Robustness Analysis of a Power-Type Varying-Parameter Recurrent Neural Network for Solving Time-Varying QM and QP Problems and Applications

Zhijun Zhang, *Member, IEEE*, Lingdong Kong, *Student Member, IEEE*, Lunan Zheng, Pengchao Zhang, Xilong Qu, Bolin Liao, and Zhuliang Yu

***Abstract*—Varying-parameter recurrent neural network, being a special kind of neural-dynamic methodology, has revealed powerful abilities to handle various time-varying problems, such as quadratic minimization (QM) and quadratic programming (QP) problems. In this paper, a novel power-type varying-parameter recurrent neural network (PT-VP-RNN) is proposed to solve the perturbed time-varying QM and QP problems. First, based on the generalization of time-varying QM and QP problems, the design process of the PT-VP-RNN is presented in detail. Second, the robustness performance of the proposed PT-VP-RNN is theoretically analyzed and proved. What is more, two numerical examples are simulated to illustrate the robustness convergence performance of PT-VP-RNN even in a large disturbance condition. Finally, two practical application examples (i.e., a robot tracking example and a venture investment example) further verify the effectiveness, accuracy, and widespread applicability of the proposed PT-VP-RNN.**

***Index Terms*—Disturbance, quadratic minimization (QM), quadratic programming (QP), recurrent neural network, robot motion planning, robustness.**

Manuscript received April 20, 2018; revised July 2, 2018; accepted August 19, 2018. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1002505, in part by the National Natural Science Foundation of China 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 2017MS049, in part by the National Key Basic Research Program of China (973 Program) under Grant 2015CB351703, and in part by the Natural Science Foundation of Guangdong Province under Grant 2014A030312005. This paper was recommended by Associate Editor S. Song. *(Corresponding author: Zhijun Zhang.)*

Z. Zhang, L. Kong, L. Zheng, and Z. Yu are with the School of Automation Science and Engineering, South China University of Technology, Guangzhou 510640, China and also with the Human–Robot Intelligence Laboratory, Center for Brain Computer Interfaces and Brain Information Processing, South China University of Technology, Guangzhou 510640, China (e-mail: auzjzhang@scut.edu.cn; scutkld@foxmail.com; aulnzheng@sina.com; zlyu@scut.edu.cn).

P. Zhang is with the Key Laboratory of Industrial Automation of Shaanxi Province, Shaanxi University of Technology, Hanzhong 710048, China (e-mail: snutzpc@126.com).

X. Qu is with the School of Information Technology and Management, Hunan University of Finance and Economics, Changsha 410006, China (e-mail: quxilong@126.com).

B. Liao is with the School of Information Science and Engineering, Jishou University, Jishou 416000, China (e-mail: bolinliao@jsu.edu.cn).

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/TSMC.2018.2866843

I. Introduction

QUADRATIC minimization (QM) [1] and quadratic programming (QP) [2]–[4] play an important role in robot motion planning [4], operation science [5], economic dispatch and finance [6], management science [7], and combinatorial optimization [8]. The traditional approaches to handle QM and QP problems are the numerical-based methods [9]–[11]. In recent years, with more and more real-time tasks arising, the traditional numerical-based methods cannot meet the requirement of solving the large-scale time-varying QM and QP problems. With the re-emergence of neural network recently, the recurrent neural network is preferred to deal with the QM and QP problems.

Recurrent neural networks have been applied in more and more fields because of the variety of remarkable properties, such as adaptive self-learning capability [12]–[15], distributed storage [16]–[19], and parallelism computation [20]–[23]. One of the most important applications is time-varying problems solving, such as QM and QP problems [4], [13], [20], [24], [25], and the corresponding of practical applications [4], [13], [25], [26]. Xia and Wang [27] proposed a primal dual neural network (PDNN) [28] that with global-exponential convergence ability for solving QP problems with unique solutions. With the PDNN, there is no need to choose the lateral-connection and self-feedback for the network [29]. This model has already been applied to obstacle avoidance of robot arms [30].

However, the design process of PDNN is complex, and a specific projection function must be obtained beforehand, which is difficult. In order to solve the aforementioned QM and QP problems in a straightway, a gradient-based neural network (GNN) was proposed [31], [32]. GNN obtains the solution by designing a scale-valued error function and making it converge to zero. In order to reach the minimum point, a negative gradient is chosen to be the descent direction. Besides, a constant parameter is designed to evaluate the convergence

speed. Nevertheless, GNN has a fatal weakness that when facing time-varying problems, it cannot track the theoretical solution in time [33].

To remedy this disadvantage, a zeroing neural network (ZNN) was proposed by Zhang *et al.* [18], Jin *et al.* [19], Xiao [34], and Li *et al.* [35]. Since the derivatives of the error function is taken into consideration, ZNN contains a predictive ability, which can be applied to tracking theoretical solutions. Howbeit, when facing external disturbance and computation errors, the traditional ZNN cannot efficiently handle the solution tracking task. Theoretical analysis demonstrates that when facing perturbation, the residual errors of ZNN can only achieve to reach an upper bound but cannot converge to zero. That is to say, the residual errors of ZNN in the case of external disturbance would oscillate all the time. Even though Jin *et al.* and Xiao proposed some modified ZNN models to improve the robustness performance [4], [19], [26], [34], the computational efficiency still should be further improved. One of the most popular modified ZNN models, which contains finite-time stability, is named finite-time recurrent neural network (FT-RNN) [36]. Xiao [37] utilized FT-RNN to solve time-varying Sylvester function and proved that this model can converge to the theoretical solutions within finite time. Gu and Cui [38] introduced a efficient algorithm for solving subset sum problems based on FT-RNN. Liu and Wang [39] used FT-RNN to handle optimization problems with piecewise objective functions and designed the block diagram of FT-RNN. Moreover, a tunable activation function named sign-bi-power activation function was led to activated FT-RNN by Shen *et al.* [40]. The different characteristics and simulative performances of these neural networks inspired us to optimize models form the aspect of designing novel parameters.

Since the convergence coefficients of the above models are fixed and set as a constant, they are called fixed-parameter recurrent neural network (termed as FP-RNN). Different from the traditional FP-RNN models, a novel varying-parameter recurrent neural network is proposed in this paper. Since its convergence coefficient is power-type and is changing with time t, it is named power-type varying-parameter recurrent neural network (PT-VP-RNN). In fact, PT-VP-RNN inherits all the advantages of FP-RNN because it is edified by the previous models. Compared with traditional FP-RNN, PT-VP-RNN can achieve faster convergence rate and stronger robustness performance. Both theoretical analysis and simulative results in this paper would illustrate the superiority of PT-VP-RNN. Specifically, the convergent rate of PT-VP-RNN is super-exponential, but that of the state-of-the-art methods are exponential. Meanwhile, the residual errors of PT-VP-RNN can always converge to zero even under large perturbation, while FP-RNN can only guarantee to approximate to an upper bond, but cannot converge to zero. It is worth mentioning that many practical problems based on time-varying QM and QP frameworks contain the potential applications of this method because the design idea of PT-VP-RNN is easy to understand and the theoretical proofs are rigorous.

The remainder of this paper is organized as follows. Section II gives the problem formulation of time-varying QM and QP problems. The design process of the PT-VP-RNN is shown in Section III. In Section IV, we analyze the robustness performance of PT-VP-RNN in detail. Simulative examples and comparisons of PT-VP-RNN and traditional FP-RNN are shown in Section V. Section VI illustrates two practical applications. The conclusion and future research are given in Section VII.

It is worth pointing out that the main contributions of this paper are listed in the following facts.

1) A PT-VP-RNN is proposed to solve the time-varying QM and QP problems in real-time. To the best of the authors' knowledge, it is the first time to propose such kind of recurrent neural network for solving time-varying QM and QP problems in a disturbance condition.
2) Theoretical analysis proves that the residual error of PT-VP-RNN can converge to zero in a disturbance condition. It is much better than the state-of-the-art methods which can only guarantee to have an upper bond, but cannot converge to zero.
3) Simulative examples verify the strong robustness performance of the PT-VP-RNN when solving perturbed time-varying QM and QP problems. Even facing large error situation and different dimensions cases, the proposed PT-VP-RNN still possesses excellent performance.
4) A robot tracking example and a venture investment example further verify the effectiveness, accuracy, and practicability of the proposed PT-VP-RNN.

II. Problem Formulations

In this section, the preliminaries and problem formulations of time-varying QM and QP problems are presented.

A. Time-Varying QM

The standard form of time-varying QM is described as

$$\text{minimize } \mathcal{M}(t) = \frac{1}{2}x^{\mathrm{T}}(t)\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)x(t) \tag{1}$$

where $\mathcal{H}(t) \in \mathbb{R}^{n\times n}$ denotes a time-varying Hessian matrix and it should be smooth and positive-definite for any time instant $t \in [0, +\infty)$; superscript $^{\mathrm{T}}$ denotes the transpose operation of a vector or a matrix; $\mathcal{P}^{\mathrm{T}}(t) \in \mathbb{R}^n$ denotes the coefficient vector; and $x(t) \in \mathbb{R}^n$ denotes an unknown vector which is solved to obtain the minimum value.

The partial derivative of $\mathcal{M}(t)$ is designed to obtain the minimum value, i.e.,

$$\nabla\mathcal{M}(x) = \frac{\partial\mathcal{M}(x)}{\partial x(t)} = \mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t). \tag{2}$$

The optimization value of QM (1) can be obtained by zeroing $\nabla\mathcal{M}(t)$ at any time instant $t \in [0, +\infty)$, i.e.,

$$\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t) = 0 \tag{3}$$

and the theoretical solution $x^*(t)$ to (1) satisfies $x^*(t) = -\mathcal{H}^{-1}(t)\mathcal{P}(t)$. Hence, the theoretical minimum value is obtained, i.e., $\mathcal{M}^* = \mathcal{M}(x^*(t)) = x^{*\mathrm{T}}(t)\mathcal{H}(t)x^*(t)/2 + \mathcal{P}^{\mathrm{T}}x^*(t)$.

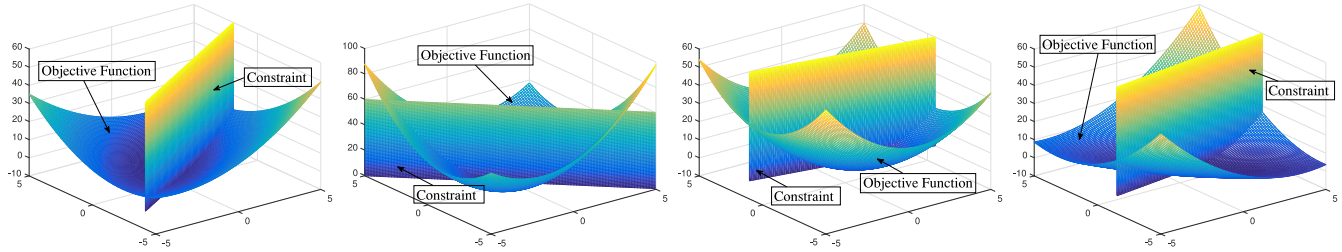


Fig. 1. Three-dimensional plots of quadratic objective function and corresponding linear constraint of a time-varying convex QP problem. (a) $t = 2.8000$. (b) $t = 5.5000$. (c) $t = 7.8000$. (d) $t = 9.9500$. We can see quite evidently that the shape and minimum value together with its minimum solution are all "moving" as time instant t goes, so that the time-varying QP problem could be considered as a "moving minimum" problem. The darker the blue, the closer to the minimum.

B. Time-Varying QP

In practical applications, time-varying QM (1) is not enough to describe the target problem in many fields, for example, robot motion planning problem. Considering a redundant robot manipulator motion tracking task, based on robot motion planning scheme, an equality constraint of time-varying QP is necessary. Hence, the standard form of time-varying QP is described as follows:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^{\mathrm{T}}(t)\mathcal{Q}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)x(t) \\ \text{subject to} \quad & \mathcal{A}(t)x(t) = \mathcal{B}(t) \end{aligned} \tag{4}$$

where vector $x(t) \in \mathbb{R}^n$ at time instant $t \in [0, +\infty)$ is unknown and to be solved in real-time; $\mathcal{Q}(t) \in \mathbb{R}^{n\times n}$ denotes the positive-definite Hessian matrix; $\mathcal{P}(t) \in \mathbb{R}^n$ denotes the coefficient vector; $\mathcal{A}(t) \in \mathbb{R}^{m\times n}$ denotes the full rank coefficient matrix; and $\mathcal{B}(t) \in \mathbb{R}^n$ denotes a coefficient vector. Besides, coefficient matrices $\mathcal{Q}(t)$, $\mathcal{A}(t)$ and vectors $\mathcal{P}(t)$, $\mathcal{B}(t)$, together with their time derivatives $\dot{\mathcal{Q}}(t)$, $\dot{\mathcal{A}}(t)$, $\dot{\mathcal{P}}(t)$, $\dot{\mathcal{B}}(t)$, are assumed to be known and smoothly time-varying, or could be estimated accurately.

To guarantee the uniqueness of the solution, such time-varying QP problem (4) should be strictly convex with positive-definite $\mathcal{Q}(t) \in \mathbb{R}^{n\times n}$ at any time instant $t \in [0, +\infty)$.

In order to solve the time-varying QP problem (4), a Lagrange form of this problem is constructed as below, i.e.,

$$\begin{aligned} \mathcal{L}(x(t), \lambda(t), t) = & \frac{1}{2}x^{\mathrm{T}}(t)\mathcal{Q}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)x(t) \\ & + \lambda^{\mathrm{T}}(t)(\mathcal{A}(t)x(t) - \mathcal{B}(t)), \quad t \in [0, +\infty) \end{aligned} \tag{5}$$

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

According to Lagrange multiplier method [41], for QP problem (4), if both $\partial\mathcal{L}(x(t), \lambda(t), t)/\partial x(t)$ and $\partial\mathcal{L}(x(t), \lambda(t), t)/\partial\lambda(t)$ exist and are continuous, then the optimum solution will be obtained when the following two equations hold truth, i.e.,

$$\begin{aligned} \frac{\partial\mathcal{L}(x(t), \lambda(t), t)}{\partial x(t)} &= \mathcal{Q}(t)x(t) + \mathcal{P}(t) + \mathcal{A}^{\mathrm{T}}(t)\lambda(t) = 0 \\ \frac{\partial\mathcal{L}(x(t), \lambda(t), t)}{\partial\lambda(t)} &= \mathcal{A}(t)x(t) - \mathcal{B}(t) = 0. \end{aligned} \tag{6}$$

Equation (6) can be further rewritten into a matrix form as

$$\mathcal{W}(t)\mathcal{Y}(t) = \mathcal{G}(t) \tag{7}$$

where

$$\begin{aligned} \mathcal{W}(t) &:= \begin{bmatrix} \mathcal{Q}(t) & \mathcal{A}^{\mathrm{T}}(t) \\ \mathcal{A}(t) & \mathbf{0}_{m\times m} \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)} \\ \mathcal{Y}(t) &:= \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix} \in \mathbb{R}^{n+m} \\ \mathcal{G}(t) &:= \begin{bmatrix} -\mathcal{P}(t) \\ \mathcal{B}(t) \end{bmatrix} \in \mathbb{R}^{n+m}. \end{aligned} \tag{8}$$

$\mathcal{W}(t)$ and $\mathcal{G}(t)$ are smoothly time-varying coefficient matrix and vector due to the smoothness and continuity of time-varying coefficient matrices $\mathcal{Q}(t)$, $\mathcal{A}(t)$ and vector $\mathcal{B}(t)$. $\mathcal{Y}(t) \in \mathbb{R}^{(n+m)}$ denotes an unknown vector and it needs to be solved at any time instant t. Solving time-varying QP problem (4) is equivalent to solving the matrix equation (7). Since the convex QP problem (4) is time-varying, i.e., coefficient vectors and matrices are changing as time t goes, the theoretical solutions will change all the time (as shown in Fig. 1). For getting better robustness property, steady states are desired if the time-varying optimal solution is expected to be obtained. In order to better illustrate the performance of the proposed algorithm, the time-varying theoretical solution can be written as

$$\mathcal{Y}^*(t) = \left[x^{*\mathrm{T}}(t), \lambda^{*\mathrm{T}}(t)\right]^{\mathrm{T}} = \mathcal{W}^{-1}(t)\mathcal{G}(t) \in \mathbb{R}^{n+m}. \tag{9}$$

III. PT-VP-RNN Model

In order to obtain the solution to the convex time-varying QM (1) and QP (4) problems, a novel neural network can be designed by the following three steps.

Step 1: A vector-type error function can be defined as follows.

1) For time-varying QM (1), according to (3)

$$\varepsilon(t) = \mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t) \in \mathbb{R}^{n+m}. \tag{10}$$

2) For time-varying QP (4), according to (7)

$$\varepsilon(t) = \mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t) \in \mathbb{R}^{n+m}. \tag{11}$$

Step 2: To make the error function $\varepsilon(t)$ approach zero, the negative time derivative of error function $\varepsilon(t)$ is necessary.

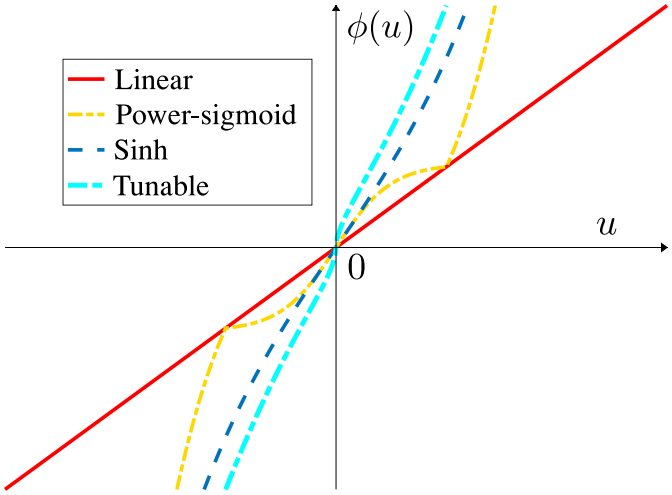


Fig. 2. Four activation functions applied in our simulations, i.e., linear activation function $\phi_1(u)$ (the red solid line), power-sigmoid activation function $\phi_2(u)$ with $\xi = 4$ and $\omega = 3$ (the yellow dashed-dotted line), sinh activation function $\phi_3(u)$ (the blue dash line), and tunable activation function $\phi_4(u)$ with $r = 0.5$ (the cyan dashed-dotted line).

Based on our previous design experience [14], [42], a power-type varying-parameter neural dynamic design formula can be described as

$$\dot{\varepsilon}(t) = \frac{\mathrm{d}\varepsilon(t)}{\mathrm{d}t} = -\big(\gamma + t^{\gamma}\big)\Phi(\varepsilon(t)) \tag{12}$$

where $\gamma > 0$ denotes the constant scalar-valued parameter, and the design parameter $\gamma + t^{\gamma}$ is utilized to scale the convergence rate of the formula. Different from the traditional neural dynamic design approach, the design parameter $\gamma + t^{\gamma}$ is changing as time instant t goes, and thus is named varying-parameter neural dynamic design method. $\Phi(\cdot)$ denotes the activation-function processing-array. In addition, each scalar-valued processing-unit $\phi(\cdot)$ of $\Phi(\cdot)$ should be a monotonically increasing odd activation function. In this paper, four different kinds of activation functions are applied and analyzed (as shown in Fig. 2), and we illustrate them as follows.

1) Linear-type activation function

$$\phi_1(u) = u.$$

2) Power-sigmoid-type activation function

$$\phi_2(u) = \begin{cases} \frac{1+\exp(-\xi)}{1-\exp(-\xi)} \cdot \frac{1-\exp(-\xi u)}{1+\exp(-\xi u)}, & \text{if } |u| \le 1 \\ u^{\omega}, & \text{otherwise} \end{cases}$$

with $\xi \ge 2$ and $\omega \ge 3$.

3) Sinh-type activation function

$$\phi_3(u) = ([\exp(u) - \exp(-u)]/2).$$

4) Tunable-type activation function $\phi_4(u) = \mathrm{sig}^r(u) + \mathrm{sig}(u) + \mathrm{sig}^{(1/r)}(u)$ with $r > 0$ and $r \ne 1$. Function $\mathrm{sig}^r(u)$ is defined as

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

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

Step 3: By expanding the neural dynamic design formula (12), a novel PT-VP-RNN is obtained.

1) For time-varying QM (1)

$$\mathcal{H}(t)\dot{x}(t) = -\dot{\mathcal{H}}(t)x(t) - \big(\gamma + t^{\gamma}\big)\Phi\big(\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)\big) - \dot{\mathcal{P}}^{\mathrm{T}}(t). \tag{13}$$

2) For time-varying QP (4)

$$\mathcal{W}(t)\dot{\mathcal{Y}}(t) = -\dot{\mathcal{W}}(t)\mathcal{Y}(t) - \big(\gamma + t^{\gamma}\big) \times \Phi(\mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)) + \dot{\mathcal{G}}(t). \tag{14}$$

The above (14) can be rewritten as

$$\dot{\mathcal{Y}}(t) = (I(t) - \mathcal{W}(t))\dot{\mathcal{Y}}(t) - \dot{\mathcal{W}}(t)\mathcal{Y}(t) - \big(\gamma + t^{\gamma}\big) \times \Phi(\mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)) + \dot{\mathcal{G}}(t) \tag{15}$$

where $I(t)$ denotes the identity matrix.

The ith-neural dynamic equation of PT-VP-RNN is

$$\dot{\mathcal{Y}}_i = \sum_{j=1}^{n+m}(I_{ij} - \mathcal{W}_{ij})\dot{\mathcal{Y}}_i - \sum_{j=1}^{n+m}\dot{\mathcal{W}}_{ij}\mathcal{Y}_j - \big(\gamma + t^{\gamma}\big) \times \phi\left(\sum_{j=1}^{n+m}\mathcal{W}_{ij}\mathcal{Y}_j - \mathcal{G}_i\right) + \dot{\mathcal{G}}_i \tag{16}$$

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

According to (16), the topological graph of the PT-VP-RNN is depicted in Fig. 3.

For the traditional neural dynamic method, the design parameter is set as a constant, and thus it is named FP-RNN. As for time-varying QM (1) and QP (4) problems, by omitting time-varying term t^{γ}, PT-VP-RNN (14) can be degenerated into the corresponding of FP-RNN model.

1) For time-varying QM (1)

$$\mathcal{H}(t)\dot{x}(t) = -\dot{\mathcal{H}}(t)x(t) - \gamma\Phi\big(\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)\big) - \dot{\mathcal{P}}^{\mathrm{T}}(t). \tag{17}$$

2) For time-varying QP (4)

$$\mathcal{W}(t)\dot{\mathcal{Y}}(t) = -\dot{\mathcal{W}}(t)\mathcal{Y}(t) - \gamma\Phi(\mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)) + \dot{\mathcal{G}}(t). \tag{18}$$

Remark 1: Mathematically, it can be seen from (14) and (18) that the PT-VP-RNN would degrade into ZNN if the time-varying parameter term t^{γ} is omitted. The PT-VP-RNN can be considered a more general form of ZNN, and ZNN can be considered as a particular case of PT-VP-RNN with $t = 0$. Besides, this difference leads to the following three distinctions.

1) The design parameter of PT-VP-RNN takes time variable t into consideration while FP-RNN does not consider it. In other words, the convergent parameter of the FP-RNN is fixed, while the proposed PT-VP-RNN is time-varying.
2) Due to the influence of time-varying parameters, parameter γ just needs to be set as a small value, and the practical performance of PT-VP-RNN would be excellent.

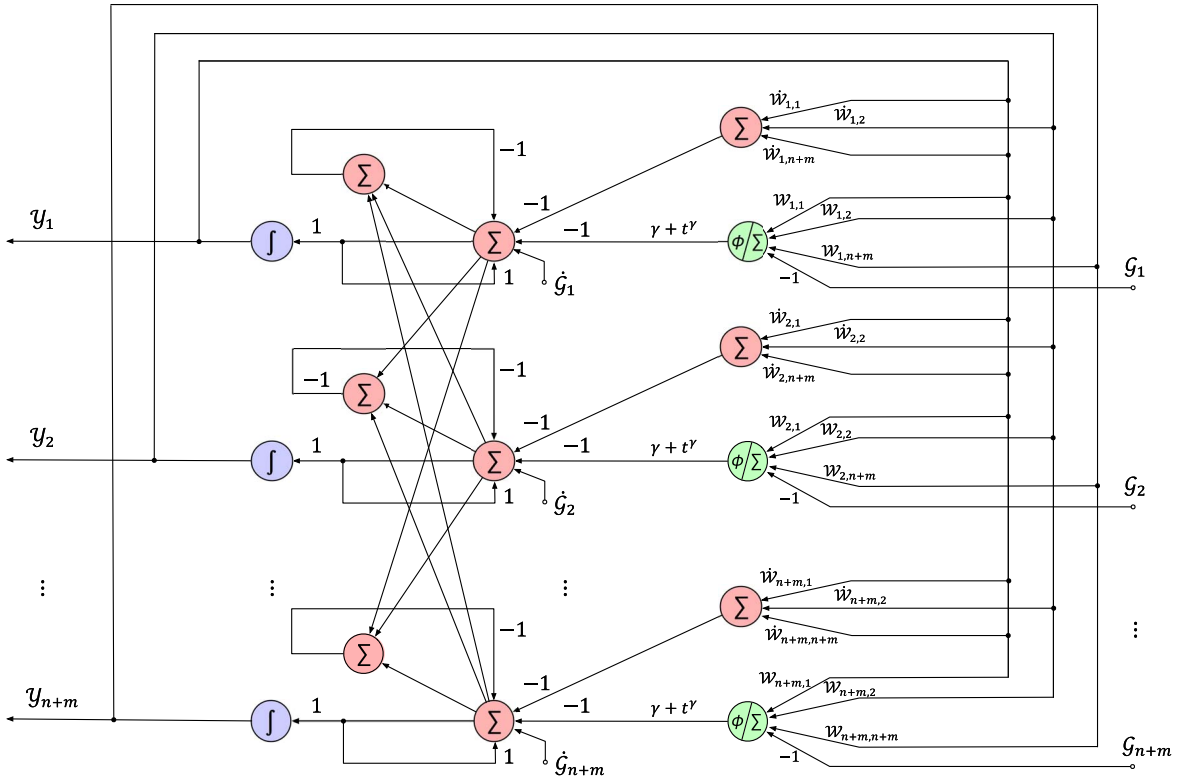


Fig. 3. Neural topological graph of the proposed PT-VP-RNN.

3) Theoretical analysis and mathematic proofs demonstrate that the robustness of PT-VP-RNN is much better than that of FP-RNN. Specifically, when solving disturbed optimization problems, the errors of FP-RNN is upper-bounded, while that of PT-VP-RNN can converge to zero.

IV. Theoretical Analysis of Robustness

In actual hardware implementation, perturbations and errors often exist, for example, differentiation error. Therefore, it is necessary to consider the robustness of a system with perturbation.

The perturbed model of PT-VP-RNN is described as follows.

1) For time-varying QM (1)

$$\mathcal{H}(t)\dot{x}(t) = -\big(\dot{\mathcal{H}}(t) + \Delta\mathcal{D}(t)\big)x(t) - \big(\gamma + t^\gamma\big) \times \Phi\big(\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)\big) - \dot{\mathcal{P}}^{\mathrm{T}}(t) + \Delta\mathcal{K}(t). \tag{19}$$

2) For time-varying QP (4)

$$\mathcal{W}(t)\dot{\mathcal{Y}}(t) = -\big(\dot{\mathcal{W}}(t) + \Delta\mathcal{D}(t)\big)\mathcal{Y}(t) - \big(\gamma + t^\gamma\big) \times \Phi(\mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)) + \dot{\mathcal{G}}(t) + \Delta\mathcal{K}(t). \tag{20}$$

where $\Delta\mathcal{D}(t) \in \mathbb{R}^{(n+m)\times(n+m)}$ denotes the differentiation error of coefficient matrix $\mathcal{W}(t)$, and $\Delta\mathcal{K}(t) \in \mathbb{R}^{n+m}$ denotes the model-implementation error of the proposed PT-VP-RNN solver.

Considering the similarity and space limitation, only the robustness analysis of PT-VP-RNN for solving QP problem (20) is discussed in this section, and the robustness analysis of QM problem (19) is omitted here.

Theorem 1 (Upper Bound Theorem): For time instant $t \in [0, +\infty)$ and $0 < \vartheta_{\mathcal{D}}, \vartheta_{\mathcal{K}}, \delta_{\mathcal{W}}, \delta_{\mathcal{G}} < +\infty$, if $||\Delta\mathcal{D}(t)||_F \leq \vartheta_{\mathcal{D}} \in \mathbb{R}$, $||\Delta\mathcal{K}(t)||_2 \leq \vartheta_{\mathcal{K}} \in \mathbb{R}$, $||\mathcal{W}^{-1}(t)||_F \leq \delta_{\mathcal{W}} \in \mathbb{R}$, $||\mathcal{G}(t)||_2 \leq \delta_{\mathcal{G}} \in \mathbb{R}$, then the upper bound of the absolute value of computation error $\varepsilon(t) = \mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)$ corresponding to the perturbed PT-VP-RNN model (20) when using a linear activation function is $[(1+\sqrt{n+m})\tau]/2$, where $\tau = (\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}})/(\alpha(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}})$ with $\alpha \geq 1$, and it will converge to zero as time instant $t \to +\infty$ with PT-VP-RNN solver in the case of $\alpha(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}} > 0$.

Proof: For further discussion, a Lyapunov function candidate [43] is defined as

$$\mathcal{V}(t) = \frac{1}{2}||\varepsilon(t)||_2^2 = \frac{1}{2}\varepsilon^{\mathrm{T}}(t)\varepsilon(t) = \sum_{i=1}^{n+m}\frac{1}{2}\varepsilon_i^2(t) \geq 0 \tag{21}$$

where $\varepsilon(t) = \mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)$ and $||\cdot||_2$ denotes the Euclidean norm of a vector. The time derivative of Lyapunov function $\mathcal{V}(t)$ is

$$\dot{\mathcal{V}}(t) = \frac{\mathrm{d}\mathcal{V}(t)}{\mathrm{d}t} = \varepsilon^{\mathrm{T}}(t)\frac{\mathrm{d}\varepsilon(t)}{\mathrm{d}t} = \varepsilon^{\mathrm{T}}(t)\dot{\varepsilon}(t). \tag{22}$$

Substituting (12) into (22), we have

$$\begin{aligned}\dot{\mathcal{V}}(t) &= -\big(\gamma + t^\gamma\big)\varepsilon^{\mathrm{T}}(t)\Phi(\varepsilon(t)) \\ &= -\big(\gamma + t^\gamma\big)\sum_{i=1}^{n+m}\varepsilon_i(t)\phi(\varepsilon_i(t))\end{aligned} \tag{23}$$

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

1) If $\varepsilon_i(t) > 0$ or $\varepsilon_i(t) < 0$, then $\varepsilon_i(t)\phi(\varepsilon_i(t)) > 0$ and $\dot{\mathcal{V}}(t) < 0$.
2) If and only if $\varepsilon_i(t) = 0$, then $\varepsilon_i(t)\phi(\varepsilon_i(t)) = 0$ and $\dot{\mathcal{V}}(t) = 0$.

Considering the definition of the error vector in (11), i.e., $\varepsilon(t) = \mathcal{W}(t)\mathcal{Y}(t) - \mathcal{G}(t)$, we have $\mathcal{Y}(t) = \mathcal{W}^{-1}(t)(\varepsilon(t) + \mathcal{G}(t))$. The time derivative of the error vector $\varepsilon(t)$ is $\dot{\varepsilon}(t) = \mathcal{W}(t)\dot{\mathcal{Y}}(t) + \dot{\mathcal{W}}(t)\mathcal{Y}(t) - \dot{\mathcal{G}}(t)$. Substituting these three equations into the perturbed PT-VP-RNN model (20) and $\dot{\varepsilon}(t)$ is reformulated as

$$\begin{aligned}\dot{\varepsilon}(t) = &-(\gamma + t^{\gamma})\Phi(\varepsilon(t)) - \Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\varepsilon(t)\\ &+ \Delta\mathcal{K}(t) - \Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\mathcal{G}(t).\end{aligned} \tag{24}$$

The time derivative of Lyapunov function with perturbation is

$$\begin{aligned}\dot{\mathcal{V}}(t) &= \varepsilon^{\mathrm{T}}(t)\dot{\varepsilon}(t)\\ &= \varepsilon^{\mathrm{T}}(t)\Big(-\big(\gamma + t^{\gamma}\big)\Phi(\varepsilon(t)) - \Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\varepsilon(t)\\ &\qquad + \Delta\mathcal{K}(t) - \Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\\ &= -\big(\gamma + t^{\gamma}\big)\varepsilon^{\mathrm{T}}(t)\Phi(\varepsilon(t)) + \varepsilon^{\mathrm{T}}(t)\mathcal{T}(t)\varepsilon(t)\\ &\quad + \varepsilon^{\mathrm{T}}(t)\Delta\mathcal{K}(t) + \varepsilon^{\mathrm{T}}(t)\Big(-\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\\ &= -\big(\gamma + t^{\gamma}\big)\varepsilon^{\mathrm{T}}(t)\Phi(\varepsilon(t)) + \varepsilon^{\mathrm{T}}(t)\frac{\mathcal{T}(t) + \mathcal{T}^{\mathrm{T}}(t)}{2}\\ &\quad + \varepsilon^{\mathrm{T}}(t)\Delta\mathcal{K}(t) + \varepsilon^{\mathrm{T}}(t)\Big(-\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\end{aligned} \tag{25}$$

where $\mathcal{T}(t) = -\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)$.

With $|\lambda_{\max}(\cdot)| \leq ||\cdot||_{\mathrm{F}}$, the second term of the right hand side of (25) can be written as the following inequalities, i.e.,

$$\begin{aligned}&\varepsilon^{\mathrm{T}}(t)\frac{\mathcal{T}(t) + \mathcal{T}^{\mathrm{T}}(t)}{2}\varepsilon(t)\\ &\leq \varepsilon^{\mathrm{T}}(t)\varepsilon(t)\left|\lambda_{\max}\left(\frac{\mathcal{T}(t) + \mathcal{T}^{\mathrm{T}}(t)}{2}\right)\right|\\ &\leq \varepsilon^{\mathrm{T}}(t)\varepsilon(t)\left|\left|\frac{\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t) + \big(\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)\big)^{\mathrm{T}}}{2}\right|\right|_{\mathrm{F}}\\ &\leq \varepsilon^{\mathrm{T}}(t)\varepsilon(t)||\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)||_{\mathrm{F}}\end{aligned} \tag{26}$$

where $||\cdot||_{\mathrm{F}}$ denotes the Frobenius norm of a vector.

According to matrix theory [44], i.e., $||\Delta\mathcal{D}(t)\mathcal{W}^{-1}(t)||_{\mathrm{F}} \leq ||\Delta\mathcal{D}(t)||_{\mathrm{F}}||\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}$, (26) is reformulated as

$$\begin{aligned}&\varepsilon^{\mathrm{T}}(t)\frac{\mathcal{T}(t) + \mathcal{T}^{\mathrm{T}}(t)}{2}\varepsilon(t)\\ &\leq \varepsilon^{\mathrm{T}}(t)\varepsilon(t)||\Delta\mathcal{D}(t)||_{\mathrm{F}}||\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}\\ &\leq \varepsilon^{\mathrm{T}}(t)\varepsilon(t)\vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\end{aligned} \tag{27}$$

where $\vartheta_{\mathcal{D}}$ and $\delta_{\mathcal{W}}$ are the upper bounds of $||\Delta\mathcal{D}(t)||_{\mathrm{F}}$ and $||\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}$, respectively.

Since each element of a vector is less than or equivalent to the maximum of elements of a vector, the third term of the right hand side of (25) satisfies the following inequality, i.e.,

$$\Delta\mathcal{K}(t) \leq \sum_{i=1}^{n+m}\max_{1\leq i\leq n+m}|\Delta\mathcal{K}_i(t)|. \tag{28}$$

From $\max_{1\leq i\leq n+m}|\Delta\mathcal{K}_i(t)| \leq ||\Delta\mathcal{K}(t)||_2$, (28) is further written as

$$\begin{aligned}\varepsilon^{\mathrm{T}}(t)\Delta\mathcal{K}(t) &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot||\Delta\mathcal{K}(t)||_2\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot\vartheta_{\mathcal{K}}\end{aligned} \tag{29}$$

where $\vartheta_{\mathcal{K}}$ is the upper bound of $||\Delta\mathcal{K}(t)||_{\mathrm{F}}$.

Similarly, the fourth term of the right hand side of (25) satisfies the following inequalities, i.e.,

$$\begin{aligned}&\varepsilon^{\mathrm{T}}(t)\cdot\Big(-\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\max_{1\leq i\leq n+m}\Big|\Big(\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)_i\Big|\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot||\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)\mathcal{G}(t)||_2.\end{aligned} \tag{30}$$

According to Cauchy–Buniakowsky–Schwarz inequality $||\beta\cdot\chi||_2 \leq ||\beta||_2\cdot||\chi||_{\mathrm{F}}$ (where β is a vector and χ is a matrix), (30) is reformulated as

$$\begin{aligned}&\varepsilon^{\mathrm{T}}(t)\cdot\Big(-\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot||\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}\cdot||\mathcal{G}(t)||_2.\end{aligned} \tag{31}$$

Based on [44], we have $||\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}} \leq ||\Delta\mathcal{D}(t)||_{\mathrm{F}}\cdot||\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}$, (31) is reformulated as

$$\begin{aligned}&\varepsilon^{\mathrm{T}}(t)\cdot\Big(-\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)\mathcal{G}(t)\Big)\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot||\Delta\mathcal{D}(t)\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}\cdot||\mathcal{G}(t)||_2\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot||\Delta\mathcal{D}(t)||_{\mathrm{F}}\cdot||\Delta\mathcal{W}^{-1}(t)||_{\mathrm{F}}\cdot||\mathcal{G}(t)||_2\\ &\leq \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot\vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}.\end{aligned} \tag{32}$$

Substituting inequalities (27), (29), and (32) into (25), the following inequality is obtained, i.e.,

$$\begin{aligned}\dot{\mathcal{V}}(t) \leq &-\big(\gamma + t^{\gamma}\big)\varepsilon^{\mathrm{T}}(t)\Phi(\varepsilon_i(t)) + \varepsilon^{\mathrm{T}}(t)\vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\varepsilon(t)\\ &+ \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot\vartheta_{\mathcal{K}} + \sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot\vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}\\ = &-\sum_{i=1}^{n+m}|\varepsilon_i(t)|\cdot\Big(\big(\gamma + t^{\gamma}\big)\Phi(|\varepsilon_i(t)|) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}|\varepsilon_i(t)|\\ &- \vartheta_{\mathcal{K}} - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}\Big).\end{aligned} \tag{33}$$

In order to analyze the above inequality easily, we can define $\Upsilon(\varepsilon_i(t)) = (\gamma + t^\gamma)\Phi(|\varepsilon_i(t)|) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}|\varepsilon_i(t)| - \vartheta_{\mathcal{K}} - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}$. Evidently, $\dot{\mathcal{V}}(t)$ is determined by $\Upsilon(\varepsilon_i(t))$ which can be discussed by the following two cases, i.e.,

$$\Upsilon(\varepsilon_i(t)) = \begin{cases} \geq 0, & \forall i = 1, 2, \ldots, n+m \\ < 0, & \exists i = 1, 2, \ldots, n+m. \end{cases} \tag{34}$$

First, If $\Upsilon(\varepsilon_i(t)) \geq 0$, then $\dot{\mathcal{V}} \leq 0$. According to the second theorem of Lyapunov stability, as $\mathcal{V}(t) \geq 0$ and $\dot{\mathcal{V}}(t) < 0$ [i.e., (21) and inequality (33)], the error vector $\varepsilon(t)$ will converge to zero, which means that the state vector $\mathcal{Y}(t)$ will converge to the theoretical solution $\mathcal{Y}^*(t)$. In addition, the steady state of the system will be reached and $\mathcal{V}(t)$ will stop decreasing when $\dot{\mathcal{V}}(t) = 0$.

Second, if $\Upsilon(\varepsilon_i(t)) < 0$, then the right hand side of (33) is positive, which means that $\dot{\mathcal{V}}(t)$ has an upper bound. There exist two conditions.

1) If $\dot{\mathcal{V}}(t) < 0$, with $\mathcal{V}(t) \geq 0$, the error vector will converge to zero, and the steady state of the system will be reached.
2) If $\dot{\mathcal{V}}(t) > 0$, without loss of generality, considering linear activation function $\phi(|\varepsilon_i(t)|) = \tau|\varepsilon_i(t)|(\tau \geq 1)$, (33) is reformulated as

$$\begin{aligned} \dot{\mathcal{V}}(t) &\leq -\sum_{i=1}^{n+m} |\varepsilon_i(t)|\big(\gamma + t^\gamma\big)\tau\big|\varepsilon_i(t)| - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}|\varepsilon_i(t)| \\ &\quad -\vartheta_{\mathcal{K}} - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}} \\ &= -\big(\tau\big(\gamma + t^\gamma\big) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\big)\sum_{i=1}^{n+m} |\varepsilon_i(t)| \\ &\quad \times \left(|\varepsilon_i(t)| - \frac{\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}}{\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}}\right). \end{aligned} \tag{35}$$

Considering $\dot{\mathcal{V}}(t) > 0$, $\mathcal{V}(t)$ will increase as time t goes, which leads to the increase of $|\varepsilon_i(t)|$. The upper bound of $\dot{\mathcal{V}}(t)$ will decrease with the increase of $|\varepsilon_i(t)|$. Therefore, there always exists a certain moment when $\dot{\mathcal{V}}(t) \leq 0$, and the system would be stable again at that moment. In other words, $|\varepsilon_i(t)|$ has an upper bound, and it will be reached when $\dot{\mathcal{V}}(t) = 0$. For the convenience of further discussion, we can define $\varpi = (\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}})/(\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}})$. When $\dot{\mathcal{V}}(t) = 0$, the right hand side of (35) is equivalent to zero, i.e., $\sum_{i=1}^{n+m} |\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi) = 0$. Since $|\varepsilon_i(t)|$ is an independent variable, function $|\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi)$ associated with $|\varepsilon_i(t)|$ has a negative minimum value, which can be obtained when $|\varepsilon_i(t)| = \varpi/2$.

In conclusion, $|\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi)$ is positive when $|\varepsilon_i(t)| > \varpi$. Since $\sum_{i=1}^{n+m} |\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi) = 0$, i.e., the sum of all the $n+m$ terms of $|\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi)$ $(i = 1, 2, \ldots, j, \ldots, n+m)$ is zero, the upper bound $|\varepsilon_j(t)|$ will be obtained if and only if the rest of the $n+m-1$ terms of $|\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi)$ reach the minimum points, which means $|\varepsilon_i(t)| = \varpi/2$. To quantitatively illustrate our discussion, when $\dot{\mathcal{V}}(t) = 0$, i.e.,

$$\begin{aligned} &\sum_{i=1}^{n+m} |\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi) \\ &= |\varepsilon_j(t)|\big(|\varepsilon_j(t)| - \varpi\big) + \sum_{i=1, i\neq j}^{n+m} |\varepsilon_i(t)|(|\varepsilon_i(t)| - \varpi) \\ &= |\varepsilon_j(t)|^2 - \varpi|\varepsilon_j(t)| + \sum_{i=1, i\neq j}^{n+m} |\varepsilon_i(t)|(|\varepsilon_i(t)| - \varpi) \\ &= 0. \end{aligned} \tag{36}$$

Substituting $\varpi = (\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}})/(\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}})$ into (36) and the following equation is obtained, i.e.,

$$\begin{aligned} &\sum_{i=1}^{n+m} |\varepsilon_i(t)| \cdot (|\varepsilon_i(t)| - \varpi) \\ &= |\varepsilon_j(t)|^2 - |\varepsilon_j(t)|\left(\frac{\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}}{\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}}\right) \\ &\quad - \frac{n+m-1}{4}\left(\frac{\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}}}{\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}}\right)^2 \\ &= 0. \end{aligned} \tag{37}$$

Hence, $|\varepsilon_i(t)|$ will reach the upper bound when $i = j$, and the upper bound is

$$\varepsilon_j(t) = \frac{1}{2}\big(1 + \sqrt{n+m}\big)\rho \tag{38}$$

where $\rho = (\vartheta_{\mathcal{K}} + \vartheta_{\mathcal{D}}\delta_{\mathcal{W}}\delta_{\mathcal{G}})/(\tau(\gamma + t^\gamma) - \vartheta_{\mathcal{D}}\delta_{\mathcal{W}})$, and $|\varepsilon_{n+m}(t)|$ will converge to zero when time $t \to +\infty$.

The proof is thus completed. ■

V. Illustrative Examples

In this section, comparative simulations are conducted to verify the aforementioned theoretical analysis and the effectiveness of the proposed PT-VP-RNN (14) for solving time-varying QM and QP problems.

The simulations are performed with MATLAB R2016a, on a MacBook Air (2015) with Intel Core i5 CPU at 1.6 GHz, 1600 MHz and 4GB of RAM.

Considering the following time-varying QM and QP problems.

1) Time-varying QM

$$\text{minimize } \frac{1}{2}x^{\mathrm{T}}(t)\mathcal{H}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)x(t). \tag{39}$$

2) Time-varying QP

$$\begin{aligned} &\text{minimize } \frac{1}{2}x^{\mathrm{T}}(t)\mathcal{Q}(t)x(t) + \mathcal{P}^{\mathrm{T}}(t)x(t) \\ &\text{subject to } \mathcal{A}(t)x(t) = \mathcal{B}(t) \end{aligned} \tag{40}$$

where

$$\mathcal{H}(t) := \begin{bmatrix} \sin t + 2 & \sin t \\ \cos t & \sin t + 2 \end{bmatrix}$$

$$\mathcal{W}(t) := \begin{bmatrix} \sin t + 2 & \sin t & \sin 3t \\ \cos t & \sin t + 2 & \cos 3t \\ \sin 3t & \cos 3t & 0 \end{bmatrix}$$

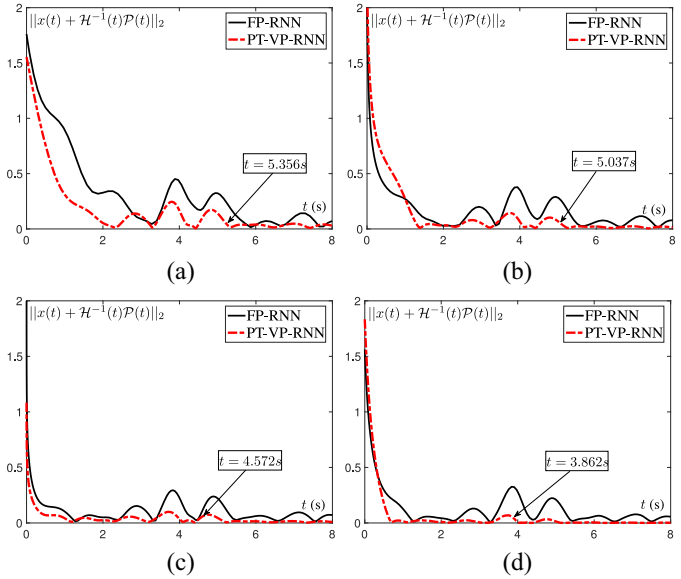


Fig. 4. Residual errors $||x(t)+\mathcal{H}^{-1}(t)\mathcal{P}(t)||_2$ of the perturbed PT-VP-RNN and FP-RNN (with $\gamma = 1$) when solving time-varying QM (39) with different activation functions during $t \in [0, 8]$. (a) Linear. (b) Power-sigmoid. (c) Sinh. (d) Tunable.

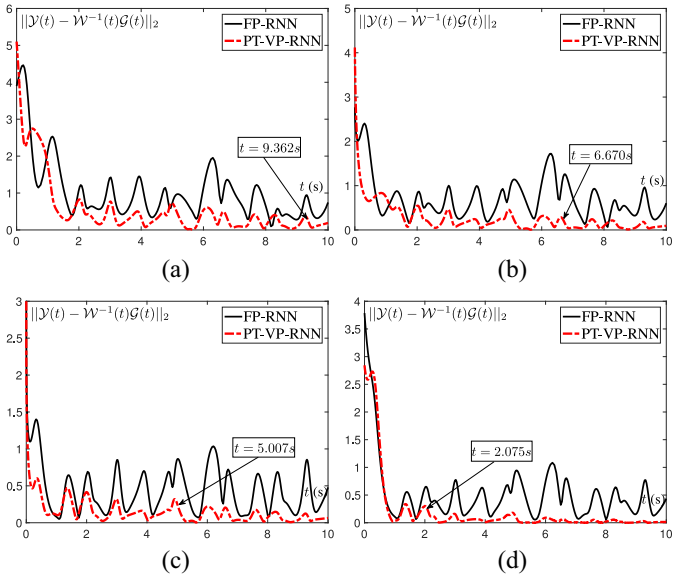


Fig. 5. Residual errors $||\mathcal{Y}(t)-\mathcal{W}^{-1}(t)\mathcal{G}(t)||_2$ of the perturbed PT-VP-RNN and FP-RNN (with $\gamma = 1$) for solving time-varying QP problem (40) with different activation functions during $t \in [0, 10]$. (a) Linear. (b) Power-sigmoid. (c) Sinh. (d) Tunable.

$$\mathcal{P}(t) \coloneqq \begin{bmatrix}\sin 2t & \cos 2t\end{bmatrix}^{\mathrm{T}}, \quad \mathcal{G}(t) \coloneqq \begin{bmatrix}\sin 2t & \cos 2t & -\cos 2t\end{bmatrix}^{\mathrm{T}}$$
$$\mathcal{A}(t) \coloneqq [\sin 3t \ \ \cos 3t], \quad \mathcal{B}(t) \coloneqq \cos 2t$$
$$x(t) \coloneqq [x_1(t) \ x_2(t)]^{\mathrm{T}}, \quad \mathcal{Y}(t) \coloneqq [x_1(t) \ x_2(t) \ \lambda_1(t)]^{\mathrm{T}}.$$

For illustrating the robustness, the model-implementation errors considered in the perturbed PT-VP-RNN (20) are as follows.

1) For time-varying QM

$$\Delta\mathcal{D}(t) \coloneqq \vartheta_{\mathcal{D}}\begin{bmatrix}\cos 4t & -\sin 4t \\ \sin 4t & \cos 4t\end{bmatrix}$$
$$\Delta\mathcal{K}(t) \coloneqq \vartheta_{\mathcal{K}}\begin{bmatrix}\sin 4t \\ \cos 4t\end{bmatrix}. \tag{41}$$

2) For time-varying QP

$$\Delta\mathcal{D}(t) \coloneqq \vartheta_{\mathcal{D}}\begin{bmatrix}\cos 4t & -\sin 4t & \cos 3t \\ \sin 4t & \cos 4t & \sin 3t \\ \cos 3t & \sin 3t & 0\end{bmatrix}$$
$$\Delta\mathcal{K}(t) \coloneqq \vartheta_{\mathcal{K}}\begin{bmatrix}\sin 4t \\ \cos 4t \\ \sin 4t\end{bmatrix}. \tag{42}$$

with $\vartheta_{\mathcal{D}} = \vartheta_{\mathcal{K}} = 0.6$.

A. Error Analysis

In this part, the error convergence of the perturbed PT-VP-RNN and the traditional FP-RNN for solving time-varying QM (39) and QP (40) problems is analyzed. A large error situation is considered to illustrate the strong robustness of PT-VP-RNN. In addition, how to set the design parameter γ is also discussed.

First, the proposed perturbed PT-VP-RNN is applied to solving a time-varying QM [i.e., considering errors $\Delta\mathcal{D}(t)$ and $\Delta\mathcal{K}(t)$ in (41)], and the simulation results are shown in Fig. 4. From the error curves shown in Fig. 4, we can see that, starting from randomly initial state $x(0)$, using whatever activation functions, the residual errors $||x(t)+\mathcal{H}^{-1}(t)\mathcal{P}(t)||_2$ of time-varying QM (39) converge to zero. During this solving process, the convergence time is very short. Specifically, if the error threshold is set as 0.05, then the convergence time of the PT-VP-RNN is only 5.356 s by using a linear activation function, which the traditional FP-RNN cannot. Under the same error threshold, the convergent time of PT-VP-RNN by using power-sigmoid, sinh, and tunable activation functions are 5.037 s, 4.572 s, and 3.862 s, respectively. This above example verifies the effectiveness of the proposed PT-VP-RNN with four different activation functions.

Second, considering an equality constrained time-varying QP problem (40) solved by the perturbed PT-VP-RNN (20) [the perturbed terms are $\Delta\mathcal{D}(t)$ and $\Delta\mathcal{K}(t)$ in (42)]. The residual errors is shown in Fig. 5. As is shown in this figure, wherever the initial state $x(0)$ starts, all the state variables converge to the theoretical solution eternally, that is to say, the residual errors converge to zero. When error threshold is set as 0.25, the convergent time of time-varying QP problem solved by the perturbed PT-VP-RNN (20) with a linear, power-sigmoid, sinh, and tunable activation functions are 9.362 s, 6.670 s, 5.007 s, and 2.075 s, respectively. In summary, the simulation results prove the excellent robustness performance of the proposed perturbed PT-VP-RNN (20) with different activation functions for solving time-varying QP problem, that is to say, the Theorem 1 in Section IV.

Third, in order to illustrate the strong robustness and superiority of the proposed perturbed PT-VP-RNN (20), strong perturbation when solving the time-varying QM and QP problems, i.e., large errors $\vartheta_{\mathcal{D}} = \vartheta_{\mathcal{K}} = 10$, are considered. For comparisons, the perturbed FP-RNN model is also illustrated.

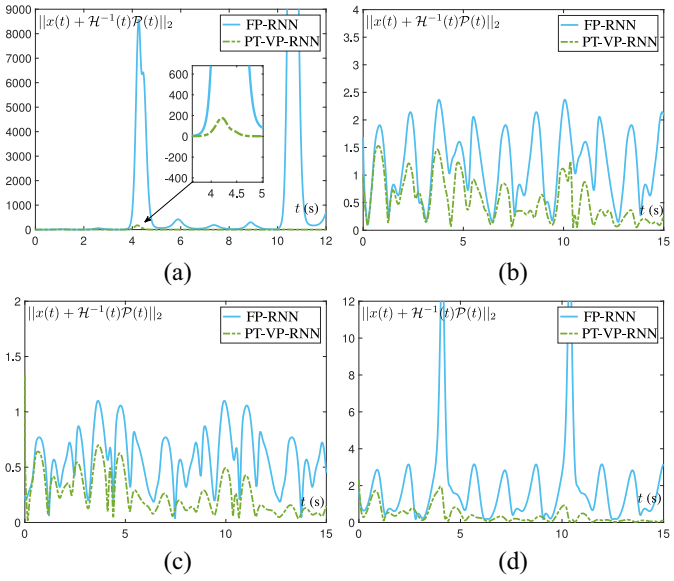


Fig. 6. Computational errors of the perturbed PT-VP-RNN and FP-RNN (with $\gamma = 1$ and large errors $\vartheta_{\mathcal{D}} = \vartheta_{\mathcal{K}} = 10$) for solving time-varying QM (39) with different activation functions. (a) Linear. (b) Power-sigmoid. (c) Sinh. (d) Tunable.

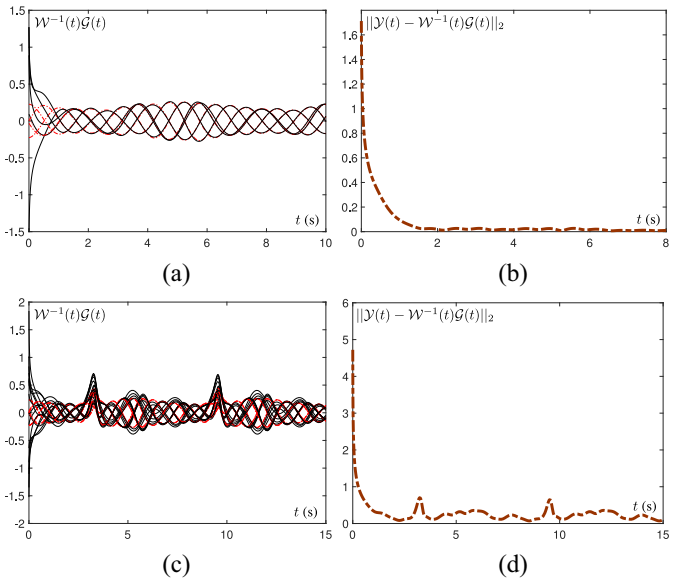


Fig. 7. Entry trajectories of the theoretical solution (the red dash-dot curves) and PT-VP-RNN computed solution (the black solid curves) using power-sigmoid activation function with different dimensions n, and the residual errors. (a) Solutions of $n = 5$. (b) Residual error of $n = 5$. (c) Solutions of $n = 15$. (d) Residual error of $n = 15$.

TABLE I
CONVERGENT TIME t (S) WHEN RESIDUAL ERROR $\varepsilon(t)$ OF QP PROBLEM (40) REACHES 0.25 VIA PT-VP-RNN WITH FOUR ACTIVATION FUNCTIONS

Activation functions	Linear	Power-sigmoid	Sinh	Tunable
$\gamma = 2$	$3.064s$	$2.092s$	$1.456s$	$0.541s$
$\gamma = 4$	$1.152s$	$1.397s$	$0.442s$	$0.358s$
$\gamma = 6$	$1.439s$	$0.446s$	$0.341s$	$0.197s$
$\gamma = 8$	$1.361s$	$0.385s$	$0.076s$	$0.084s$
$\gamma = 10$	$0.451s$	$0.158s$	$0.035s$	$0.082s$

The simulative results are shown in Fig. 6. Compared with the traditional FP-RNN model, we can see that the residual errors $||x(t) + \mathcal{H}^{-1}(t)\mathcal{P}(t)||_2$ synthesized by the perturbed PT-VP-RNN (20) are much smaller than that of FP-RNN. Specifically, when using PT-VP-RNN with linear, power-sigmoid, sinh, and tunable activation functions, the maximum residual errors are 176.9, 1.528, 0.697, and 1.951. In addition, from Fig. 6, we can also see that the residual errors synthesized by the FP-RNN fluctuate at the end. Contrastively, the residual errors synthesized by the PT-VP-RNN always converge to zero finally, which verifies the robustness and superiority of the proposed PT-VP-RNN when solving time-varying QP problem in strong perturbed situations.

It is worth pointing out that the robustness of the PT-VP-RNN can be further improved by increasing γ, which can be seen from Table I.

B. Time-Varying Toeplitz Matrix With Different Dimensions

Compared with the traditional serial-processing algorithms, recurrent neural network provides a powerful alternative to solve time-varying problems in real-time due to the distribution processing and parallel character, for example, large-scale computation situation. Therefore, analysis of different dimensions n of time-varying coefficients $\mathcal{W}(t)$ and $\mathcal{G}(t)$ is necessary.

Suppose that time-varying Toeplitz matrices (diagonal-constant matrices) $\mathcal{W}(t)$ and $\mathcal{G}(t)$ are obtained from the QP problem. The specific matrices $\mathcal{W}_{\mathcal{T}}(t)$ and $\mathcal{G}(t)$ are

$$\begin{bmatrix} \mathcal{W}_1(t) & \mathcal{W}_2(t) & \mathcal{W}_3(t) & \cdots & \mathcal{W}_n(t) \\ \mathcal{W}_2(t) & \mathcal{W}_1(t) & \mathcal{W}_2(t) & \cdots & \mathcal{W}_{n-1}(t) \\ \mathcal{W}_3(t) & \mathcal{W}_2(t) & \mathcal{W}_1(t) & \cdots & \mathcal{W}_{n-2}(t) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{W}_n(t) & \mathcal{W}_{n-1}(t) & \mathcal{W}_{n-2}(t) & \cdots & \mathcal{W}_1(t) \end{bmatrix} \tag{43}$$

where $\mathcal{W}_{\mathcal{T}}(t) = [\mathcal{W}_1(t), \mathcal{W}_2(t), \mathcal{W}_3(t), \ldots, \mathcal{W}_n(t)]^{\mathrm{T}} \in \mathbb{R}^{n\times 1}$ denotes the first column vector of the above matrix $\mathcal{W}(t)$ (43).

Assume that $\mathcal{W}_1(t) = \sin t + 5$ and $\mathcal{W}_\ell(t) = \cos t/(\ell - 1)$ $(\ell = 2, 3, \ldots, n)$, and the time-varying vector $\mathcal{G}(t) \in \mathbb{R}^{n\times 1}$ is

$$\left[\sin 3t \quad \sin\left(3t + \tfrac{\pi}{2}\right) \quad \sin\left(3t + \tfrac{(n-1)\pi}{2}\right)\right]^{\mathrm{T}}. \tag{44}$$

The perturbed PT-VP-RNN (20) with time-varying model-implementation errors $\Delta\mathcal{D}(t) \in \mathbb{R}^{n\times n}$ and $\Delta\mathcal{K}(t) \in \mathbb{R}^{n\times 1}$ (with $\vartheta_{\mathcal{D}} = \vartheta_{\mathcal{K}} = 0.6$) is illustrated as the following forms, i.e.,

$$\vartheta_{\mathcal{D}} \begin{bmatrix} \mathcal{D}_1(t) & \mathcal{D}_2(t) & \mathcal{D}_3(t) & \cdots & \mathcal{D}_n(t) \\ \mathcal{D}_2(t) & \mathcal{D}_1(t) & \mathcal{D}_2(t) & \cdots & \mathcal{D}_{n-1}(t) \\ \mathcal{D}_3(t) & \mathcal{D}_2(t) & \mathcal{D}_1(t) & \cdots & \mathcal{D}_{n-2}(t) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{D}_n(t) & \mathcal{D}_{n-1}(t) & \mathcal{D}_{n-2}(t) & \cdots & \mathcal{D}_1(t) \end{bmatrix} \tag{45}$$

$$\vartheta_{\mathcal{K}}\left[\cos 4t \quad \cos\left(4t - \tfrac{\pi}{2}\right) \quad \cos\left(3t - \tfrac{(n-1)\pi}{2}\right)\right]^{\mathrm{T}} \tag{46}$$

where $\mathcal{D}_{\mathcal{T}}(t) = [\mathcal{D}_1(t), \mathcal{D}_2(t), \mathcal{D}_3(t), \ldots, \mathcal{D}_n(t)]^{\mathrm{T}} \in \mathbb{R}^{n\times 1}$ denotes the first column vector of matrix $\mathcal{D}(t)$ (45). Assume that $\mathcal{D}_\ell(t) = \cos(3t + (\ell - 1)\pi/2)$ $(\ell = 2, 3, \ldots, n)$, the entry trajectories of the theoretical solution $\mathcal{W}^{-1}(t)\mathcal{G}(t)$ with dimensions $n = 5$ and $n = 15$ are shown in Fig. 7. From this figure

we can see that even if the dimension n changes from $n = 5$ to $n = 15$, the state variables generated from the perturbed PT-VP-RNN (20), starting from randomly initial states, can always converge to the theoretical solutions.

In summary, all the above two simulation results verify the effectiveness and robustness of the proposed PT-VP-RNN (20) with different activation functions for solving time-varying QM (39) and QP (40) problems under perturbations. In addition, compared with the traditional FP-RNN model, these examples illustrate the superiority of the proposed PT-VP-RNN for solving time-varying problems. These conclusions also hold true in different matrix dimensions situation.

VI. Applications

In this section, the proposed PT-VP-RNN is applied to the inverse-kinematics motion planning of a Kinova JACO^2 manipulator and a venture investment problem.

A. Robot Tracking Problem

The joint-angle vector of the Kinova JACO^2 manipulator (with six degrees-of-freedom) is $\theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]^{\mathrm{T}} \in \mathbb{R}^6$. In order to illustrate the strong robustness performance of the proposed PT-VP-RNN (20), the digital computation errors and the external perturbations are taken into consideration.

The forward kinematic equation is $r = f(\theta)$ [18], where $\theta(t) \in \mathbb{R}^6$ denotes the joint-angular vector, and $r(t) \in \mathbb{R}^3$ denotes the end-effector path. Due to the redundancy and nonlinearity of a robot manipulator, it is difficult to straightly obtain its inverse kinematic solution $\theta(t)$ through known $r(t)$ [45]–[47]. A general approach is to resolve it at the velocity level as

$$\mathcal{J}(\theta(t))\dot{\theta}(t) = \dot{r}(t) \tag{47}$$

where $\dot{\theta}(t)$ denotes the joint-angular velocity; $\dot{r}(t)$ denotes the end-effector velocity; $\mathcal{J}(\theta) = \partial f(\theta)/\partial\theta \in \mathbb{R}^{3\times 6}$ denotes the Jacobian matrix.

In order to obtain the solution to (47), a QP-based feed-back control and repetitive motion (FCRM) scheme is written as

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}||\dot{\theta}(t) + \mathcal{C}(t)||_2^2 \\ &\text{subject to} \quad \mathcal{J}(\theta(t))\dot{\theta}(t) = \dot{r}(t) + \mathcal{U}(r(t) - f(\theta)) \end{aligned} \tag{48}$$

where $\mathcal{C}(t) = \kappa(\theta(t) - \theta(0))$ with $\kappa > 0$ being the magnitude of the response to the joint drift $\theta(t) - \theta(0)$; $\mathcal{U}(t) \in \mathbb{R}^{3\times 3}$ denotes the feedback-controlled matrix; and $||\cdot||_2$ denotes the Euclidean norm of a vector.

For comparison and illustration, the motion planning of the six degrees-of-freedom manipulator can be achieved by solving the above QP problem (48) via the proposed perturbed PT-VP-RNN and the traditional FP-RNN. Comparisons of tracking trajectories and final results synthesized by PT-VP-RNN and FP-RNN when Kinova JACO^2 manipulator tracking a "butterfly" path are shown in Fig. 8. As can be seen from Fig. 8(a), when using FP-RNN, although the end-effector task is completed, the tracking trajectories cannot coincide with the expected path. By contrast, Fig. 8(b) shows the same tracking task when using the proposed PT-VP-RNN (20). Evidently, the task is finished very well, and the tracking trajectories well-match the expected path.

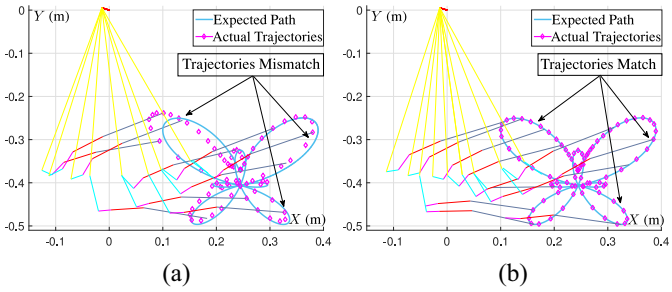


Fig. 8. Comparisons of tracking trajectories synthesized by perturbed FP-RNN and PT-VP-RNN when a Kinova JACO^2 manipulator tracks a butterfly path. (a) Tracking trajectories with the perturbed FP-RNN. (b) Tracking results with the perturbed FP-RNN.

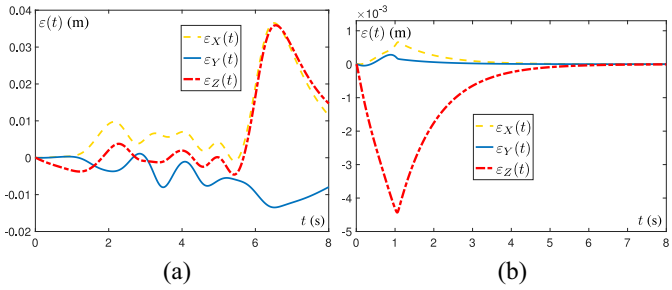


Fig. 9. Comparisons of position errors when the Kinova JACO^2 manipulator tracks a butterfly path synthesized by PT-VP-RNN and FP-RNN via FCRM scheme (48) (with linear activation function and $\gamma = 100$). (a) Position errors with the perturbed FP-RNN. (b) Position errors with the perturbed PT-VP-RNN.

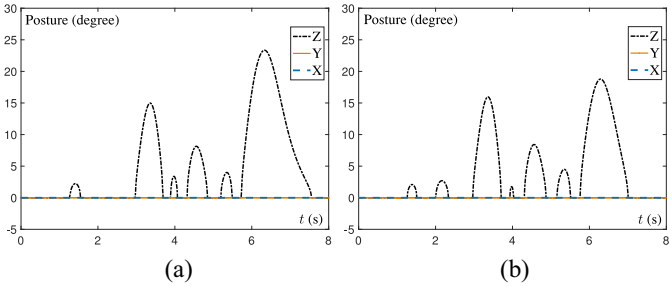


Fig. 10. Orientation behaviors of the Kinova JACO^2 robot manipulator when tracking a butterfly shape trajectory synthesized by FP-RNN and PT-VP-RNN.

The tracking performance with the two neural networks can be further verified by position errors of the end-effector shown in Fig. 9. From the position error curves, we can see quite evidently that the errors synthesized by PT-VP-RNN can converge to zero as time t passes by, but the position errors synthesized by FP-RNN are diverging. Furthermore, the corresponding orientation behaves are shown in Fig. 10. The simulative results prove the advantage of the proposed PT-VP-RNN when solving robot motion planning problems.

In summary, this application to inverse-kinematics control of the six degrees-of-freedom Kinova JACO^2 robot manipulator demonstrates the effectiveness and high accuracy of the proposed PT-VP-RNN (14) on solving time-varying QP

TABLE II
AVERAGE RETURN r_i OF SECURITIES

Serial number i	Rate-of-return r_i
$i = 1$	$r_1 = 0.100162 + 0.01\sin(t)$
$i = 2$	$r_2 = 0.164244 + 0.01\sin(2t)$
$i = 3$	$r_3 = 0.182082 + 0.01\sin(3t)$

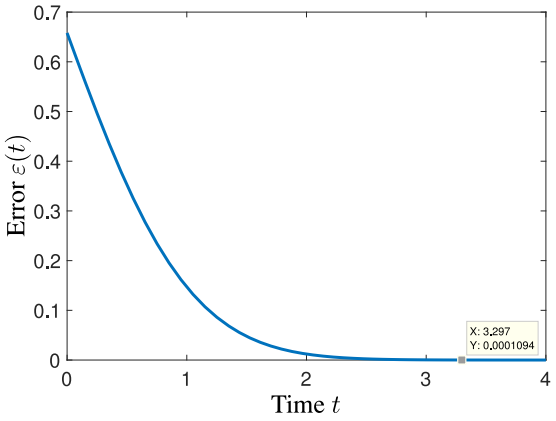


Fig. 11. Error of average expected rate-of-return of investment securities synthesized by the PT-VP-RNN method.

problems and availability of handling robot motion planning problems.

B. Venture Investment Problem

According to Markowits' mean-variance portfolio model [48], we have

$$\begin{aligned} &\text{minimize} \quad \sigma^2(t) = x^{\mathrm{T}}(t)\Sigma(t)x(t) \\ &\text{subject to} \quad d^{\mathrm{T}}(t)x(t) = \nu(t) \end{aligned} \tag{49}$$

where variance $\sigma^2(t) = \sum_i^n \sum_j^n x_i(t)\sigma_{i,j}(t)x_j(t) = x^{\mathrm{T}}(t)\Sigma(t)x(t)$ with $\Sigma(t)$ denoting the covariance matrix, and it represents the investment risk; $x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ denotes the coefficient vector of investment proportion; and x_i $(i = 1, 2, \ldots, n)$ is the proportion of each individual security in the total investment (i.e., the portfolio); $\sum_{i=1}^n x_i(t) = 1$ $(n \geq 2)$ denotes the normalized initial investment; $d(t) = (d_1(t), d_2(t), \ldots, d_n)^{\mathrm{T}}$ denotes the vector of the desired rate-of-return; and $\nu(t)$ denotes the profitability measure index of securities.

Considering that we would like to invest three securities. The average return $r_i(i = 1, 2, 3)$ of each security is fluctuating within a small certain range (i.e., the time-varying sine functions), and the data are shown in Table II. The risk related covariance matrix $\Sigma(t)$ can be calculated as

$$\begin{aligned} \Sigma(t) &= \mathrm{cov}\big(r_i(t), r_i^{\mathrm{T}}(t)\big) \\ &= \begin{bmatrix} 0.100162 & 0.045864 & 0.005712 \\ 0.045864 & 0.210773 & 0.028283 \\ 0.005712 & 0.028283 & 0.066884 \end{bmatrix} \end{aligned} \tag{50}$$

and in order to simplify the operation, the small fluctuation is omitted.

The simulations are conducted in MATLAB R2017b, on a MacBook Pro (15-inch, 2017) with Intel Core i7 CPU at 2.8 GHz, 2133 MHz LPDDR3 and 16 GB of RAM. Simply considering that the average expected return ν is $0.15 + 0.01\ln(1 + t)$, which is raising in a slow trend with time-varying logarithmic function. The error condition of the simulative result is shown in Fig. 11. As we can see from this figure, the error converges to zero (error accuracy assumption is set as 0.0001) in the range of $t \in [0, 3.3]$, and the optimal venture portfolio of the three funds can be seen from Table III (the sampling time is chosen as 0.4125 s). For example, if we want to make a decision of investment to the three securities at time instant $t = 2.4750$ s, the proportions of the money invested are 20.77%, 7.00%, and 72.23%, respectively.

TABLE III
OPTIMAL VENTURE PORTFOLIO OF INVESTMENT SECURITIES

Serial number i	x_1	x_2	x_3
$t = 0.0000s$	0.3333	0.3333	0.3333
$t = 0.4125s$	0.3214	0.3063	0.3723
$t = 0.8250s$	0.2700	0.2104	0.5196
$t = 1.2375s$	0.2482	0.1248	0.6270
$t = 1.6500s$	0.2373	0.0732	0.6895
$t = 2.0625s$	0.2188	0.0551	0.7261
$t = 2.4750s$	0.2077	0.0700	0.7223
$t = 2.8875s$	0.1973	0.1037	0.6990
$t = 3.3000s$	0.1849	0.1088	0.7063

In summary, both the robot tracking example and the venture investment example verify the effectiveness, accuracy, and widespread availability of the proposed PT-VP-RNN method.

VII. CONCLUSION

In this paper, a PT-VP-RNN is proposed for solving time-varying QM and QP problems. With the Lagrange and neural dynamic design method, the PT-VP-RNN is derived in detail. Theoretical analysis shows that the residual errors of PT-VP-RNN can converge to zero in the case of various interferences. Two numerical computational examples and comparisons with the traditional FP-RNN prove that the state solutions to the time-varying QM and QP problems generated by PT-VP-RNN can converge to the theoretical solution efficiently and accurately. All the numerical experiments with the large error conditions, different design parameters, and different dimensions of time-varying coefficients verify the effectiveness, accuracy and robustness of the proposed PT-VP-RNN when solving time-varying QM and QP problems. Finally, a perturbed redundant robot tracking example and a venture investment example further demonstrate the high-efficiency, accuracy, and practicability of the proposed PT-VP-RNN. Our future research is to explore a discrete-time varying-parameter neural model, and to study its practical applications.

REFERENCES

[1] Y. Zhang, B. Mu, and H. Zheng, "Link between and comparison and combination of Zhang neural network and quasi-Newton BFGS method for time-varying quadratic minimization," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 490–503, Apr. 2013.

[2] A. Sideris and J. E. Bobrow, "An efficient sequential linear quadratic algorithm for solving nonlinear optimal control problems," *IEEE Trans. Autom. Control*, vol. 50, no. 12, pp. 2043–2047, Dec. 2005.

[3] M. Gachpazan, "Solving of time varying quadratic optimal control problems by using Bézier control points," *Comput. Appl. Math.*, vol. 30, no. 2, pp. 367–379, 2011.

[4] 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," *Nonlin. Dyn.*, vol. 85, no. 1, pp. 245–261, 2016.

[5] M. Fu, Z.-Q. Luo, and Y. Ye, "Approximation algorithms for quadratic programming," *J. Comb. Optim.*, vol. 2, no. 1, pp. 29–50, 1998.

[6] G. F. Reid and L. Hasdorff, "Economic dispatch using quadratic programming," *IEEE Trans. Power App. Syst.*, vol. PAS-92, no. 6, pp. 2015–2023, Nov. 1973.

[7] S. Huang, Q. Wu, S. S. Oren, R. Li, and Z. Liu, "Distribution locational marginal pricing through quadratic programming for congestion management in distribution networks," *IEEE Trans. Power Syst.*, vol. 30, no. 4, pp. 2170–2178, Jul. 2015.

[8] J. Malick, "The spherical constraint in Boolean quadratic programs," *J. Glob. Optim.*, vol. 39, no. 4, pp. 609–622, 2007.

[9] N. Dyn and W. E. Ferguson, "The numerical solution of equality-constrained quadratic programming problems," *Math. Comput.*, vol. 41, no. 163, pp. 165–170, 1983.

[10] J. J. Moré and G. Toraldo, "Algorithms for bound constrained quadratic programming problems," *Numerische Mathematik*, vol. 55, no. 4, pp. 377–400, 1989.

[11] Z. Zhang, L. Kong, and Y. Niu, "A time-varying-constrained motion generation scheme for humanoid robot arms," in *Proc. 15th Int. Symp. Neural Netw.*, 2018, pp. 757–767.

[12] 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.

[13] 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.

[14] Z. Zhang, S. Li, and X. Zhang, "Simulink comparison of varying-parameter convergent-differential neural-network and gradient neural network for solving online linear time-varying equations," in *Proc. Intell. Control Autom.*, 2016, pp. 887–894.

[15] 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.

[16] W. Meng, Q. Yang, J. Sarangapani, and Y. Sun, "Distributed control of nonlinear multiagent systems with asymptotic consensus," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 5, pp. 749–757, May 2017.

[17] S. Hosseini, A. Chapman, and M. Mesbahi, "Online distributed convex optimization on dynamic networks," *IEEE Trans. Autom. Control*, vol. 61, no. 11, pp. 3545–3550, Nov. 2014.

[18] 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.

[19] 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.

[20] J. Feng, S. Qin, F. Shi, and X. Zhao, "A recurrent neural network with finite-time convergence for convex quadratic bilevel programming problems," in *Neural Computing and Applications*. London, U.K.: Springer, 2017, pp. 1–10.

[21] S. Qin, X. Le, and J. Wang, "A neuraldynamic optimization approach to bilevel quadratic programming," in *Advances in Neural Networks*, vol. 9377. Cham, Switzerland: Springer, 2015, pp. 418–425, doi: 10.1007/978-3-319-25393-0_46.

[22] Y.-J. Liu, S. Lu, and S. Tong, "Neural network controller design for an uncertain robot with time-varying output constraint," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 8, pp. 2060–2068, Aug. 2017.

[23] 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.

[24] 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.

[25] 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.

[26] L. Xiao and Y. Zhang, "Solving time-varying inverse kinematics problem of wheeled mobile manipulators using Zhang neural network with exponential convergence," *Nonlin. Dyn.*, vol. 76, no. 2, pp. 1543–1559, 2014.

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

[28] Y. Xia, "A new neural network for solving linear and quadratic programming problems," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1544–1548, Nov. 1996.

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

[30] 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.

[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. Adv. Intell. Comput. Theories Appl.*, 2008, pp. 68–75.

[32] Y. Zhang, K. Chen, and H.-Z. Tan, "Performance analysis of gradient neural network exploited for online time-varying matrix inversion," *IEEE Trans. Autom. Control*, vol. 54, no. 8, pp. 1940–1945, Aug. 2009.

[33] Y. Zhang, W. Ma, and B. Cai, "From Zhang neural network to Newton iteration for matrix inversion," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 7, pp. 1405–1415, Jul. 2009.

[34] 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.

[35] Z. Li *et al.*, "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.

[36] S. Li, S. Chen, and B. Liu, "Accelerating a recurrent neural network to finite-time convergence for solving time-varying Sylvester equation by using a sign-bi-power activation function," *Neural Process. Lett.*, vol. 37, no. 2, pp. 189–205, 2013.

[37] L. Xiao, "A finite-time recurrent neural network for solving online time-varying Sylvester matrix equation based on a new evolution formula," *Nonlin. Dyn.*, vol. 90, no. 3, pp. 1581–1591, 2017.

[38] S. Gu and R. Cui, "An efficient algorithm for the subset sum problem based on finite-time convergent recurrent neural network," *Neurocomputing*, vol. 149, pp. 13–21, Feb. 2015.

[39] Q. Liu and J. Wang, "Finite-time convergent recurrent neural network with a hard-limiting activation function for constrained optimization with piecewise-linear objective functions," *IEEE Trans. Neural Netw.*, vol. 22, no. 4, pp. 601–613, Apr. 2011.

[40] Y. Shen, P. Miao, Y. Huang, and Y. Shen, "Finite-time stability and its application for solving time-varying Sylvester equation by recurrent neural network," *Neural Process. Lett.*, vol. 42, no. 3, pp. 763–784, 2015.

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

[42] Z. Zhang, S. Chen, L. Zheng, and J. Zhang, "MATLAB simulink of varying-parameter convergent-differential neural-network for solving online time-varying matrix inverse," in *Proc. 9th Int. Symp. Comput. Intell. Design*, vol. 1, 2016, pp. 320–325.

[43] C. M. Kellett, "Classical converse theorems in Lyapunov's second method," *Discr. Continuous Dyn. Syst. B*, vol. 20, no. 8, pp. 2333–2360, 2015.

[44] M. Hazewinkel, *Advanced Multivariate Statistics With Matrices*. Dordrecht, The Netherlands: Springer, 2005.

[45] J. J. Craig, "Introduction to robotics—Mechanics and control," *Automatica*, vol. 23, no. 2, pp. 263–264, 2005.

[46] J. Baillieul, "Introduction to ROBOTICS mechanics and control," *IEEE Trans. Autom. Control*, vol. 32, no. 5, pp. 463–464, May 2003, doi: 10.1109/TAC.1987.1104613.

[47] F. Merat, "Introduction to robotics: Mechanics and control," *IEEE J. Robot. Autom.*, vol. 3, no. 2, p. 166, Sep. 2003, doi: 10.1109/ISSM.1995.524390.

[48] L. Chincarini and D. Kim, *Quantitative Equity Portfolio Management*. Boca Raton, FL, USA: Chapman and Hall, 2011.

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

He was a Post-Doctoral Research Fellow with the Institute for Media Innovation, Nanyang Technological University, Singapore, from 2013 to 2015. He is currently 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 Laboratory, Center for Brain Computer Interfaces and Brain Information Processing. His current research interests include neural network, robotics, and human–robot interaction.

Lingdong 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 Laboratory, Center for Brain Computer Interfaces and Brain Information Processing, SCUT. His current research interests include automatic control, neural network, and robotics.

Lunan Zheng 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.

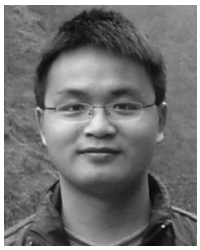
He is currently with the Human–Robot Intelligence Laboratory, Center for Brain Computer Interfaces and Brain Information Processing, SCUT. His current research interests include neural network, machine learning, and robotics.

Pengchao Zhang received the B.Eng. degree in automation from the Shaanxi University of Technology (SNUT), Hanzhong, China, and the M.Eng. degree in traffic control engineering from Northwestern Polytechnical University (NPU), Xi'an, China, where he is currently pursuing the Doctoral degree.

He is currently an Associate Professor with SNUT. His current research interests include industrial robot and mobile robotics.

Xilong Qu received the Ph.D. degree in traffic information engineering and control from Southwest Jiaotong University, Chengdu, China, in 2006.

He is currently the Dean of the School of Information Technology and Management, Hunan University of Finance and Economics, Changsha, China. His current research interests include networked manufacture, agile supply chain, and integration of distributed system.



Bolin Liao received the B.Eng. degree in electronic information engineering from Jishou University (JSU), Jishou, China, in 2003, the M.Eng. degree in control theory and control engineering from the Anshan University of Science and Technology, Anshan, China, in 2006, and the Ph.D. degree in communication and information systems from the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China, in 2015.

He is currently an Associate Professor with the School of Information Science and Engineering, JSU. His current research interests include neural network, robotics, and process control.

Zhuliang Yu received the B.Eng. degree in electronic engineering and M.Eng. degree in signal processing from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1995 and 1998, respectively, and the Ph.D. degree in electronic engineering from Nanyang Technological University (NTU), Singapore, in 2006.

He joined the Centre of Signal Processing (CSP), NTU, as a Research Engineer in 2000, and then as a Group Leader in 2001. In 2008, he joined the School of Automation Science and Engineering, South China University of Technology, Guangzhou, China, where he became a Full Professor in 2010. His current research interests include signal processing, pattern recognition, and intelligent system techniques.

Dr. Yu was a recipient of the Best Researcher Award of CSP, and the Silver Award of Tan KahKee Young Inventor Award, Singapore, as well as the Meritorious in International Mathematical Contest in Modeling administered by the Consortium for Mathematics and Its Applications of the USA. He has been served as a reviewer for several prestigious international journals, such as the IEEE TRANSACTIONS ON SIGNAL PROCESSING, IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I: REGULAR PAPERS, and *IEEE Signal Processing Letters*.