and B. Cui, "A novel neural network for solving convex quadratic programming problems subject to equality and inequality constraints," *Neurocomputing*, vol. 214, pp. 23–31, Nov. 2016.

[33] S. Qin and X. Xue, "A two-layer recurrent neural network for nonsmooth convex optimization problems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 6, pp. 1149–1160, Jun. 2015.

[34] J. Feng, S. Qin, F. Shi, and X. Zhao, "A recurrent neural network with finite-time convergence for convex quadratic bilevel programming problems," *Neural Comput. Appl.*, vol. 30, no. 11, pp. 3399–3408, Dec. 2018.

[35] X. Le and J. Wang, "A two-time-scale neurodynamic approach to constrained minimax optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 620–629, Mar. 2017.

[36] X. Liu and T. Chen, "Global exponential stability for complex-valued recurrent neural networks with asynchronous time delays," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 593–606, Mar. 2016.

[37] H. Zhang, Z. Wang, and D. Liu, "A comprehensive review of stability analysis of continuous-time recurrent neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 7, pp. 1229–1262, Jul. 2014.

[38] J. Y. Wang, L. Zhang, Q. Guo, and Z. Yi, "Recurrent neural networks with auxiliary memory units," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1652–1661, May 2018.

[39] Z. Yan and J. Wang, "Nonlinear model predictive control based on collective Neurodynamic optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 840–850, Apr. 2015.

[40] Y. Zhang and S. S. Ge, "Design and analysis of a general recurrent neural network model for time-varying matrix inversion," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1477–1490, Nov. 2005.

[41] L. Xiao, "A new design formula exploited for accelerating Zhang neural network and its application to time-varying matrix inversion," *Theor. Comput. Sci.*, vol. 647, pp. 50–58, Sep. 2016.

[42] Y. Zhang, W. Ma, and C. Yi, "The link between newton iteration for matrix inversion and Zhang neural network (ZNN)," in *Proc. IEEE Int. Conf. Ind. Technol.*, Apr. 2008, pp. 1–6.

[43] L. Jin, Y. Zhang, S. Li, and Y. Zhang, "Modified ZNN for time-varying quadratic programming with inherent tolerance to noises and its application to kinematic redundancy resolution of robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 6978–6988, Nov. 2016.

[44] L. Xiao and Y. Zhang, "Zhang neural network versus gradient neural network for solving time-varying linear inequalities," *IEEE Trans. Neural Netw.*, vol. 22, no. 10, pp. 1676–1684, Oct. 2011.

[45] Y. Zhang and Z. Li, "Zhang neural network for online solution of time-varying convex quadratic program subject to time-varying linear-equality constraints," *Phys. Lett. A*, vol. 373, nos. 18–19, pp. 1639–1643, 2009.

[46] Y. Zhang, G. Ruan, K. Li, and Y. Yang, "Robustness analysis of the Zhang neural network for online time-varying quadratic optimization," *J. Phys. A, Math. General*, vol. 43, no. 24, pp. 202–245, 2010.

[47] Z. Zhang, L. Zheng, J. Weng, Y. Mao, W. Lu, and L. Xiao, "A new varying-parameter recurrent neural-network for online solution of time-varying Sylvester equation," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3135–3148, Nov. 2018, doi: 10.1109/TCYB.2017.2760883.

[48] D. R. Glandorf, "Lagrange multipliers and the state transition matrix for coasting arcs," *AIAA J.*, vol. 7, no. 2, pp. 363–365, 2015.

[49] S. Boyd, L. Vandenberghe, and L. Faybusovich, "Convex optimization," *IEEE Trans. Autom. Control*, vol. 51, no. 11, p. 1859, Nov. 2006.

[50] S. Li, H. Wang, and M. U. Rafique, "A novel recurrent neural network for manipulator control with improved noise tolerance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1908–1918, May 2018, doi: 10.1109/TNNLS.2017.2672989.
[51] S. W. S. Tang and J. Wang, "A primal-dual neural network for kinematic control of redundant manipulators subject to joint velocity constraints," in *Proc. Int. Conf. Neural Inf. Process.*, vol. 2, Nov. 1999, pp. 801–806.
[52] Y. Zhang, X. Lv, Z. Li, and Z. Yang, "Repetitive motion planning of redundant robots based on LVI-based primal-dual neural network and puma 560 example," in *Proc. Int. Conf. Life Syst. Modeling Simulation*, 2007, pp. 536–545.
[53] S. Li, J. He, Y. Li, and M. U. Rafique, "Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 415–426, Feb. 2017.
[54] L. Jin, S. Li, H. M. La, and X. Luo, "Manipulability optimization of redundant manipulators using dynamic neural networks," *IEEE Trans. Ind. Electron.*, vol. 64, no. 6, pp. 4710–4720, Jun. 2017.
[55] L. Jin, S. Li, J. Yu, and J. He, "Robot manipulator control using neural networks: A survey," *Neurocomputing*, vol. 235, pp. 23–34, Apr. 2018.
[56] C.-F. Juang and Y.-T. Yeh, "Multiobjective evolution of biped robot gaits using advanced continuous ant-colony optimized recurrent neural networks," *IEEE Trans. Cybern.*, vol. 48, no. 6, pp. 1910–1922, Jun. 2017.
[57] J. Craig, *Introduction to Robotics: Mechanics and Control*. New York, NY, USA: Pearson Education, 1989.
[58] R. Kalman and J. Bertram, "Control system analysis and design via the second method of Lyapunov: (I) continuous-time systems (II) discrete time systems," *IEEE Trans. Autom. Control*, vol. 4, no. 3, p. 112, Dec. 2003.
[59] M. Hazewinkel, *Advanced Multivariate Statistics With Matrices*. Dordrecht, The Netherlands: Springer, 2005.
[60] E. O. Roxin, "Ordinary differential equations," *Ordinary Differ. Equ.*, vol. 24, no. 3, pp. 82–122, 1972.
[61] S. Li and Y. Li, "Nonlinearly activated neural network for solving time-varying complex Sylvester equation," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1397–1407, Aug. 2014.

Zhijun Zhang (M'12) received the Ph.D. degree in communication and information systems from Sun Yat-sen University, Guangzhou, China, in 2012.

From 2013 to 2015, he was a Post-Doctoral Research Fellow with the Institute for Media Innovation, Nanyang Technological University, Singapore. Since 2015, he has been an Associate Professor with the School of Automation Science and Engineering, South China University of Technology, Guangzhou, where he is currently with the Human-Robot Intelligence Lab, Center for Brain Computer Interfaces and Brain Information Processing. His current research interests include neural networks, automatic control, humanoid robots, and human–robot interaction.

Ling-Dong Kong (S'18) is currently pursuing the B.Sc. degree in intelligence science and technology with the School of Automation Science and Engineering, South China University of Technology (SCUT), Guangzhou, China.

He is currently with the Human-Robot Intelligence Lab, Center for Brain Computer Interfaces and Brain Information Processing, SCUT. His current research interests include neural networks, machine learning, and robotics.

Mr. Kong was a recipient of the National Scholarship of the Year 2017–2018 awarded by the Ministry of Education of China.

Lunan Zheng (S'18) received the B.Eng. degree in automation from the South China University of Technology (SCUT), Guangzhou, China, in 2017, where he is currently pursuing the M.Sc. degree in pattern recognition and intelligence system with the School of Automation Science and Engineering.

He is currently with the Human-Robot Intelligence Lab, Center for Brain Computer Interfaces and Brain Information Processing, SCUT. His current research interests include neural networks, machine learning, and robotics.