Soft Computing (2019) 23:7769–7803
https://doi.org/10.1007/s00500-018-3405-5

METHODOLOGIES AND APPLICATION

A Runge–Kutta neural network-based control method for nonlinear MIMO systems

Kemal Uçak[1]

Published online: 24 July 2018

Abstract
In this paper, a novel Runge–Kutta neural network (RK-NN)-based control mechanism is introduced for multi-input multi-output (MIMO) nonlinear systems. The overall architecture embodies an online Runge–Kutta model which computes a forward model of the system, an adaptive controller with tunable parameters and an adjustment mechanism realized by separate online Runge–Kutta neural networks to identify the dynamics of each tunable controller parameter. Runge–Kutta identification block has the competency to approximate the time-varying parameters of the model and unmeasurable states of the controlled system. Thus, the strengths of radial basis function (RBF) neural network structure and Runge–Kutta integration method are combined in this structure. Adaptive MIMO proportional–integral–derivative (PID) controller is deployed in the controller block. The control performance of the proposed adaptive control method has been evaluated via simulations performed on a nonlinear three-tank system and Van de Vusse benchmark system for different cases, and the obtained results reveal that the RK-NN-based control mechanism and Runge–Kutta model attain good control and modelling performances.

Keywords Adaptive controller · MIMO PID-type RK-NN controller · Runge–Kutta EKF · Runge–Kutta identification · Runge–Kutta neural network · Runge–Kutta parameter estimator

1 Introduction

The vital physical or behavioural characteristic which provides living organisms to be prosperous in a particular circumstance is called as adaptation. Adaptation skill is one of the most crucial milestones (keystone) of the evolutionary process for living organisms. Without the adaptation skill, the vitality could have come to an end. Hence, although most of the species have come from similar genetic origins, they have acquired various abilities and exhibit various differences with respect to the ecological system they are in, due to their adaptation skills. When examined psychologically, individuals with adaptation skills are more successful in social life than strict thoughters. Therefore, in a very broad sense, adaptation can be considered as the most basic inevitable element of success in life.

Similarly, when examined in terms of control system theory, adaptive control systems generally exhibit better performance with respect to their fixed-parameter counterparts. The control of nonlinear systems, especially multi-input multi-output (MIMO) nonlinear systems, is a challenging task since the controlled dynamics of the system interact, and fixed controller structures cannot follow and approximate alternations on dynamics behaviour of system, which ensnarl to force the system dynamics to desired reference signals. The complexity of nonlinear systems necessitates the utilization of flexible controller structures. Therefore, considering the impact of the adaptation in success, it is required to interfuse adaptation ability to the controller structures particularly designed for nonlinear systems, which enhance the control performance and dynamics of the controller parameters in response to unpredictable changes in system dynamics. By introducing adaptation to a conventional controller, it is possible to deploy it to cope with strong nonlinearities, time delays and time-varying dynamics of systems (Uçak and Günel 2017).

The parameter adaptive control can be roughly examined under three main headings in a common framework according to Aström (1983): gain scheduling, model reference adaptive control and self-adaptive controllers.

Communicated by V. Loia.

✉ Kemal Uçak
ucak@mu.edu.tr

[1] Department of Electrical and Electronics Engineering, Faculty of Engineering, Muğla Sıtkı Koçman University Kötekli, 48000 Muğla, Turkey

Gain scheduling can be thought of as a mapping between the current state of the system and the appropriate controller parameters for this state (Uçak and Günel 2017; Aström and Wittenmark 2008). In gain scheduling, firstly, the wide operating range in which the system is controlled is partitioned into small uncertain subregions via a priori information, then robust and optimal controllers are designed for each small range (Uçak and Günel 2017). Decision trees or lookup tables are deployed in order to constitute a model of the relationship between predefined system operating conditions and designed controller parameters. Thus, the appropriate controller parameters to the current situation of the controlled system can be deployed when system is running. Since gain scheduling is designed by taking into account the previously defined scenarios, the control performance deteriorates and even the controllability of the system is rarified when the system exhibits unpredictable behaviour or encounters an unpredictable situation. Another drawback is that it is an open-loop compensation since there is no feedback structure which compensates for an incorrect schedule (Aström 1983). Therefore, gain scheduling can be considered as a feedback control method where controller parameters are adjusted by feedforward compensation. The other trouble of gain scheduling is the time-consuming computations carried out to determine the appropriate controller parameters for many operating conditions and extensive simulations utilized to check the control performance (Aström 1983; Aström and Wittenmark 2008). In spite of the mentioned drawbacks, the controller parameters can be changed very quickly in response to the alternations on system behaviour since most of the chores in controller design steps are completed before control process.

Model reference adaptive control (MRAC) structures consist of two loops. The inner loop includes a controller with adjustable parameters and the system to be controlled. The outer loop embodies the update rules for the controller, reference model to be followed and system model to approximate future behaviours of system dynamics. In MRAC, the aim is that the closed-loop system exhibits the same behaviour as a reference model. Therefore, the transient and steady-state specifications of the closed-loop system are defined on a reference model in which closed-loop system is compelled to track. The adjustment rules for the control algorithm are derived, in such a way that the error between reference model and closed-loop system output is minimized.

Self-adaptive controllers (SAC) are one of the most effective adaptive control structures for nonlinear systems (Uçak and Günel 2017). Notwithstanding differences in their origin, MRAC and SAC have similar properties with regard to the number of the feedback loops in adjustment mechanism (Aström 1983). Both adaptive control methods consist of two feedback loops: inner and outer feedback loops. The inner loop consists of the system to be controlled and a controller with adjustable parameters, and the controller parameters are adjusted via outer loop. Nevertheless, the methods to design the inner loop and the techniques used to adjust the parameters in the outer loop may be different (Aström 1983). For SAC, controller design alternatives can be enriched since a variety of controllers and parameter estimators can be deployed in controller and estimator blocks, by combining the powerful features of these components. For instance, by combining the nonlinear function approximation ability of artificial neural networks (ANN) and robustness of PID controllers, PID-type ANN controllers can be designed to effectively control nonlinear systems (Akhyar and Omatu 1993; Wang et al. 2001).

In technical literature, various effective adaptive control structures based on soft computing methods have been proposed for nonlinear systems (Uçak and Günel 2017; Akhyar and Omatu 1993; Wang et al. 2001; Flynn et al. 1997; Pham and Karaboga 1999; Sharkawy 2010; Bouallégue et al. 2012; Bishr et al. 2000; Zhao et al. 2016a, b). However, the main drawback is the computational load of the system identification procedure. In model-based control structures, the accuracy of the system model and computational load of system identification are significant issues in the implementation of the control algorithm. Whereas the accuracy of the controller parameters is directly affected by system model, computational load of the identification step restricts the implementation of the algorithm for various kinds of systems. In order to overcome these drawbacks, a novel system identification technique based on Runge–Kutta (RK) model has been proposed by Iplikci (2013) for nonlinear MIMO systems to be deployed in a nonlinear model predictive control (NMPC) structure. The method requires the differential equations of the system to be derivable. Since this is possible for many kinds of dynamical systems, the controller structures based on RK model can be successfully deployed for wide ranges of nonlinear systems.

There exist various controller structures based on RK-system identification technique, in technical literature. The precessor form of the RK-based identification technique has been proposed by Iplikci (2013) to be deployed in the NMPC framework. NMPC structures require the future behaviour of the controlled system in response to the candidate control signals to optimize a finite-horizon open-loop optimal control problem during each sampling period. Using the Taylor expansion of the objective function, the adjustment rules for control signal vector can be derived. In order to approximate the sensitivity of the controlled system outputs with respect to control signals (system Jacobian), the dynamics of the system is identified via RK model. The Runge–Kutta (RK) identification block comprises raw RK system model, RK-based model parameter estimator block and RK-based EKF block. RK model of the system is utilized for control, state estimation and model parameter adjustment (Iplikci 2013; Beyhan 2013). In order to approximate future behaviours of the states, the current states of the system estimated via RK-based EKF are required. The identification block subsumes RK model parameter estimator block to estimate the model parameters which cannot be determined accurately. In auto-tuning PID mechanism proposed in Cetin and Iplikci (2015), the adjustment mechanism based on support vector regression (SVR) for SISO nonlinear systems proposed in Iplikci (2010a) has been expanded and adapted for nonlinear MIMO systems by using Runge–Kutta (RK) model in place of SVR identification technique. The proposed auto-tuning PID mechanism incorporates the robustness of PID structure, fast convergence from the MPC framework and gradient-based adaptation ability (Cetin and Iplikci 2015). RK model of the system is deployed to estimate K-step ahead future system behaviour and Jacobian of the system utilized in Levenberg–Marquardt adjustment algorithm. In the nonlinear observer introduced in Beyhan (2013), the states are adjusted using Levenberg–Marquardt algorithm where the proposed RK-based identification method in Iplikci (2013) is deployed to approximate the sensitivity of the system outputs with respect to system states.

In this paper, a novel Runge–Kutta neural network-based control mechanism has been proposed for multi-input multi-output (MIMO) nonlinear systems. The adjustment mechanism is composed of Runge–Kutta neural network to approximate the optimal parameter values of an adaptive controller and Runge–Kutta model to acquire the system Jacobian information. Neural networks such as multilayer perceptrons (MLP) which are constructed by considering input–output system states cannot catch the long-term behaviour of the identified systems well, and long-term prediction accuracy is usually not good since the network learns the system states, instead of the changing rates of system states (Wang and Lin 1998), which motivates us to deploy Runge–Kutta neural network to approximate the changing rates of the controller parameters. Therefore, Runge–Kutta neural network, which subsumes the strong sides of the Runge–Kutta integration method and artificial neural networks, is preferred in the controller parameter estimator block. In order to identify the dynamics of the controlled nonlinear system, RK-based identification method proposed by Iplikci (2013) is deployed owing to its low computational load and high identification accuracy. The adjustment mechanism can be deployed to any controller with adjustable parameters.

The main contribution of this paper is to propose a strong nonlinear adaptive controller adjustment mechanism which embodies the strong sides of the Runge–Kutta integration method and artificial neural networks to approximate the optimal parameter values of any controller with adjustable parameters. The proposed mechanism is utilized to optimize the parameters of a MIMO PID controller. In existing literature, MIMO PID controller parameters are obtained incrementally and therefore they can not be attained mathematically. This study differentiates from the studies in the literature in terms of mathematically and physically obtaining nonlinear MIMO controller parameters. Thus, the main novelty of this paper is that the parameters of the nonlinear MIMO controller can be identified as mathematical expressions for nonlinear MIMO systems. The performance of the proposed control method has been assessed on nonlinear three-tank system and Van de Vusse benchmark system. The obtained results indicate that the proposed Runge–Kutta neural network-based control method and Runge–Kutta model achieve good identification and closed-loop control performances.

The rest of the paper is organized as follows: Sect. 2 overviews the proposed Runge–Kutta neural network-based adaptive controller. The basic principles of Runge–Kutta model utilized in system identification block proposed by Iplikci (2013) is described in Sect. 3. Construction of the optimization problem and adjustment rules to utilize Runge–Kutta neural network directly as an adaptive controller parameter estimator and the proposed adjustment mechanism are explained in detail in Sect. 4. In Sect. 5, the control performance evaluation of the proposed method on a nonlinear three-tank system and Van de Vusse benchmark system is presented. Also, a comparison with Runge–Kutta model-based PID is provided. The paper is concluded with a brief conclusion part in Sect. 6.

2 The proposed Runge–Kutta neural network-based adaptive control structure

In adaptive control, it is aimed to interfuse flexibility to the controller parameters to attune to the alterations occurring in system dynamics. Therefore, it is required to adjust the controller parameters in accordance with the change in system dynamics. This concord depends on accurate approximation of the system dynamics and adjustment mechanism deployed to obtain controller parameters. Adaptive control structures with controller parameter estimator are frequently deployed as an adaptive control method since they intend to fit a nonlinear function to the controller parameters. Therefore, in this section, firstly, the mechanism and requirements of SACs are overviewed in Sect. 2.1. Then, the proposed nonlinear control method based on Runge–Kutta model is outlined in Sect. 2.2.

2.1 An overview of self-adaptive control

A self-adaptive controller (SAC) comprises system model, parameter estimator and controller blocks as depicted in Fig. 1 where $\boldsymbol{\beta}$ and $\boldsymbol{C}_{in}$ represent the controller parameters and input of the controller, respectively. Accurate adjustment of the controller parameters depends on the preciseness with which the future behaviour of the system dynamics can be foreseen. Therefore, system model block is crucial to esti-

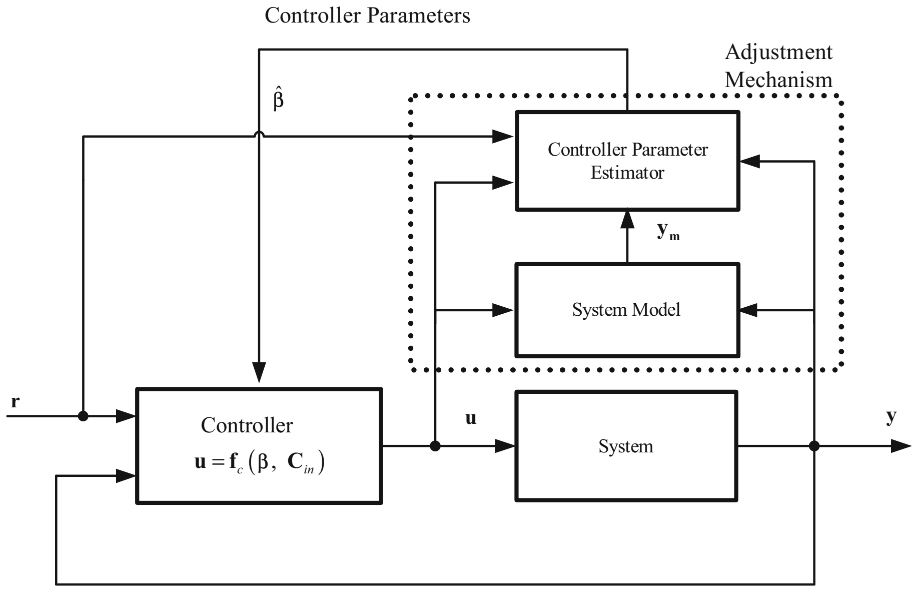


Fig. 1 Self-adaptive controller

mate the dynamics of the system. In controller parameter estimator block, the dynamics of the new controller parameters which obtrude the output of the system to the reference trajectory are identified by taking into account the history of the system dynamics and the future behaviour of the system via the obtained system model. By using the appropriate controller parameters in the controller block, the control signal which forces the system dynamics to the reference signal can be accurately attained. As can be seen from Fig. 1, different adaptive controller structures can be proposed by combining different controller, system model and parameter estimator types. Any controller with adjustable parameters can be executed in the controller block given in Fig. 1 (Uçak and Günel 2017). In this work, MIMO PID controller is implemented in the proposed control structure. Similarly, depending on the controlled systems and design techniques, numerous adaptive architectures are possible in the parameter estimation block (Aström et al. 1977). As for the system model part, various intelligent modelling techniques such as artificial neural networks (ANN) (Efe and Kaynak 2000; Hagan et al. 2002; Efe and Kaynak 1999; Efe 2011), adaptive neuro fuzzy inference system (ANFIS) (Denai et al. 2004; Jang 1993) and SVR (Iplikci 2010a, b, 2006) have been utilized to identify the system dynamics.

In the proposed controller structure, the dynamics of the controlled system is identified via Runge–Kutta system model. Subsequently, Runge–Kutta neural network is employed as a parameter estimator to approximate controller parameters.

2.2 Runge–Kutta neural network-based adjustment mechanism

The adjustment mechanism of the proposed control architecture based on Runge–Kutta model is illustrated in Fig. 2, where R is the dimension of the input signal and Q represents the dimension of the controlled output. There are two main structures to be carefully examined in the proposed mechanism: Runge–Kutta neural network controller parameter estimator to identify the controller parameters and Runge–Kutta system model to approximate the future behaviour of the controlled system. For simplicity, Runge–Kutta neural network controller parameter estimator is abbreviated as RK-NN$_{\text{estimator}}$ and Runge–Kutta system model is RK$_{\text{model}}$. RK-NN$_{\text{estimator}}$ and RK$_{\text{model}}$ are both utilized online to perform learning, prediction and control consecutively. In the proposed mechanism, firstly, the controller parameters ($\boldsymbol{\beta}$) are estimated using the current weights ($\boldsymbol{\Theta}^{old} = [\alpha_1^{old} \cdots \alpha_M^{old}]^T$) of the RK-NN$_{\text{estimator}}$, and then the control signal is attained as follows:

$$\boldsymbol{u}[\boldsymbol{n}] = f_c(\hat{\boldsymbol{\beta}}, \boldsymbol{C}_{in}) \tag{1}$$

where $\hat{\boldsymbol{\beta}}$ represents the approximated controller parameters and $\boldsymbol{C}_{in}$ is the input signal of the control law. The obtained control signal ($\boldsymbol{u}[\boldsymbol{n}]$) is repeatedly applied to the RK$_{\text{model}}$ K-times in order to approximate K-step ahead future behaviour of the controlled system. For this purpose, in order to attain K-step ahead future behaviour, firstly, it is required to obtain the current states of the controlled system. Therefore, in Runge–Kutta model-based EKF block, using the previous control inputs and system outputs, the current states of the system ($[\tilde{x}_1[n] \cdots \tilde{x}_N[n]]$) are attained. Then, by taking into consideration the possibility that the system parameters may change, using the obtained current states of the system ($[\tilde{x}_1[n] \cdots \tilde{x}_N[n]]$) and control signals, optimal model parameters ($\boldsymbol{\theta}$) for Runge–Kutta model can be approximated via Runge–Kutta-based model parameter estimation block. Consequently, using the current values of

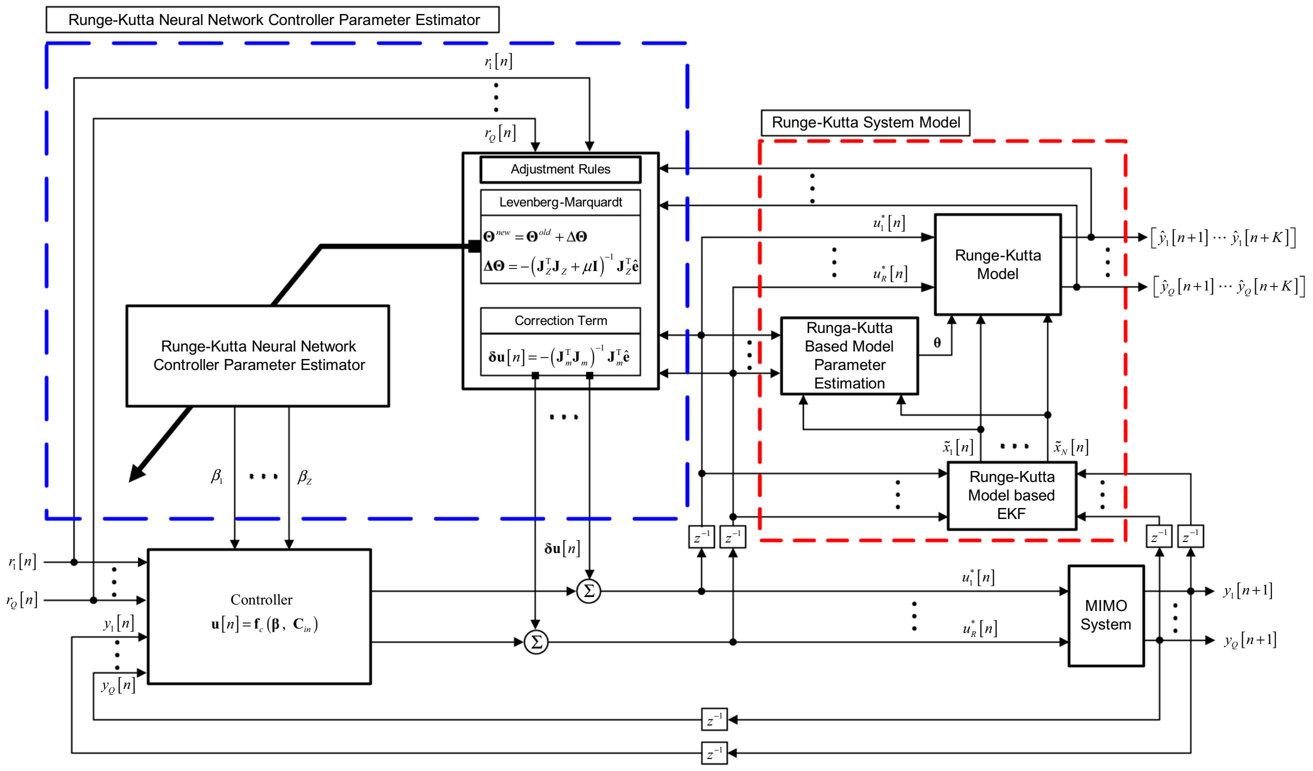

Runge-Kutta Neural Network Controller Parameter Estimator
Adjustment Rules
Levenberg-Marquardt
$\Theta^{new} = \Theta^{old} + \Delta\Theta$
$\Delta\Theta = -(\mathbf{J}_Z^T\mathbf{J}_Z + \mu\mathbf{I})^{-1}\mathbf{J}_Z^T\hat{\mathbf{e}}$
Correction Term
$\delta\mathbf{u}[n] = -(\mathbf{J}_m^T\mathbf{J}_m)^{-1}\mathbf{J}_m^T\hat{\mathbf{e}}$
Runge-Kutta Neural Network Controller Parameter Estimator
Runge-Kutta System Model
Runge-Kutta Model
Runga-Kutta Based Model Parameter Estimation
Runge-Kutta Model based EKF
Controller
$\mathbf{u}[n] = \mathbf{f}_c(\boldsymbol{\beta}, \mathbf{C}_{in})$
MIMO System
$r_1[n]$
$r_Q[n]$
$y_1[n]$
$y_Q[n]$
β_1
β_Z
$\delta\mathbf{u}[n]$
$u_1^*[n]$
$u_R^*[n]$
θ
$\tilde{x}_1[n]$
$\tilde{x}_N[n]$
$[\hat{y}_1[n+1] \cdots \hat{y}_1[n+K]]$
$[\hat{y}_Q[n+1] \cdots \hat{y}_Q[n+K]]$
$y_1[n+1]$
$y_Q[n+1]$
z^{-1}

Fig. 2 Proposed Runge–Kutta neural network-based control structure

model parameters ($\boldsymbol{\theta}$), system states ($[\tilde{x}_1[n] \cdots \tilde{x}_N[n]]$) and control signals ($[u_1^\star[n] \cdots u_R^\star[n]]$) in Runge–Kutta model, K-step ahead future behaviour of the controlled system can be computed. After the future behaviour of the system dynamics is acquired via RK$_{\text{model}}$, it is required to optimize the weights of the RK-NN$_{\text{estimator}}$ so as to obtain the feasible controller parameters that force the system output to track the reference signal. For this purpose, the objective function in (2) is minimized where K is the prediction horizon, Q is number of the controlled outputs, R is the number of the control signals and λs are penalty terms utilized to restrict the deviation of the control signals.

$$\begin{aligned} F(\boldsymbol{u}[\boldsymbol{n}], \hat{\boldsymbol{e}}_q) &= \sum_{q=1}^{Q}\sum_{k=1}^{K}\Big[r_q[n+k] - \hat{y}_q[n+k]\Big]^2 \\ &\quad + \sum_{r=1}^{R}\lambda_r\Big[u_r[n] - u_r[n-1]\Big]^2 \\ &= \sum_{q=1}^{Q}\Big[\hat{e}_q[n+k]\Big]^2 \\ &\quad + \sum_{r=1}^{R}\lambda_r\Big[u_r[n] - u_r[n-1]\Big]^2 \end{aligned} \tag{2}$$

With the network parameters of RK-NN$_{\text{estimator}}$ expressed as $\boldsymbol{\Theta}^{old} = [\alpha_1^{old} \cdots \alpha_M^{old}]^T$, the weights of the RK-NN$_{\text{estimator}}$ can be optimized using Levenberg–Marquardt optimization rule as follows:

$$\begin{aligned} \boldsymbol{\Theta}^{new} &= \boldsymbol{\Theta}^{old} + \Delta\boldsymbol{\Theta} \\ \Delta\boldsymbol{\Theta} &= -\left(\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I}\right)^{-1}\boldsymbol{J}^T\hat{\boldsymbol{e}} \end{aligned} \tag{3}$$

where $\boldsymbol{J}$ is a $(QK+R)xM$ dimension Jacobian matrix given as

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial\hat{e}_1[n+1]}{\partial\alpha_1^{old}} & \cdots & \frac{\partial\hat{e}_1[n+1]}{\partial\alpha_M^{old}} \\ \vdots & \ddots & \vdots \\ \frac{\partial\hat{e}_Q[n+K]}{\partial\alpha_1^{old}} & \cdots & \frac{\partial\hat{e}_Q[n+K]}{\partial\alpha_M^{old}} \\ \sqrt{\lambda_1}\frac{\partial\Delta u_1[n]}{\partial\alpha_1^{old}} & \cdots & \sqrt{\lambda_1}\frac{\partial\Delta u_1[n]}{\partial\alpha_M^{old}} \\ \vdots & \ddots & \vdots \\ \sqrt{\lambda_R}\frac{\partial\Delta u_R[n]}{\partial\alpha_1^{old}} & \cdots & \sqrt{\lambda_R}\frac{\partial\Delta u_R[n]}{\partial\alpha_M^{old}} \end{bmatrix}$$

$$= -\begin{bmatrix} \frac{\partial \hat{y}_1[n+1]}{\partial \alpha_1^{old}} & \cdots & \frac{\partial \hat{y}_1[n+1]}{\partial \alpha_M^{old}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}_Q[n+K]}{\partial \alpha_1^{old}} & \cdots & \frac{\partial \hat{y}_Q[n+K]}{\partial \alpha_M^{old}} \\ -\sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \alpha_1^{old}} & \cdots & -\sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \alpha_M^{old}} \\ \vdots & \ddots & \vdots \\ -\sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \alpha_1^{old}} & \cdots & -\sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \alpha_M^{old}} \end{bmatrix}_{[(QK+R)xM]} \quad (4)$$

and $\hat{\boldsymbol{e}}$ is the vector of the prediction errors

$$\hat{\boldsymbol{e}} = \begin{bmatrix} \hat{e}_1[n+1] \\ \vdots \\ \hat{e}_1[n+K] \\ \hat{e}_2[n+1] \\ \vdots \\ \hat{e}_2[n+K] \\ \vdots \\ \hat{e}_Q[n+1] \\ \vdots \\ \hat{e}_Q[n+K] \\ \sqrt{\lambda_1}\Delta u_1[n] \\ \vdots \\ \sqrt{\lambda_R}\Delta u_R[n] \end{bmatrix} = \begin{bmatrix} \hat{e}[n+1] \\ \vdots \\ \hat{e}[n+K] \\ \hat{e}[n+K+1] \\ \vdots \\ \hat{e}[n+2K] \\ \vdots \\ \hat{e}[n+(Q-1)K+1] \\ \vdots \\ \hat{e}[n+QK] \\ \sqrt{\lambda_1}\Delta u_1[n] \\ \vdots \\ \sqrt{\lambda_R}\Delta u_R[n] \end{bmatrix}$$

$$= \begin{bmatrix} r_1[n+1] - \hat{y}_1[n+1] \\ \vdots \\ r_1[n+K] - \hat{y}_1[n+K] \\ r_2[n+1] - \hat{y}_2[n+1] \\ \vdots \\ r_2[n+K] - \hat{y}_2[n+K] \\ \vdots \\ r_Q[n+1] - \hat{y}_Q[n+1] \\ \vdots \\ r_Q[n+K] - \hat{y}_Q[n+K] \\ \sqrt{\lambda_1}\left[u_1[n] - u_1[n-1]\right] \\ \vdots \\ \sqrt{\lambda_R}\left[u_R[n] - u_R[n-1]\right] \end{bmatrix}_{[(QK+R)x1]} \quad (5)$$

Since the objective function in (2) is nonconvex and optimized via Levenberg–Marquardt algorithm, the weights of the network which force the output of the system to track the reference input can be obtained locally. Since obtaining global solutions requires more computational load in comparison with local counterparts, utilization of local solutions in online control is more convenient and effective than global ones. The term $\frac{\partial \hat{e}_Q[n+K]}{\partial \alpha_M^{old}}$ in Jacobian matrix (4) can be expanded via chain rule as follows:

$$\frac{\partial \hat{e}_Q[n+K]}{\partial \alpha_M^{old}} = \frac{\partial \hat{e}_Q[n+K]}{\partial y_Q[n+K]}\left[\sum_{r=1}^{R} \frac{\partial y_Q[n+K]}{\partial u_r[n]} \frac{\partial u_r[n]}{\partial \alpha_M^{old}}\right]$$
$$= -\left[\sum_{r=1}^{R} \frac{\partial y_Q[n+K]}{\partial u_r[n]} \frac{\partial u_r[n]}{\partial \alpha_M^{old}}\right] \quad (6)$$

where $\frac{\partial y_Q[n+K]}{\partial u_r[n]}$ indicates the sensitivity of the system outputs with respect to control inputs and $\frac{\partial u_r[n]}{\partial \alpha_M^{old}}$ is the sensitivity of the control signals with respect to RK-NN$_{\text{estimator}}$ parameters. The term $\frac{\partial u_r[n]}{\partial \alpha_M^{old}}$ can be easily derived using the relationship between control signals and RK-NN$_{\text{estimator}}$; however, $\frac{\partial y_Q[n+K]}{\partial u_r[n]}$ is an unknown term which is difficult to attain. Ideally, during the course of online working, it is expected that $\hat{y}_q[n+1]$, $q \in \{1, \ldots, Q\}$ converges to $y_q[n+1]$, $q \in \{1, \ldots, Q\}$ (Uçak and Günel 2017). Therefore, as can be seen from this expansion, the RK$_{\text{model}}$ can be utilized to approximate the K-step ahead future system Jacobian information (in other words sensitivity of the system outputs with respect to control signals ($\frac{\partial y_Q[n+K]}{\partial u_r[n]}$) so as to construct the Jacobian matrix for Levenberg–Marquardt algorithm. As a result of the adjustment rule based on Levenberg–Marquardt algorithm given in (3), RK-NN$_{\text{estimator}}$ parameters and also inherently the controller parameters are anticipated to iteratively converge to their optimal values in the long run (Iplikci 2010a). However, especially because of modelling inaccuracies and external disturbances, mostly in the transient state and to some extent in the steady state, the control action $\boldsymbol{u}[n]$ may not be adequate to force the system output towards the desired trajectory as a result of the non-optimal controller parameters (Iplikci 2010a). In order to overcome this situation, a correction term $\delta\boldsymbol{u}[n]$ to be added to the control action is proposed to enhance control performance (Iplikci 2010a). Thus, the deteriorations in control performance can be reduced. The suboptimal correction term $\delta\boldsymbol{u}[n]$ which aims to minimize the objective function F with respect to $\delta\boldsymbol{u}[n]$ can be derived using the second-order Taylor approximation of the objective function F as follows (Iplikci 2010a):

$$F(\boldsymbol{u}[\boldsymbol{n}] + \delta\boldsymbol{u}[\boldsymbol{n}]) \cong F(\boldsymbol{u}[\boldsymbol{n}]) + \frac{\partial F(\boldsymbol{u}[\boldsymbol{n}])}{\partial \boldsymbol{u}[\boldsymbol{n}]}\delta\boldsymbol{u}[\boldsymbol{n}]$$
$$+ \frac{1}{2}\frac{\partial^2 F(\boldsymbol{u}[\boldsymbol{n}])}{\partial^2 \boldsymbol{u}[\boldsymbol{n}]}(\delta\boldsymbol{u}[\boldsymbol{n}])^2 \quad (7)$$

Using the first-order optimality conditions, the derivative of the approximate F with respect to $\delta\boldsymbol{u}[n]$ can be acquired as

$$\frac{\partial F(\boldsymbol{u}[n]+\delta\boldsymbol{u}[n])}{\partial\delta\boldsymbol{u}[n]} \cong \frac{\partial F(\boldsymbol{u}[n])}{\partial\boldsymbol{u}[n]} + \frac{\partial^2 F(\boldsymbol{u}[n])}{\partial^2\boldsymbol{u}[n]}\delta\boldsymbol{u}[n] = 0 \tag{8}$$

Thus, $\delta\boldsymbol{u}[n]$ can be obtained as

$$\delta\boldsymbol{u}[n] = -\frac{\frac{\partial F(\boldsymbol{u}[n])}{\partial\boldsymbol{u}[n]}}{\frac{\partial^2 F(\boldsymbol{u}[n])}{\partial^2\boldsymbol{u}[n]}} \tag{9}$$

$\delta\boldsymbol{u}[n]$ depends on gradient ($\frac{\partial F(\boldsymbol{u}[n])}{\partial\boldsymbol{u}[n]}$) and Hessian ($\frac{\partial^2 F(\boldsymbol{u}[n])}{\partial^2\boldsymbol{u}[n]}$) terms. The gradient vector can be easily derived using (2) as

$$\frac{\partial F(\boldsymbol{u}[n])}{\partial\boldsymbol{u}[n]} = 2\boldsymbol{J}_m^T\hat{\boldsymbol{e}} \tag{10}$$

where

$$\boldsymbol{J}_m = \begin{bmatrix} -\frac{\partial\hat{y}_1[n+1]}{\partial u_1[n]} & \cdots & -\frac{\partial\hat{y}_1[n+1]}{\partial u_R[n]} \\ \vdots & \ddots & \vdots \\ -\frac{\partial\hat{y}_Q[n+K]}{\partial u_1[n]} & \cdots & -\frac{\partial\hat{y}_Q[n+K]}{\partial u_R[n]} \\ \sqrt{\lambda_1} & \cdots & \sqrt{\lambda_1} \\ \vdots & \ddots & \vdots \\ \sqrt{\lambda_R} & \cdots & \sqrt{\lambda_R} \end{bmatrix}_{[(QK+R)x1]} \tag{11}$$

In order to reduce the computational load and complexity of the Hessian ($\frac{\partial^2 F(\boldsymbol{u}[n])}{\partial^2\boldsymbol{u}[n]}$) term, the Hessian can be approximated as

$$\frac{\partial^2 F(\boldsymbol{u}[n])}{\partial^2\boldsymbol{u}[n]} = 2\boldsymbol{J}_m^T\boldsymbol{J}_m \tag{12}$$

Substituting (10) and (12) in (9), $\delta\boldsymbol{u}[n]$ can be computed as

$$\delta\boldsymbol{u}[n] = -(\boldsymbol{J}_m^T\boldsymbol{J}_m)^{-1}\boldsymbol{J}_m^T\hat{\boldsymbol{e}} \tag{13}$$

By substituting the adjusted network parameters ($\boldsymbol{\Theta}^{new} = [\alpha_1^{new}\cdots\alpha_M^{new}]^T$) in RK-NN$_{\text{estimator}}$, adjusted controller parameters ($\boldsymbol{\beta}^{new}$) and new control signal can be computed via (1). By adding the correction term, the optimal control signal which compels the system output to track reference signal can be attained as $\boldsymbol{u}^\star[n] = \boldsymbol{u}[n] + \delta\boldsymbol{u}[n]$ and applied to the real system. Up to this point, the outline of the adjustment mechanism is extracted. The working principle of each block in RK$_{\text{model}}$ and RK-NN$_{\text{estimator}}$ is detailed in Sects. 3 and 4, respectively. The detailed pseudocode of the proposed adaptive control architecture is presented in Sect. 4.4.

3 System identification via Runge–Kutta system model

In this section, the RK-based nonlinear system identification block proposed by Iplikci 2013 is presented. The idea behind the RK-based identification method is to discretize the continuous-time MIMO system dynamics via fourth-order Runge–Kutta integration method in order to attain an adaptive, data sampled identification technique. Therefore, firstly, Runge–Kutta discretization method is given in Sect. 3.1. Runge–Kutta discretization method approximates one-step ahead future behaviour of the system in the case that the current value of system states and system parameters utilized in state functions are available. Therefore, the current states of the system and model parameters are two significant components of the method to be determined. Since the identification method is data sampled and correct approximation of system states dramatically depends on the accuracy of the current states, RK-model-based EKF method is utilized to estimate current states of the system. Therefore, RK-model-based EKF is detailed in Sect. 3.2. Because of the modelling inaccuracies in system parameters, it is required to deploy a system parameter estimator to estimate system parameters. In order to adjust the RK-model parameter when the dynamics of the system are altered owing to internal or external factors such as uncertainty or disturbance, the Runge–Kutta model-based online model parameter estimation block is deployed as presented in Sect. 3.3. After all fundamental components of the RK-based nonlinear system identification block proposed by Iplikci 2013 are examined, in Sect. 3.4, the approximation of the future system behaviour via RK-system model is investigated.

3.1 An overview of MIMO systems and Runge–Kutta system model

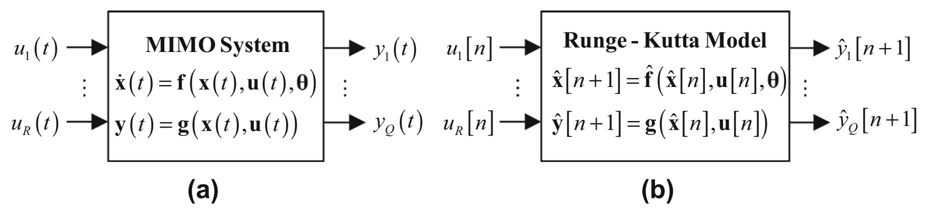
Let us consider an N-dimensional continuous-time MIMO system as depicted in Fig. 3a. The state equations of the system are expressed as

$$\begin{aligned} \dot{x}_1(t) &= f_1(x_1(t),\ldots,x_N(t),u_1(t),\ldots,u_R(t),\boldsymbol{\theta}) \\ &\vdots \\ \dot{x}_N(t) &= f_N(x_1(t),\ldots,x_N(t),u_1(t),\ldots,u_R(t),\boldsymbol{\theta}) \end{aligned} \tag{14}$$

subject to state and input constraints of the form

$$\begin{aligned} &x_1(t)\in X_1,\ldots,x_N(t)\in X_N, \forall t\geq 0 \\ &u_1(t)\in U_1,\ldots,u_R(t)\in U_R, \forall t\geq 0 \end{aligned} \tag{15}$$

where X_is and U_is are box constraints for the states and inputs as given below, respectively



MIMO System
$\dot{\mathbf{x}}(t)=\mathbf{f}(\mathbf{x}(t),\mathbf{u}(t),\boldsymbol{\theta})$
$\mathbf{y}(t)=\mathbf{g}(\mathbf{x}(t),\mathbf{u}(t))$
$u_1(t)$
$u_R(t)$
$y_1(t)$
$y_Q(t)$
Runge - Kutta Model
$\hat{\mathbf{x}}[n+1]=\hat{\mathbf{f}}(\hat{\mathbf{x}}[n],\mathbf{u}[n],\boldsymbol{\theta})$
$\hat{\mathbf{y}}[n+1]=\mathbf{g}(\hat{\mathbf{x}}[n],\mathbf{u}[n])$
$u_1[n]$
$u_R[n]$
$\hat{y}_1[n+1]$
$\hat{y}_Q[n+1]$
(a)
(b)

Fig. 3 **a** A continuous-time multi-input multi-output (MIMO) system and **b** its Runge–Kutta model

$$X_i \in \left\{ x_i \in \Re \mid x_{i_{min}} \le x_i \le x_{i_{max}} \right\},\ \ for\ i = 1, \ldots, N$$
$$U_i \in \left\{ u_i \in \Re \mid u_{i_{min}} \le u_i \le u_{i_{max}} \right\},\ \ for\ i = 1, \ldots, R \tag{16}$$

and the output equations are

$$\begin{aligned} y_1(t) &= g_1(x_1(t), \ldots, x_N(t), u_1(t), \ldots, u_R(t)) \\ &\vdots \\ y_Q(t) &= g_Q(x_1(t), \ldots, x_N(t), u_1(t), \ldots, u_R(t)) \end{aligned} \tag{17}$$

where R denotes the number of inputs, N stands for the number of states, Q is the number of outputs and $\boldsymbol{\theta}$ emblematizes the parameters of the system (Iplikci 2013).The above system equations for MIMO system can be given in a more compact form as (Iplikci 2013)

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\theta}) \\ \boldsymbol{y} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{x} &\in \boldsymbol{X}\ ,\ \boldsymbol{u} \in \boldsymbol{U} \end{aligned} \tag{18}$$

where it is assumed that terms f_i and g_i are known and continuously differentiable with respect to their input variables, the state variables and $\boldsymbol{\theta}$, and that the state and input constraint sets $\boldsymbol{X}$ and $\boldsymbol{U}$ are compact. The current states and inputs of the system can be discretized as $x_1[n] \cdots x_N[n]$ and $u_1[n] \cdots u_R[n]$ where n symbolizes the sampling instant as $t = nT_s$. One-step ahead system states and outputs, i.e. $x_i[n+1]$ and $y_i[n+1]$, can be approximated via the fourth-order Runge–Kutta integration algorithm as follows.

$$\begin{aligned} \hat{x}_1[n+1] &= \hat{x}_1[n] + \frac{1}{6}K_{1X_1}[n] + \frac{2}{6}K_{2X_1}[n] + \frac{2}{6}K_{3X_1}[n] + \frac{1}{6}K_{4X_1}[n] \\ &\vdots \\ \hat{x}_N[n+1] &= \hat{x}_N[n] + \frac{1}{6}K_{1X_N}[n] + \frac{2}{6}K_{2X_N}[n] + \frac{2}{6}K_{3X_N}[n] + \frac{1}{6}K_{4X_N}[n] \end{aligned} \tag{19}$$

and

$$\begin{aligned} y_1[n+1] &= g_1(\hat{x}_1[n+1], \ldots, \hat{x}_N[n+1], u_1[n], \ldots, u_R[n]) \\ &\vdots \\ y_Q[n+1] &= g_Q(\hat{x}_1[n+1], \ldots, \hat{x}_N[n+1], u_1[n], \ldots, u_R[n]) \end{aligned} \tag{20}$$

where

$$\begin{aligned} K_{1X_1}[n] &= T_s f_1(\hat{x}_1[n], \ldots, \hat{x}_N[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}) \\ &\vdots \\ K_{1X_N}[n] &= T_s f_N(\hat{x}_1[n], \ldots, \hat{x}_N[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}) \end{aligned} \tag{21}$$

$$\begin{aligned} K_{2X_1}[n] &= T_s f_1\left(\hat{x}_1[n] + \frac{1}{2}K_{1X_1}[n], \ldots, \hat{x}_N[n] + \frac{1}{2}K_{1X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}\right) \\ &\vdots \\ K_{2X_N}[n] &= T_s f_N\left(\hat{x}_1[n] + \frac{1}{2}K_{1X_1}[n], \ldots, \hat{x}_N[n] + \frac{1}{2}K_{1X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}\right) \end{aligned} \tag{22}$$

$$\begin{aligned} K_{3X_1}[n] &= T_s f_1\left(\hat{x}_1[n] + \frac{1}{2}K_{2X_1}[n], \ldots, \hat{x}_N[n] + \frac{1}{2}K_{2X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}\right) \\ &\vdots \\ K_{3X_N}[n] &= T_s f_N\left(\hat{x}_1[n] + \frac{1}{2}K_{2X_1}[n], \ldots, \hat{x}_N[n] + \frac{1}{2}K_{2X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}\right) \end{aligned} \tag{23}$$

$$\begin{aligned} K_{4X_1}[n] &= T_s f_1(\hat{x}_1[n] + K_{3X_1}[n], \ldots, \hat{x}_N[n] + K_{3X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}) \\ &\vdots \\ K_{4X_N}[n] &= T_s f_N(\hat{x}_1[n] + K_{3X_1}[n], \ldots, \hat{x}_N[n] + K_{3X_N}[n], u_1[n], \ldots, u_R[n], \boldsymbol{\theta}) \end{aligned} \tag{24}$$

The Runge–Kutta integration method in (19) and (20) can be expressed in a more compact form as

$$\begin{aligned}\hat{\boldsymbol{x}}[n+1] &= \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}[n], \boldsymbol{u}[n], \boldsymbol{\theta}) \\ \hat{\boldsymbol{y}}[n+1] &= \boldsymbol{g}(\hat{\boldsymbol{x}}[n], \boldsymbol{u}[n])\end{aligned} \tag{25}$$

Thus, for the given values of current state variables $x_1[n] \cdots x_N[n]$ and input signals $u_1[n] \cdots u_R[n]$ at the sampling instants $t = nT_s$, one-step ahead system states and outputs can be estimated via (25). In a nutshell, applying the obtained states iteratively to (25), K-step ahead approximate future behaviour of the system and also system Jacobian (sensitivity of the system outputs with respect to control signal) which is a very significant part of the model-based control structures can be acquired. As can be seen from the compact form in (25), determination of the current states of the system ($\hat{x}[n]$) and system model parameters ($\boldsymbol{\theta}$) are crucial to obtain K-step ahead future predictions. Therefore, in the following section (Sect. 3.2), firstly, in order to estimate the current system states, Runge–Kutta model-based EKF is proposed. Then, in order to approximate the unknown system parameters($\boldsymbol{\theta}$), Runge–Kutta model-based online system model parameter estimator is examined in Sect. 3.3.

3.2 Runge–Kutta model-based EKF

In Runge–Kutta identification block, the correct estimation of the current system states $\hat{x}_1[n] \cdots \hat{x}_N[n]$ at any time during the control period is required to attain future behaviour of the system states. Therefore, Runge–Kutta model-based EKF is deployed to approximate current states. For this reason, it is crucial to bethink EKF. The EKF has become just about the most popular tool for state estimation owing to its simplicity and its computational efficiency (Thrun et al. 2005). Let us consider a nonlinear discrete MIMO system as follows:

$$\begin{aligned}\boldsymbol{x}[n+1] &= \boldsymbol{h}(\boldsymbol{x}[n], \boldsymbol{u}[n]) + \boldsymbol{w}[n] \\ \boldsymbol{y}[n+1] &= \boldsymbol{g}(\boldsymbol{x}[n], \boldsymbol{u}[n]) + \boldsymbol{v}[n]\end{aligned} \tag{26}$$

where $\boldsymbol{x}$ represents the N-dimensional state vector to be approximated, $\boldsymbol{u} \in \Re^R$ denotes the input vector and $\boldsymbol{y} \in \Re^Q$ is the output vector, $\boldsymbol{w}$ is the vector of system noise with covariance matrix $\boldsymbol{Q}$ and $\boldsymbol{v}$ denotes the vector of measurement noise with covariance matrix $\boldsymbol{R}$. In EKF, estimation of the system states consists of two main steps: prediction and correction. In prediction step, the states and covariance matrix of the states are computed as follows:

$$\begin{aligned}\tilde{\boldsymbol{x}}^-[n] &= \boldsymbol{h}(\tilde{\boldsymbol{x}}[n-1], \boldsymbol{u}[n-1]) \\ \boldsymbol{P}^-[n] &= \boldsymbol{A}[n]\boldsymbol{P}[n-1]\boldsymbol{A}^T[n] + \boldsymbol{Q}\end{aligned} \tag{27}$$

where $\tilde{\boldsymbol{x}}^-[n]$ and $\boldsymbol{P}^-[n]$ denote the predicted state and covariance matrix at time n, $\tilde{\boldsymbol{x}}[n-1]$ and $\boldsymbol{P}[n-1]$ stand for the corrected state and covariance matrix at time $n-1$ and $\boldsymbol{A}[n]$ is the state transition matrix of linearized system (Iplikci 2013; Thrun et al. 2005). In correction step, using the measurements from system, the predicted states $\tilde{\boldsymbol{x}}^-[n]$ and covariance matrix of the states $\boldsymbol{P}^-[n]$ are corrected as follows:

$$\begin{aligned}\boldsymbol{K}[n] &= \boldsymbol{P}^-[n]\boldsymbol{H}^T[n]\left(\boldsymbol{H}[n]\boldsymbol{P}^-[n]\boldsymbol{H}^T[n] + \boldsymbol{R}\right)^{-1} \\ \tilde{\boldsymbol{x}}[n] &= \tilde{\boldsymbol{x}}^-[n] + \boldsymbol{K}[n]\left(\boldsymbol{y}[n] - \boldsymbol{g}(\tilde{\boldsymbol{x}}^-[n], \boldsymbol{u}[n-1])\right) \\ \boldsymbol{P}[n] &= \left(\boldsymbol{I} - \boldsymbol{K}[n]\boldsymbol{H}[n]\right)\boldsymbol{P}^-[n]\end{aligned} \tag{28}$$

where $\boldsymbol{K}[n]$ is the Kalman gain of filter, $\tilde{\boldsymbol{x}}[n]$ and $\boldsymbol{P}[n]$ are corrected and estimated system state vector and corresponding covariance matrix. Jacobian $\boldsymbol{A}[n]$ and $\boldsymbol{H}[n]$ for EKF can be acquired as follows:

$$\begin{aligned}\boldsymbol{A}[n] &= \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\begin{bmatrix}\boldsymbol{x} = \tilde{\boldsymbol{x}}[n-1] \\ \boldsymbol{u} = \boldsymbol{u}[n-1]\end{bmatrix}} \\ \boldsymbol{H}[n] &= \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\begin{bmatrix}\boldsymbol{x} = \tilde{\boldsymbol{x}}[n-1] \\ \boldsymbol{u} = \boldsymbol{u}[n-1]\end{bmatrix}}\end{aligned} \tag{29}$$

In this study, since the systems under investigation are continuous and EKF is convenient for systems in discrete form, Runge–Kutta discretization method given in (25) can be utilized as discrete model of the controlled system. Thus, the Jacobian $\boldsymbol{A}[n]$ and $\boldsymbol{H}[n]$ matrices can be obtained as follows:

$$\begin{aligned}\boldsymbol{A}[n] &= \left.\frac{\partial \hat{\boldsymbol{f}}}{\partial \boldsymbol{x}}\right|_{\begin{bmatrix}\boldsymbol{x} = \tilde{\boldsymbol{x}}[n-1] \\ \boldsymbol{u} = \boldsymbol{u}[n-1]\end{bmatrix}} \\ \boldsymbol{H}[n] &= \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\begin{bmatrix}\boldsymbol{x} = \tilde{\boldsymbol{x}}[n-1] \\ \boldsymbol{u} = \boldsymbol{u}[n-1]\end{bmatrix}}\end{aligned} \tag{30}$$

where

$$\begin{aligned}\left.\frac{\partial \hat{\boldsymbol{f}}}{\partial \boldsymbol{x}}\right|_{\begin{bmatrix}\boldsymbol{x} = \tilde{\boldsymbol{x}}[n-1] \\ \boldsymbol{u} = \boldsymbol{u}[n-1]\end{bmatrix}} &= \left[\frac{\partial \hat{f}_i(\tilde{\boldsymbol{x}}[n-1], \boldsymbol{u}[n-1])}{\partial \tilde{x}_j[n-1]}\right] \\ &= \left[\frac{\partial \tilde{x}_i[n]}{\partial \tilde{x}_j[n-1]}\right] \text{ for } i = 1, \ldots, N \text{ and } j = 1, \ldots, N\end{aligned} \tag{31}$$

and

$$\frac{\partial \tilde{x}_i[n]}{\partial \tilde{x}_j[n-1]} = \delta_{i,j} + \frac{1}{6}\frac{\partial K_{1X_i}[n-1]}{\partial \tilde{x}_j[n-1]} + \frac{2}{6}\frac{\partial K_{2X_i}[n-1]}{\partial \tilde{x}_j[n-1]} + \frac{2}{6}\frac{\partial K_{3X_i}[n-1]}{\partial \tilde{x}_j[n-1]} + \frac{1}{6}\frac{\partial K_{4X_i}[n-1]}{\partial \tilde{x}_j[n-1]} \tag{32}$$

$$\frac{\partial K_{1X_i}[n-1]}{\partial \tilde{x}_j[n-1]} = T_s \frac{\partial f_i(\tilde{x}_1[n-1],\ldots,\tilde{x}_N[n-1],u_1[n-1],\ldots,u_R[n-1])}{\partial \tilde{x}_j[n-1]} = Ts\frac{\partial f_i}{\partial x_j}\bigg|_{\begin{bmatrix}\boldsymbol{x}=\tilde{\boldsymbol{x}}[n-1]\\ \boldsymbol{u}=\boldsymbol{u}[n-1]\end{bmatrix}} \tag{33}$$

$$\frac{\partial K_{2X_i}[n-1]}{\partial \tilde{x}_j[n-1]} = T_s\left(\sum_{p=1}^{N}\frac{\partial f_i}{\partial x_p}\left(\delta_{p,j}+\frac{1}{2}\frac{\partial K_{1X_p}[n-1]}{\partial \tilde{x}_j[n-1]}\right)\right)\Bigg|_{\begin{bmatrix} x_1=\tilde{x}_1[n-1]+\frac{1}{2}K_{1X_1}[n-1]\\ \vdots \\ x_N=\tilde{x}_N[n-1]+\frac{1}{2}K_{1X_N}[n-1]\\ \boldsymbol{u}=\boldsymbol{u}[n-1]\end{bmatrix}} \tag{34}$$

$$\frac{\partial K_{3X_i}[n-1]}{\partial \tilde{x}_j[n-1]} = T_s\left(\sum_{p=1}^{N}\frac{\partial f_i}{\partial x_p}\left(\delta_{p,j}+\frac{1}{2}\frac{\partial K_{2X_p}[n-1]}{\partial \tilde{x}_j[n-1]}\right)\right)\Bigg|_{\begin{bmatrix} x_1=\tilde{x}_1[n-1]+\frac{1}{2}K_{2X_1}[n-1]\\ \vdots \\ x_N=\tilde{x}_N[n-1]+\frac{1}{2}K_{2X_N}[n-1]\\ \boldsymbol{u}=\boldsymbol{u}[n-1]\end{bmatrix}} \tag{35}$$

$$\frac{\partial K_{4X_i}[n-1]}{\partial \tilde{x}_j[n-1]} = T_s\left(\sum_{p=1}^{N}\frac{\partial f_i}{\partial x_p}\left(\delta_{p,j}+\frac{\partial K_{3X_p}[n-1]}{\partial \tilde{x}_j[n-1]}\right)\right)\Bigg|_{\begin{bmatrix} x_1=\tilde{x}_1[n-1]+K_{3X_1}[n-1]\\ \vdots \\ x_N=\tilde{x}_N[n-1]+K_{3X_N}[n-1]\\ \boldsymbol{u}=\boldsymbol{u}[n-1]\end{bmatrix}} \tag{36}$$

where

$$\delta_{i,j} = \begin{cases}1, & i=j\\ 0, & i\neq j\end{cases}$$

Consequently, the current states of the system can be approximated by its Runge–Kutta model utilized in the EKF algorithm via (27–36) (Iplikci 2013).

3.3 The Runge–Kutta model-based online parameter estimation

In case any system model parameter deviates from its actual value, the identification performance of RK$_{\text{model}}$ deteriorates. Therefore, online estimation of system parameters is a vital step to enhance identification performance of RK$_{\text{model}}$. If the Runge–Kutta model of the system is utilized, the current state of the system to its previous state $(x_1[n],\ldots,x_N[n])$, inputs $(u_1[n],\ldots,u_R[n])$ and parameters $(\boldsymbol{\theta})$ can be easily related by (19, 21–24). The parameter vector of the system can be adjusted as

$$\boldsymbol{\theta}[n+1] = \boldsymbol{\theta}[n] - \frac{\boldsymbol{J}_{\boldsymbol{\theta}}^T\boldsymbol{e}}{\boldsymbol{J}_{\boldsymbol{\theta}}^T\boldsymbol{J}_{\boldsymbol{\theta}}} \tag{37}$$

where

$$\begin{aligned}\boldsymbol{J}_\theta &= \left[\frac{\partial e_1[n+1]}{\partial\boldsymbol{\theta}[n]}\ \cdots\ \frac{\partial e_N[n+1]}{\partial\boldsymbol{\theta}[n]}\right]^T\\ &= -\left[\frac{\partial \hat{x}_1[n+1]}{\partial\boldsymbol{\theta}[n]}\ \cdots\ \frac{\partial \hat{x}_N[n+1]}{\partial\boldsymbol{\theta}[n]}\right]^T\end{aligned} \tag{38}$$

and

$$\boldsymbol{e} = \begin{bmatrix} e_1[n+1]\\ \vdots\\ e_N[n+1]\end{bmatrix} = \begin{bmatrix} x_1[n+1]-\hat{x}_1[n+1]\\ \vdots \\ x_N[n+1]-\hat{x}_N[n+1]\end{bmatrix} \tag{39}$$

by assuming that previous state $\boldsymbol{x}[n]$ and current state $\boldsymbol{x}[n+1]$ of a nonlinear system (14) are given directly (or estimated by EKF) at time $(n+1)T_s$ and that the previous control input $\boldsymbol{u}[n]$ is known (Iplikci 2013). The term $\frac{\partial \hat{x}_i[n+1]}{\partial \boldsymbol{\theta}[n]}$ required for the construction of Jacobian in (38) can be acquired as

$$\frac{\partial \hat{x}_i[n+1]}{\partial \boldsymbol{\theta}[n]} = \frac{\partial \hat{x}_i[n]}{\partial \boldsymbol{\theta}[n]} + \frac{1}{6}\frac{\partial K_{1X_i}[n]}{\partial \boldsymbol{\theta}[n]} + \frac{2}{6}\frac{\partial K_{2X_i}[n]}{\partial \boldsymbol{\theta}[n]} + \frac{2}{6}\frac{\partial K_{3X_i}[n]}{\partial \boldsymbol{\theta}[n]} + \frac{1}{6}\frac{\partial K_{4X_i}[n]}{\partial \boldsymbol{\theta}[n]} \tag{40}$$

where

$$\begin{aligned}\frac{\partial K_{1X_i}[n]}{\partial \boldsymbol{\theta}[n]} &= T_s \frac{\partial f_i(\tilde{x}_1[n],\ldots,\tilde{x}_N[n],u_1[n],\ldots,u_R[n]),\boldsymbol{\theta}[n]}{\partial \boldsymbol{\theta}[n]}\\ &= T_s \frac{\partial f_i}{\partial \boldsymbol{\theta}}\Bigg|_{\begin{bmatrix}\boldsymbol{x}=\tilde{\boldsymbol{x}}[n]\\ \boldsymbol{u}=\boldsymbol{u}[n]\\ \boldsymbol{\theta}=\boldsymbol{\theta}[n]\end{bmatrix}}\end{aligned} \tag{41}$$

$$\frac{\partial K_{2X_i}[n]}{\partial \boldsymbol{\theta}[n]} = T_s\left(\frac{\partial f_i}{\partial \boldsymbol{\theta}} + \frac{1}{2}\sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\frac{\partial K_{1X_j}[n]}{\partial \boldsymbol{\theta}}\right)\Bigg|_{\begin{bmatrix}x_1=\hat{x}_1[n]+\frac{1}{2}K_{1X_1}[n]\\ \vdots\\ x_N=\hat{x}_N[n]+\frac{1}{2}K_{1X_N}[n]\\ \boldsymbol{u}=\boldsymbol{u}[n]\\ \boldsymbol{\theta}=\boldsymbol{\theta}[n]\end{bmatrix}} \tag{42}$$

$$\frac{\partial K_{3X_i}[n]}{\partial \boldsymbol{\theta}[n]} = T_s\left(\frac{\partial f_i}{\partial \boldsymbol{\theta}} + \frac{1}{2}\sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\frac{\partial K_{2X_j}[n]}{\partial \boldsymbol{\theta}}\right)\Bigg|_{\begin{bmatrix}x_1=\hat{x}_1[n]+\frac{1}{2}K_{2X_1}[n]\\ \vdots\\ x_N=\hat{x}_N[n]+\frac{1}{2}K_{2X_N}[n]\\ \boldsymbol{u}=\boldsymbol{u}[n]\\ \boldsymbol{\theta}=\boldsymbol{\theta}[n]\end{bmatrix}} \tag{43}$$

$$\frac{\partial K_{4X_i}[n]}{\partial \boldsymbol{\theta}[n]} = T_s\left(\frac{\partial f_i}{\partial \boldsymbol{\theta}} + \sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\frac{\partial K_{3X_j}[n]}{\partial \boldsymbol{\theta}}\right)\Bigg|_{\begin{bmatrix}x_1=\hat{x}_1[n]+K_{3X_1}[n]\\ \vdots\\ x_N=\hat{x}_N[n]+K_{3X_N}[n]\\ \boldsymbol{u}=\boldsymbol{u}[n]\\ \boldsymbol{\theta}=\boldsymbol{\theta}[n]\end{bmatrix}} \tag{44}$$

Thus, the derivatives for Runge–Kutta-based model parameter estimator can be attained.

3.4 K-step ahead future system behaviour predictions and Jacobian computations

The K-step ahead future behaviour of the system can be approximated by feeding back the obtained values of states to RK model given in (25), and by assuming that the control signal vector $\boldsymbol{u}[n]$ remains unchanged during the prediction process between time instants $[t+T_s \quad t+KT_s]$:

$$\begin{aligned}\hat{\boldsymbol{x}}[n+k] &= \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}[n+k-1],\boldsymbol{u}[n],\boldsymbol{\theta})\\ \hat{\boldsymbol{y}}[n+k] &= \boldsymbol{g}(\hat{\boldsymbol{x}}[n+k-1],\boldsymbol{u}[n]) \text{ for } k=1,\ldots,K\end{aligned} \tag{45}$$

Thus, a series of future predictions is obtained for each output as (Iplikci 2013)

$$\left[\hat{y}_q[n+1]\cdots\hat{y}_q[n+K]\right] \quad \text{for} \quad q=1,\ldots,Q \tag{46}$$

In order to attain the system Jacobian which is the most significant part of the model-based adaptive mechanism, firstly, (19)–(24) can be expressed in an iterative way as follows:

$$\begin{aligned}\hat{x}_i[n+k] &= \hat{x}_i[n+k-1] + \frac{1}{6}K_{1X_i}[n+k-1] + \frac{2}{6}K_{2X_i}[n+k-1]\\ &\quad + \frac{2}{6}K_{3X_i}[n+k-1] + \frac{1}{6}K_{4X_i}[n+k-1]\end{aligned} \tag{47}$$

for $i=1,\ldots,N$ and

$$\begin{aligned}\hat{y}_q[n+k] = g_q(&\hat{x}_1[n+k-1],\ldots,\hat{x}_N[n+k-1],\\ &u_1[n],\ldots,u_R[n])\end{aligned} \tag{48}$$

for $q=1,\ldots,Q$ where

$$\begin{aligned}K_{1X_i}[n+k-1] &= T_s f_i(\hat{x}_1[n+k-1],\ldots,\hat{x}_N[n+k-1],\\ &\qquad u_1[n],\ldots,u_R[n],\boldsymbol{\theta})\\ K_{2X_i}[n+k-1] &= T_s f_i(\hat{x}_1[n+k-1]\\ &\quad + \frac{1}{2}K_{1X_1}[n+k-1],\ldots,\hat{x}_N[n+k-1]\\ &\quad + \frac{1}{2}K_{1X_N}[n+k-1],u_1[n],\ldots,u_R[n],\boldsymbol{\theta})\\ K_{3X_i}[n+k-1] &= T_s f_i(\hat{x}_1[n+k-1]\\ &\quad + \frac{1}{2}K_{2X_1}[n+k-1],\ldots,\hat{x}_N[n+k-1]\\ &\quad + \frac{1}{2}K_{2X_N}[n+k-1],u_1[n],\ldots,u_R[n],\boldsymbol{\theta})\\ K_{4X_i}[n+k-1] &= T_s f_i(\hat{x}_1[n+k-1]\\ &\quad + K_{3X_1}[n+k-1],\ldots,\hat{x}_N[n+k-1]\\ &\quad + K_{3X_N}[n+k-1],u_1[n],\ldots,u_R[n],\boldsymbol{\theta})\end{aligned} \tag{49}$$

Thus, $\frac{\partial \hat{y}_q[n+k]}{\partial u_r[n]}$ term can be derived as follows:

$$\frac{\partial \hat{y}_q[n+k]}{\partial u_r[n]} = \left(\frac{\partial g_q}{\partial u_r} + \sum_{i=1}^{N} \frac{\partial g_q}{\partial x_i}\frac{\partial \hat{x}_i[n+k]}{\partial u_r[n]}\right)\Bigg|\begin{bmatrix} x_1 = \hat{x}_1[n+k] \\ \vdots \\ x_N = \hat{x}_N[n+k] \\ g_q = g_q(\hat{x}_1[n+k],\ldots,\hat{x}_N[n+k],u_1[n],\ldots,u_R[n]) \\ u_r = u_r[n] \end{bmatrix} \tag{50}$$

$\frac{\partial \hat{x}_i[n+k]}{\partial u_r[n]}$ can be computed as follows:

$$\frac{\hat{x}_i[n+k]}{\partial u_r[n]} = \frac{\hat{x}_i[n+k-1]}{\partial u_r[n]} + \frac{1}{6}\frac{K_{1X_i}[n+k-1]}{\partial u_r[n]} + \frac{2}{6}\frac{K_{2X_i}[n+k-1]}{\partial u_r[n]} + \frac{2}{6}\frac{K_{3X_i}[n+k-1]}{\partial u_r[n]} + \frac{1}{6}\frac{K_{4X_i}[n+k-1]}{\partial u_r[n]} \tag{51}$$

where

$$\frac{\partial K_{1X_i}[n]}{\partial u_r[n]} = T_s \frac{\partial f_i(\tilde{x}_1[n],\ldots,\tilde{x}_N[n],u_1[n],\ldots,u_R[n]),\boldsymbol{\theta}[n]}{\partial u_r[n]} = T_s\left(\frac{\partial f_i}{\partial u_r} + \sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\frac{\partial \hat{x}_j[n+k-1]}{\partial u_r[n]}\right)\Bigg|\begin{bmatrix} x_1 = \hat{x}_1[n+k-1] \\ \vdots \\ x_N = \hat{x}_N[n+k-1] \end{bmatrix} \tag{52}$$

and

$$\frac{\partial K_{2X_i}[n]}{\partial u_r[n]} = T_s\left(\frac{\partial f_i}{\partial u_r} + \sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\left(\frac{\partial \hat{x}_j[n+k-1]}{\partial u_r[n]} + \frac{1}{2}\frac{\partial K_{1X_j}[n+k-1]}{\partial u_r[n]}\right)\right)\Bigg|\begin{bmatrix} x_1 = \hat{x}_1[n] + \frac{1}{2}K_{1X_1}[n] \\ \vdots \\ x_N = \hat{x}_N[n] + \frac{1}{2}K_{1X_N}[n] \end{bmatrix} \tag{53}$$

$$\frac{\partial K_{3X_i}[n]}{\partial u_r[n]} = T_s\left(\frac{\partial f_i}{\partial u_r} + \sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\left(\frac{\partial \hat{x}_j[n+k-1]}{\partial u_r[n]} + \frac{1}{2}\frac{\partial K_{2X_j}[n+k-1]}{\partial u_r[n]}\right)\right)\Bigg|\begin{bmatrix} x_1 = \hat{x}_1[n] + \frac{1}{2}K_{2X_1}[n] \\ \vdots \\ x_N = \hat{x}_N[n] + \frac{1}{2}K_{2X_N}[n] \end{bmatrix} \tag{54}$$

$$\frac{\partial K_{4X_i}[n]}{\partial u_r[n]} = T_s\left(\frac{\partial f_i}{\partial u_r} + \sum_{j=1}^{N}\frac{\partial f_i}{\partial x_j}\left(\frac{\partial \hat{x}_j[n+k-1]}{\partial u_r[n]} + \frac{\partial K_{3X_j}[n+k-1]}{\partial u_r[n]}\right)\right)\Bigg|\begin{bmatrix} x_1 = \hat{x}_1[n] + K_{3X_1}[n] \\ \vdots \\ x_N = \hat{x}_N[n] + K_{3X_N}[n] \end{bmatrix} \tag{55}$$

As a result, all derivations needed to constitute system Jacobian information can be obtained.

4 Runge–Kutta neural network-based adaptive control structure

4.1 An overview of Runge–Kutta neural network

Let us consider a nonlinear system characterized by the following ODE

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) \tag{56}$$

with the initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$. In the case that f is known, using fourth-order Runge–Kutta integration formulas, one-step ahead behaviour of the system dynamics can be computed as follows:

$$\boldsymbol{x}[n+1] = \boldsymbol{x}[n] + \frac{1}{6}h\left[K_{1\boldsymbol{x}}[n] + 2K_{2\boldsymbol{x}}[n] + 2K_{3\boldsymbol{x}}[n] + K_{4\boldsymbol{x}}[n]\right] \tag{57}$$

where h stands for Runge–Kutta integration step size (Efe and Kaynak 1999), $K_{1\boldsymbol{x}}[n]$, $K_{2\boldsymbol{x}}[n]$, $K_{3\boldsymbol{x}}[n]$ and $K_{4\boldsymbol{x}}[n]$ are the slopes utilized to obtain the changing rates of the system states and are given as (Iplikci 2013; Wang and Lin 1998; Efe and Kaynak 1999)

$$\begin{aligned} K_{1\boldsymbol{x}}[n] &= f(\boldsymbol{x}_c[n])\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]} \\ K_{2\boldsymbol{x}}[n] &= f(\boldsymbol{x}_c[n])\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+\frac{1}{2}hK_{1\boldsymbol{x}}[n]} \\ K_{3\boldsymbol{x}}[n] &= f(\boldsymbol{x}_c[n])\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+\frac{1}{2}hK_{2\boldsymbol{x}}[n]} \\ K_{4\boldsymbol{x}}[n] &= f(\boldsymbol{x}_c[n])\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+hK_{3\boldsymbol{x}}[n]} \end{aligned} \tag{58}$$

If function f is unknown, a neural network (NN) structure can be constructed to precisely identify f so as to approximate these four slopes such that NN can successfully perform long-term prediction of the state trajectory $\boldsymbol{x}(t)$ of the system described in (56). Thus, the powerful integration feature of Runge–Kutta method and powerful approximation and generalization abilities of NN structure can be combined in RK-NN network structure. The input and output relationship of the fourth-order RK-NN can be expressed as

$$\boldsymbol{x}[n+1] = \boldsymbol{x}[n] + \frac{1}{6}h\Big[K_{1\boldsymbol{x}}[n] + 2K_{2\boldsymbol{x}}[n] + 2K_{3\boldsymbol{x}}[n] + K_{4\boldsymbol{x}}[n]\Big] \tag{59}$$

where

$$\begin{aligned} K_{1\boldsymbol{x}}[n] &= N_f(\boldsymbol{x}_c[n], \boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]} \\ K_{2\boldsymbol{x}}[n] &= N_f(\boldsymbol{x}_c[n], \boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+\frac{1}{2}hK_{1\boldsymbol{x}}[n]} \\ K_{3\boldsymbol{x}}[n] &= N_f(\boldsymbol{x}_c[n], \boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+\frac{1}{2}hK_{2\boldsymbol{x}}[n]} \\ K_{4\boldsymbol{x}}[n] &= N_f(\boldsymbol{x}_c[n], \boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n]=\boldsymbol{x}[n]+hK_{3\boldsymbol{x}}[n]} \end{aligned} \tag{60}$$

$N_f(\boldsymbol{x}[n], \boldsymbol{\Theta})$ with $\boldsymbol{x}[n]$ and weights $\boldsymbol{\Theta}$ can be selected to be a multilayer perceptron network (MLP) as given in Fig. 4 or radial basis function network given in Fig. 6 or any nonlinear regression network. The network topology of the RK-NN is illustrated in Fig. 5. It is significant to note that the four $N_f(\boldsymbol{x}[n], \boldsymbol{\Theta})$ subnetworks in Fig. 5 are identical, meaning that they have the same network structure and utilize the same corresponding weights (Wang and Lin 1998). As can be seen from Fig. 5 and (60), in order to obtain slopes $K_{1\boldsymbol{x}}[n]$, $K_{2\boldsymbol{x}}[n]$, $K_{3\boldsymbol{x}}[n]$ and $K_{4\boldsymbol{x}}[n]$, the output of the constituent subnetwork is consecutively applied to itself. The fact that n subnetworks of an $n-order$ RK-NN are identical facilitates the realization of the RK-NN in both software or hardware implementations (Wang and Lin 1998). That is, the real network size of an n-order RK-NN is the same as that of its constituent subnetwork (Wang and Lin 1998). As a constituent network, MLP or RBF neural network structures can be used. The input–output relationship of the MLP-NN model is described by

$$K_{m\boldsymbol{x}}[n] = \sum_{j=1}^{S} w^o_{1,j}\boldsymbol{\Phi}(d_j[n]) + b^o_1\ ,\ m \in \{1,2,3,4\} \tag{61}$$

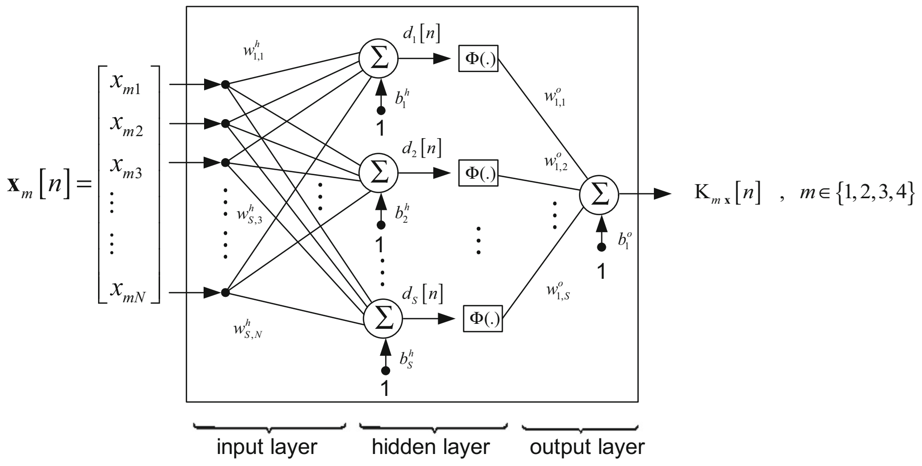
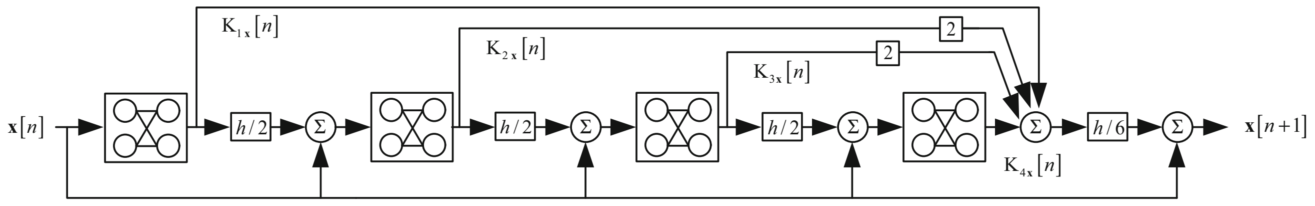


Fig. 4 MLP neural network structure



$\mathbf{x}[n]$
$\mathrm{K}_{1\mathbf{x}}[n]$
$\mathrm{K}_{2\mathbf{x}}[n]$
$\mathrm{K}_{3\mathbf{x}}[n]$
$\mathrm{K}_{4\mathbf{x}}[n]$
$h/2$
$h/6$
2
Σ
$\mathbf{x}[n+1]$

Fig. 5 Runge–Kutta neural network structure

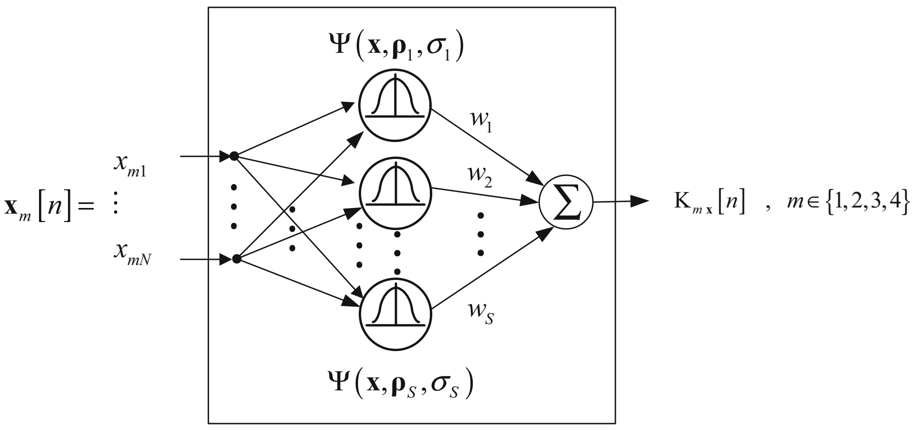

$\Psi(\mathbf{x},\boldsymbol{\rho}_1,\sigma_1)$
w_1
w_2
w_S
x_{m1}
$\mathbf{x}_m[n]=$
x_{mN}
$\Psi(\mathbf{x},\boldsymbol{\rho}_S,\sigma_S)$
$\mathrm{K}_{m\mathbf{x}}[n]$, $m\in\{1,2,3,4\}$

Fig. 6 RBF neural network structure

where $\boldsymbol{\Phi}(,)$ is the activation function, S is the number of the neurons in hidden layer, and

$$d_j[n]=\sum_{i=1}^{N}w_{j,i}^{h}\,x_{m,i}+b_j^h \tag{62}$$

N denotes the entry number of the MLP-NN. The regression function of RBF NN model can be attained as

$$K_{m\boldsymbol{x}}[n]=\sum_{i=1}^{S}w_i\boldsymbol{\Psi}(\boldsymbol{x}_m[n],\boldsymbol{\rho}_i[n],\sigma_i[n])\;,\;m\in\{1,2,3,4\} \tag{63}$$

where S denotes number of the neurons, $\boldsymbol{\rho}_i[n]$ and $\sigma_i[n]$ stand for centre vector and the bandwidth of neurons, respectively, and

$$\boldsymbol{\Psi}(\boldsymbol{x}_m[n],\boldsymbol{\rho}_i[n],\sigma_i[n])=exp\left(\frac{-\|\boldsymbol{x}_m[n]-\boldsymbol{\rho}_i[n]\|^2}{\sigma_i^2[n]}\right) \tag{64}$$

The tuning rules for the weights of the RK-NN structure used for estimation of the controller parameters are detailed in the following subsections.

4.2 Identification of controller parameters via Runge–Kutta neural network

Consider that the control law performed by the generalized controller is given as

$$\boldsymbol{u}[n]=\boldsymbol{f}_c(\boldsymbol{\beta}[n],\boldsymbol{C}_{in})=\begin{bmatrix}u_1[n]\\ \vdots\\ u_R[n]\end{bmatrix}=\boldsymbol{f}_c\left(\begin{bmatrix}\beta_1\\ \vdots\\ \beta_Z\end{bmatrix},\begin{bmatrix}c_1\\ \vdots\\ c_I\end{bmatrix}\right) \tag{65}$$

where R indicates the number of control inputs, Z denotes the number of adjustable controller parameters and I represents the number of controller inputs. In order to force the output of the controlled system to the desired reference signal, the dynamics of the controller parameters are identified by adjusting the weights of the RK-NN structure using Levenberg–Marquardt algorithm-based update rules given in (3–5, 11, 13). Thus, the dynamic behaviour of the controller parameters can be identified using RK-NN subnetworks illustrated in Fig. 5 as follows:

$$\begin{aligned}\boldsymbol{\beta}[n]=\boldsymbol{\beta}[n-1]+\frac{1}{6}h\Big[&K_{1\boldsymbol{\beta}}[n-1]+2K_{2\boldsymbol{\beta}}[n-1]\\ &+2K_{3\boldsymbol{\beta}}[n-1]+K_{4\boldsymbol{\beta}}[n-1]\Big]\end{aligned} \tag{66}$$

where

$$K_{1\boldsymbol{\beta}}[n-1]=N_f(\boldsymbol{x}_c[n-1],\boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n-1]=\boldsymbol{\beta}[n-1]}$$
$$K_{2\boldsymbol{\beta}}[n-1]=N_f(\boldsymbol{x}_c[n-1],\boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n-1]=\boldsymbol{\beta}[n-1]+\frac{1}{2}hK_{1\boldsymbol{\beta}}[n-1]}$$
$$K_{3\boldsymbol{\beta}}[n-1]=N_f(\boldsymbol{x}_c[n-1],\boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n-1]=\boldsymbol{\beta}[n-1]+\frac{1}{2}hK_{2\boldsymbol{\beta}}[n-1]}$$
$$K_{4\boldsymbol{\beta}}[n-1]=N_f(\boldsymbol{x}_c[n-1],\boldsymbol{\Theta})\Big|_{\boldsymbol{x}_c[n-1]=\boldsymbol{\beta}[n-1]+hK_{3\boldsymbol{\beta}}[n-1]}$$

Since RK-NN$_{\text{estimator}}$ with RBF has multi-input single-output (MISO) structure, a separate RK-NN$_{\text{estimator}}$ is deployed for each approximated controller parameter. Therefore, the number of the RK-NN$_{\text{estimator}}$ structures to be used in parameter estimator block depends on the number of adjustable parameters in the controller. For instance, three RK-NN$_{\text{estimator}}$ structures are employed for a SISO PID controller to forecast K_p, K_i and K_d parameters. If it is assumed that RBF network is employed in constituent subnetwork, the network parameters of the zth estimator to be adjusted are given as follows:

$$\boldsymbol{\Theta}_z = \left[w_1 \cdots w_s \ \rho_{11} \cdots \rho_{1N} \cdots\cdots \rho_{S1} \cdots \rho_{SN} \ \sigma_1 \cdots \sigma_S\right]^T \tag{67}$$

Using Levenberg–Marquardt rule in (3), the weights of the constituent subnetwork of zth parameter estimator can be optimized as

$$\begin{aligned} \boldsymbol{\Theta}_z^{new} &= \boldsymbol{\Theta}_z^{old} + \Delta\boldsymbol{\Theta}_z \\ \Delta\boldsymbol{\Theta}_z &= -\left(\boldsymbol{J}_z^T\boldsymbol{J}_z + \mu\boldsymbol{I}\right)^{-1}\boldsymbol{J}_z^T\hat{\boldsymbol{e}} \end{aligned} \tag{68}$$

where $\boldsymbol{J}_z$ is a $\left(QK+R\right)x\left(N\left(S+2\right)\right)$ dimension Jacobian matrix given as

$$\boldsymbol{J}_z = \begin{bmatrix} \frac{\partial \hat{e}_1[n+1]}{\partial w_1} & \cdots & \frac{\partial \hat{e}_1[n+1]}{\partial w_S} & \frac{\partial \hat{e}_1[n+1]}{\partial \rho_{11}} & \cdots & \frac{\partial \hat{e}_1[n+1]}{\partial \rho_{SN}} & \frac{\partial \hat{e}_1[n+1]}{\partial \sigma_1} & \cdots & \frac{\partial \hat{e}_1[n+1]}{\partial \sigma_S} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{e}_Q[n+K]}{\partial w_1} & \cdots & \frac{\partial \hat{e}_Q[n+K]}{\partial w_S} & \frac{\partial \hat{e}_Q[n+K]}{\partial \rho_{11}} & \cdots & \frac{\partial \hat{e}_Q[n+K]}{\partial \rho_{SN}} & \frac{\partial \hat{e}_Q[n+K]}{\partial \sigma_1} & \cdots & \frac{\partial \hat{e}_Q[n+K]}{\partial \sigma_S} \\ \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial w_1} & \cdots & \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial w_S} & \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \rho_{11}} & \cdots & \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \rho_{SN}} & \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \sigma_1} & \cdots & \sqrt{\lambda_1}\frac{\partial \Delta u_1[n]}{\partial \sigma_S} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial w_1} & \cdots & \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial w_S} & \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \rho_{11}} & \cdots & \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \rho_{SN}} & \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \sigma_1} & \cdots & \sqrt{\lambda_R}\frac{\partial \Delta u_R[n]}{\partial \sigma_S} \end{bmatrix}_{[(QK+R)x(Nx(S+2))]} \tag{69}$$

and $\hat{\boldsymbol{e}}$ is the vector of the prediction errors

$$\hat{\boldsymbol{e}} = \begin{bmatrix} \hat{e}_1[n+1] \\ \vdots \\ \hat{e}_1[n+K] \\ \hat{e}_2[n+1] \\ \vdots \\ \hat{e}_2[n+K] \\ \vdots \\ \hat{e}_Q[n+1] \\ \vdots \\ \hat{e}_Q[n+K] \\ \sqrt{\lambda_1}\Delta u_1[n] \\ \vdots \\ \sqrt{\lambda_R}\Delta u_R[n] \end{bmatrix} = \begin{bmatrix} \hat{e}[n+1] \\ \vdots \\ \hat{e}[n+K] \\ \hat{e}[n+K+1] \\ \vdots \\ \hat{e}[n+2K] \\ \vdots \\ \hat{e}[n+(Q-1)K+1] \\ \vdots \\ \hat{e}[n+QK] \\ \sqrt{\lambda_1}\Delta u_1[n] \\ \vdots \\ \sqrt{\lambda_R}\Delta u_R[n] \end{bmatrix}$$

$$= \begin{bmatrix} r_1[n+1]-\hat{y}_1[n+1] \\ \vdots \\ r_1[n+K]-\hat{y}_1[n+K] \\ r_2[n+1]-\hat{y}_2[n+1] \\ \vdots \\ r_2[n+K]-\hat{y}_2[n+K] \\ \vdots \\ r_Q[n+1]-\hat{y}_Q[n+1] \\ \vdots \\ r_Q[n+K]-\hat{y}_Q[n+K] \\ \sqrt{\lambda_1}\left[u_1[n]-u_1[n-1]\right] \\ \vdots \\ \sqrt{\lambda_R}\left[u_R[n]-u_R[n-1]\right] \end{bmatrix}_{[(QK+R)x1]} \tag{70}$$

In order to construct the Jacobian given in (70), it is required to derive the $\frac{\partial \hat{e}_Q[n+K]}{\partial w_S}$, $\frac{\partial \hat{e}_Q[n+K]}{\partial \rho_{SN}}$, $\frac{\partial \hat{e}_Q[n+K]}{\partial \sigma_S}$, $\frac{\partial \Delta u_R[n]}{\partial w_S}$, $\frac{\partial \Delta u_R[n]}{\partial \rho_{SN}}$ and $\frac{\partial \Delta u_R[n]}{\partial \sigma_S}$ terms. By using chain rule, the mentioned terms can be attained as follows:

$$\begin{aligned} \frac{\partial \hat{e}_Q[n+K]}{\partial w_S} &= \frac{\partial \hat{e}_Q[n+K]}{\partial y_Q[n+K]}\left[\sum_{r=1}^{R}\frac{\partial y_Q[n+K]}{\partial u_r[n]}\frac{\partial u_r[n]}{\partial \beta_Z[n]}\right]\frac{\partial \beta_Z[n]}{\partial w_S} \\ \frac{\partial \hat{e}_Q[n+K]}{\partial \rho_{SN}} &= \frac{\partial \hat{e}_Q[n+K]}{\partial y_Q[n+K]}\left[\sum_{r=1}^{R}\frac{\partial y_Q[n+K]}{\partial u_r[n]}\frac{\partial u_r[n]}{\partial \beta_Z[n]}\right]\frac{\partial \beta_Z[n]}{\partial \rho_{SN}} \\ \frac{\partial \hat{e}_Q[n+K]}{\partial \sigma_S} &= \frac{\partial \hat{e}_Q[n+K]}{\partial y_Q[n+K]}\left[\sum_{r=1}^{R}\frac{\partial y_Q[n+K]}{\partial u_r[n]}\frac{\partial u_r[n]}{\partial \beta_Z[n]}\right]\frac{\partial \beta_Z[n]}{\partial \sigma_S} \\ \frac{\partial \Delta u_R[n]}{\partial w_S} &= \frac{\partial u_R[n]}{\partial \beta_Z[n]}\frac{\partial \beta_Z[n]}{\partial w_S} \\ \frac{\partial \Delta u_R[n]}{\partial \rho_{SN}} &= \frac{\partial u_R[n]}{\partial \beta_Z[n]}\frac{\partial \beta_Z[n]}{\partial \rho_{SN}} \\ \frac{\partial \Delta u_R[n]}{\partial \sigma_S} &= \frac{\partial u_R[n]}{\partial \beta_Z[n]}\frac{\partial \beta_Z[n]}{\partial \sigma_S} \end{aligned} \tag{71}$$

where

$$\frac{\partial \beta_Z[n]}{\partial w_S} = \frac{1}{6}h\Bigg[\frac{\partial K_{1\beta_Z}[n-1]}{\partial w_S} + 2\frac{\partial K_{2\beta_Z}[n-1]}{\partial w_S} + 2\frac{\partial K_{3\beta_Z}[n-1]}{\partial w_S} + \frac{\partial K_{4\beta_Z}[n-1]}{\partial w_S}\Bigg]$$
$$\frac{\partial \beta_Z[n]}{\partial \rho_{SN}} = \frac{1}{6}h\Bigg[\frac{\partial K_{1\beta_Z}[n-1]}{\partial \rho_{SN}} + 2\frac{\partial K_{2\beta_Z}[n-1]}{\partial \rho_{SN}} + 2\frac{\partial K_{3\beta_Z}[n-1]}{\partial \rho_{SN}} + \frac{\partial K_{4\beta_Z}[n-1]}{\partial \rho_{SN}}\Bigg]$$
$$\frac{\partial \beta_Z[n]}{\partial \sigma_S} = \frac{1}{6}h\Bigg[\frac{\partial K_{1\beta_Z}[n-1]}{\partial \sigma_S} + 2\frac{\partial K_{2\beta_Z}[n-1]}{\partial \sigma_S} + 2\frac{\partial K_{3\beta_Z}[n-1]}{\partial \sigma_S} + \frac{\partial K_{4\beta_Z}[n-1]}{\partial \sigma_S}\Bigg] \tag{72}$$

$\frac{\partial K_{m\beta_Z}[n-1]}{\partial w_S}$, $\frac{\partial K_{m\beta_Z}[n-1]}{\partial \rho_{SN}}$ and $\frac{\partial K_{m\beta_Z}[n-1]}{\partial \sigma_S}$ terms can be derived as given in (73–75):

$$\begin{aligned}
\frac{\partial K_{1\beta_Z}[n-1]}{\partial w_S} &= \left[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_S}\right]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]}\\
\frac{\partial K_{2\beta_Z}[n-1]}{\partial w_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{1\beta_Z}[n-1]}\frac{\partial K_{1\beta_Z}[n-1]}{\partial w_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{1\beta_Z}[n-1]}\\
\frac{\partial K_{3\beta_Z}[n-1]}{\partial w_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{2\beta_Z}[n-1]}\frac{\partial K_{2\beta_Z}[n-1]}{\partial w_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{2\beta_Z}[n-1]}\\
\frac{\partial K_{4\beta_Z}[n-1]}{\partial w_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{3\beta_Z}[n-1]}\frac{\partial K_{3\beta_Z}[n-1]}{\partial w_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+hK_{3\beta_Z}[n-1]}
\end{aligned} \tag{73}$$

$$\begin{aligned}
\frac{\partial K_{1\beta_Z}[n-1]}{\partial \rho_{SN}} &= \left[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{SN}}\right]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]}\\
\frac{\partial K_{2\beta_Z}[n-1]}{\partial \rho_{SN}} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{SN}} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{1\beta_Z}[n-1]}\frac{\partial K_{1\beta_Z}[n-1]}{\partial \rho_{SN}}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{1\beta_Z}[n-1]}\\
\frac{\partial K_{3\beta_Z}[n-1]}{\partial \rho_{SN}} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{SN}} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{2\beta_Z}[n-1]}\frac{\partial K_{2\beta_Z}[n-1]}{\partial \rho_{SN}}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{2\beta_Z}[n-1]}\\
\frac{\partial K_{4\beta_Z}[n-1]}{\partial \rho_{SN}} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{SN}} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{3\beta_Z}[n-1]}\frac{\partial K_{3\beta_Z}[n-1]}{\partial \rho_{SN}}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+hK_{3\beta_Z}[n-1]}
\end{aligned} \tag{74}$$

$$\begin{aligned}
\frac{\partial K_{1\beta_Z}[n-1]}{\partial \sigma_S} &= \left[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_S}\right]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]}\\
\frac{\partial K_{2\beta_Z}[n-1]}{\partial \sigma_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{1\beta_Z}[n-1]}\frac{\partial K_{1\beta_Z}[n-1]}{\partial \sigma_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{1\beta_Z}[n-1]}\\
\frac{\partial K_{3\beta_Z}[n-1]}{\partial \sigma_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{2\beta_Z}[n-1]}\frac{\partial K_{2\beta_Z}[n-1]}{\partial \sigma_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+\frac{1}{2}hK_{2\beta_Z}[n-1]}\\
\frac{\partial K_{4\beta_Z}[n-1]}{\partial \sigma_S} &= \Bigg[\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_S} + \frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]} \times \frac{\partial \boldsymbol{x}_Z[n-1]}{\partial K_{3\beta_Z}[n-1]}\frac{\partial K_{3\beta_Z}[n-1]}{\partial \sigma_S}\Bigg]\Bigg|_{\boldsymbol{x}_Z[n-1]=\beta_Z[n-1]+hK_{3\beta_Z}[n-1]}
\end{aligned} \tag{75}$$

In order to obtain the above derivations, it is required to derive $\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_S}$, $\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{SN}}$, $\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_S}$ and $\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \boldsymbol{x}_Z[n-1]}$ terms via the regression function of constituent RBF subnetwork. Using the regression function of RBF network in (63–64), the above terms can be computed as

$$\begin{aligned}
\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial w_j} &= \boldsymbol{\Psi}(\boldsymbol{x}_z[n-1],\boldsymbol{\rho}_j[n-1],\sigma_j[n-1])\\
&= exp\left(\frac{-\|\boldsymbol{x}_Z[n-1]-\boldsymbol{\rho}_j[n-1]\|^2}{\sigma_j^2[n-1]}\right)\\
\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{ji}} &= 2\,w_j\,\boldsymbol{\Psi}(\boldsymbol{x}_z[n-1],\boldsymbol{\rho}_j[n-1],\sigma_j[n-1]) \times\left(\frac{x_i[n-1]-\rho_{ji}[n-1]}{\sigma_j^2[n-1]}\right)\\
\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \sigma_j} &= 2\,w_j\,\boldsymbol{\Psi}(\boldsymbol{x}_z[n-1],\boldsymbol{\rho}_j[n-1],\sigma_j[n-1]) \times\left(\frac{\|\boldsymbol{x}_Z[n-1]-\boldsymbol{\rho}_j[n-1]\|^2}{\sigma_j^3[n-1]}\right)\\
\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial x_i[n-1]} &= -\sum_{j=1}^{S}\frac{\partial N_f(\boldsymbol{x}_Z[n-1],\boldsymbol{\Theta})}{\partial \rho_{ji}}\\
&= -2\sum_{j=1}^{S} w_j\,\boldsymbol{\Psi}(\boldsymbol{x}_z[n-1],\boldsymbol{\rho}_j[n-1],\sigma_j[n-1])\left(\frac{x_i[n-1]-\rho_{ji}[n-1]}{\sigma_j^2[n-1]}\right)
\end{aligned} \tag{76}$$

In a nutshell, derivations except for $\frac{\partial \Delta u_R[n]}{\partial w_S}$, $\frac{\partial \Delta u_R[n]}{\partial \rho_{SN}}$ and $\frac{\partial \Delta u_R[n]}{\partial \sigma_S}$ terms can be successfully carried out since the $\frac{\partial u_R[n]}{\partial \beta_Z[n]}$ term depends on the chosen controller structure. In the next subsection, $\frac{\partial u_R[n]}{\partial \beta_Z[n]}$ term is derived for MIMO PID controller structure.

4.3 PID-type Runge–Kutta neural network controller

PID controller has supplanted most of the controller structures in industry owing to its robustness, effectiveness for wide range of operating conditions and its functional simplicity (Akhyar and Omatu 1993; Sung et al. 2009). Implementation simplicity, good control performance and excellent robustness to uncertainties are the main reasons for preference of PID controllers in industry (Cetin and Iplikci 2015; Iplikci 2010a; Sung et al. 2009). The classical incremental MIMO PID controller produces a control signal as follows (Cetin and Iplikci 2015; Sung et al. 2009; Bobal et al. 2005; Wanfeng et al. 2008; Aström and Hagglund 1995; Visioli 2006):

$$\boldsymbol{u}[n] = \boldsymbol{f}_c(\hat{\boldsymbol{\beta}}[n], \boldsymbol{C}_{in}) = \begin{bmatrix} u_1[n] \\ \vdots \\ u_R[n] \end{bmatrix}_{Rx1} = \begin{bmatrix} u_1[n-1] \\ \vdots \\ u_R[n-1] \end{bmatrix}_{Rx1} + [\boldsymbol{K_{PID}}]_{Rx3Q} \begin{bmatrix} e_1[n] - e_1[n-1] \\ e_1[n] \\ e_1[n] - 2e_1[n-1] + e_1[n-1] \\ \vdots \\ e_Q[n] - e_Q[n-1] \\ e_Q[n] \\ e_Q[n] - 2e_Q[n-1] + e_Q[n-2] \end{bmatrix}_{3Qx1} \tag{77}$$

where

$$[\boldsymbol{K_{PID}}]_{Rx3Q} = \begin{bmatrix} K_{P_{11}} & K_{I_{11}} & K_{D_{11}} & K_{P_{12}} & K_{I_{12}} & K_{D_{12}} & \cdots & K_{P_{1Q}} & K_{I_{1Q}} & K_{D_{1Q}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\ K_{P_{R1}} & K_{I_{R1}} & K_{D_{R1}} & K_{P_{R2}} & K_{I_{R2}} & K_{D_{R2}} & \cdots & K_{P_{RQ}} & K_{I_{RQ}} & K_{D_{RQ}} \end{bmatrix}_{Rx3Q} \tag{78}$$

and $K_{P_{rq}}, K_{I_{rq}}, K_{D_{rq}}$, $r \in \{1 \cdots R\}$, $q \in \{1 \cdots Q\}$ denote the parameters of proportional, integral and derivative parts of the controller to be tuned, respectively. In an adaptive control scheme, the initially assigned values of the controller parameters will generally not be optimal (Iplikci 2010a); hence, it is required to adjust the parameters via optimization methods (Iplikci 2010a; Luenberger and Ye 2008). The controller parameters $K_{P_{rq}}, K_{I_{rq}}$, and $K_{D_{rq}}$, $r \in \{1 \cdots R\}$, $q \in \{1 \cdots Q\}$ are computed via RK-NN$_{\text{estimator}}$. For this purpose, an online RK-NN has been deployed for each controller parameter since RBF network in Fig. 6 has multi-input single-output structure, so parameter estimator is composed of $Rx3Q$ separate RK-NN identifiers. The zth parameter of the PID controller parameters can be estimated via RK-NN$_{\text{estimator}}$ as:

$$\beta_Z[n] = \beta_Z[n-1] + \frac{1}{6}h\Big[K_{1\beta_Z}[n-1] + 2K_{2\beta_Z}[n-1] + 2K_{3\beta_Z}[n-1] + K_{4\beta_Z}[n-1]\Big] \tag{79}$$

where

$$K_{m\beta_Z}[n-1] = \sum_{i=1}^{S} w_i \boldsymbol{\Psi}(\boldsymbol{x}_{Zm}[n-1], \boldsymbol{\rho}_i[n-1], \sigma_i[n-1]), \quad m \in \{1,2,3,4\}$$
$$\boldsymbol{\Psi}(\boldsymbol{x}_{Zm}[n-1], \boldsymbol{\rho}_i[n-1], \sigma_i[n-1]) = exp\left(\frac{-\|\boldsymbol{x}_{Zm}[n-1] - \boldsymbol{\rho}_i[n-1]\|^2}{\sigma_i^2[n-1]}\right) \tag{80}$$

Since the parameters of the controller are estimated via RK-NN, it is named as ***PID-type Runge–Kutta neural network controller***. This structure inherits both the robustness of PID controllers and the nonlinear generalization performance of RK-NN method. In MIMO PID controller, controller parameters and controller inputs are multiplied. Therefore, the $\frac{\partial u_R[n]}{\partial \beta_Z[n]}$ term is equal to the corresponding controller input affecting the Rth control signal.

4.4 Pseudocode for proposed adjustment mechanism

The pseudocode of the proposed adjustment mechanism algorithm is given step by step as follows. In the control procedure given below, $\boldsymbol{u}^-[n]$ stands for the control signal predicted with controller parameters obtained at the previous step and $\boldsymbol{u}^+[n]$ denotes the control signal estimated with trained controller parameters at the current step.

```
Step 1: Initialization.
-Initialize RK-NN$_{\text{estimator}}$, RK$_{\text{EKF}}$ and RK$_{\text{model}}$ parameters.
Step 2: Prediction step for parameter estimator ( $\boldsymbol{\Theta}^-$)
-Set time step $n$.
- Calculate the approximated controller parameters $\hat{\boldsymbol{\beta}}$ by RK-NN$_{\text{estimator}}$ ($\boldsymbol{\Theta}^-$) trained at previous step ($n-1$) via (66).
Step 3: Computation of control signal ($\boldsymbol{u}[n]^-$)
-Calculate the control signal $\boldsymbol{u}[n]^-$ via (65).
Step 4: Runge–Kutta model-based EKF (Prediction Phase)
-Apply candidate control signal ($\boldsymbol{u}^-[n]$) once to Runge–Kutta model-based EKF to obtain current states via (27–36) $\boldsymbol{x}\tilde{[n]} = [\tilde{x}_1[n] \cdots \tilde{x}_N[n]]$
Step 5: Runge–Kutta model-based Parameter Estimation (Prediction Phase)
-Deploy model parameters obtained at current step ($n$) $\boldsymbol{\theta}_n$.
Step 6: Runge–Kutta Model of the system (Prediction Phase)
-Apply candidate control signal ($\boldsymbol{u}^-[n]$) K-times to Runge–Kutta model using estimated states ($\tilde{\boldsymbol{x}}[n]$) and estimated model parameters ($\boldsymbol{\theta}_n$) so as to;
6.1 Obtain K-step ahead future behaviour of the system
```

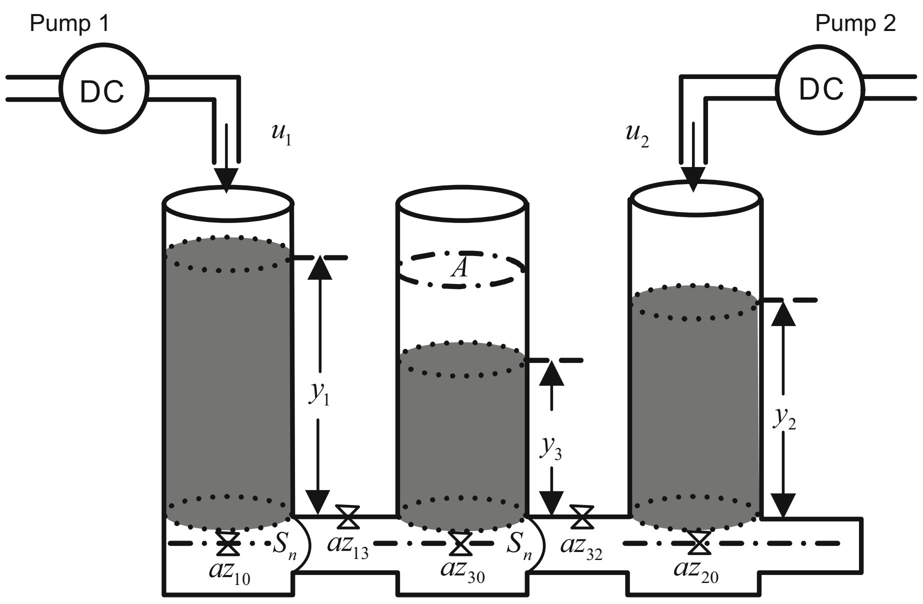


Fig. 7 Three-tank system

dynamics and prediction errors vector via (45–49, 70)
6.2 Compute objective function given by (2)
If $F\left(\boldsymbol{u}[n], \hat{\boldsymbol{e}}_{tq}\right) > \varepsilon_{closed-loop}$
Jump to next substep 6.3.
else
Continue with RK-NN$_{\text{estimator}}$ parameter trained at previous step and jump to step 10
end
6.3 Obtain K-step ahead Jacobian of the system via (50–55) which is required to construct Jacobian matrix for RK-NN$_{\text{estimator}}$
6.4 Construct $\boldsymbol{J}_m$ via (11) and compute the correction term $\delta\boldsymbol{u}[n]$ for control action via (13)
Step 7: Construction of Jacobian matrix for zth RK-NN$_{\text{estimator}}$
-Constitute the Jacobian matrix in (69) for zth RK-NN$_{\text{estimator}}$ via (71–76) and K- step ahead Jacobian of the system via (50–55) which is attained at step 6.
-Perform the above operation for all RK-NN$_{\text{estimator}}$'s.
Step 8: Training phase for RK-NN$_{\text{estimator}}$'s ($\boldsymbol{\Theta}$) (Levenberg–Marquardt Algorithm)
-Adjust the network parameters of the all RK-NN$_{\text{estimator}}$s via (68)
Step 9: Prediction step for parameter estimator ($\boldsymbol{\Theta}$)
- Calculate the approximated controller parameters $\hat{\boldsymbol{\beta}}$ by RK-NN$_{\text{estimator}}$ trained at step 8 ($\boldsymbol{\Theta}$) via (66).
Step 10: Computation of control signal ($\boldsymbol{u}^{+}[n]$) using trained controller parameter
-Calculate the control signal $\boldsymbol{u}[n]$ via (65)
Step 11: Control of Real System
- Apply the control action ($\boldsymbol{u}^{+}[n] + \delta\boldsymbol{u}[n]$) to the real system and obtain real system states and system outputs
Step 12: Runge–Kutta model-based Parameter Estimation (Training Phase)
-Obtain $\boldsymbol{J}_{\boldsymbol{\theta}}$ via (38, 40–44) and $\boldsymbol{e}$ via (39)
-Update system model parameters via Runge–Kutta model-based Parameter Estimation block as $\boldsymbol{\theta}[n+1] = \boldsymbol{\theta}[n] - \frac{\boldsymbol{J}_{\boldsymbol{\theta}}^T \boldsymbol{e}}{\boldsymbol{J}_{\boldsymbol{\theta}}^T \boldsymbol{J}_{\boldsymbol{\theta}}}$ given in (37).
Step 13: Incrementation of time step
-Increment $n = n + 1$ and back to step 2.

5 Simulation results

The performance of the proposed Runge–Kutta neural network-based adjustment mechanism with adaptive PID is examined on nonlinear three-tank system (TTS) and Van de Vusse System. Nonetheless, a diverse range of nonlinear systems can be controlled and fundamental control problems that frequently appear in practice such as nonlinearity and instability can be solved via the proposed adjustment mechanism.

5.1 Three-tank system

The three-tank system (TTS) is frequently employed in technical literature as a MIMO nonlinear system so as to peruse and compare the performances of proposed control methodologies for MIMO systems (Iplikci 2013; Cetin and Iplikci 2015; Iplikci 2010b). A three-tank system is a tank system in which three ideal cylindrical tanks are serially interconnected to each other. The schematic diagram of the TTS is illustrated in Fig. 7. The system is composed of three interconnected cylindrical tanks, two pumps, valves between tanks and water reservoir in the bottom. The water at the bottom of the reservoir is pumped to tank 1 and tank 2 via pump 1 and pump

Table 1 System parameters for three-tank system

Parameter description	Value
az_{13}: outflow coefficient between tank 1 and tank 3	0.52
az_{32}: outflow coefficient between tank 3 and tank 2	0.55
az_{10}: outflow coefficient from tank 1 to reservoir	0.26
az_{20}: outflow coefficient from tank 2 to reservoir	0.28
az_{30}: outflow coefficient from tank 3 to reservoir	0.45
A: cross section of the cylinders	0.15 (m^2)
S_n: section of connection pipe n	5×10^{-5} (m^2)
g: gravitation coefficient	9.81 (m/s^2)

2, respectively. The differential equations describing the dynamic behaviour of the system are expressed as follows:

$$\begin{aligned}\dot{y}_1 &= \frac{1}{A}\Big[u_1(t) - Q_{13}(t) - Q_{10}(t)\Big]\\ \dot{y}_2 &= \frac{1}{A}\Big[u_2(t) + Q_{32}(t) - Q_{20}(t)\Big]\\ \dot{y}_3 &= \frac{1}{A}\Big[Q_{13}(t) - Q_{32}(t) - Q_{30}(t)\Big]\end{aligned} \tag{81}$$

where

$$\begin{aligned}Q_{13}(t) &= az_{13}\ S_n\ sgn\big(y_1(t) - y_3(t)\big)\sqrt{2g|y_1(t) - y_3(t)|}\\ Q_{32}(t) &= az_{32}\ S_n\ sgn\big(y_3(t) - y_2(t)\big)\sqrt{2g|y_3(t) - y_2(t)|}\\ Q_{10}(t) &= az_{10}\ S_n\ \sqrt{2g|y_1(t)|}\\ Q_{20}(t) &= az_{20}\ S_n\ \sqrt{2g|y_2(t)|}\\ Q_{30}(t) &= az_{30}\ S_n\ \sqrt{2g|y_3(t)|}\end{aligned} \tag{82}$$

and $u_i(t)$ is the supply flow rate of the ith pump as the ith input, $y_i(t)$ is the liquid level of the ith tank as the ith output and $Q_{ji}(t)$ is the flow rate between tank j and i (Iplikci 2013; Cetin and Iplikci 2015; AMIRA 2009). The descriptions of the symbols and their numerical values in Fig. 7 and (81, 82) are tabulated in Table 1. In the closed-loop TTS control problem, the aim is to independently keep the liquid levels of tank 1 and tank 2 at the desired levels by adjusting the flow rates of pump 1 ($u_1(t)$) and pump 2 ($u_2(t)$) within allowed intervals (Iplikci 2013; Cetin and Iplikci 2015; AMIRA 2009; Theilliol et al. 2002). Therefore, $y_1(t)$ and $y_2(t)$ are the controlled outputs of the system, while $u_1(t)$ and $u_2(t)$ are control inputs. The third output of the system ($y_3(t)$), in the middle tank, is uncontrollable (Theilliol et al. 2002). In the simulations, sampling time is chosen as $T_s = 1sec$ and the limitations for the magnitude of the control signal are $u_{1_{min}} = u_{2_{min}} = 0m^3/s$ and $u_{1_{max}} = u_{2_{max}} = 10^{-4}m^3/s$ (Iplikci 2013). The continuation period of control signals is kept constant at $\tau_{1_{min}} = \tau_{2_{min}} = \tau_{1_{max}} = \tau_{2_{max}} = T_s = 1.0$ sec. Since states of the system may not be available for measurement, Runge–Kutta-based EKF is utilized to approximate the dynamic behaviour of the states (Iplikci 2013). Simulations have been performed for three separate cases:

1) Nominal case with no measurement noise and parametric uncertainty
2) Measurement noise is added to the controlled outputs of the system
3) Parametric uncertainty is imposed on a system parameter.

For all cases, the number of the neurons in RBF networks (S) is chosen as 2 and input vectors are settled as $X_z = \big[\beta_z[n-1]\ \beta_z[n-2]\big]^T$ for all controller parameters. The prediction horizon is assigned as $K = 5$.

5.1.1 Nominal case with no noise and parametric uncertainty

The tracking performance of the controller for the case when no noise and parametric uncertainty is applied to the system and all system parameters are fully known is illustrated in Fig. 8a, d for staircase reference signals. The control signals produced by MIMO PID controller and the correction terms are also given in Fig. 8b, c, e, f. It is seen that the system outputs can be successfully derived to the desired reference inputs. As can be explicitly seen from Fig. 8, the control task can be carried off by only small transient and steady-state error. In addition, while the second reference ($r_2(t)$) signal is fixed at 0.15 m, the first reference signal is changed between 0-800 sec in a stepwise fashion. The fluctuation in tank 1 interacts with tank 2 through tank 3. If the behaviour of the system outputs and control signals between 0-800 sec are attentively examined, it can be explicitly seen that the controller effectively overcomes and rejects the coupling between the tank 1 and 2. The MIMO PID controller parameters are depicted in Figs. 9 and 10. In order to exemplify the inner learning mechanism of the RK-NN$_{\text{estimator}}$, convergence of the weights of $K_{d_{11}}$ controller parameter to

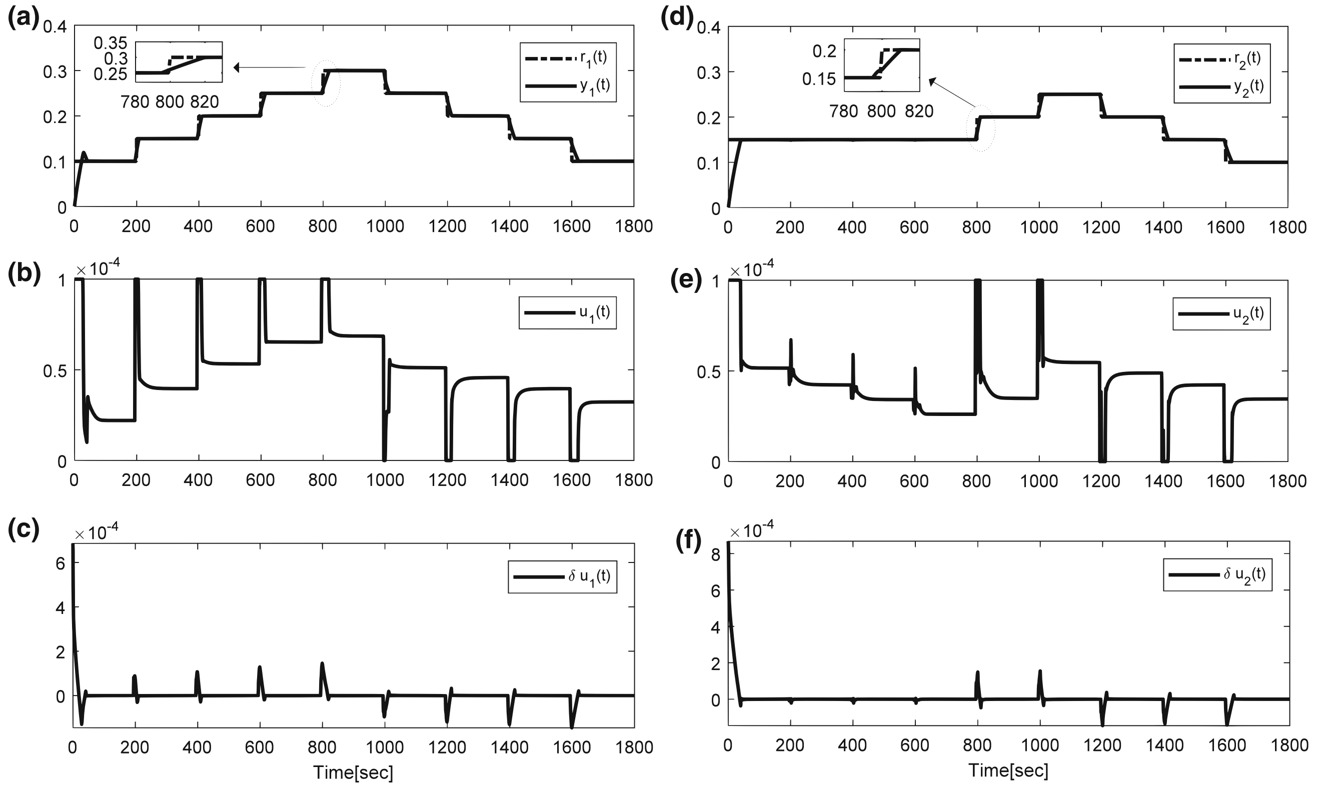


Fig. 8 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms(**c**, **f**) of three-tank system for the nominal case with no measurement noise and parametric uncertainty (staircase reference inputs)

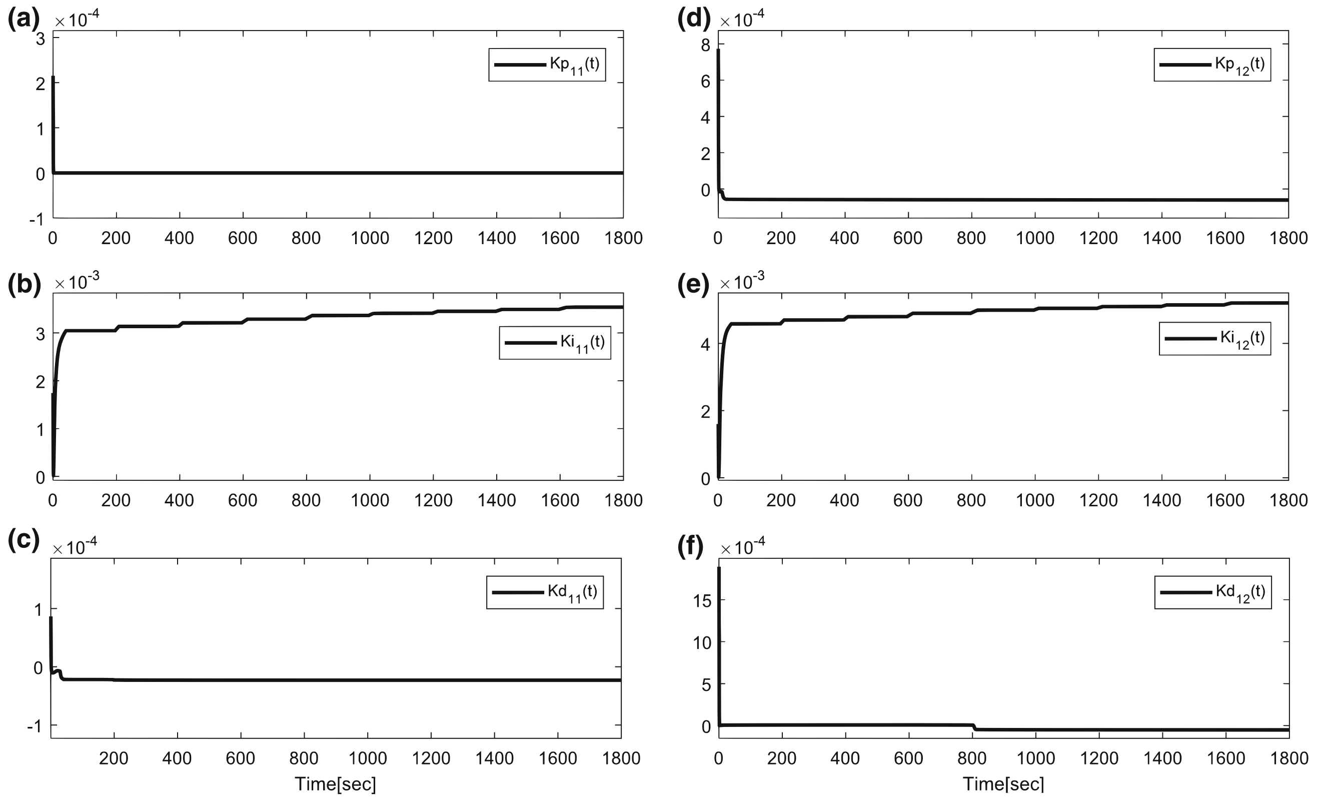


Fig. 9 MIMO PID controller parameters for three-tank system

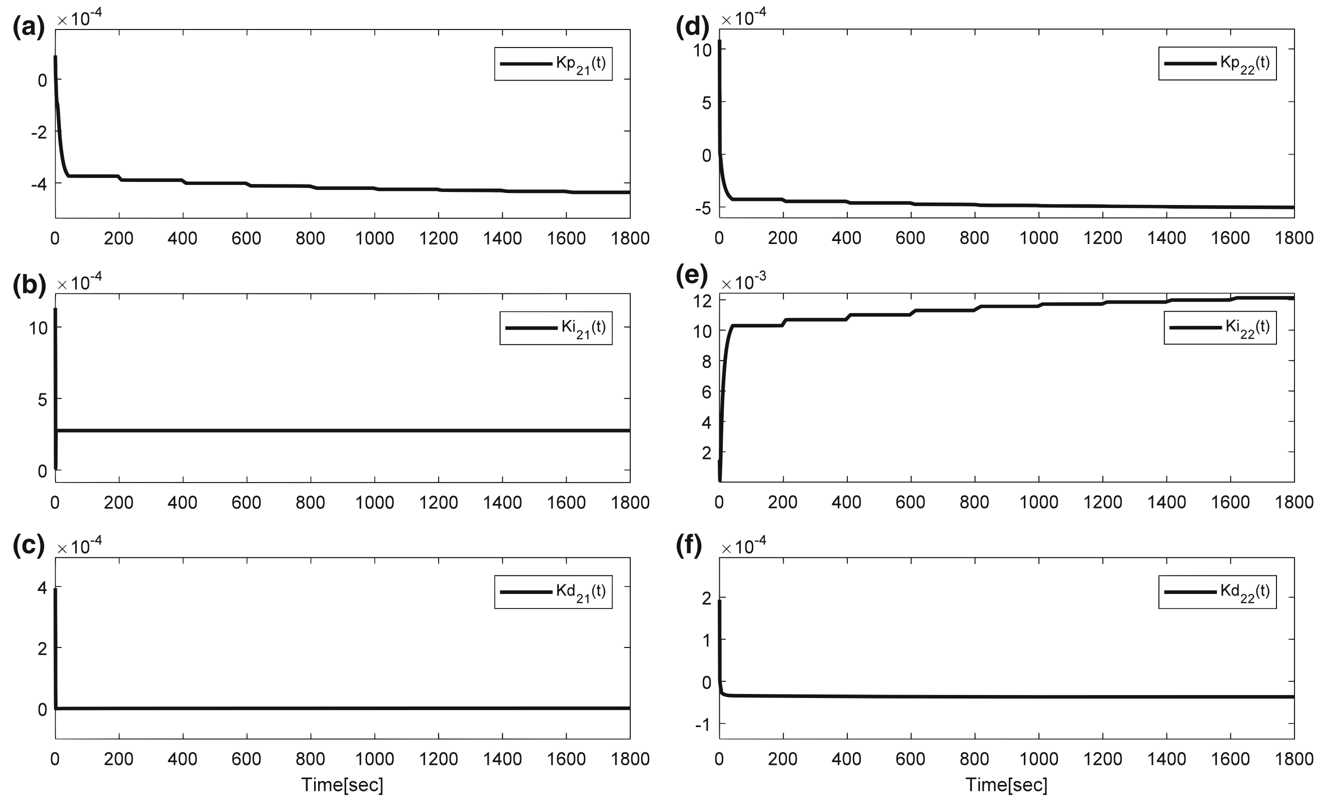


Fig. 10 MIMO PID controller parameters for three-tank system

their optimal values depending on alternation on reference signals is illustrated in Fig. 11. In Fig. 11, it is palpably seen that the RK-NN$_{\text{estimator}}$ can adapt itself so as to learn the required controller dynamics forcing the outputs of the system to reference signals. The performance of the controller for sinusoidal reference signals is depicted in Fig. 12. As can be seen from Fig. 12, the controller tracks the sinusoidal reference inputs accurately.

5.1.2 Measurement noise

In order to evaluate the performance and robustness of the controller under the influence of measurement noise, an additive zero mean Gaussian noises with $\sigma_{y_1(t)} = \sigma_{y_2(t)} = 0.003$ standard deviations are added to the measured controlled outputs of the system$(y_1(t), y_2(t))$. The tracking performance and control inputs produced by the controller are depicted in Fig. 13. As can be seen from Fig. 13, although there exists measurement noise, the control task can be carried off by only small transient and steady-state errors, and it is observed that the controller accurately tracks the desired reference signals.

5.1.3 Uncertainty in system parameters

In RK model-based control, it is difficult to obtain RK model parameters accurately, which directly affects the control performance of the model-based controller. Therefore, it is crucial that the adjustment mechanism subsumes the model parameter estimator. As can be seen from Fig. 2, the controller structure has Runge–Kutta-based model parameter estimation block. Finally, in this subsection, the model parameter estimation performance and also tracking performance of the proposed adjustment mechanism are examined. For this purpose, the desired reference signals are assigned as 0.25 and 0.2 m for tank 1 and tank 2, respectively, and the outflow parameter az_{13} is selected as the uncertain system parameter, which varies as $az_{13} = 0.52 + 0.28sin(0.0133\pi t)$. The tracking performance of the controller structure for the uncertainty in system parameter case is illustrated in Fig. 14. As can be seen from Fig. 14e, the proposed Runge–Kutta-based model parameter estimation block accurately approximates the correct values of the uncertain parameter in a timely manner and then maintains it in the long run (Iplikci 2013).

5.2 Van de Vusse chemical reaction

The second nonlinear benchmark system to query the effectiveness of the proposed adaptive controller is Van de Vusse chemical reaction. In nonlinear control theory, Van de Vusse chemical reaction has ofttimes been deployed as a nonlinear benchmark problem to assess the performances of developed adaptive control methodologies since the uncon-

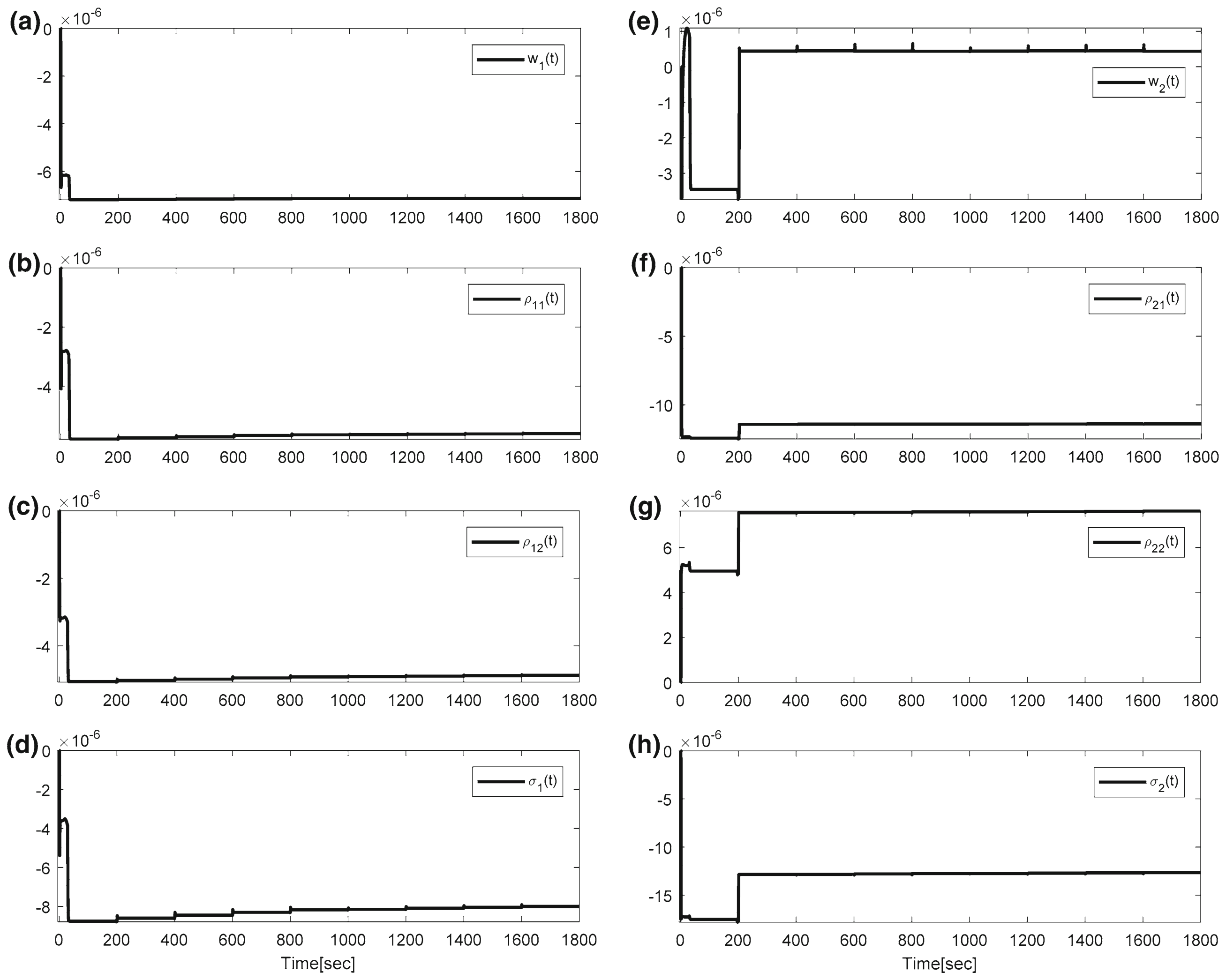


Fig. 11 Adaptation of RK-NN$_{\text{estimator}}$ parameters for $K_{d_{11}}$ controller parameter (three-tank system)

trolled equations of this system are highly nonlinear, the system is a non-isothermal process affected by thermal effect, and the resulting system shows strictly non-minimum-phase behaviour (Iplikci 2013). It is required to control the system actively in order to hinder divergent behaviour since the system involves severe nonlinearity with strong coupling between its dynamics. The chemical reaction mechanism attributed to Van de Vusse is described by the following reaction scheme.

$$\begin{aligned} &\boldsymbol{A} \xrightarrow{k_1} \boldsymbol{B} \xrightarrow{k_2} \boldsymbol{C} \\ &2A \xrightarrow{k_3} \boldsymbol{D} \end{aligned} \tag{83}$$

where $\boldsymbol{A}$ is the inlet reactant, $\boldsymbol{B}$ is the desired product, $\boldsymbol{C}$ and $\boldsymbol{D}$ are unwanted by products and k_i's denote the reaction rates (Chen et al. 1995; Engell and Klatt 1993; Vojtesek and Dostal 2010; Nørregard 2007; Kulikov and Kulikova 2014). The reaction considered in (83) is the production of cyclopentenol ($\boldsymbol{B}$) from cyclopentadiene ($\boldsymbol{A}$) by acid-catalyzed electrophilic addition of water in dilute solution (Engell and Klatt 1993). As a result of the strong reactivity of the cyclopentadiene ($\boldsymbol{A}$) and the cyclopentenol ($\boldsymbol{B}$), dicyclopentadiene ($\boldsymbol{D}$) is produced by Diels–Alder reaction as a side product, and cyclopentanediol ($\boldsymbol{C}$) as a consecutive product by addition of another water molecule (Engell and Klatt 1993). The complete reaction scheme is

$$\begin{aligned} &\underset{Cyclopentadiene(\boldsymbol{A})}{\boldsymbol{C_5H_6}} \xrightarrow{+H_2O(k_1)} \underset{Cyclopentenol(\boldsymbol{B})}{\boldsymbol{C_5H_7OH}} \xrightarrow{+H_2O(k_2)} \underset{Cyclopentanediol(\boldsymbol{C})}{\boldsymbol{C_5H_8(OH)_2}} \\ &\underset{Cyclopentadiene(\boldsymbol{A})}{2\boldsymbol{C_5H_6}} \xrightarrow{k_3} \underset{Dicyclopentadiene(\boldsymbol{D})}{\boldsymbol{C_{10}}H_{12}} \end{aligned} \tag{84}$$

The differential equations describing the mole balances for species $\boldsymbol{A}$ and $\boldsymbol{B}$, and the energy balance for the reactor given to realize the chemical reaction scheme in (83–84) are given as follows:

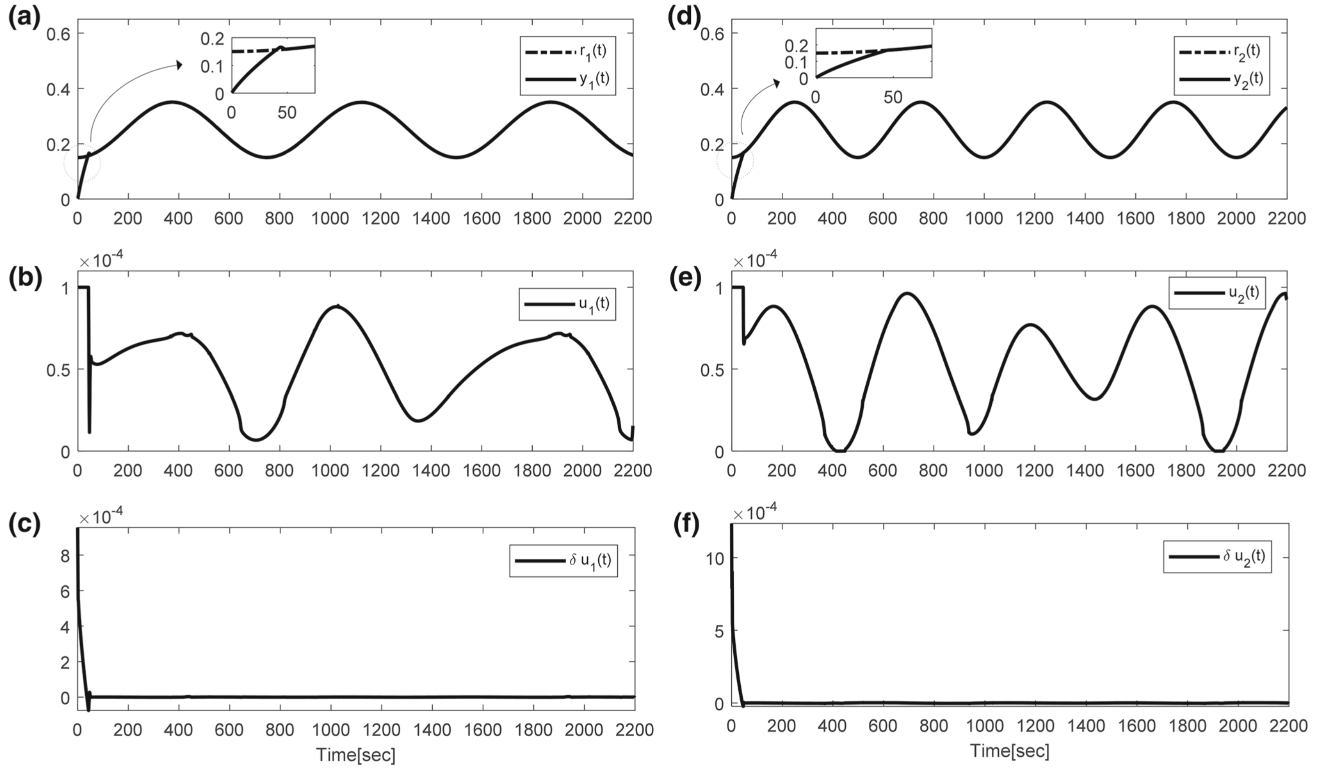


Fig. 12 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of three-tank system for the nominal case with no measurement noise and parametric uncertainty (sinusoidal reference inputs)

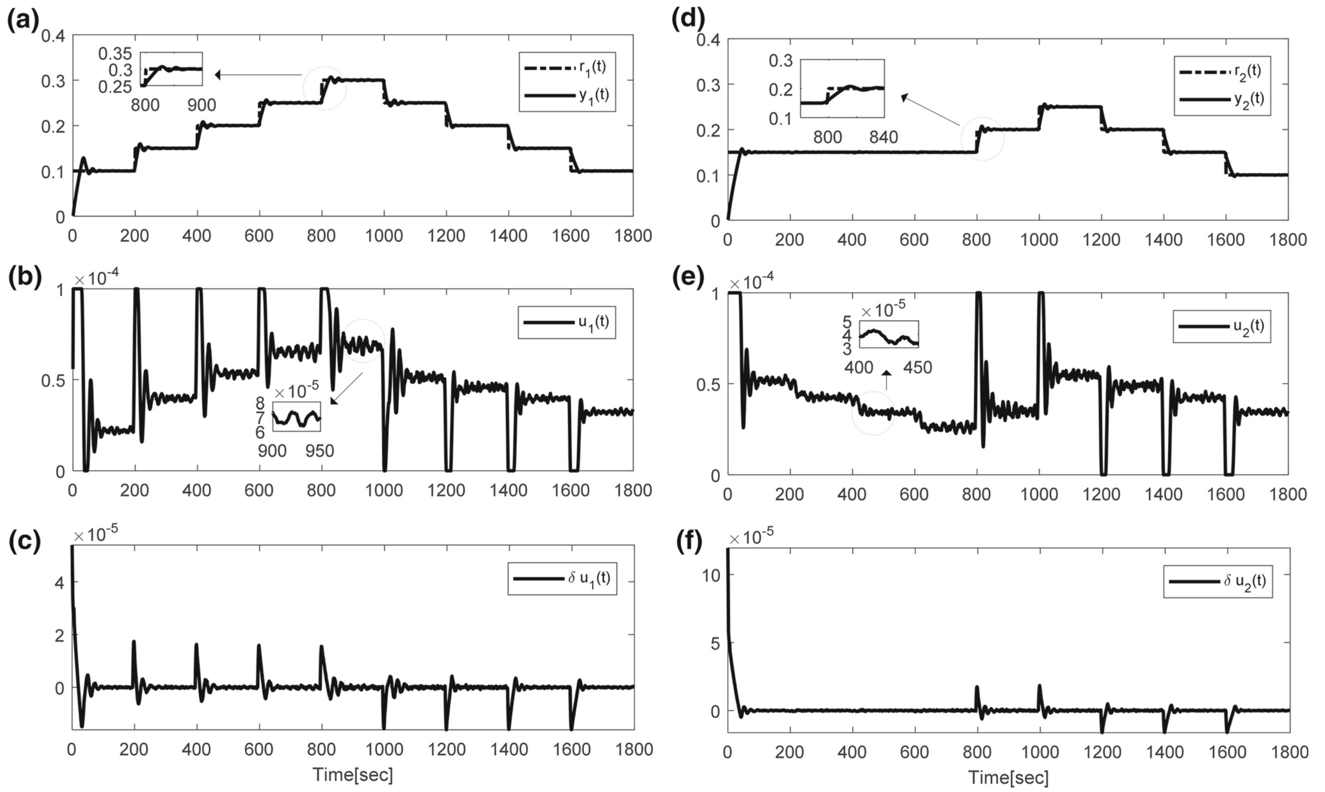


Fig. 13 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of three-tank system for case with measurement noise (staircase reference inputs)

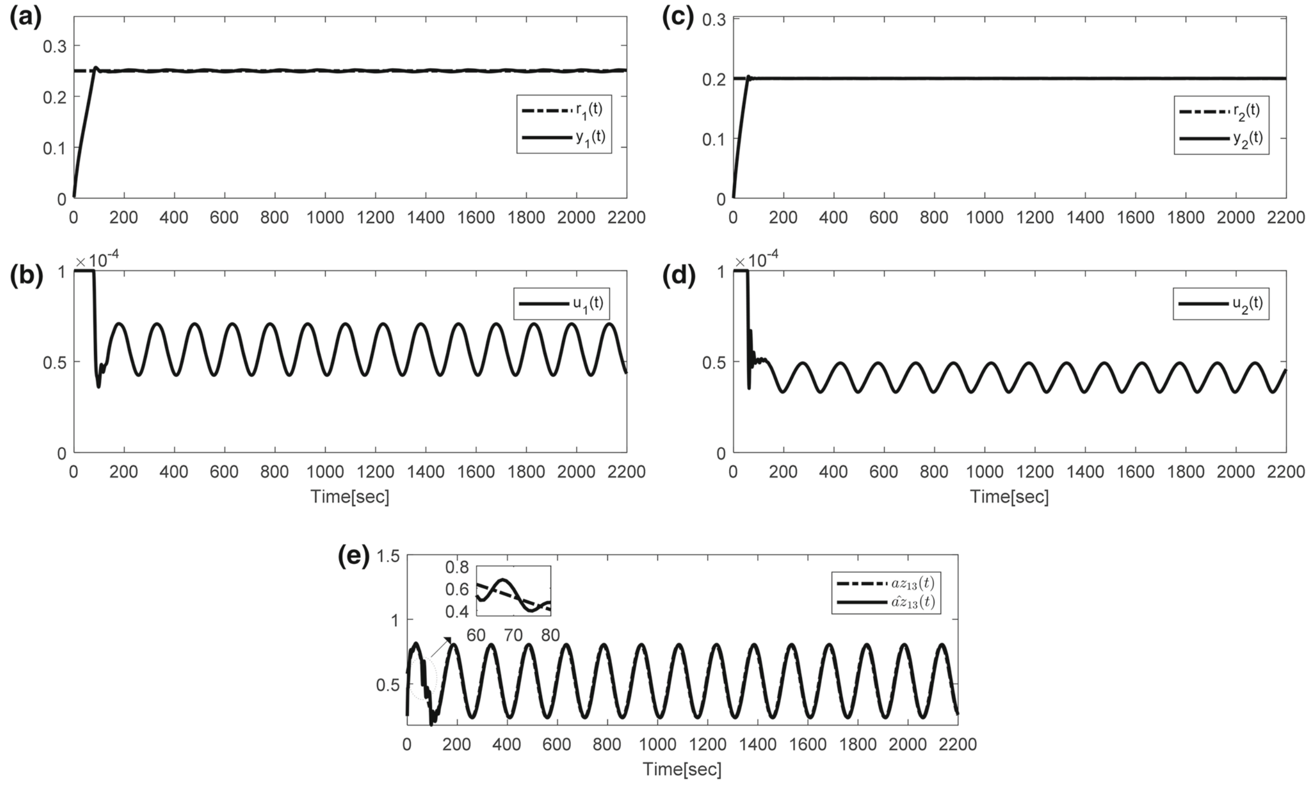


Fig. 14 System outputs (**a**, **c**), control signals (**b**, **d**) and uncertain outflow parameter ($az_{13}(t)$) (**e**) and its estimation for the case with parametric uncertainty (Three-tank system)

$$\dot{C}_A(t) = \frac{F}{V}\left(C_{A0} - C_A(t)\right) - k_{10}e^{\frac{-E_1}{T}}C_A(t) - k_{30}e^{\frac{-E_3}{T}}C_A^2(t)$$
$$\dot{C}_B(t) = -\frac{F}{V}C_B(t) + k_{10}e^{\frac{-E_1}{T}}C_A(t) - k_{20}e^{\frac{-E_2}{T}}C_B(t)$$
$$\dot{T}(t) = \frac{1}{\rho C_p}\Big[k_{10}e^{\frac{-E_1}{T}}C_A(t)(-\Delta H_1) + k_{20}e^{\frac{-E_2}{T}}C_B(t)(-\Delta H_2)$$
$$+k_{30}e^{\frac{-E_3}{T}}C_A^2(t)(-\Delta H_3)\Big] + \frac{F}{V}(T_0 - T(t)) + \frac{Q}{\rho C_p} \quad (85)$$

where C_A and C_B are the molar concentrations of $\boldsymbol{A}$ and $\boldsymbol{B}$, T is the reactor temperature, F/V is the dilution rate and Q is the rate of the heat added or removed per unit volume, C_p and ρ are the heat capacity and density of the reacting mixture, respectively, ΔH_i are the heats of the reaction and E are activation energies (Iplikci 2013; Vojtesek and Dostal 2010; Niemiec and Kravaris 2003; Kravaris et al. 1998). The descriptions and values of the physical and chemical parameters are given in Table 2 (Iplikci 2013; Chen et al. 1995; Niemiec and Kravaris 2003; Kravaris et al. 1998). In the closed-loop system, it is aimed to independently control the molar concentration of $\boldsymbol{B}$ ($y_1 = C_B$) and the temperature of the reactor ($y_2 = T$) by adjusting the dilution rate ($u_1 = F/V$) and the rate of heat added or removed per unit volume ($u_2 = Q$) within allowed intervals (Iplikci 2013; Niemiec and Kravaris 2003). In the simulations, sampling time is assigned as $T_s = 0.01h$ and the allowable limits for the magnitude of the control signals are $u_1 = [0, 500]$ h^{-1} and $u_2 = [-1000, 0]$ kJ/l h (Iplikci 2013). The continuation period of the control signals is kept constant at $\tau 1_{min} = \tau 2_{min} = \tau 1_{max} = \tau 2_{max} = T_s = 0.01$ h. The performance of the closed-loop system has been examined for three separate cases:

1) Nominal case with no measurement noise and parametric uncertainty
2) Measurement noise is added to the controlled output of the systems
3) Parametric uncertainty is imposed on a system parameter.

For all cases, the number of the neurons in RBF networks (S) is chosen as 2 and input vectors are settled as $X_z = \left[\beta_z[n-1]\ \beta_z[n-2]\right]^T$ for all controller parameters. The prediction horizon is assigned as $K = 5$.

5.2.1 Nominal case with no noise and parametric uncertainty

The control performance for the case when no noise and parametric uncertainty is applied to the system and all system parameters are fully known is depictured in Fig. 15a, d for staircase reference signals. The system tracks the reference

Table 2 Physicochemical parameters for Van de Vusse (Iplikci 2013; Chen et al. 1995; Kravaris et al. 1998)

Description of parameter	Symbol	Value of parameter
Collision factor for reaction k_1	k_{10}	1.287×10^{12} (h^{-1})
Collision factor for reaction k_2	k_{20}	1.287×10^{12} (h^{-1})
Collision factor for reaction k_3	k_{30}	9.043×10^{9} (h^{-1}l/mol)
Activation energy for reaction k_1	E_1	9758.3 (K)
Activation energy for reaction k_2	E_2	9758.3 (K)
Activation energy for reaction k_3	E_3	8560.0 (K)
Enthalpies of reaction k_1	ΔH_1	4.2 (kJ/mol)
Enthalpies of reaction k_2	ΔH_2	−11 (kJ/mol)
Enthalpies of reaction k_3	ΔH_3	−41.85 (kJ/mol)
The concentration of A in the feed stream	C_{A0}	5.0 (mol/l)
Feed temperature	T_0	403.15 (K)
Density	ρ	0.9342 (kg/l)
Heat capacity	C_p	3.01 (kJ/kg K)
Reactor volume	V	10 (l)

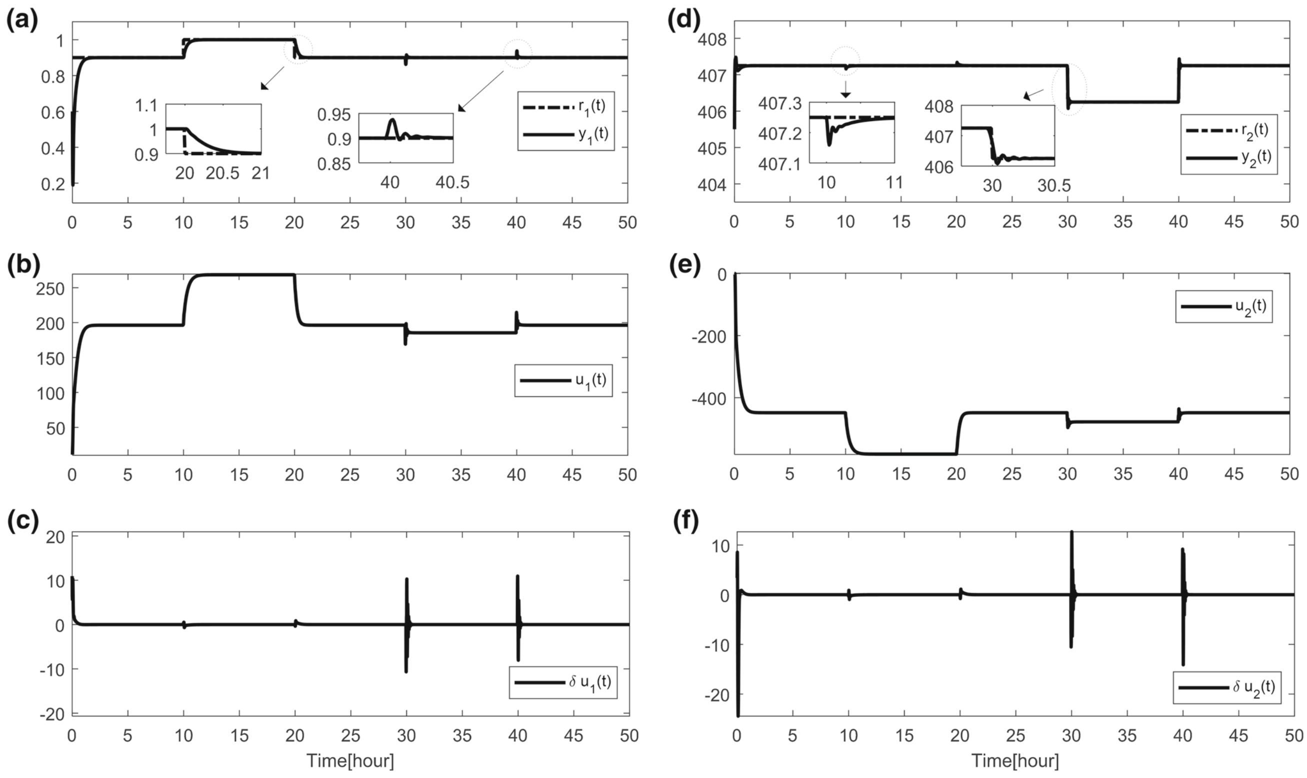


Fig. 15 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of Van de Vusse system for the nominal case with no measurement noise and parametric uncertainty (staircase reference inputs)

signals with very small steady-state errors except for transient states occurring as a result of the abrupt changes on reference signals. The control signals and correction terms are shown in Fig. 15b, c, e, f as well. As can be seen from control signals produced especially between [10, 20] and [30, 40] h, the controller successfully tolerates the strong coupling between C_B and reactor temperature (T). The alternation of the MIMO PID controller parameters are illustrated in Figs. 16 and 17. The inner learning mechanism of the RK-NN$_{\text{estimator}}$ is exemplified in Fig. 18 for the weights of controller parameter $K_{d_{11}}$. In order to appraise the performance of the controller for sinusoidal reference signals, while the reactor temperature is chosen as constant during control, C_B is forced to change sinusoidally as given in Fig. 19. As can be seen from Fig. 19, the controller tracks the sinusoidal reference inputs accurately.

5.2.2 Meaurement noise

It is required to evaluate the performance of the controller with respect to measurement noise since systems can generally be exposed to noise resulting from measurement mechanisms. For this purpose, zero mean Gaussian noises with $\sigma_{C_B(t)} = \sigma_{T(t)} = 0.0003$ standard deviations for C_B and for T are added to the measured outputs of the system$(y_1(t), y_2(t))$. The tracking performance of the controller for the noisy case is given in Fig. 20a, d. The control signals and correction terms are illustrated in Fig. 20b, c, e, f. It is observed that the controller accurately tracks the desired reference signals as given in Fig. 20 for the case where measurement noise is added.

5.2.3 Uncertainty in system parameters

In order to examine the adaptation ability of the proposed mechanism for parametric uncertainty case, $C_{A0}(t)$ parameter, which varies as $C_{A0}(t) = 5 + 0.5sin(0.2\pi t)$, is chosen as an uncertain time-varying parameter while the desired reference signals are assigned as 0.95 and 407.25 for $C_B(t)$ and T, respectively. The model parameter estimation performance and also tracking performance of the proposed adjustment mechanism are depicted in Fig. 21. As can be seen from Fig. 21e, the proposed Runge–Kutta-based model parameter estimation block accurately approximates the correct values of the uncertain parameter in a timely manner and then maintains it in the long run (Iplikci 2013).

5.3 Computation times

The applicability potential of the proposed mechanism in real-time applications is as crucial as its closed-loop tracking performance. Therefore, in order to evaluate the applicability of the proposed RK-NN$_{\text{estimator}}$ in real-time systems, computation times of each operation in the control algorithm have been registered for each case during every sampling period, and the maximum response times of the each operation have been tabulated in Table 3. Since the sampling times for three-tank system and Van de Vusse systems are 1 sec and 0.01 hour, respectively, and the maximum response times of the proposed controller for both systems are less than 35 ms, it can be rendered that RK-NN$_{\text{estimator}}$ can be conveniently deployed in real-time applications. Moreover, computation time of RK-NN$_{\text{estimator}}$ can be minimized by optimizing the control algorithm and implementing it utilizing effective hardwares such as FPGA. In simulations, a PC with 2.2 GHz core i7 CPU and 8 GB RAM has been utilized to implement the control algorithm and codes are not optimized.

5.4 Comparison with Runge–Kutta model-based PID

The performance of the proposed controller has been compared with Runge–Kutta model-based PID proposed by Cetin and Iplikci (2015). The main difference between controller structures is that the dynamics of the PID parameters can be identified mathematically in the proposed RK-NN structure whereas the PID parameters in Cetin and Iplikci (2015) are updated incrementally and can not be mathematically obtained. The parameters of the controller in Cetin and Iplikci (2015) are adjusted using Levenberg–Marquardt optimization algorithm. K-step ahead future system Jacobian information which is required to construct Jacobian Matrix deployed in Levenberg–Marquardt algorithm is attained using Runge–Kutta system model. The tracking performance of the Runge–Kutta model-based PID controller for three-tank system in all cases is illustrated in Figs. 22, 23 and 24. As can be seen from Fig. 23b, e, if the control signals in Figs. 12 and 23 are compared, the control signals produced by the proposed RK-NN-based PID are more realizable since RK model-based PID has more chattering. The tracking performance of the controller structure for the uncertainty in system parameter case is shown in Fig. 24. The tracking performance of the Runge–Kutta model-based PID controller for Van de Vusse system in all cases is depictured in Figs. 25, 26 and 27. The model parameter estimation performance and tracking performance of the RK model-based PID mechanism are illustrated in Fig. 27.

The performances of the controllers are compared using the following performance index in Tables 4 and 5:

$$P_q = \sum_{n=1}^{t_f} \left[r_q[n] - y_q[n] \right]^2, \quad q \in \{1, 2\} \tag{86}$$

where t_f is the final time index. As a consequence, it is clearly seen that the tracking performance of the proposed adjustment mechanism (RK-NN-based PID) is generally better than RK model-based PID proposed in Cetin and Iplikci (2015).

6 Conclusion

In this paper, a novel adaptive control architecture is introduced where Runge–Kutta integration method is deployed both in the controller parameter estimator and the system identification blocks. The main novelty of this paper is that the parameters of the any nonlinear controller can be iden-

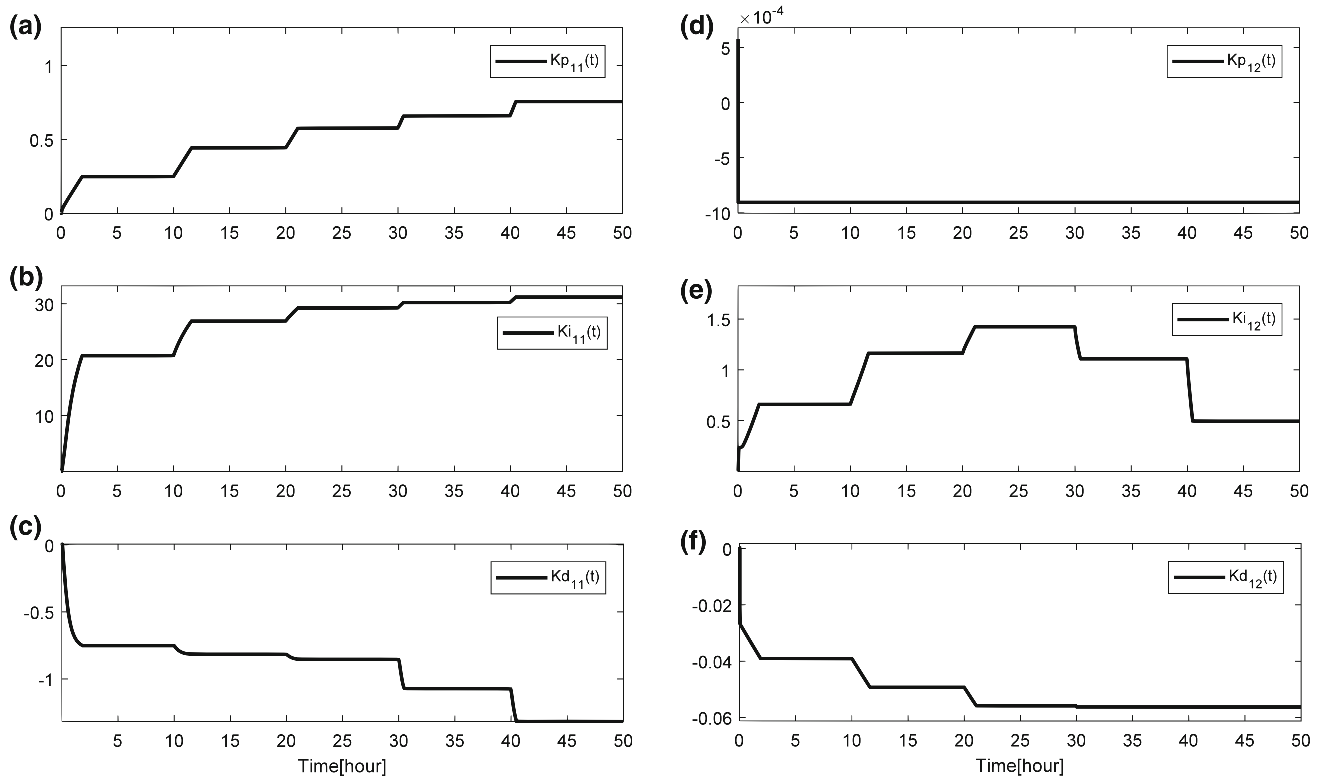


Fig. 16 MIMO PID controller parameters for Van de Vusse system

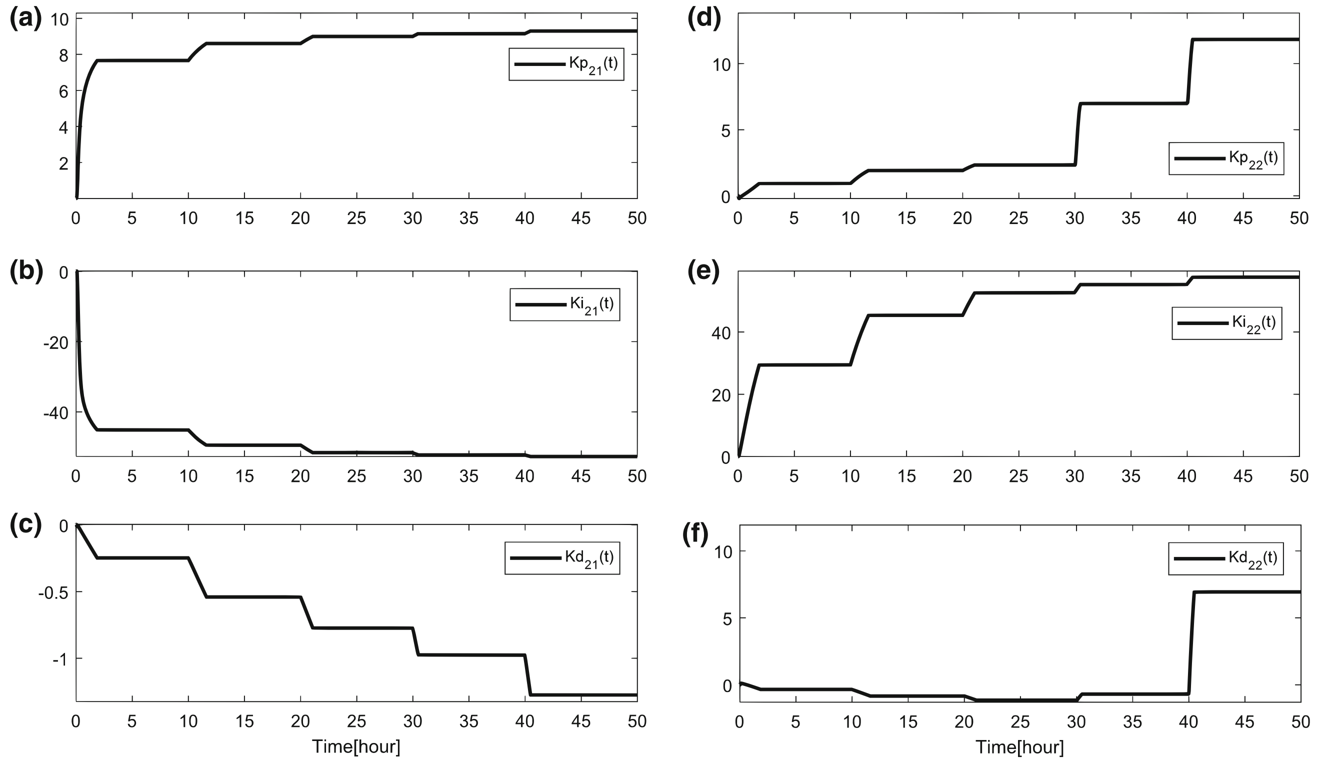


Fig. 17 MIMO PID controller parameters for Van de Vusse system

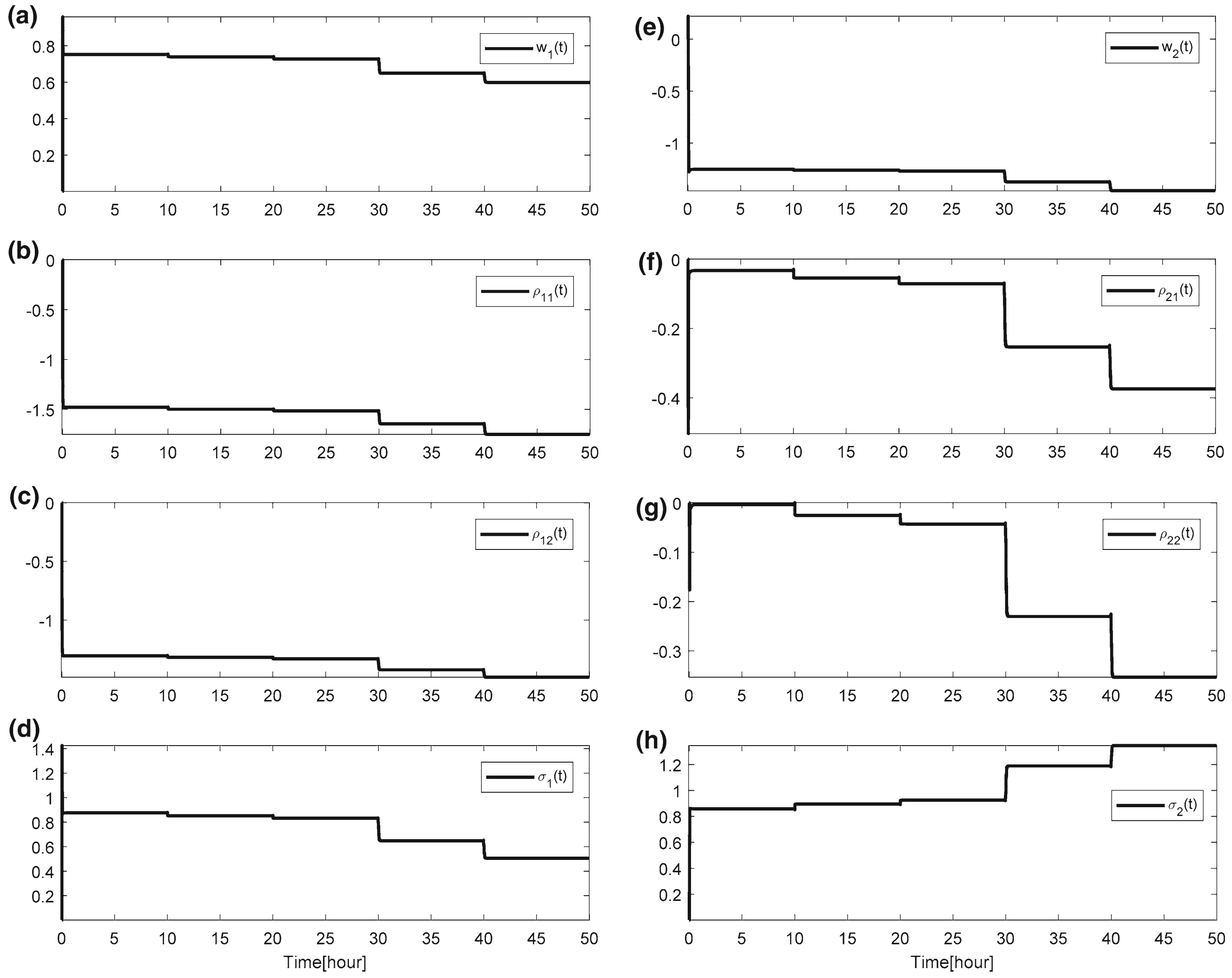


Fig. 18 Adaptation of RK-NN$_{\text{estimator}}$ parameters for $K_{d_{11}}$ controller parameter for Van de Vusse system

Table 3 Computation times (ms) for proposed controller

Systems	Three-tank system			Van de Vusse		
Operations	Noiseless	Noisy	Uncertain	Noiseless	Noisy	Uncertain
EKF state estimation	0.9083	1.0221	0.57008	0.75995	1.5073	1.3501
K-step prediction	1.7219	2.6969	1.3193	1.065	1.9113	2.1497
RK-NN$_{\text{estimator}}$ Training (LM)	14.5846	25.7949	14.7273	15.4252	27.0363	29.9557
Controller law	0.40633	1.8707	0.3107	0.48984	3.079	0.89944
RK-based model parameter						
Estimator training	–	–	0.52389	–	–	1.0399
Miscellaneous tasks	0.15721	0.21506	0.090503	0.26964	0.46185	0.26125
Total loop time	17.1056	29.3628	16.6937	16.8448	32.3247	34.5257

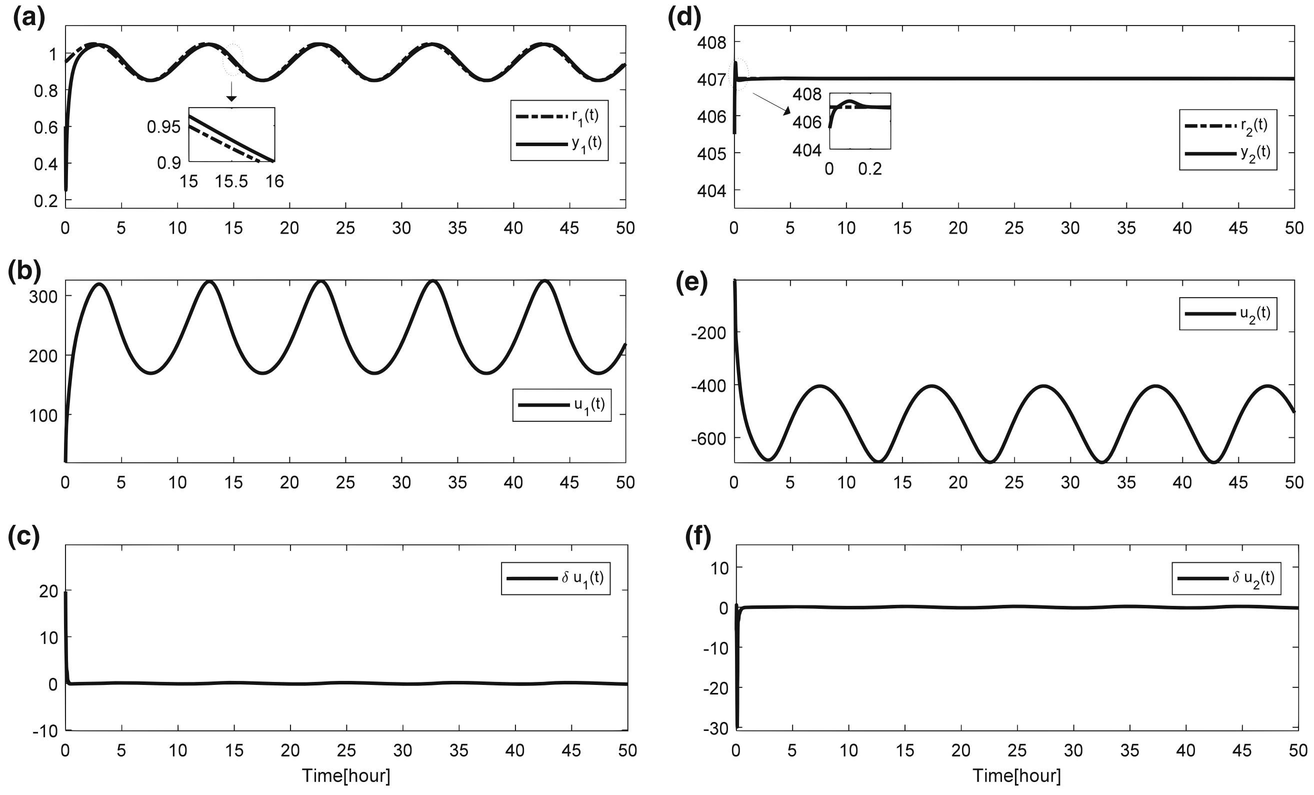


Fig. 19 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of Van de Vusse system for the nominal case with no measurement noise and parametric uncertainty (sinusoidal reference input)

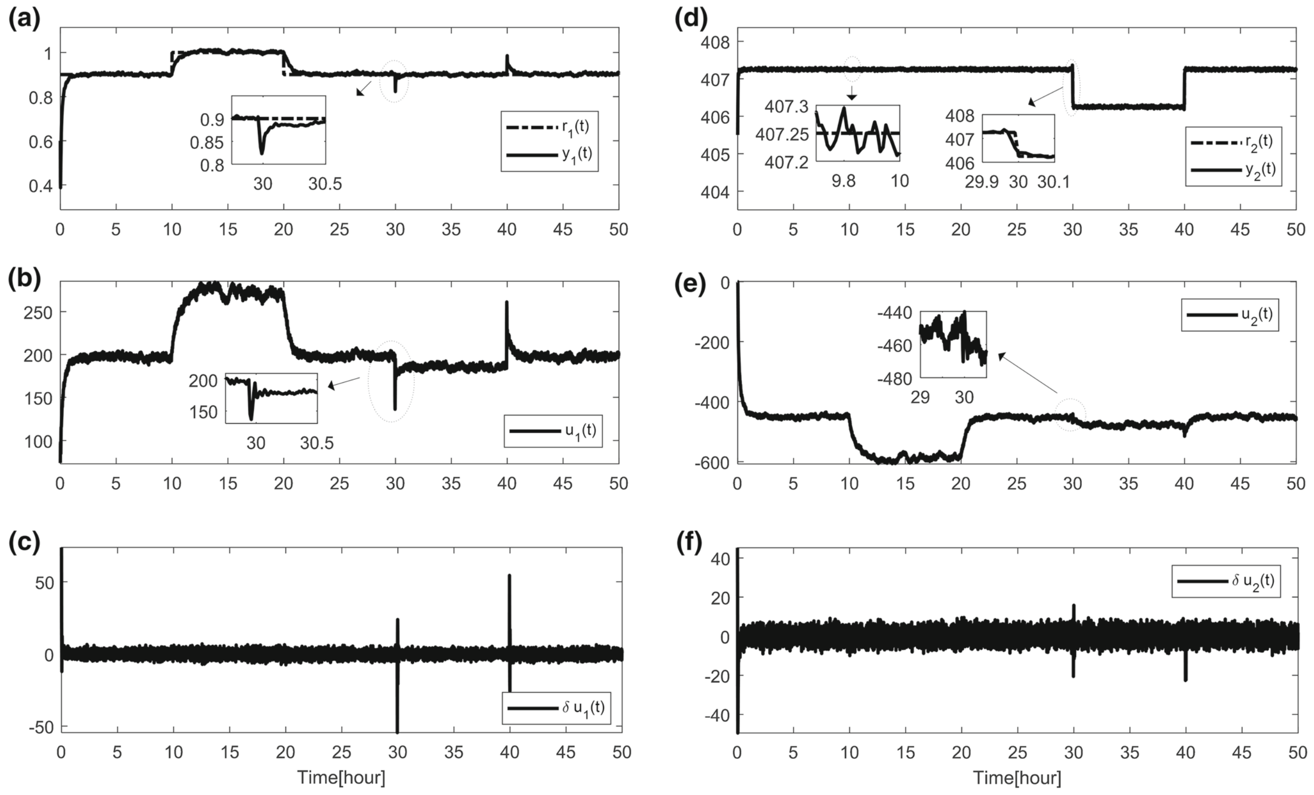


Fig. 20 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of Van de Vusse system for case with measurement noise (staircase reference inputs)

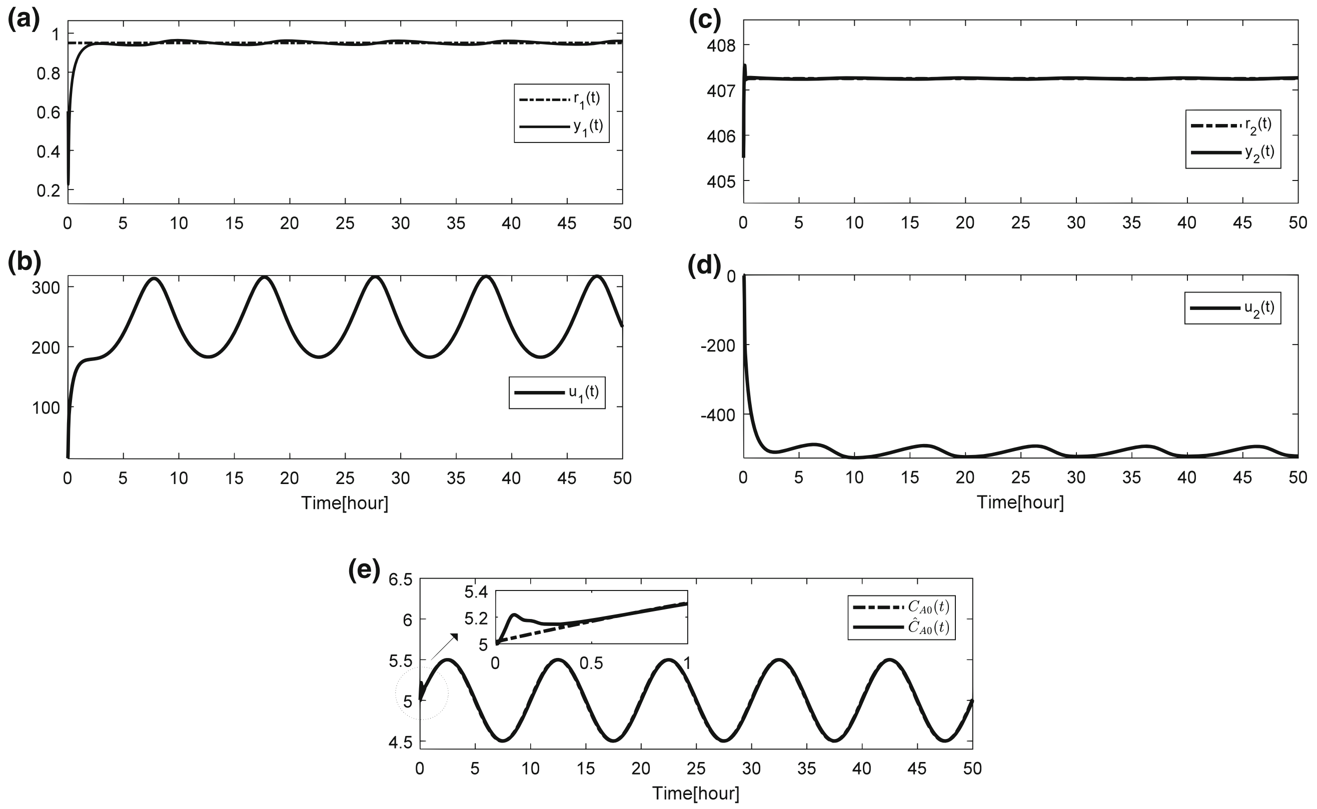


Fig. 21 System outputs (**a**, **c**), control signals (**b**, **d**) and uncertain system parameter ($C_{A0}(t)$) (**e**) and its estimation for the case with parametric uncertainty (Van de Vusse system)

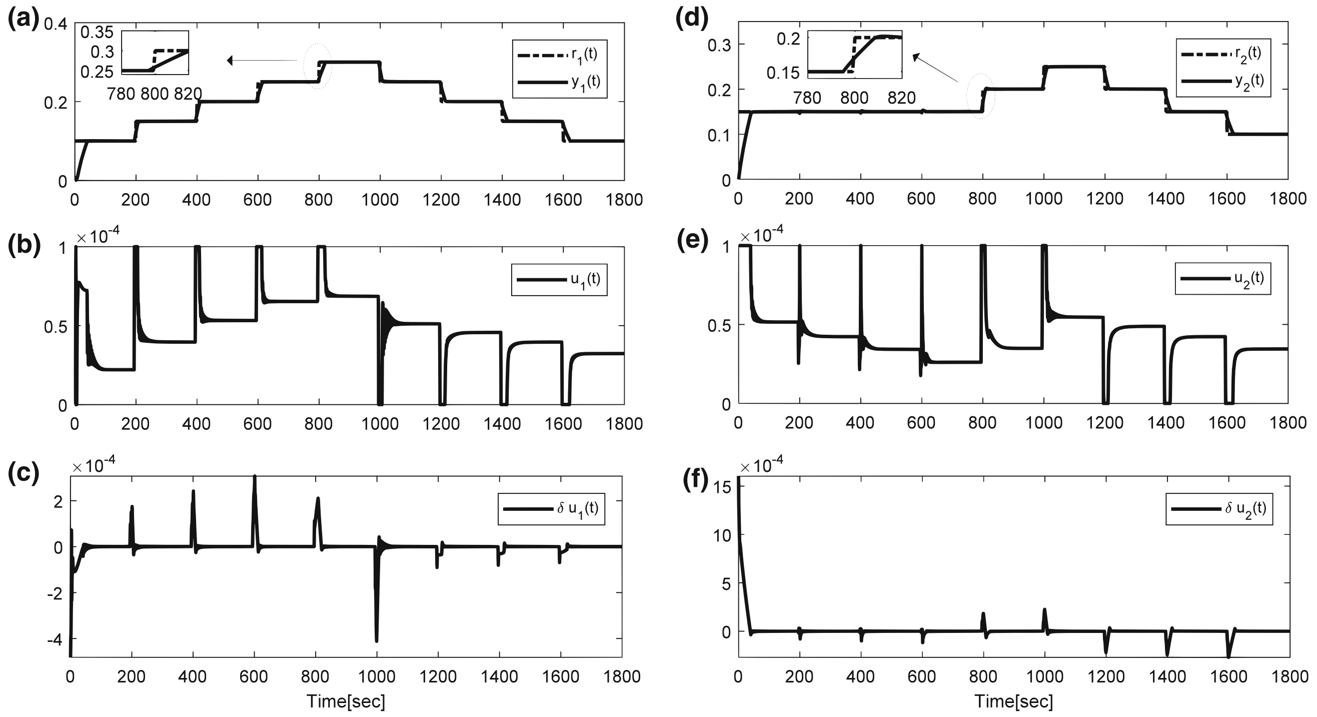


Fig. 22 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of three-tank system for the nominal case with no measurement noise and parametric uncertainty (staircase reference inputs) (RK model-based PID)

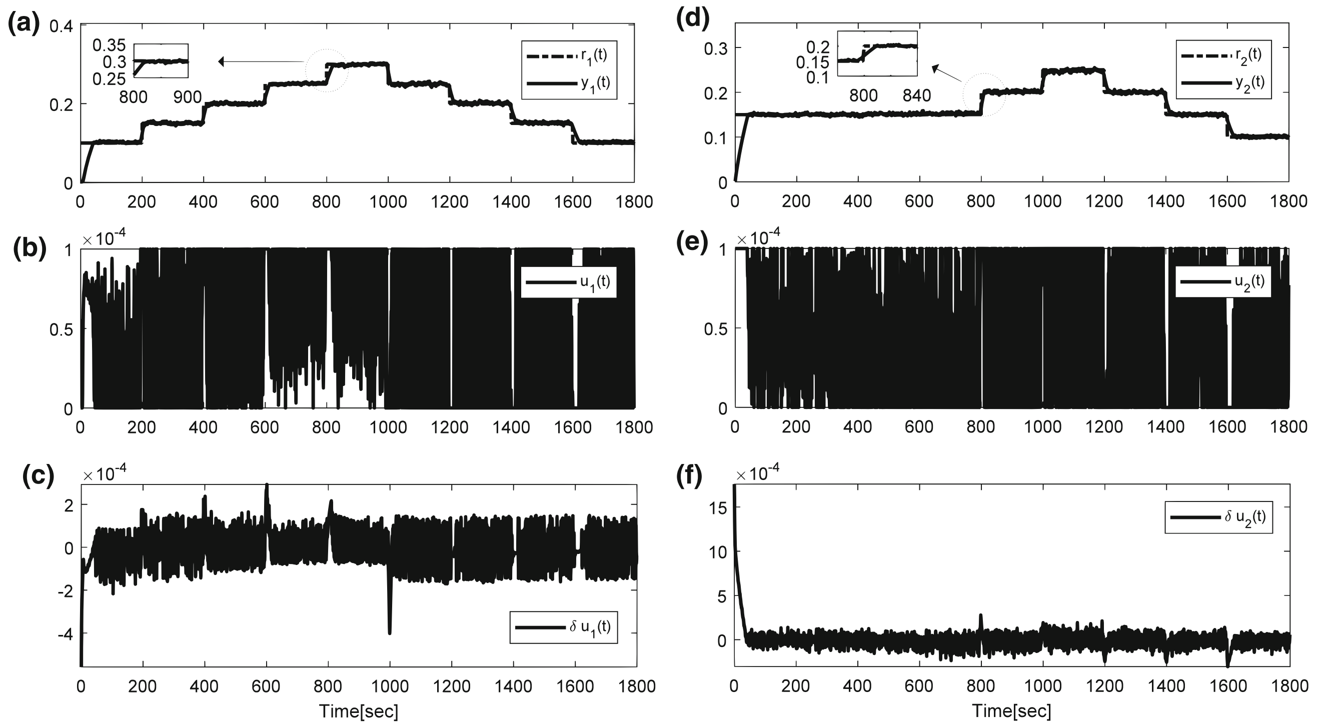


Fig. 23 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of three-tank system for the case with measurement noise (staircase reference inputs) (RK model-based PID)

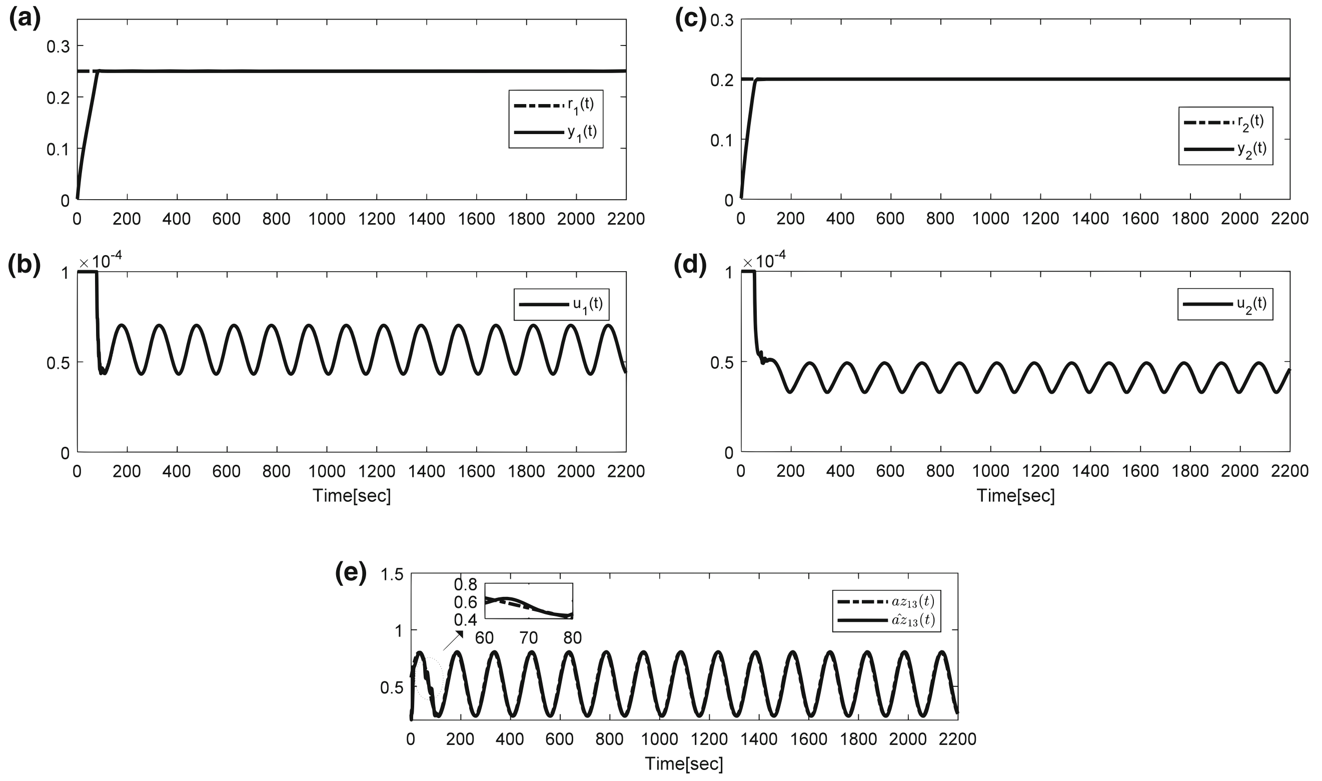


Fig. 24 System outputs (**a**, **c**), control signals (**b**, **d**) and uncertain outflow parameter ($az_{13}(t)$) (**e**) and its estimation for the case with parametric uncertainty (Three-tank system) (RK model based PID)

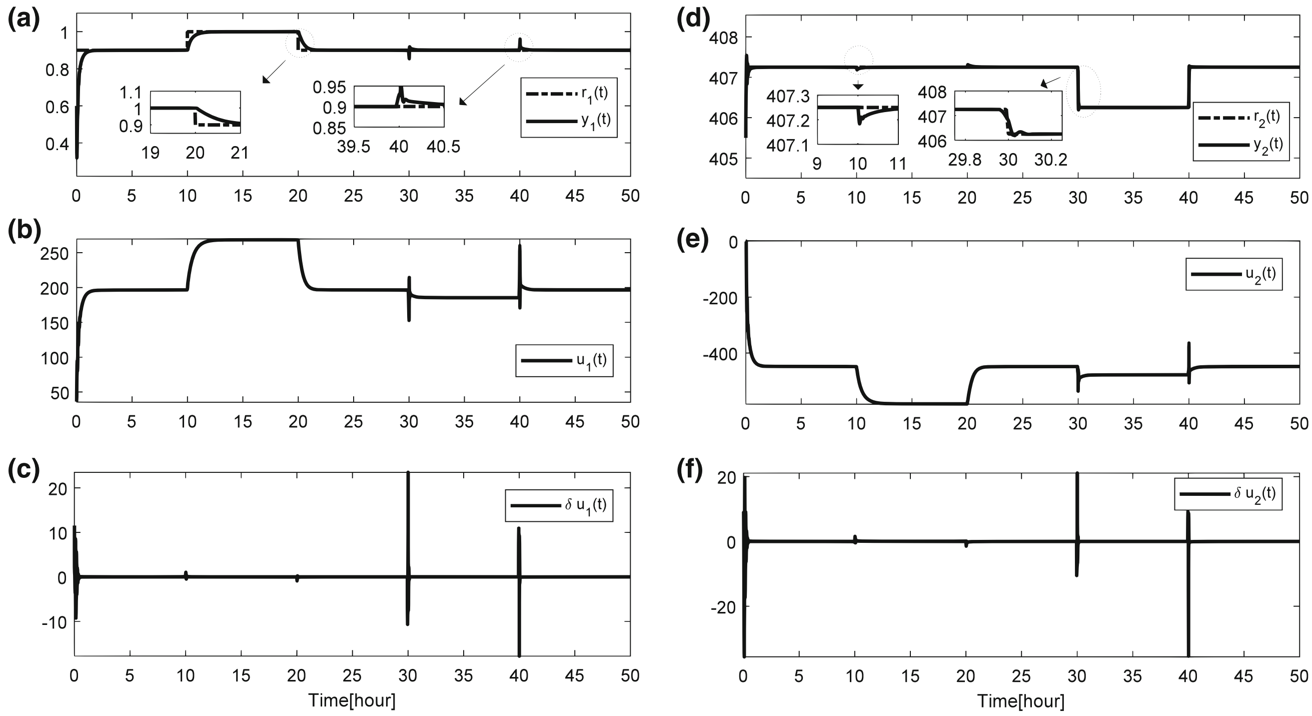


Fig. 25 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of Van de Vusse system for the nominal case with no measurement noise and parametric uncertainty (staircase reference inputs) (RK model-based PID)

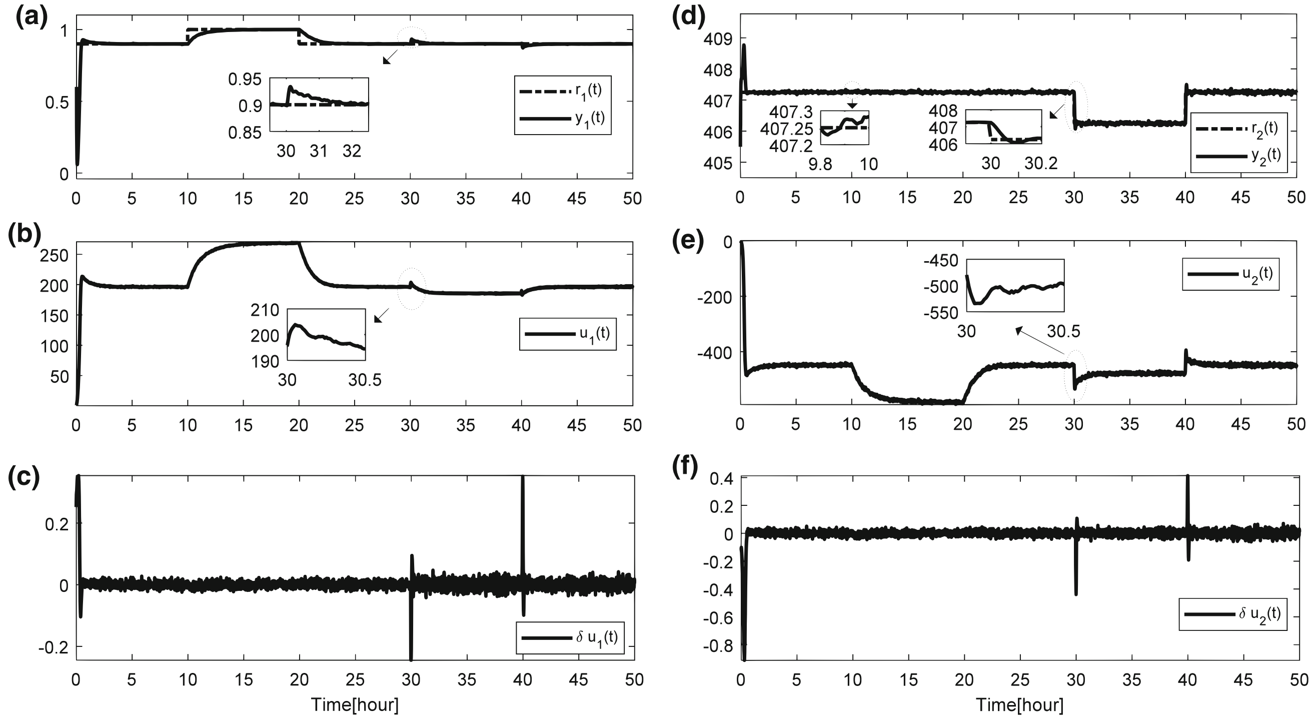


Fig. 26 System outputs (**a**, **d**), control signals (**b**, **e**) and correction terms (**c**, **f**) of Van de Vusse system for the case with measurement noise (staircase reference inputs) (RK model-based PID)

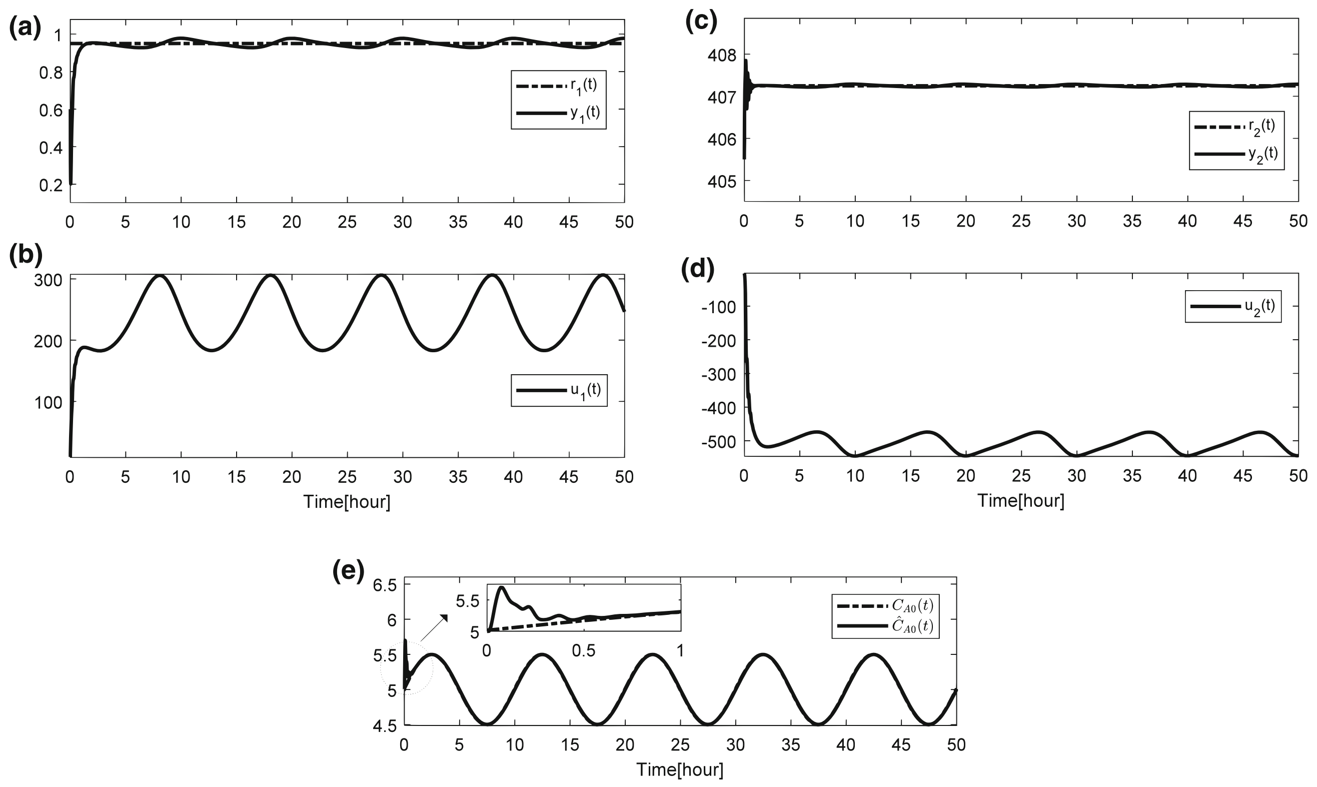


Fig. 27 System outputs (**a**, **c**), control signals (**b**, **d**) and uncertain system parameter ($C_{A0}(t)$) (**e**) and its estimation for the case with parametric uncertainty (Van de Vusse system) (RK model based PID)

Table 4 Tracking performance comparison for P_1 in (86)

Systems	Three-tank system			Van de Vusse		
Controllers	Noiseless	Noisy	Uncertain	Noiseless	Noisy	Uncertain
RK-NN-based PID	0.1253	0.1584	1.4157	1.3338	2.7958	0.9562
RK model-based PID	0.2152	0.2358	1.4118	3.3539	15.0886	8.7319

Table 5 Tracking performance comparison for P_2 in (86)

Systems	Three-tank system			Van de Vusse		
Controllers	Noiseless	Noisy	Uncertain	Noiseless	Noisy	Uncertain
RK-NN-based PID	0.2915	0.3026	0.6411	20.3835	9.2205	17.8882
RK model-based PID	0.2952	0.3032	0.6304	11.1640	49.7886	11.1324

tified as mathematical expressions for nonlinear systems. The adjustment mechanism is composed of two main blocks based on Runge–Kutta method: RK-NN$_{\text{estimator}}$ to identify and compute the controller parameters and RK$_{\text{model}}$ to reveal the K-step ahead future dynamical behaviour of the controlled system. RK-NN$_{\text{estimator}}$ inherits the powerful features of RBF neural network structure and Runge–Kutta integration method. Levenberg–Marquardt update rule is deployed to adjust the network parameters of RK-NN$_{\text{estimator}}$. RK$_{\text{model}}$ embodies three subblocks: RK raw system model deployed to extract gradient information required for Jacobian calculation; RK-based model parameter estimator employed for online estimation of time-varying parameters of the system and RK-based EKF utilized to approximate the unmeasurable states of the controlled system.

The performance evaluation of the proposed control method (or architecture) is carried out on nonlinear three-tank system and Van de Vusse benchmark system. The robustness of the controllers has also been examined for the noiseless, measurement noise and parametric uncertainty cases. Additionally, the performance of the controller has been compared with Runge–Kutta model-based PID controller.

The results indicate that the proposed adaptation mechanism for nonlinear MIMO systems accomplishes successful tracking performance as well as good noise rejection and high toleration to parametric uncertainties. In future works, it is planned to extend Runge–Kutta numerical integration method to develop new Runge–Kutta-type adaptive controller design methods for nonlinear MIMO systems.

Compliance with ethical standards

Conflict of interest The author declares that there is no conflict of interests regarding the publication of this paper.

Human and animal rights statement This article does not contain any studies with human participants or animals performed by any of the authors.

References

Akhyar S, Omatu S (1993) Self-tuning PID control by neural networks. In: International joint conference on neural network (IJCNN'93). Nagoya

AMIRA (2000) Lab Manual DTS200—laboratory setup three-tank system. Amira GmbH, Duisburg

Aström KJ (1983) Theory and applications of adaptive control—a survey. Automatica 19(5):471–486

Aström KJ, Hagglund T (1995) PID controllers: theory, design and tuning. Instrument Society of America, USA

Aström KJ, Wittenmark B (2008) Adaptive control. Dover Publications, Mineola

Aström KJ, Borisson U, Ljung L, Wittenmark B (1977) Theory and applications of self-tuning regulators. Automatica 13(5):457–476

Beyhan S (2013) Runge–Kutta model-based nonlinear observer for synchronization and control of chaotic systems. ISA Trans 52(4):501–509

Bouallégue S, Haggege J, Ayadi M, Benrejeb M (2012) PID-type fuzzy logic controller tuning based on particle swarm optimization. Eng Appl Artif Intell 25(3):484–493. https://doi.org/10.1016/j.engappai.2011.09.018

Bishr M, Yang YG, Lee G (2000) Self-tuning PID control using an adaptive network based fuzzy inference system. Intell Autom Soft Comput 6(4):271–280

Bobal V, Bohm J, Fessl J, Machacek J (2005) Digital self-tuning controllers. Advanced textbooks in control and signal processing. Springer, London

Cetin M, Iplikci S (2015) A novel auto-tuning PID control mechanism for nonlinear systems. ISA Trans 58:292–308

Chen H, Kremling A, Allgöwer F (1995) Nonlinear predictive control of a benchmark CSTR. In: European control conference (ECC'95). Rome

Denai MA, Palis F, Zeghbib A (2004) ANFIS based modelling and control of non-linear systems: a tutorial. In: IEEE international conference on systems, man and cybernetics

Efe MO (2011) Neural-network-based control. In: Wilamowski BM, Irwin JD (eds) The industrial electronics handbook: intelligent systems. CRC Press, Boca Raton

Efe MO, Kaynak O (1999) A comparative study of neural network structures in identification of nonlinear systems. Mechatronics 9(3):287–300. https://doi.org/10.1016/S0957-4158(98)00047-6

Efe MO, Kaynak O (2000) A comparative study of soft-computing methodologies in identification of robotic manipulators. Robot Auton Syst 30(3):221–230

Engell S, Klatt KU (1993) Nonlinear control of a non-minimum-phase CSTR. In: American control conference. San Francisco

Flynn D, McLoone S, Irwin GW, Brown MD, Swidenbank E, Hogg BW (1997) Neural control of turbogenerator systems. Automatica 33(11):1961–1973. https://doi.org/10.1016/S0005-1098(97)00142-8

Hagan MT, Demuth HB, De Jesus O (2002) An introduction to the use of neural networks in control systems. Int J Robust Nonlinear Control 12(11):959–985. https://doi.org/10.1002/rnc.727

Iplikci S (2006) Online trained support vector machines-based generalized predictive control of non-linear systems. Int J Adapt Control Signal Process 20(10):599–621. https://doi.org/10.1002/acs.919

Iplikci S (2010a) A comparative study on a novel model-based PID tuning and control mechanism for nonlinear systems. Int J Robust Nonlinear Control 20(13):1483–1501. https://doi.org/10.1002/rnc.1524

Iplikci S (2010b) A support vector machine based control application to the experimental three-tank system. ISA Trans 49(3):376–386. https://doi.org/10.1016/j.isatra.2010.03.013

Iplikci S (2013) Runge-Kutta model-based adaptive predictive control mechanism for non-linear processes. Trans Inst Meas Control 35(2):166–180

Jang JSR (1993) ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans Syst Man Cybern 23(3):665–685. https://doi.org/10.1109/21.256541

Kravaris C, Niemiec M, Berber R, Brosilow CB (1998) Nonlinear model-based control of nonminimum-phase processes. In: Kravaris C, Berber R (eds) Nonlinear model based process control. Springer, Dordrecht, pp 115–142

Kulikov GY, Kulikova MV (2014) Accurate state estimation in the Van Der Vusse reaction. In: IEEE international conference on control applications (CCA). Nice

Luenberger DG, Ye Y (2008) Linear and nonlinear programming. Springer, New York

Niemiec MP, Kravaris C (2003) Nonlinear model-state feedback control for nonminimum-phase processes. Automatica 39(7):1295–1302. https://doi.org/10.1016/S0005-1098(03)00103-1

Nørregard JB (2007) A critical discussion of the continuous-discrete extended Kalman filter. In: European congress of chemical engineering-6. Copenhagen

Pham DT, Karaboga D (1999) Self-tuning fuzzy controller design using genetic optimisation and neural network modelling. Artif Intell Eng 13(2):119–130. https://doi.org/10.1016/S0954-1810(98)00017-X

Sharkawy AB (2010) Genetic fuzzy self-tuning PID controllers for antilock braking systems. Eng Appl Artif Intell 23(7):1041–1052. https://doi.org/10.1016/j.engappai.2010.06.011

Sung SW, Lee J, Lee IB (2009) Process identification and PID control. IEEE Press, Wiley, Singapore

Theilliol D, Noura H, Ponsart JC (2002) Fault diagnosis and accommodation of a three-tank system based on analytical redundancy. ISA Trans 41(3):365–382. https://doi.org/10.1016/S0019-0578(07)60094-9

Thrun S, Burgard W, Fox D (2005) Probabilistic robotics. The MIT Press, Cambridge

Uçak K, Günel GO (2017) Generalized self-tuning regulator based on online support vector regression. Neural Comput Appl 28(Suppl 1):775–801. https://doi.org/10.1007/s00521-016-2387-4

Wanfeng S, Shengdun Z, Yajing S (2008) Adaptive PID controller based on online LSSVM identification. In: IEEE/ASME international conference on advanced intelligent mechatronics (AIM 2008). Xian

Wang YJ, Lin CT (1998) Runge-Kutta neural network for identification of dynamical systems in high accuracy. IEEE Trans Neural Netw 9(2):294–307. https://doi.org/10.1109/72.661124

Wang GJ, Fong CT, Chang KJ (2001) Neural-network-based self-tuning PI controller for precise motion control of PMAC motors. IEEE Trans Ind Electron 48(2):408–415

Vojtesek J, Dostal P (2010) Adaptive control of chemical reactor. In: International conference cybernetics and informatics. Vysna Boca

Visioli A (2006) Practical PID control. Springer, London

Zhao XD, Yang HJ, Karimi HR, Zhu YZ (2016a) Adaptive neural control of MIMO nonstrict-feedback nonlinear systems with time delay. IEEE Trans Cybern 46(6):1337–1349. https://doi.org/10.1109/TCYB.2015.2441292

Zhao XD, Shi P, Zheng XL (2016b) Fuzzy adaptive control design and discretization for a class of nonlinear uncertain systems. IEEE Trans Cybern 46(6):1476–1483. https://doi.org/10.1109/TCYB.2015.2447153