INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
*Int. J. Robust. Nonlinear Control* 2013; **23**:991–1012
Published online 16 April 2012 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/rnc.2814

# Computationally efficient simultaneous policy update algorithm for nonlinear $H_\infty$ state feedback control with Galerkin's method

Biao Luo and Huai-Ning Wu*,†

*Science and Technology on Aircraft Control Laboratory, School of Automation Science and Electrical Engineering, Beihang University (Beijing University of Aeronautics and Astronautics), Beijing 100191, P. R. China*

SUMMARY

The main bottleneck for the application of $H_\infty$ control theory on practical nonlinear systems is the need to solve the Hamilton–Jacobi–Isaacs (HJI) equation. The HJI equation is a nonlinear partial differential equation (PDE) that has proven to be impossible to solve analytically, even the approximate solution is still difficult to obtain. In this paper, we propose a simultaneous policy update algorithm (SPUA), in which the nonlinear HJI equation is solved by iteratively solving a sequence of Lyapunov function equations that are linear PDEs. By constructing a fixed point equation, the convergence of the SPUA is established rigorously by proving that it is essentially a Newton's iteration method for finding the fixed point. Subsequently, a computationally efficient SPUA (CESPUA) based on Galerkin's method, is developed to solve Lyapunov function equations in each iterative step of SPUA. The CESPUA is simple for implementation because only one iterative loop is included. Through the simulation studies on three examples, the results demonstrate that the proposed CESPUA is valid and efficient. Copyright © 2012 John Wiley & Sons, Ltd.

Received 10 September 2011; Revised 26 December 2011; Accepted 16 February 2012

KEY WORDS: Hamilton–Jacobi–Isaacs equation; simultaneous policy update algorithm; $H_\infty$ state feedback control; Galerkin's method; convergence

## 1. INTRODUCTION

An important paradigm in control theory is $H_\infty$ control, which is used to synthesize controllers achieving robust performance or stabilization. Over the past few decades, a large number of theoretical results on $H_\infty$ control have been reported [1–8]. Although the formulation of the nonlinear theory of $H_\infty$ control has been well developed, the main bottleneck for the practical application of the theory is the need to solve the Hamilton–Jacobi–Isaacs (HJI) equation. The HJI equation, similar with the Hamilton– Jacobi–Bellman (HJB) equation of nonlinear optimal control, is a first-order nonlinear partial differential equation (PDE), which is difficult or impossible to solve, and may not have global analytic solutions even in simple cases.

In recent years, some works [9–12] have been conducted for the solution of HJB equation in nonlinear optimal control. However, it is clear that HJI equations are generally more difficult to solve than HJB equations, because the disturbance inputs are additionally reflected in HJI equations. The HJI equation is intractable to solve directly, thus, many works have be directed toward approximating their solutions. In [8], it was proven that there exists a sequence of policy iterations on the control input to pursue the smooth solution of the HJI equation, where the HJI equation was successively

*Correspondence to: Huai-Ning Wu, Science and Technology on Aircraft Control Laboratory, School of Automation Science and Electrical Engineering, Beihang University (Beijing University of Aeronautics and Astronautics) Beijing 100191, P. R. China.

†E-mail: whn@buaa.edu.cn

approximated with a sequence of HJB equations (more actually, they are HJB-like equations, see Section 2.2). Then, the methods for solving HJB equation can be used for the solution of HJI equation. In [9], the HJB equation was successively approximated by a sequence of linear generalized HJB (GHJB) equations, which were solved with Galerkin's approximation in [10, 11]. On the basis of these works [8–11], iteration in policy space was used to approximate the HJI equation in [13], where each HJB equation in [8] was further successively approximated by a sequence of generalized HJI (GHJI) equations (that are essentially the same as GHJB equations, see Section 2.2) and solved with Galerkin's approximation. This obviously results in two iterative loops for the solution of HJI equation, that is, the inner loop solves an HJB equation by iteratively solving a sequence of GHJB equations, and the outer loop solves the HJI equation by iteratively solving a sequence of HJB equations. In [12], a revised generalized policy iteration method was developed to solve the HJB equation, in which the weights of both actor and critic networks tune at the same time. This work [12] was extended to solve the HJI equation online in [14]. Similar to [13], a policy iteration scheme was developed in [15] for the constrained input system, and a practical way to implement this scheme was developed on the basis of neuro-dynamic programming in [16], where neural networks were used for value function approximation. However, this approach can only be applied to the case that the saddle point exists, thus, a situation that the saddle point does not exist was considered in [17]. In [18], Feng and colleagues extended the method in [19] (for $H_\infty$algebra Riccati equation (ARE)) to solve the HJI equation, in which the approaches in [20] and [21] (as described in [18]) were used to solve the associated HJB equations (or HJB-like equations).

Essentially, the works in [13–18] follow such a thought that the HJI equation is first successively approximated with a sequence of HJB equations, and then each HJB equation is solved by the existing methods. This often brings two iterative loops, because the updates of control and disturbance policies are asynchronous, that is, one player updates its policy while the other remains invariant. Such a procedure may lead to redundant equation solutions (i.e., redundant iterations), and thus waste of resources and result in low efficiency. In this paper, we propose a simultaneous policy update algorithm (SPUA) for solving the HJI equation in nonlinear $H_\infty$ state feedback control, where the control and disturbance policies update simultaneously. The SPUA avoids solving the HJB equations, because the HJI equation is directly successively approximated by a sequence of Lyapunov function equations (LFEs). This leads to only one iterative loop in the SPUA rather than two. Hence, the SPUA is much simpler and easier to implement than the existing methods. Because the SPUA is essentially different with the algorithm framework in [13–18], the proof of its convergence is also different, which needs a new course of action. By constructing a fixed point equation, the convergence of the SPUA is established by showing that it is a Newton's method in a Banach space. For implementation purposes, we employ Galerkin's method to solve LFEs, and further develop a computationally efficient SPUA (CESPUA). In Galerkin's method, amounts of integrals need to be computed, which are often time-consuming and will increase exponentially with the increase of the number of basis functions and the dimension of system states. In the CESPUA, all integrals are computed once for all. Furthermore, we use Monte Carlo integration together with the Latin hypercube sampling (LHS) to evaluate all integrals, which further improves the efficiency of the CESPUA.

The rest of this paper is organized as follows. In Section 2, we give the problem description and briefly present the preliminary results. In Section 3, we propose the SPUA and discuss some related issues, where a CESPUA is also developed. Simulation studies are conducted in Section 4. Finally, a brief conclusion is derived in Section 5.

*Notations:* $\mathbb{R}$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ are the set of real numbers, the $n$-dimensional Euclidean space and the set of all real $n\times m$ matrices, respectively. $\|\cdot\|$ denotes the vector norm or matrix norm in $\mathbb{R}^n$ or $\mathbb{R}^{n\times m}$, respectively. For a symmetric matrix $\boldsymbol{M}$, $\boldsymbol{M} > (\geqslant) 0$ means that it is a positive (semi-positive) definite matrix. The superscript $T$ is used for the transpose and $I$ denotes the identity matrix of appropriate dimension. $L_2[0,\infty)$ is a Banach space, for $\forall \boldsymbol{w}(t) \in L_2[0,\infty)$, $\int_0^\infty \|\boldsymbol{w}(t)\|^2 \,\mathrm{d}t < \infty$. For a column vector function $\boldsymbol{s}(\boldsymbol{x})$, $\boldsymbol{x} \in \Omega \subset \mathbb{R}^n$, $\|\boldsymbol{s}(\boldsymbol{x})\|_\Omega \triangleq \left(\int_\Omega \boldsymbol{s}^T(\boldsymbol{x})\boldsymbol{s}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\right)^{1/2}$.



DOI: 10.1002/rnc

## 2. BACKGROUND

In this section, we present some background knowledge. First, we give the problem description of continuous-time nonlinear $H_\infty$ control. Second, we briefly review the method in [13] for solving the HJI equation.

### 2.1. Problem description

Let us consider the following continuous-time nonlinear system with external disturbance:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}(t) + \boldsymbol{k}(\boldsymbol{x})\boldsymbol{w}(t) \tag{1}$$

$$z(t) = \boldsymbol{h}(\boldsymbol{x}) \quad , \tag{2}$$

where $\boldsymbol{x} \in \Omega \subset \mathbb{R}^n$ is the state, $\boldsymbol{u} \in \mathbb{R}^m$ is the control input and $\boldsymbol{u}(t) \in L_2[0,\infty)$, $\boldsymbol{w} \in \mathbb{R}^q$ is the external disturbance and $\boldsymbol{w}(t) \in L_2[0,\infty)$, and $z \in \mathbb{R}^p$ is the objective output. $\boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{g}(\boldsymbol{x})$, $\boldsymbol{k}(\boldsymbol{x})$, and $\boldsymbol{h}(\boldsymbol{x})$ are vector or matrix functions of appropriate dimension. We assume $\boldsymbol{x}_e = 0$ be an equilibrium point.

The $H_\infty$ control problem under consideration is to find a state feedback control law $\boldsymbol{u}(t) = \boldsymbol{u}(\boldsymbol{x}(t))$ such that the systems (1) and (2) are closed-loop asymptotically stable, and has $L_2$-gain less than or equal to $\gamma$, that is,

$$\int_0^\infty \left( \|z(t)\|^2 + \|\boldsymbol{u}(t)\|_{\boldsymbol{R}}^2 \right) \mathrm{d}t \leqslant \gamma^2 \int_0^\infty \|\boldsymbol{w}(t)\|^2 \, \mathrm{d}t \tag{3}$$

for all $\boldsymbol{w}(t) \in L_2[0,\infty)$, where $\|\boldsymbol{u}(t)\|_{\boldsymbol{R}}^2 = \boldsymbol{u}^T \boldsymbol{R}\boldsymbol{u}$, $\boldsymbol{R} > 0$ and $\gamma > 0$ is some prescribed level of disturbance attenuation.

*Lemma 1 (see Theorem 16 and Corollary 17 in* [8]*)*
Assume the systems (1) and (2) are zero-state observable. Let $\gamma > 0$. Suppose there exists a smooth solution $V^*(\boldsymbol{x}) \geqslant 0$ to the HJI equation

$$\begin{aligned} \mathcal{G}(V^*) = {} & \left( \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} \right)^T \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{h}^T(\boldsymbol{x})\boldsymbol{h}(\boldsymbol{x}) - \frac{1}{4} \left( \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} \right)^T \boldsymbol{g}(\boldsymbol{x}) \boldsymbol{R}^{-1} \boldsymbol{g}^T(\boldsymbol{x}) \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} \\ & + \frac{1}{4\gamma^2} \left( \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} \right)^T \boldsymbol{k}(\boldsymbol{x}) \boldsymbol{k}^T(\boldsymbol{x}) \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} = 0. \end{aligned} \tag{4}$$

Then, the closed-loop system for the state feedback control

$$\boldsymbol{u}(t) = \boldsymbol{u}^*(\boldsymbol{x}(t)) = -\frac{1}{2} \boldsymbol{R}^{-1} \boldsymbol{g}^T(\boldsymbol{x}) \frac{\partial V^*(\boldsymbol{x})}{\partial \boldsymbol{x}} \tag{5}$$

has $L_2$-gain less than or equal to $\gamma$, and the closed-loop systems (1), (2), and (5) (when $\boldsymbol{w}(t) \equiv 0$) is locally asymptotically stable.

### 2.2. Preliminary results

It is noted from Lemma 1 that the nonlinear $H_\infty$ control problem hinges on the solution of the HJI equation (4). In [13], the HJI equation was solved by using two iterative loops. Here, we briefly review this approach in Algorithm 1 without deduction.

**Algorithm 1.**

*Step 1:* Give an initial control law $\boldsymbol{u}^{(0)}$ that can stabilize the system (1) (with $\boldsymbol{w} = 0$). Let $i = 0$.
*Step 2:* Let $\boldsymbol{w}^{(i+1,0)} = 0$, and $j = 0$.
*Step 3:* Solve the following GHJI equation for $V^{(i+1,j+1)}$:



DOI: 10.1002/rnc

$$\left(\frac{\partial V^{(i+1,j+1)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f}+\boldsymbol{g}\boldsymbol{u}^{(i)}+\boldsymbol{k}\boldsymbol{w}^{(i+1,j)}\right)+\boldsymbol{h}^T\boldsymbol{h}+\left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} - \gamma^2\left(\boldsymbol{w}^{(i+1,j)}\right)^T \boldsymbol{w}^{(i+1,j)}=0. \tag{6}$$

*Step 4:* Update the disturbance policy with

$$\boldsymbol{w}^{(i+1,j+1)}=\frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\frac{\partial V^{(i+1,j+1)}}{\partial \boldsymbol{x}}. \tag{7}$$

*Step 5:* Set $j=j+1$. If $\left\|V^{(i+1,j)}-V^{(i+1,j-1)}\right\|_\Omega \leqslant \varepsilon_1$ ($\varepsilon_1$ is a small positive real number), let $V^{(i+1,*)}=V^{(i+1,j)}$ and go to step 6, else, go back to step 3.

*Step 6:* Update the control policy with

$$\boldsymbol{u}^{(i+1)}=-\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i+1,*)}}{\partial \boldsymbol{x}}. \tag{8}$$

*Step 7:* Set $i=i+1$. If $\left\|\boldsymbol{u}^{(i)}-\boldsymbol{u}^{(i-1)}\right\|_\Omega \leqslant \varepsilon$ ($\varepsilon$ is a small positive real number), stop and output $V^{(i,*)}$ as the solution of the HJI equation (4) $\left(\text{i.e., } V^*=V^{(i,*)}\right)$, else, go back to step 2 and continue.

*Remark 1*

It is observed that Algorithm 1 has two iterative loops. The inner iterative loop (i.e., from step 2 to step 5) is to solve the following HJB-like equation for $V^{(i+1,*)}$:

$$\left(\frac{\partial V^{(i+1,*)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f}+\boldsymbol{g}\boldsymbol{u}^{(i)}\right)+\boldsymbol{h}^T\boldsymbol{h}+\left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)}+\frac{1}{4\gamma^2}\left(\frac{\partial V^{(i+1,*)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i+1,*)}}{\partial \boldsymbol{x}}=0. \tag{9}$$

Then, the (9) is successively approximated with a sequence of GHJI equations (6), which are LFEs (i.e., linear PDEs). The outer iterative loop (i.e., from step 1 to step 7) is to solve the HJI equation by solving a sequence of HJB-like equations (9). In other words, the inner loop is for the update of disturbance policy $\boldsymbol{w}$, and the outer loop is for the update of control policy $\boldsymbol{u}$.

*Remark 2*

If we exchange the roles of $\boldsymbol{w}$ and $\boldsymbol{u}$ in Algorithm 1, the (9) is an HJB equation for optimal control (with $\boldsymbol{w}$ fixed) and the GHJI equations (6) is the GHJB equation. It is worth pointing out that the HJB equation and HJB-like equation, and the GHJB equation and GHJI equation are not foundationally different. For simplicity, throughout the whole paper, we denote the HJB-like equation and GHJI equation as HJB equation and GHJB equation, respectively.

*Remark 3*

In [13], Galerkin's method was used to solve the LFEs (6). Of course, any other method for the solution of linear PDE can also be used for this purpose, such as, using approximate dynamic programming with neural networks in [14, 16, 17]. That is to say, the methods in [13–17] follow the same procedure as in Algorithm 1.

## 3. COMPUTATIONALLY EFFICIENT SPUA WITH GALERKIN'S METHOD

We notice that Algorithm 1 brings two iterative loops, that is to say, while one player's policy is updated the other remains invariant. Thus, an intuitive question is that 'Can we update both control policy and disturbance policy simultaneously?' This section gives a positive answer. A SPUA is proposed, where both policies update simultaneously.



DOI: 10.1002/rnc

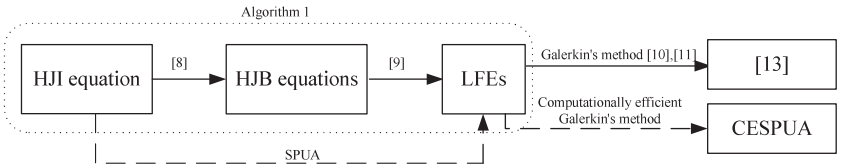



Figure 1. The main thoughts of the Algorithm 1 and the proposed SPUA.

We use Figure 1 to show the main differences between the Algorithm 1 and the proposed SPUA intuitively. In [8], it was shown that the HJI equation can be successively approximated by a sequence of HJB equations (9). In [9], the HJB equation was successively approximated by a sequence of LFEs. Thus, Algorithm 1 is in fact a direct extension of the works in [8] and [9] for solving the HJI equation. In [13], Galerkin's approximation developed in [10] and [11] was used to solve LFEs. Just as mentioned in Remark 3, the methods in [13–17] follow the same procedure as in Algorithm 1, thus, their convergence is guaranteed by References [8, 9, 13]. However, in the SPUA, we use a sequence of LFEs to directly approximate the HJI equation and develop an efficient method for solving LFEs. Because only one iterative loop is included in the SPUA, a new course of action is needed for the proof of its convergence.

### *3.1. Simultaneous policy update algorithm and its convergence*

In this subsection, we propose the SPUA for the solution of HJI equation (4) and establish its convergence. We first give the procedure of the SPUA.

**Algorithm 2 (SPUA).**

*Step 1*: Give an initial function $V^{(0)} \in \mathbb{V}_0$ ($\mathbb{V}_0 \subset \mathbb{V}$ is determined by Lemma 5). Let $\boldsymbol{u}^{(0)} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(0)}}{\partial \boldsymbol{x}}$, $\boldsymbol{w}^{(0)} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\frac{\partial V^{(0)}}{\partial \boldsymbol{x}}$, and $i = 0$.

*Step 2*: Solve the following LFE for the cost function $V^{(i+1)}$:

$$\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T\left(\boldsymbol{f} + \boldsymbol{g}\boldsymbol{u}^{(i)} + \boldsymbol{k}\boldsymbol{w}^{(i)}\right) + \boldsymbol{h}^T\boldsymbol{h} + \left(\boldsymbol{u}^{(i)}\right)^T\boldsymbol{R}\boldsymbol{u}^{(i)} - \gamma^2\left(\boldsymbol{w}^{(i)}\right)^T\boldsymbol{w}^{(i)} = 0 \quad (10)$$

*Step 3*: Update the control and disturbance policies with

$$\boldsymbol{u}^{(i+1)} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}} \quad (11)$$

$$\boldsymbol{w}^{(i+1)} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}} \quad (12)$$

*Step 4*: Set $i = i + 1$. If $\left\|V^{(i)} - V^{(i-1)}\right\|_\Omega \leqslant \varepsilon$ ($\varepsilon$ is a small positive real number), stop and output $V^{(i)}$ as the solution of the HJI equation (4) $\left(\text{i.e., } V^* = V^{(i)}\right)$, else, go back to step 2 and continue.

*Remark 4*

Compared with Algorithm 1, the proposed SPUA reduces two iterative loops to one and updates the control and disturbance policies simultaneously, which avoids solving the HJB equation (9) by using a sequence of LFEs (10) to directly successively approximate the HJI equation (4).

Next, we will establish the convergence of SPUA. Namely, we want to show that the solution of LFEs (10) converges to the solution of HJI equation (4) when $i$ goes to infinity.



DOI: 10.1002/rnc

To this end, let us consider such a Banach space $\mathbb{V} \subset \{V(\boldsymbol{x}) : \Omega \to \mathbb{R}\}$ equipped with a norm $\|\cdot\|_\Omega$, and consider the mapping $\mathcal{G} : \mathbb{V} \to \mathbb{V}$ defined in (4). Define a mapping $\mathcal{T} : \mathbb{V} \to \mathbb{V}$ as follows:

$$\mathcal{T}V = V - (\mathcal{G}'(V))^{-1}\mathcal{G}(V), \tag{13}$$

where $\mathcal{G}'(V)$ is the Fréchet derivative of $\mathcal{G}(\cdot)$ at point $V$. It should be noticed that both $\mathcal{G}'(V)$ and $(\mathcal{G}'(V))^{-1}$ are operators on Banach space $\mathbb{V}$.

The Fréchet derivative is often difficult to compute directly, thus we introduce the Gâteaux derivative.

*Definition 1 (Gâteaux derivative* [22]*)*
Let $\mathcal{G} : \mathbb{U}(V) \subseteq \mathbb{X} \to \mathbb{Y}$ be a given map, with $\mathbb{X}$ and $\mathbb{Y}$ Banach spaces. Here, $\mathbb{U}(V)$ denotes a neighborhood of $V$. The map $\mathcal{G}$ is *Gâteaux differentiable* at $V$ if there exists a bounded linear operator $\mathcal{L} : \mathbb{X} \to \mathbb{Y}$ such that

$$\mathcal{G}(V + sW) - \mathcal{G}(V) = s\mathcal{L}(W) + o(s), \quad s \to 0, \tag{14}$$

for all $W$ with $\|W\|_\Omega = 1$ and all real numbers $s$ in some neighborhood of zero, where $\lim_{s\to 0}(o(s)/s) = 0$. $\mathcal{L}$ is called the *Gâteaux derivative* of $\mathcal{G}$ at $V$. The *Gâteaux differential* at $V$ is defined by $\mathcal{L}(W)$.

From (14), the Gâteaux differential at $V$ can be defined equivalently through the following expression [22]:

$$\mathcal{L}(W) = \lim_{s\to 0} \frac{\mathcal{G}(V + sW) - \mathcal{G}(V)}{s}. \tag{15}$$

The formula (15) gives a method to compute the Gâteaux derivative, rather than Fréchet derivative required in (13). Thus, we introduce the following Lemma to give the relationship between them.

*Lemma 2 (*[22]*)*
If $\mathcal{G}'$ exists as Gâteaux derivative in some neighborhood of $V$, and if $\mathcal{G}'$ is continuous at $V$, then $\mathcal{L} = \mathcal{G}'(V)$ is also a Fréchet derivative at $V$.

Now, it follows from Lemma 2 that we can compute the Fréchet derivative $\mathcal{G}'(V)$ in (13) via (15). We have the following result.

*Lemma 3*
Let $\mathcal{G} : \mathbb{V} \to \mathbb{V}$ be a mapping defined as (4), then, for $\forall V \in \mathbb{V}$, the Fréchet differential of $\mathcal{G}$ at $V$ is

$$\begin{aligned}\mathcal{G}'(V)W = \mathcal{L}(W) = &\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{f} - \frac{1}{4}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V}{\partial \boldsymbol{x}} - \frac{1}{4}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial W}{\partial \boldsymbol{x}} \\ &+ \frac{1}{4\gamma^2}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial V}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial W}{\partial \boldsymbol{x}}.\end{aligned} \tag{16}$$

*Proof*
See Appendix. □

Given an initial function $V^{(0)}$, we construct a Newton iterative sequence $\{V^{(i)}\}$ as

$$V^{(i+1)} = \mathcal{T}V^{(i)}, \quad i = 0, 1, 2, \cdots. \tag{17}$$

Under some proper assumptions, this sequence (17) can converge to the unique solution of the fixed-point equation $\mathcal{T}V^* = V^*$, that is, the solution of equation $\mathcal{G}(V^*) = 0$. The convergence and the error bound of sequence $\{V^{(i)}\}$ are guaranteed by the following Kantorovtich's Theorem [23, 24].



DOI: 10.1002/rnc

*Lemma 4 (Kantorovtich's Theorem)*
Assume for some $V^{(0)} \in \mathbb{V}_1 \subset \mathbb{V}$ such that $\left(\mathcal{G}'\left(V^{(0)}\right)\right)^{-1}$ exists and that

1)

$$\left\|\left(\mathcal{G}'\left(V^{(0)}\right)\right)^{-1}\right\|_\Omega \leqslant B_0, \tag{18}$$

2)

$$\left\|\left(\mathcal{G}'(V^{(0)})\right)^{-1}\mathcal{G}\left(V^{(0)}\right)\right\|_\Omega \leqslant \eta, \tag{19}$$

3)

$$\left\|\mathcal{G}'\left(V^{(1)}\right)-\mathcal{G}'\left(V^{(2)}\right)\right\|_\Omega \leqslant K\left\|V^{(1)}-V^{(2)}\right\|_\Omega, \text{for all } V^{(1)}, V^{(2)} \in \mathbb{V}_1 \tag{20}$$

with $h = B_0 K\eta \leqslant \frac{1}{2}$. Let

$$\mathbb{V}_2 = \left\{V \mid \left\|V - V^{(0)}\right\|_\Omega \leqslant \sigma\right\}, \text{where } \sigma = \left(\frac{1-\sqrt{1-2h}}{h}\right)\eta. \tag{21}$$

Now, if $\mathbb{V}_2 \subset \mathbb{V}_1$, then, the sequence $\left\{V^{(i)}\right\}$ given in (17) is well defined, remains in $\mathbb{V}_2$, and converges to $V^* \in \mathbb{V}_2$ such that $\mathcal{G}(V^*) = 0$. In addition,

$$\left\|V^* - V^{(i)}\right\|_\Omega \leqslant \frac{\eta}{h}\left(\frac{\left(1-\sqrt{1-2h}\right)^{2^i}}{2^i}\right), \quad i = 0, 1, 2, \cdots \tag{22}$$

It is observed from Lemma 4 that the $\mathbb{V}_1$ must be suitably chosen. The following lemma gives a method to determine a $\mathbb{V}_0$ satisfying$\mathbb{V}_0 \subset \mathbb{V}_1$, so that $\mathbb{V}_0$ conversely guarantees the hypotheses of Lemma 4.

*Lemma 5 (*[25]*)*
Suppose $V^* \geqslant 0$ is the solution of HJI equation $\mathcal{G}(V^*) = 0$. If $\left\|(\mathcal{G}'(V^*))^{-1}\right\|_\Omega \leqslant B^*$, and

$$\mathbb{V}_3 = \{V \mid \|V - V^*\|_\Omega \leqslant (1/B^*K)\} \subset \mathbb{V}_1, \tag{23}$$

then, the hypotheses of Lemma 4 is satisfied, that is, for each $V^{(0)} \in \mathbb{V}_0$, $h \leqslant 1/2$, $B_0$ and $\eta$ in conditions (18) and (19) are

$$B_0 = \frac{B^*}{1 - B^*K\left\|V^{(0)} - V^*\right\|_\Omega} \geqslant \left\|\left(\mathcal{G}'\left(V^{(0)}\right)\right)^{-1}\right\|_\Omega \text{ and}$$
$$\eta = \frac{1 - \frac{1}{2}B^*K\left\|V^{(0)} - V^*\right\|_\Omega}{1 - B^*K\left\|V^{(0)} - V^*\right\|_\Omega}\left\|V^{(0)} - V^*\right\|_\Omega \geqslant \left\|\left(\mathcal{G}'\left(V^{(0)}\right)\right)^{-1}\mathcal{G}\left(V^{(0)}\right)\right\|_\Omega,$$

where $\mathbb{V}_0$ is

$$\mathbb{V}_0 = \left\{V \mid \|V - V^*\|_\Omega \leqslant \left(2-\sqrt{2}\right)/\left(2B^*K\right)\right\}. \tag{24}$$

Lemmas 4 and 5 imply that if $V^{(0)}$ is chosen in $\mathbb{V}_0$ defined by (24), the Newton sequence $\left\{V^{(i)}\right\}$ generated by (17) can converge to the fixed point of (13), that is, the solution of HJI equation $\mathcal{G}(V^*) = 0$, and the error bound is given in (22). With Lemmas 4 and 5, we will prove that the sequence $\left\{V^{(i)}\right\}$ generated by Algorithm 2 is essentially a Newton sequence in Theorem 1.



DOI: 10.1002/rnc

*Theorem 1*
Let $\mathcal{T}$ be a mapping defined by (13). Then, the sequence $\{V^{(i)}\}$ generated by Algorithm 2 and the Newton iteration (17) are equivalent.

*Proof*
See Appendix. □

*Remark 5*
Theorem 1 shows that the sequence $\{V^{(i)}\}$ generated by Algorithm 2 is equivalent to the Newton sequence obtained by (17), the convergence of which can be guaranteed by Lemma 4. Therefore, the sequence $\{V^{(i)}\}$ also converges to the solution $V^*$ of HJI equation (4), that is, $V^{(i)} \to V^*$, when $i \to \infty$.

### *3.2. Galerkin's method for Lyapunov function equations*

Note that in Algorithm 1 and 2, one needs to solve the LFEs (6) and (10), respectively. In [10], Galerkin's approximation was used to solve the LFE (6). Because both LFEs (6) and (10) are first-order, linear PDEs, the methods for solving them are similar. For completeness, we briefly recall the Galerkin's method for the solution of LFE (10).

Galerkin's method assumes that there exist a complete set of linearly independent basis functions $\boldsymbol{\Psi}(\boldsymbol{x}) = \{\psi_k(\boldsymbol{x})\}_{k=1}^{\infty}$ such that $\psi_k(0) = 0, \forall k$, and for the LFE (10), its solution can be expressed as a linear combination of basis functions $\boldsymbol{\Psi}(\boldsymbol{x})$, that is, $V^{(i+1)}(\boldsymbol{x}) = \sum_{k=1}^{\infty} c_k^{(i+1)} \psi_k(\boldsymbol{x})$, where the sum is assumed to converge pointwise in $\Omega$. An approximation of $V^{(i+1)}(\boldsymbol{x})$can be obtained by truncating the series to

$$\hat{V}^{(i+1)}(\boldsymbol{x}) = \left(\boldsymbol{c}^{(i+1)}\right)^T \boldsymbol{\Psi}_N(\boldsymbol{x}) = \boldsymbol{\Psi}_N^T(\boldsymbol{x})\boldsymbol{c}^{(i+1)}. \tag{25}$$

where $\boldsymbol{c}^{(i+1)} = \left(c_1^{(i+1)}, \cdots, c_N^{(i+1)}\right)^T$, $\boldsymbol{\Psi}_N(\boldsymbol{x}) = (\psi_1, \cdots, \psi_N)^T$. Then, the coefficients $\boldsymbol{c}^{(i+1)}$ are computed by solving the following algebraic equation:

$$\int_{\Omega} \left( \left( \frac{\partial \hat{V}^{(i+1)}}{\partial \boldsymbol{x}} \right)^T \left( \boldsymbol{f} + \boldsymbol{g}\hat{\boldsymbol{u}}^{(i)} + \boldsymbol{k}\hat{\boldsymbol{w}}^{(i)} \right) + \boldsymbol{h}^T\boldsymbol{h} + \left(\hat{\boldsymbol{u}}^{(i)}\right)^T \boldsymbol{R}\hat{\boldsymbol{u}}^{(i)} - \gamma^2 \left(\hat{\boldsymbol{w}}^{(i)}\right)^T \hat{\boldsymbol{w}}^{(i)} \right) \psi_k(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 0, \quad k = 1, \cdots, N, \tag{26}$$

that is,

$$\boldsymbol{Z}^{(i)}\boldsymbol{c}^{(i+1)} + \boldsymbol{s}^{(i)} = 0 \tag{27}$$

where

$$\boldsymbol{Z}^{(i)} = \boldsymbol{Z}_1 + \boldsymbol{Z}_2^{(i)}, \quad \boldsymbol{Z}_1 = \int_{\Omega} \boldsymbol{\Psi}_N \boldsymbol{f}^T \nabla \boldsymbol{\Psi}_N^T \mathrm{d}\boldsymbol{x}, \quad \boldsymbol{Z}_2^{(i)} = \int_{\Omega} \boldsymbol{\Psi}_N \left( \boldsymbol{g}\hat{\boldsymbol{u}}^{(i)} + \boldsymbol{k}\hat{\boldsymbol{w}}^{(i)} \right)^T \nabla \boldsymbol{\Psi}_N^T \mathrm{d}\boldsymbol{x} \tag{28}$$

$$\boldsymbol{s}^{(i)} = \boldsymbol{s}_1 + \boldsymbol{s}_2^{(i)}, \quad \boldsymbol{s}_1 = \int_{\Omega} \boldsymbol{\Psi}_N \boldsymbol{h}^T \boldsymbol{h} \mathrm{d}\boldsymbol{x}, \quad \boldsymbol{s}_2^{(i)} = \int_{\Omega} \boldsymbol{\Psi}_N \left( \hat{\boldsymbol{u}}^{(i)T} \boldsymbol{R}\hat{\boldsymbol{u}}^{(i)} - \gamma^2 \hat{\boldsymbol{w}}^{(i)T} \hat{\boldsymbol{w}}^{(i)} \right) \mathrm{d}\boldsymbol{x} \tag{29}$$

in which $\nabla \boldsymbol{\Psi}_N(\boldsymbol{x}) = ((\partial\psi_1/\partial\boldsymbol{x}), \cdots, (\partial\psi_N/\partial\boldsymbol{x}))^T$ is the Jacobian of $\boldsymbol{\Psi}_N$. From (27), the coefficients $\boldsymbol{c}^{(i+1)}$ can be computed with

$$\boldsymbol{c}^{(i+1)} = -\left(\boldsymbol{Z}^{(i)}\right)^{-1} \boldsymbol{s}^{(i)}, \tag{30}$$

where $\boldsymbol{Z}^{(i)}$ is invertible [10]. Then, the solution of LFE (10) is obtained by (25).



DOI: 10.1002/rnc

*Remark 6*
From the well-known high-order Weierstrass approximation theorem [26], it follows that a continuous function can be uniformly approximated to any degree of accuracy by a complete set of linear independent basis functions. This means that the solution of HJI equation can be uniformly approximated to any degree of accuracy by any infinite-dimensional linear independent basis function set $\boldsymbol{\Psi}(\boldsymbol{x}) = \{\psi_k(\boldsymbol{x})\}_{k=1}^{\infty}$. For the truncated case (25), the selections of finite-dimensional basis function set $\boldsymbol{\Psi}_N(\boldsymbol{x})$ and its size are often experience-based, which can affect the convergence of HJI equation. Like all function approximation techniques, large size of the basis function set can improve the convergence at the price of highly computational effort. Thus, an appropriate selection of a basis function set and its size are useful to balance the convergence and computation. However, it is still difficult to develop a general optimal selection method for all systems. This is simply because the optimal selection is often different for different systems. In a word, for a specific system, prior experiences would be helpful for the selection of basis function set and its size.

Notice that there does not exist any fundamental difference for solving LFEs (6) and (10) with Galerkin's method. The convergence of the Galerkin approximation for solving LFEs was established mathematically in [10]. Here, we introduce this result briefly without deduction.

*Lemma 6 (*[10]*)*
If the conditions 1–6 in [10] hold, then $\left\| V^{(i+1)}(\boldsymbol{x}) - \hat{V}^{(i+1)}(\boldsymbol{x}) \right\|_{\Omega} \to 0$, when $N \to \infty$.

### *3.3. Galerkin simultaneous policy update algorithm and convergence*

By using Galerkin's method to solve LFEs (10) in each iteration, we can derive a Galerkin SPUA (GSPUA). With (25), the control policy (11) and disturbance policy (12) can be rewritten as

$$\hat{\boldsymbol{u}}^{(i+1)} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial \hat{V}^{(i+1)}}{\partial \boldsymbol{x}} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\nabla\boldsymbol{\Psi}_N^T\boldsymbol{c}^{(i+1)} \tag{31}$$

$$\hat{\boldsymbol{w}}^{(i+1)} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\frac{\partial \hat{V}^{(i+1)}}{\partial \boldsymbol{x}} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\nabla\boldsymbol{\Psi}_N^T\boldsymbol{c}^{(i+1)}. \tag{32}$$

Then, the GSPUA for the solution of HJI equation is presented as follows.

**Algorithm 3 (GSPUA)**

*Step 1:* Select an independent basis function set $\boldsymbol{\Psi}_N(\boldsymbol{x})$, and evaluate $\boldsymbol{Z}_1$ and $\boldsymbol{s}_1$. Give initial coefficients $\boldsymbol{c}^{(0)}$ such that $\hat{V}^{(0)} \in \mathbb{V}_0$. Let $\hat{\boldsymbol{u}}^{(0)} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\nabla\boldsymbol{\Psi}_N^T\boldsymbol{c}^{(0)}$, $\hat{\boldsymbol{w}}^{(0)} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\nabla\boldsymbol{\Psi}_N^T\boldsymbol{c}^{(0)}$, and $i = 0$.

*Step 2:* Compute $\boldsymbol{Z}^{(i)}$ and $\boldsymbol{s}^{(i)}$ according to (28) and (29), and then solve the equation (30) for $\boldsymbol{c}^{(i+1)}$.

*Step 3:* Update the control and disturbance policies $\hat{\boldsymbol{u}}^{(i+1)}$ and $\hat{\boldsymbol{w}}^{(i+1)}$ with (31) and (32).

*Step 4:* Set $i = i + 1$. If $\left\|\boldsymbol{c}^{(i)} - \boldsymbol{c}^{(i-1)}\right\| \leqslant \varepsilon$ ($\varepsilon$ is a small positive real number), stop and use $\boldsymbol{c}^{(i)}$ as the coefficients of $V^*$ $\left(\text{i.e., } V^* \text{ is approximated expressed with } V^* = \left(\boldsymbol{c}^{(i)}\right)^T\boldsymbol{\Psi}_N\right)$, else, go back to step 2 and continue.

The convergence of GSPUA is guaranteed by the following theorem.

*Theorem 2*
Suppose $V^*(\boldsymbol{x}) \geqslant 0$ is the solution of the HJI equation (4). Then, for $\forall\zeta > 0$, there exist positive integers $I$ and $\bar{N}$, such that when $i > I, N > \bar{N}$,

$$\left\| V^*(\boldsymbol{x}) - \hat{V}^{(i)}(\boldsymbol{x}) \right\|_{\Omega} < \zeta. \tag{33}$$

*Proof*
See Appendix. □



DOI: 10.1002/rnc

### 3.4. A computationally efficient Galerkin simultaneous policy update algorithm

To solve the (30) for coefficients $\boldsymbol{c}^{(i+1)}$ in the GSPUA, we need to compute integrals $\boldsymbol{Z}^{(i)}$ and $\boldsymbol{s}^{(i)}$ in each iterative step, which are often time-consuming and will increase exponentially with the increase of the number of basis functions and the dimension of system states. In this subsection, we derive a CESPUA, which improves the efficiency from two aspects: (1) we derive a method that evaluates all integrals once for all, which is to reduce the number of integral evaluations, and (2) we use Monte Carlo method together with the Latin hypercube sampling (LHS) to evaluate integrals, which is to improve the efficiency of integral evaluations. Now, we discuss the first one.

We rewrite $\hat{\boldsymbol{u}}^{(i)}$ and $\hat{\boldsymbol{w}}^{(i)}$ as

$$\hat{\boldsymbol{u}}^{(i)} = -\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T\left(\sum_{k=1}^{N} c_k^{(i)}\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right) = -\frac{1}{2}\sum_{k=1}^{N} c_k^{(i)}\left(\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right) \tag{34}$$

$$\hat{\boldsymbol{w}}^{(i)} = \frac{1}{2}\gamma^{-2}\boldsymbol{k}^T\left(\sum_{k=1}^{N} c_k^{(i)}\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right) = \frac{1}{2}\gamma^{-2}\sum_{k=1}^{N} c_k^{(i)}\left(\boldsymbol{k}^T\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right). \tag{35}$$

Substituting (34) and (35) into $\boldsymbol{Z}_2^{(i)}$ in (28), yields

$$\begin{aligned}
\boldsymbol{Z}_2^{(i)} &= \int_\Omega \boldsymbol{\Psi}_N\left(\boldsymbol{g}\hat{\boldsymbol{u}}^{(i)} + \boldsymbol{k}\hat{\boldsymbol{w}}^{(i)}\right)^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x}\\
&= \int_\Omega \boldsymbol{\Psi}_N\left(\boldsymbol{g}\left(-\frac{1}{2}\sum_{k=1}^{N} c_k^{(i)}\left(\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right)\right) + \boldsymbol{k}\left(\frac{1}{2}\gamma^{-2}\sum_{k=1}^{N} c_k^{(i)}\left(\boldsymbol{k}^T\frac{\partial\psi_k}{\partial\boldsymbol{x}}\right)\right)\right)^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x}\\
&= -\frac{1}{2}\sum_{k=1}^{N} c_k^{(i)}\int_\Omega \boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x} + \frac{1}{2}\gamma^{-2}\sum_{k=1}^{N} c_k^{(i)}\int_\Omega \boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x}.
\end{aligned} \tag{36}$$

Let us consider

$$\begin{aligned}
\boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\nabla\boldsymbol{\Psi}_N^T &= \begin{bmatrix}\psi_1\\ \vdots\\ \psi_N\end{bmatrix}\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\begin{bmatrix}\frac{\partial\psi_1}{\partial\boldsymbol{x}} & \cdots & \frac{\partial\psi_N}{\partial\boldsymbol{x}}\end{bmatrix}\\
&= \begin{bmatrix}
\psi_1\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_1}{\partial\boldsymbol{x}} & \cdots & \psi_1\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_l}{\partial\boldsymbol{x}} & \cdots & \psi_1\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_N}{\partial\boldsymbol{x}}\\
\vdots & \ddots & \vdots & \iddots & \vdots\\
\psi_j\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_1}{\partial\boldsymbol{x}} & \cdots & \psi_j\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_l}{\partial\boldsymbol{x}} & \cdots & \psi_j\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_N}{\partial\boldsymbol{x}}\\
\vdots & \iddots & \vdots & \ddots & \vdots\\
\psi_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_1}{\partial\boldsymbol{x}} & \cdots & \psi_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_l}{\partial\boldsymbol{x}} & \cdots & \psi_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial\psi_N}{\partial\boldsymbol{x}}
\end{bmatrix}.
\end{aligned}$$

A similar expression can be derived for $\boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\nabla\boldsymbol{\Psi}_N^T$. Thus, we write (36) as

$$\begin{aligned}
\boldsymbol{Z}_2^{(i)} &= -\frac{1}{2}\sum_{k=1}^{N} c_k^{(i)}\boldsymbol{X}_k + \frac{1}{2}\gamma^{-2}\sum_{k=1}^{N} c_k^{(i)}\boldsymbol{Y}_k\\
&= \frac{1}{2}\sum_{k=1}^{N} c_k^{(i)}\left(-\boldsymbol{X}_k + \gamma^{-2}\boldsymbol{Y}_k\right),
\end{aligned} \tag{37}$$

where

$$\boldsymbol{X}_k = \int_\Omega \boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x}, \tag{38}$$

$$\boldsymbol{Y}_k = \int_\Omega \boldsymbol{\Psi}_N\frac{\partial\psi_k^T}{\partial\boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\nabla\boldsymbol{\Psi}_N^T\mathrm{d}\boldsymbol{x}. \tag{39}$$



DOI: 10.1002/rnc

$$\boldsymbol{X}_k = (X_{k(j,l)})_{N\times N} \in \mathbb{R}^{N\times N}, \quad \boldsymbol{Y}_k = (Y_{k(j,l)})_{N\times N} \in \mathbb{R}^{N\times N}, \quad k = 1, \cdots, N,$$

$X_{k(j,l)} = \int_\Omega \psi_j \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T \frac{\partial \psi_l}{\partial \boldsymbol{x}} \mathrm{d}\boldsymbol{x}$ and $Y_{k(j,l)} = \int_\Omega \psi_j \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{k}\boldsymbol{k}^T \frac{\partial \psi_l}{\partial \boldsymbol{x}} \mathrm{d}\boldsymbol{x}$. Similarly, the $\boldsymbol{s}_2^{(i)}$ in (29) can be written as

$$\begin{aligned}
\boldsymbol{s}_2^{(i)} &= \int_\Omega \boldsymbol{\Psi}_N \left(-\frac{1}{2}\sum\nolimits_{k=1}^N c_k^{(i)} \left(\boldsymbol{R}^{-1}\boldsymbol{g}^T \frac{\partial \psi_k}{\partial \boldsymbol{x}}\right)\right)^T \boldsymbol{R} \left(-\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^T \nabla \boldsymbol{\Psi}_N^T \boldsymbol{c}^{(i)}\right) \mathrm{d}\boldsymbol{x} \\
&\quad - \gamma^2 \int_\Omega \boldsymbol{\Psi}_N \left(\frac{1}{2}\gamma^{-2}\sum\nolimits_{k=1}^N c_k^{(i)} \left(\boldsymbol{k}^T \frac{\partial \psi_k}{\partial \boldsymbol{x}}\right)\right)^T \left(\frac{1}{2}\gamma^{-2}\boldsymbol{k}^T \nabla \boldsymbol{\Psi}_N^T \boldsymbol{c}^{(i)}\right) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{4}\sum\nolimits_{k=1}^N c_k^{(i)} \int_\Omega \boldsymbol{\Psi}_N \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T \nabla \boldsymbol{\Psi}_N^T \boldsymbol{c}^{(i)} \mathrm{d}\boldsymbol{x} - \frac{1}{4}\gamma^{-2}\sum\nolimits_{k=1}^N c_k^{(i)} \int_\Omega \boldsymbol{\Psi}_N \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{k}\boldsymbol{k}^T \nabla \boldsymbol{\Psi}_N^T \boldsymbol{c}^{(i)} \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{4}\sum\nolimits_{k=1}^N c_k^{(i)} \left(\int_\Omega \boldsymbol{\Psi}_N \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T \nabla \boldsymbol{\Psi}_N^T \mathrm{d}\boldsymbol{x}\right) \boldsymbol{c}^{(i)} \\
&\quad - \frac{1}{4}\gamma^{-2}\sum\nolimits_{k=1}^N c_k^{(i)} \left(\int_\Omega \boldsymbol{\Psi}_N \frac{\partial \psi_k^T}{\partial \boldsymbol{x}} \boldsymbol{k}\boldsymbol{k}^T \nabla \boldsymbol{\Psi}_N^T \mathrm{d}\boldsymbol{x}\right) \boldsymbol{c}^{(i)} \\
&= \frac{1}{4}\left(\sum\nolimits_{k=1}^N c_k^{(i)} \left(\boldsymbol{X}_k - \gamma^{-2}\boldsymbol{Y}_k\right)\right) \boldsymbol{c}^{(i)}. \qquad (40)
\end{aligned}$$

The substitution of (37) and (40) into (30), yields

$$\boldsymbol{c}^{(i+1)} = -\left(\boldsymbol{Z}_1 - \frac{1}{2}\sum\nolimits_{k=1}^N c_k^{(i)} \left(\boldsymbol{X}_k - \gamma^{-2}\boldsymbol{Y}_k\right)\right)^{-1} \left(\boldsymbol{s}_1 + \frac{1}{4}\left(\sum\nolimits_{k=1}^N c_k^{(i)} \left(\boldsymbol{X}_k - \gamma^{-2}\boldsymbol{Y}_k\right)\right) \boldsymbol{c}^{(i)}\right). \qquad (41)$$

*Remark 7*

From (41), it is observed that $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ $(k = 1, \cdots, N)$ are invariant in each iteration. They can be computed once for all, thus, we need not update $\boldsymbol{Z}^{(i)}$ and $\boldsymbol{s}^{(i)}$ in each iteration as in Algorithm 3.

Now, we use Monte Carlo method together with LHS to compute integrals $\boldsymbol{Z}_1$, $\boldsymbol{s}_1$, $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ $(k = 1, \cdots, N)$. Monte Carlo integration is usually employed to estimate integrals over multidimensional domains. It is especially competitive in high dimensional domains. LHS is a uniform sampling method that has small variance, which was first proposed as a Monte Carlo integration technique by McKay *et al.* [27]. Here, we use $\boldsymbol{X}_k$ as an example to illustrate this procedure. First, select $H$ samples $\boldsymbol{x}_i \in \Omega$ $(i = 1, \cdots, H)$ with LHS. Then, evaluate $\boldsymbol{X}_k$ with

$$\boldsymbol{X}_k = \frac{1}{H}\sum_{i=1}^H \boldsymbol{\Psi}_N(\boldsymbol{x}_i) \frac{\partial \psi_k^T(\boldsymbol{x}_i)}{\partial \boldsymbol{x}} \boldsymbol{g}(\boldsymbol{x}_i) \boldsymbol{R}^{-1}\boldsymbol{g}^T(\boldsymbol{x}_i) \nabla \boldsymbol{\Psi}_N^T(\boldsymbol{x}_i) \quad . \qquad (42)$$

On the basis of (41) and (42), we derive the CESPUA.

**Algorithm 4 (CESPUA).**

*Step 1:* Select an independent basis function set $\boldsymbol{\Psi}_N(\boldsymbol{x})$, compute $\boldsymbol{Z}_1, \boldsymbol{s}_1, \boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ $(k = 1, \mathrm{L}, N)$ with the above Monte Carlo integration. Give initial coefficients $\boldsymbol{c}^{(0)}$ such that $\hat{V}^{(0)} \in \mathbb{V}_0$. Let $i = 0$.

*Step 2:* Solve the equation (41) for $\boldsymbol{c}^{(i+1)}$.

*Step 3:* Set $i = i + 1$. If $\left\|\boldsymbol{c}^{(i)} - \boldsymbol{c}^{(i-1)}\right\| \leq \varepsilon$ ($\varepsilon$ is a small positive real number), stop and use $\boldsymbol{c}^{(i)}$ as the coefficients of $V^*$ $\left(\text{i.e., } V^* = \left(\boldsymbol{c}^{(i)}\right)^T \boldsymbol{\Psi}_N\right)$, else, go back to step 2 and continue.

*Remark 8*

It is observed that the CESPUA is the SPUA (i.e., Algorithm 2) by using the developed efficient Galerkin's method to solve LEF (10) in each iteration. In CESPUA, as far as the basis functions are



DOI: 10.1002/rnc

selected, the integrals $\boldsymbol{Z}_1$, $\boldsymbol{s}_1$, $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ $(k = 1, \cdots, N)$ can be precomputed directly and remain invariant during the iterative procedure. Thus, it is very convenient for offline controller design.

## 4. COMPARATIVE SIMULATION STUDIES

In this section, we give comparative simulation studies between CESPUA (i.e., Algorithm 4) and Algorithm 1 (with the efficient Galerkin's method) on three examples. Because the analytic solution of HJI equation is often not available, we first compare CESPUA and Algorithm 1 on a linear system to illustrate their validity, and then apply them to a simple nonlinear system and a rotational/translational actuator (RTAC) nonlinear benchmark problem.

*Remark 9*

For fairness reason, the method for solving LFEs (6) and (10) should be the same. Moreover, by comparing (6) and (10), we observe that they have the same form. Thus, if the same method is employed to solve (6) and (10), the time required for one iterative step in Algorithm 1 and 2 is also the same. Therefore, it is equivalent to compare the efficiency of two algorithms by observing the iterative steps they needed. Here, we use the efficient Galerkin's method (described in Subsections 3.2 and 3.4) to solve (6) and (10). Of course, as mentioned in Remark 3, other methods can also be used to solve (6) and (10). For example, using neural network (as in [16]), our simulation results show that the proposed SPUA still performs better than Algorithm 1, which are omitted here.

### *4.1. Simulations on linear system*

We consider the $H_\infty$ control problem of a power system in [28] described by the following linear state-space model:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}_1 w + \boldsymbol{B}_2 u, \quad \boldsymbol{z} = \boldsymbol{C}\boldsymbol{x} \tag{43}$$

where

$$\boldsymbol{A} = \begin{bmatrix} -0.0665 & 8 & 0 & 0 \\ 0 & -3.663 & 3.663 & 0 \\ -6.86 & 0 & -13.736 & -13.736 \\ 0.6 & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{B}_1 = \begin{bmatrix} -8 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{B}_2 = \begin{bmatrix} 0 \\ 0 \\ 13.736 \\ 0 \end{bmatrix}, \quad \boldsymbol{C} = \boldsymbol{I}.$$

Select $\boldsymbol{R} = \boldsymbol{I}$ and $\gamma = 3$. Letting $V(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P}\boldsymbol{x}$, the HJI equation (4) for the linear system (43) is the following $H_\infty$ ARE:

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{C}^T\boldsymbol{C} + \gamma^{-2}\boldsymbol{P}\boldsymbol{B}_1\boldsymbol{B}_1^T\boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}_2\boldsymbol{R}^{-1}\boldsymbol{B}_2^T\boldsymbol{P} = 0 \tag{44}$$

and the corresponding $H_\infty$ control law (5) is

$$\boldsymbol{u}^*(\boldsymbol{x}) = -\boldsymbol{R}^{-1}\boldsymbol{B}_2^T\boldsymbol{P}\boldsymbol{x}. \tag{45}$$

Solving the ARE (44) with the MATLAB (MathWorks, Inc., Natick, MA, USA) command CARE, we obtain

$$\boldsymbol{P} = \begin{bmatrix} 1.2606 & 1.5616 & 0.2341 & 1.1793 \\ 1.5616 & 2.3229 & 0.3734 & 1.3223 \\ 0.2341 & 0.3734 & 0.0920 & 0.1782 \\ 1.1793 & 1.3223 & 0.1782 & 3.0258 \end{bmatrix}. \tag{46}$$



DOI: 10.1002/rnc

Select the state domain $\Omega$ for integrals evaluation in Galerkin's method as $\Omega = \{\boldsymbol{x} \mid -0.5 \leqslant x_i \leqslant 0.5,\ i = 1, 2, 3, 4\}$, and 10 polynomials as basis functions as follows:

$$\boldsymbol{\Psi}_{10}(\boldsymbol{x}) = \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 & x_1x_4 & x_2^2 & x_2x_3 & x_2x_4 & x_3^2 & x_3x_4 & x_4^2 \end{bmatrix}^T.$$

Then, the true values of coefficients $\boldsymbol{c}$ are

$$\begin{aligned} \boldsymbol{c} &= \begin{bmatrix} P_{11} & 2P_{12} & 2P_{13} & 2P_{14} & P_{22} & 2P_{23} & 2P_{24} & P_{33} & 2P_{34} & P_{44} \end{bmatrix}^T \\ &= \begin{bmatrix} 1.2606 & 3.1232 & 0.4682 & 2.3586 & 2.3229 & 0.7468 & 2.6446 & 0.0920 & 0.3564 & 3.0258 \end{bmatrix}^T. \end{aligned} \tag{47}$$

The value of $\varepsilon_1$ in Algorithm 1 is selected as $10^{-5}$, and the value of stop criterion $\varepsilon$ in Algorithm 1 and CESPUA is set to be $10^{-7}$. Select the initial coefficients as $\boldsymbol{c}^{(0)} = 0$.

*Remark 10*
As mentioned in Remark 6, the selection of basis function set and its size are often experience-based. However, for a linear system, the basis function set is known because the solution of HJI equation is $V^*(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P} \boldsymbol{x}$, which can be accurately represented with polynomial basis function set $\boldsymbol{\Psi}_N = \{x_i x_j, i, j = 1, \ldots, n\}$ via (25).

We first use Algorithm 1 to solve the $H_\infty$ control problem under the above settings. However, the simulation results show that the coefficients are divergent. To test the ability of Algorithm 1, we gradually increase the value of $\gamma$ and run Algorithm 1 in 'PC' and 'FS' modes again, where the symbol 'PC' means that Algorithm 1 terminates when *precisions* $\varepsilon_1$ and $\varepsilon$ are attained, and 'FS' means that *fixed iteration steps*: 100 are used for both the inner and outer loops of Algorithm 1. Table I gives the simulation results obtained by Algorithm 1 for different values of $\gamma$, where the symbol —' denotes divergence of the iteration. From Table I, we find that Algorithm 1 is not convergent until $\gamma = 12$ for 'PC' mode, and $\gamma = 8$ for 'FS' mode. Figure 2 gives coefficients $\boldsymbol{c}^{(i)}$ at each iterative step obtained by Algorithm 1 in 'PC' case with $\gamma = 12$, where the dash lines represent the true values of coefficients $\boldsymbol{c}$.

*Remark 11*
It is observed from Table I that Algorithm 1 in 'FS' mode converges at a lower value of $\gamma$ than in 'PC' mode. The reason for this is that when the inner loop of Algorithm 1 cannot yield a convergent solution with precision of $\varepsilon_1$, the algorithm in 'PC' mode, cannot terminate the inner loop and thus results in divergence, while in 'FS' mode, by terminating forcibly the inner loop after fixed steps, it is likely to obtain a convergent solution in the outer loop in the end. Generally, Algorithm 1 is bound to converge in 'FS' mode if it is convergent in 'PC' mode, and the results in Table I demonstrate this fact.

The proposed CESPUA is also used to solve the $H_\infty$ control problem (with $\gamma = 3$). Figure 3 shows coefficients $\boldsymbol{c}^{(i)}$ in each iterative step, where the dashed lines represent the true values of coefficients $\boldsymbol{c}$ (i.e., (47)). It is observed that Algorithm 4 can find the coefficients of precision $\varepsilon$ at the $13^{\text{th}}$ iterative step.

Obviously, the results demonstrate that the convergence of CESPUA is much better than Algorithm 1.

Table I. Simulation results obtained by Algorithm 1.

| $\gamma$ | 3 | 4 | 6 | 8 | 10 | 12 | 14 |
|---|---|---|---|---|---|---|---|
| PC | — | — | — | — | — | 529 | 489 |
| FS | — | — | — | 926 | 725 | 529 | 489 |



DOI: 10.1002/rnc

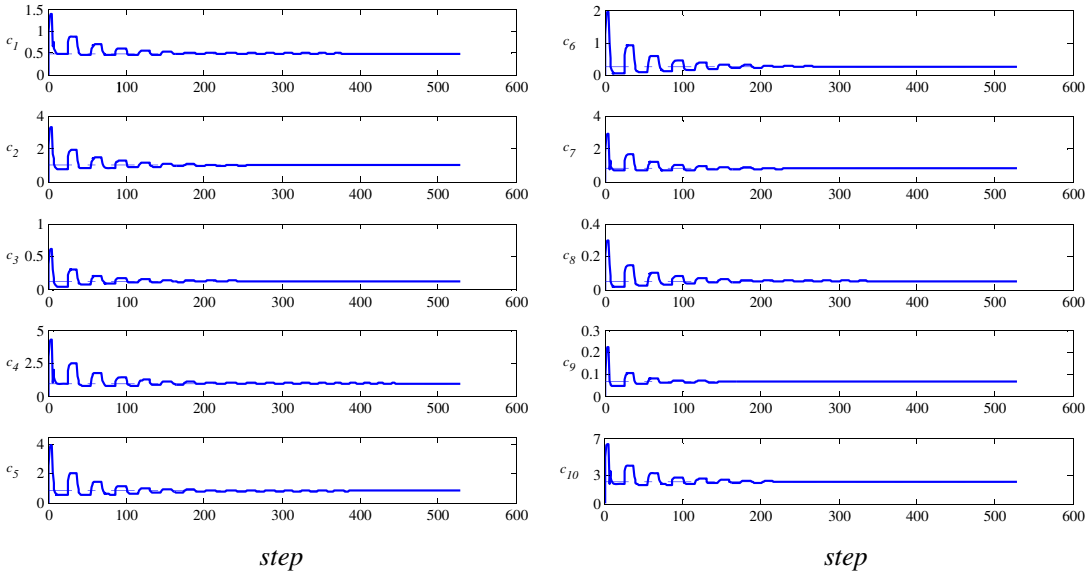



Figure 2. Coefficients (with$\gamma = 12$) obtained by Algorithm 1 in 'PC' mode.

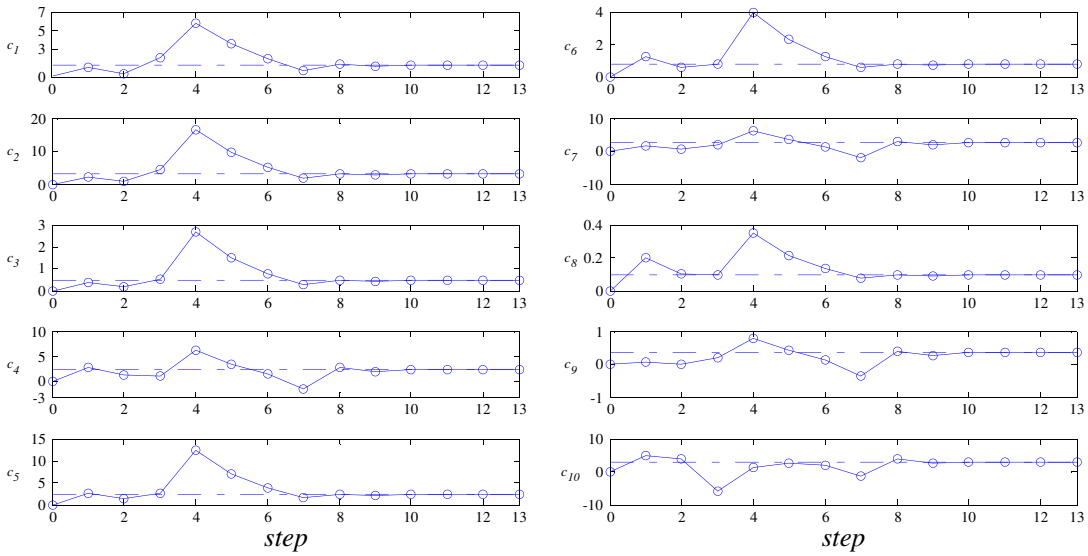



Figure 3. Coefficients (with $\gamma = 3$) obtained by CESPUA.

### *4.2. Simulations on nonlinear system*

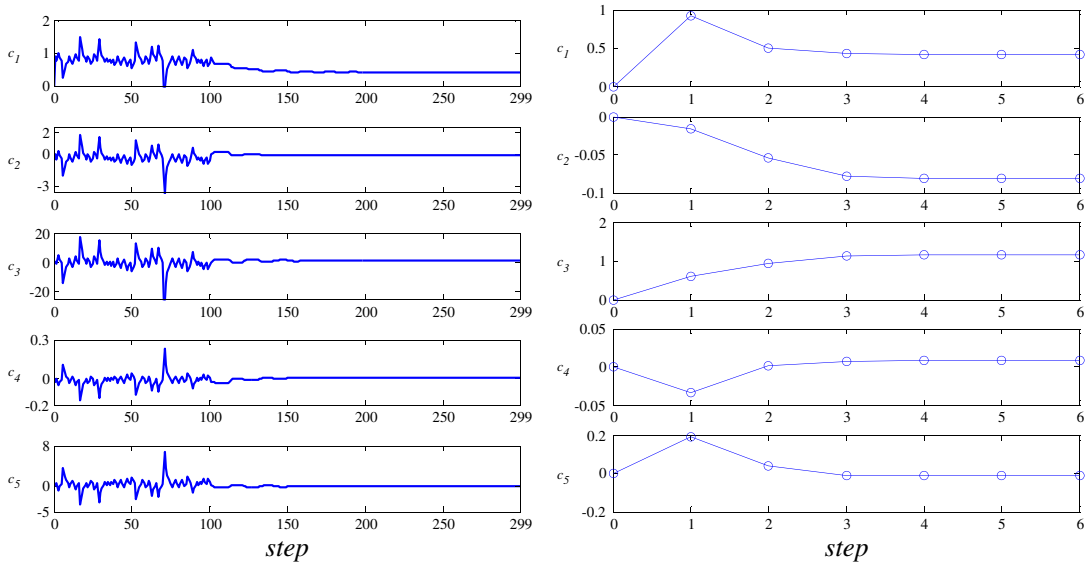
This section considers a revised version of the example in [29]. The system model is given as follows:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} -x_1 + x_2 \\ -0.5(x_1 + x_2) + 0.5x_2 \sin^2(x_1) \end{bmatrix} + \begin{bmatrix} 0 \\ \sin(x_1) \end{bmatrix} w + \begin{bmatrix} 0 \\ \cos(x_1) \end{bmatrix} u$$
$$\boldsymbol{z} = \boldsymbol{x}.$$

Select the state domain $\Omega$ for integrals evaluation in Galerkin's method as $\Omega = \{\boldsymbol{x} \mid -2 \leqslant x_i \leqslant 2,\ i = 1, 2\}$, the weight matrix $\boldsymbol{R} = \boldsymbol{I}$ and $\gamma = 2$. We select five polynomials as basis functions $\boldsymbol{\Psi}_5(\boldsymbol{x}) = \begin{bmatrix} x_1^2 & x_1x_2 & x_2^2 & x_1^4 & x_2^4 \end{bmatrix}^T$, and the initial coefficients $\boldsymbol{c}^{(0)} = 0$. The value of $\varepsilon_1$ in Algorithm 1 is selected as $10^{-3}$, and the value of stop criterion $\varepsilon$ in Algorithm 1 and CESPUA is set as $10^{-7}$.

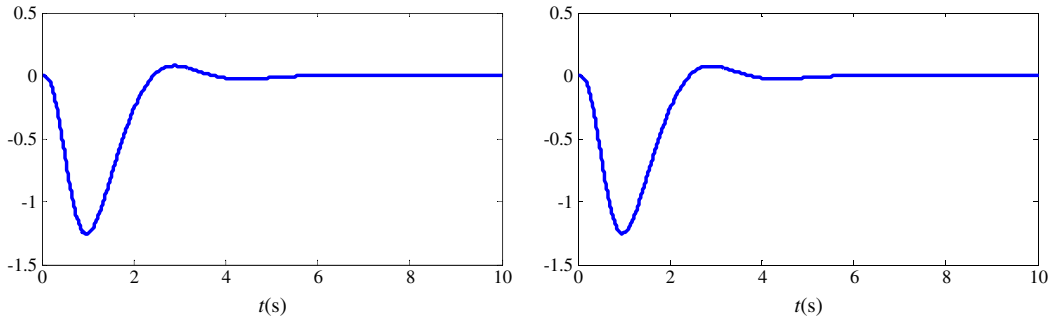


DOI: 10.1002/rnc




(a) Algorithm 1 (b) CESPUA

Figure 4. Coefficients obtained by Algorithm 1 and CESPUA.




(a) Algorithm 1 (b) CESPUA

Figure 5. Control actions corresponding to Algorithm 1 and CESPUA.

We use Algorithm 1 and CESPUA to solve the nonlinear $H_\infty$ control problem, respectively. Figure 4 shows the coefficients $\boldsymbol{c}^{(i)}$ in each iterative step. It is observed that CESPUA converges much more quickly than Algorithm 1. Algorithm 1 terminates at the 299$^{\text{th}}$ iterative step, and the coefficients are $\begin{bmatrix} 0.4129 & -0.0819 & 1.1554 & 0.0090 & -0.0163 \end{bmatrix}$. CESPUA terminates at the 6$^{\text{th}}$ iterative step, and the coefficients are $\begin{bmatrix} 0.4163 & -0.0812 & 1.1452 & 0.0086 & -0.0149 \end{bmatrix}$.

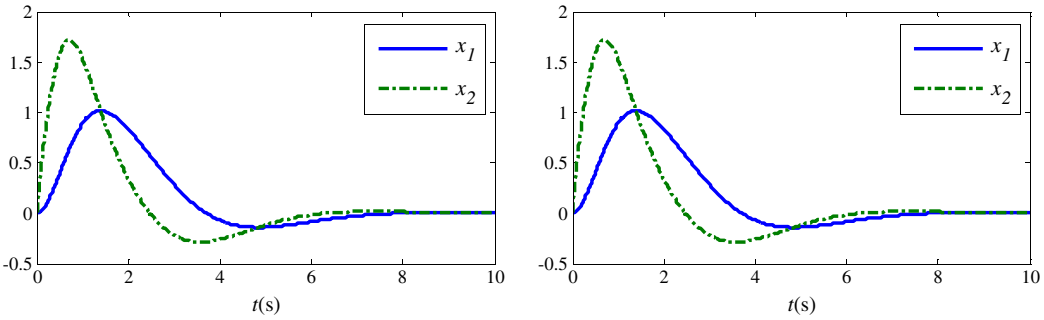
With the coefficients, the solution of HJI (4) is obtained by (25), and then the $H_\infty$ controller is designed by (5). Select a disturbance signal as $w(t) = 5e^{-t}\cos(t)$, and use the $H_\infty$ controller for the closed-loop system simulations. Figures 5 and 6 show the control actions and the closed-loop state trajectories corresponding to Algorithm 1 and CESPUA, respectively. It can be seen from Figure 6 that the closed-loop systems are asymptotically stable. Figure 7 shows the evolutions of $r(t)$ corresponding to Algorithm 1 and CESPUA, where

$$r(t) \triangleq \frac{\int_0^t \left( \|z(\tau)\|^2 + \|\boldsymbol{u}(\tau)\|_{\boldsymbol{R}}^2 \right) \mathrm{d}\tau}{\int_0^t \|\boldsymbol{w}(\tau)\|^2 \, \mathrm{d}\tau}. \tag{45}$$

The $r(t)$ corresponding to Algorithm 1 and CESPUA converge to 0.6237 and 0.6235, respectively, both satisfy the $L_2$-gain requirement (i.e., $r(t) < \gamma^2 = 4$) when $t \to \infty$.

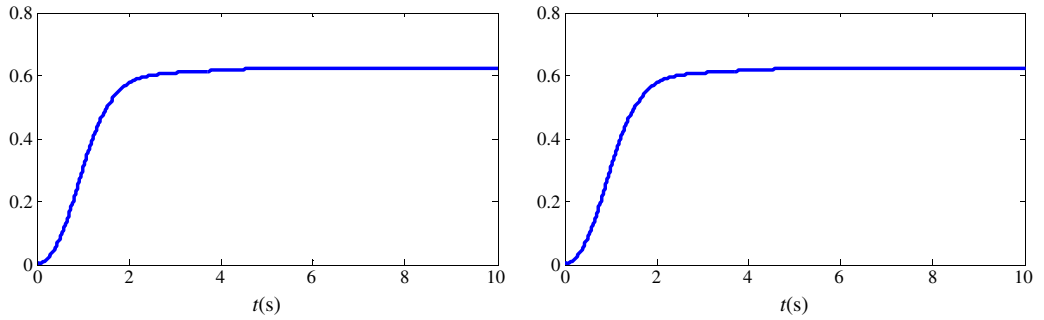


DOI: 10.1002/rnc




(a) Algorithm 1 (b) CESPUA

Figure 6. Closed-loop state trajectories corresponding to Algorithm 1 and CESPUA.




(a) Algorithm 1 (b) CESPUA

Figure 7. Evolutions of $r(t)$ corresponding to Algorithm 1 and CESPUA.

### *4.3. Application to the rotational/translational actuator nonlinear benchmark problem*

The RTAC nonlinear benchmark problem has been widely used to test the abilities of control methods [16, 30, 31]. The dynamics of this nonlinear plant poses challenges because the rotational and translation motions are coupled. The RTAC system is given as follows:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} x_2 \\ \frac{-x_1+\varsigma x_4^2 \sin x_3}{1-\varsigma^2\cos^2 x_3} \\ x_4 \\ \frac{\varsigma\cos x_3\left(x_1-\varsigma x_4^2\sin x_3\right)}{1-\varsigma^2\cos^2 x_3} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{1-\varsigma^2\cos^2 x_3} \\ 0 \\ \frac{-\varsigma\cos x_3}{1-\varsigma^2\cos^2 x_3} \end{bmatrix} w + \begin{bmatrix} 0 \\ \frac{-\varsigma\cos x_3}{1-\varsigma^2\cos^2 x_3} \\ 0 \\ \frac{1}{1-\varsigma^2\cos^2 x_3} \end{bmatrix} u$$

$$\boldsymbol{z} = \boldsymbol{x},$$

where $\varsigma = 0.2$.

Select the state domain $\Omega$ for integrals evaluation in Galerkin's method as $\Omega = \{\boldsymbol{x} | -0.5 \leqslant x_i \leqslant 0.5,\ i = 1, 2, 3, 4\}$, the weight matrix $\boldsymbol{R} = \boldsymbol{I}$ and $\gamma = 12$. We select 20 polynomials as basis functions

$$\boldsymbol{\Psi}_{20}(\boldsymbol{x}) = \begin{bmatrix} x_1^2 & x_1x_2 & x_1x_3 & x_1x_4 & x_2^2 & x_2x_3 & x_2x_4 & x_3^2 & x_3x_4 & x_4^2 \\ x_1^3x_2 & x_1^3x_3 & x_1^3x_4 & x_1^2x_2^2 & x_1^2x_2x_3 & x_1^2x_2x_4 & x_1^2x_3^2 & x_1^2x_3x_4 & x_1^2x_4^2 & x_1x_2^3 \end{bmatrix}^T,$$

and the initial coefficients

$$\boldsymbol{c}^{(0)} = \begin{bmatrix} 25.9955 & -4.9320 & -0.1281 & -5.9837 & 26.3805 & 5.7509 & 11.1264 \\ 2.5446 & 3.6809 & 3.5886 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T.$$

The value of $\varepsilon_1$ in Algorithm 1 is selected as $10^{-3}$, and the value of stop criterion $\varepsilon$ in Algorithm 1 and CESPUA is set as $10^{-7}$.



DOI: 10.1002/rnc

We first use the Algorithm 1 to solve the nonlinear $H_\infty$ control problem under the above settings, but the simulation results show that the coefficients are divergent. However, applying CESPUA to the problem, we can obtain the convergent coefficients. Figure 8 shows some representative coefficients $c_1$, $c_4$, $c_6$, $c_7$, and $c_{11}$ in each iterative step. Note that CESPUA terminates at the 6$^{\text{th}}$ iterative step, and the coefficients converge to

$$\begin{bmatrix} 19.6565 & -3.4249 & -0.6704 & -4.5848 & 19.6744 & 4.0408 & 7.7612 \\ 1.6390 & 2.5310 & 2.5962 & -2.1325 & 0.5822 & -0.5073 & 0.0522 & 0.3060 \\ 0.0255 & 0.2150 & 0.5853 & -0.0377 & -1.6824 \end{bmatrix}^T.$$

By using the above coefficients, the solution of HJI equation (4) is obtained by (25), and then the $H_\infty$ controller is designed by (5). Select a disturbance signal as $w(t) = 0.5e^{-t}\cos(t)$, and use the resulting $H_\infty$ controller for the closed-loop system simulations. Figures 10(a) and 9 demonstrate the control actions and the closed-loop state trajectories corresponding to CESPUA. It can be seen from Figure 9 that the closed-loop system is asymptotically stable. Figure 10(b) gives the evolutions of $r(t)$ corresponding to CESPUA. It is observed that the $r(t)$ converges to 15.1105 ($< \gamma^2 = 144$), which satisfies the $L_2$-gain requirement when $t \to \infty$.

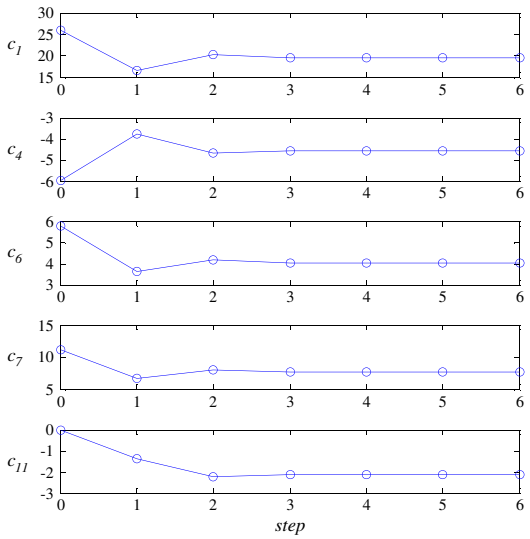



Figure 8. Coefficients $c_1$, $c_4$, $c_6$, $c_7$, and $c_{11}$ and obtained by CESPUA.

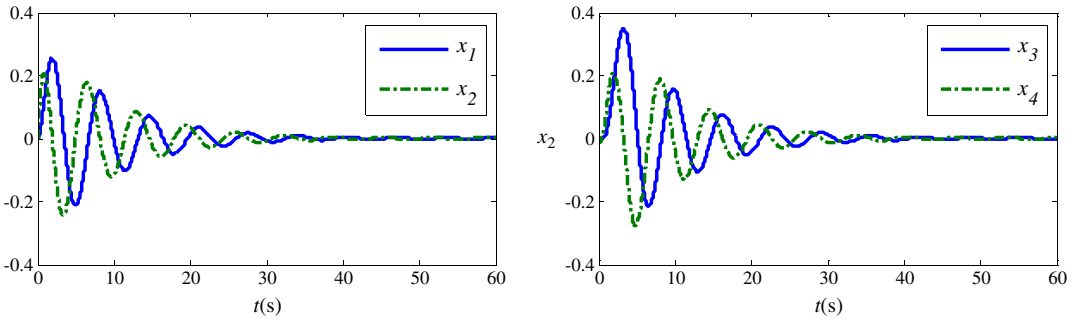



Figure 9. Closed-loop state trajectories corresponding to CESPUA.



DOI: 10.1002/rnc

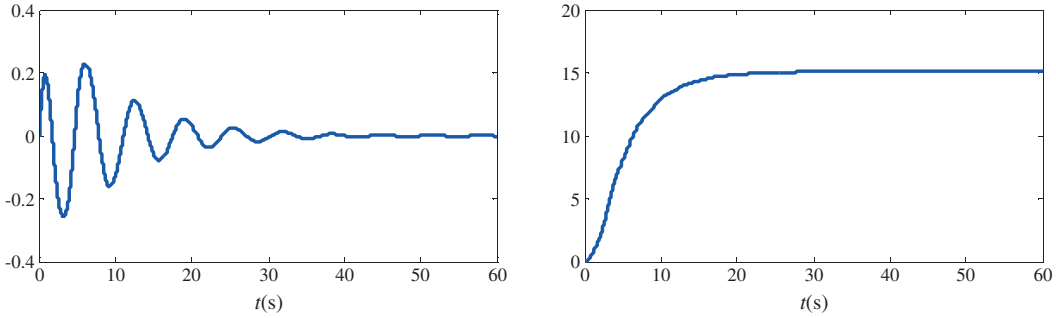



Figure 10. (a) Control action, (b) evolution of $r(t)$ corresponding to CESPUA.

## 5. CONCLUSIONS

In this paper, we have proposed a SPUA to solve the $H_\infty$ state feedback control problem for nonlinear systems, where the HJI equation is directly successively approximated with a sequence of LFEs. Different from the existing methods, the proposed SPUA updates control and disturbance policies simultaneously, which avoids solving a series of HJB equations. We discover that the SPUA is essentially a Newton's method for finding the solution of a fixed point equation, thus, its convergence is proven with the help of the Kantorovtich's Theorem. For implementation purposes, we employ Galerkin's method to solve LFEs, and derive a GSPUA. Moreover, we further improve the efficiency of GSPUA from two aspects, and provide a CESPUA. Finally, through the simulation studies on a linear power system and two nonlinear systems, the achieved results demonstrate that CESPUA is valid and efficient.

## APPENDIX

*Proof of Lemma 3*

For $\forall V \in \mathbb{V}$, we have

$$\begin{aligned}\mathcal{G}(V+sW)-\mathcal{G}(V) &= \frac{\partial (V+sW)^T}{\partial \boldsymbol{x}}\boldsymbol{f}+\boldsymbol{h}^T\boldsymbol{h}-\frac{\partial (V+sW)^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial (V+sW)}{\partial \boldsymbol{x}}\\ &\quad+\frac{1}{4\gamma^2}\frac{\partial (V+sW)^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial (V+sW)}{\partial \boldsymbol{x}}\\ &\quad-\left(\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{f}+\boldsymbol{h}^T\boldsymbol{h}-\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V}{\partial \boldsymbol{x}}+\frac{1}{4\gamma^2}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial V}{\partial \boldsymbol{x}}\right)\\ &= s\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{f}-\frac{s}{4}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V}{\partial \boldsymbol{x}}-\frac{s}{4}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial W}{\partial \boldsymbol{x}}\\ &\quad-\frac{s^2}{4}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial W}{\partial \boldsymbol{x}}+\frac{s}{4\gamma^2}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial V}{\partial \boldsymbol{x}}+\frac{s}{4\gamma^2}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial W}{\partial \boldsymbol{x}}\\ &\quad+\frac{s^2}{4\gamma^2}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial W}{\partial \boldsymbol{x}}.\end{aligned}$$

Thus, the Gâteaux differential at $V$ is

$$\begin{aligned}\mathcal{L}(W) &= \lim_{s\to 0}\frac{\mathcal{G}(V+sW)-\mathcal{G}(V)}{s}\\ &= \frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{f}-\frac{1}{4}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V}{\partial \boldsymbol{x}}-\frac{1}{4}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial W}{\partial \boldsymbol{x}}+\frac{1}{4\gamma^2}\frac{\partial W^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial V}{\partial \boldsymbol{x}}\\ &\quad+\frac{1}{4\gamma^2}\frac{\partial V^T}{\partial \boldsymbol{x}}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial W}{\partial \boldsymbol{x}}.\end{aligned}\tag{A1}$$



DOI: 10.1002/rnc

Next, we will prove that the map $\mathcal{L} = \mathcal{G}'(V)$ is continuous. For $\forall W_0 \in \mathbb{V}$, it is immediate that

$$\mathcal{L}(W) - \mathcal{L}(W_0) = \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{f} - \frac{1}{4} \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial V}{\partial \boldsymbol{x}} - \frac{1}{4} \frac{\partial V^T}{\partial \boldsymbol{x}} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial (W - W_0)}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2} \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial V}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2} \frac{\partial V^T}{\partial \boldsymbol{x}} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial (W - W_0)}{\partial \boldsymbol{x}}$$

Then, we have

$$\begin{aligned}
&\| \mathcal{L}(W) - \mathcal{L}(W_0) \|_\Omega \\
&\leqslant \left\| \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{f} \right\|_\Omega + \left\| \frac{1}{4} \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega + \left\| \frac{1}{4} \frac{\partial V^T}{\partial \boldsymbol{x}} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial (W - W_0)}{\partial \boldsymbol{x}} \right\|_\Omega \\
&\quad + \left\| \frac{1}{4\gamma^2} \frac{\partial (W - W_0)^T}{\partial \boldsymbol{x}} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega + \left\| \frac{1}{4\gamma^2} \frac{\partial V^T}{\partial \boldsymbol{x}} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial (W - W_0)}{\partial \boldsymbol{x}} \right\|_\Omega \\
&= \left( \| \boldsymbol{f} \|_\Omega + \left\| \frac{1}{2} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega + \left\| \frac{1}{2\gamma^2} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega \right) \left\| \frac{\partial (W - W_0)}{\partial \boldsymbol{x}} \right\|_\Omega \\
&\leqslant \left( \| \boldsymbol{f} \|_\Omega + \left\| \frac{1}{2} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega + \left\| \frac{1}{2\gamma^2} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega \right) m_1 \| W - W_0 \|_\Omega
\end{aligned} \tag{A2}$$

where $m_1 > 0$. Let

$$M = m_1 \left( \| \boldsymbol{f} \|_\Omega + \left\| \frac{1}{2} \boldsymbol{g} \boldsymbol{R}^{-1} \boldsymbol{g}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega + \left\| \frac{1}{2\gamma^2} \boldsymbol{k} \boldsymbol{k}^T \frac{\partial V}{\partial \boldsymbol{x}} \right\|_\Omega \right),$$

then, for $\forall \varepsilon > 0$, there exists a $\delta = \varepsilon / M$, when $\| W - W_0 \|_\Omega < \delta$, we have

$$\| \mathcal{L}(W) - \mathcal{L}(W_0) \|_\Omega \leqslant M \| W - W_0 \|_\Omega < \varepsilon. \tag{A3}$$

This means $\mathcal{L} = \mathcal{G}'(V)$ is continuous on $\mathbb{V}$, thus according to Lemma 2, $\mathcal{L}(W) = \mathcal{G}'(V)W$ (i.e., (A1)) is the Fréchet differential, and $\mathcal{L} = \mathcal{G}'(V)$ is the Fréchet derivative at $V$. □

*Proof of Theorem 1*
It follows from (17) that

$$V^{(i+1)} = \mathcal{T} V^{(i)} = V^{(i)} - \left( \mathcal{G}' \left( V^{(i)} \right) \right)^{-1} \mathcal{G} \left( V^{(i)} \right),$$

which can be rewritten as

$$G' \left( V^{(i)} \right) V^{(i+1)} = G' \left( V^{(i)} \right) V^{(i)} - G \left( V^{(i)} \right). \tag{A4}$$



DOI: 10.1002/rnc

From (10)–(12) and (16), we have

$$\begin{aligned}\mathcal{G}'(V^{(i)})V^{(i+1)} &= \left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} - \frac{1}{4}\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} - \frac{1}{4}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}} \\ &\quad + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}} \\ &= \left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} - \frac{1}{4}\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} \\ &\quad - \left(\frac{1}{4}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \\ &\quad + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\left(\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \\ &= \left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f} - \frac{1}{2}\boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} + \frac{1}{2\gamma^2}\boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right) \\ &= \left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f} + \boldsymbol{g}\boldsymbol{u}^{(i)} + \boldsymbol{k}\boldsymbol{w}^{(i)}\right), \end{aligned} \tag{A5}$$

$$\begin{aligned}\mathcal{G}'(V^{(i)})V^{(i)} &= \left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} - \frac{1}{4}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} - \frac{1}{4}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} \\ &\quad + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} \\ &= \left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} - 2\left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} + 2\gamma^2\left(\boldsymbol{w}^{(i)}\right)^T \boldsymbol{w}^{(i)}, \end{aligned} \tag{A6}$$

$$\begin{aligned}\mathcal{G}(V^{(i)}) &= \left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} + \boldsymbol{h}^T\boldsymbol{h} - \frac{1}{4}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{g}\boldsymbol{R}^{-1}\boldsymbol{g}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{k}\boldsymbol{k}^T\frac{\partial V^{(i)}}{\partial \boldsymbol{x}} \\ &= \left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} + \boldsymbol{h}^T\boldsymbol{h} - \left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} + \gamma^2\left(\boldsymbol{w}^{(i)}\right)^T \boldsymbol{w}^{(i)}. \end{aligned} \tag{A7}$$

Substituting (A5)–(A7) into (A4), yields

$$\begin{aligned}\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f} + \boldsymbol{g}\boldsymbol{u}^{(i)} + \boldsymbol{k}\boldsymbol{w}^{(i)}\right) &= \left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} - 2\left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} + 2\gamma^2\left(\boldsymbol{w}^{(i)}\right)^T \boldsymbol{w}^{(i)} \\ &\quad - \left(\left(\frac{\partial V^{(i)}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f} + \boldsymbol{h}^T\boldsymbol{h} - \left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} + \gamma^2\left(\boldsymbol{w}^{(i)}\right)^T \boldsymbol{w}^{(i)}\right)\end{aligned}$$

that is

$$\left(\frac{\partial V^{(i+1)}}{\partial \boldsymbol{x}}\right)^T \left(\boldsymbol{f} + \boldsymbol{g}\boldsymbol{u}^{(i)} + \boldsymbol{k}\boldsymbol{w}^{(i)}\right) + \boldsymbol{h}^T\boldsymbol{h} + \left(\boldsymbol{u}^{(i)}\right)^T \boldsymbol{R}\boldsymbol{u}^{(i)} - \gamma^2\left(\boldsymbol{w}^{(i)}\right)^T \boldsymbol{w}^{(i)} = 0.$$

This completes the proof. □



DOI: 10.1002/rnc

*Proof of Theorem 2*

From (22) in Lemma 4, we have that, when $i \to \infty$,

$$\left\| V^* - V^{(i)} \right\|_\Omega \leqslant \frac{\eta}{h} \left( \frac{\left(1 - \sqrt{1-2h}\right)^{2^i}}{2^i} \right) \to 0,$$

then, for $\forall \zeta > 0$, $\exists i > I$ such that

$$\left\| V^* - V^{(i)} \right\|_\Omega \leqslant \frac{\zeta}{2}. \tag{A8}$$

From Lemma 6, $\exists N > \bar{N}$ such that

$$\left\| V^{(i)} - \hat{V}^{(i)} \right\|_\Omega < \frac{\zeta}{2}. \tag{A9}$$

Then,

$$\left\| V^* - \hat{V}^{(i)} \right\|_\Omega = \left\| \left( V^* - V^{(i)} \right) + \left( V^{(i)} - \hat{V}^{(i)} \right) \right\|_\Omega \leqslant \left\| V^* - V^{(i)} \right\|_\Omega + \left\| V^{(i)} - \hat{V}^{(i)} \right\|_\Omega < \frac{\zeta}{2} + \frac{\zeta}{2} = \zeta.$$

Moreover, when $i \to \infty$ and $N \to \infty$, $\left\| V^* - \hat{V}^{(i)} \right\|_\Omega \to 0$. □

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by the National Basic Research Program of China (973 Program) (2012CB720003), in part by the National Natural Science Foundation of China under Grants 61074057, 61121003, and 91016004.

REFERENCES

1. Başar T, Bernhard P. *$H^\infty$ Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach*, 2nd edn. Birkhäuser: Boston, 1995.
2. Van Der Schaft AJ. *$L_2$-gain and Passivity Techniques in Nonlinear Control*. Springer-Verlag: Berlin, 1996.
3. Zhou K, Doyle JC, Glover K. *Robust and Optimal Control*. Prentice Hall Upper Saddle River: NJ, 1996.
4. Isidori A, Astolfi A. Disturbance attenuation and $H_\infty$ control via measurement feedback in nonlinear systems. *IEEE Transactions on Automatic Control* 1992; **37**(9):1283–1293.
5. Isidori A, Kang W. $H_\infty$ control via measurement feedback for general nonlinear systems. *IEEE Transactions on Automatic Control* 1995; **40**(3):466–472.
6. Bianchini G, Genesio R, Parenti A, Tesi A. Global $H_\infty$ controllers for a class of nonlinear systems. *IEEE Transactions on Automatic Control* 2004; **49**(2):244–249.
7. Ankelhed D, Helmersson A, Hansson A. A quasi-Newton interior point method for low order H-infinity controller synthesis. *IEEE Transactions on Automatic Control* 2011; **56**(6):1462–1467.
8. Van Der Schaft AJ. $L_2$-gain analysis of nonlinear systems and nonlinear state-feedback $H_\infty$ control. *IEEE Transactions on Automatic Control* 1992; **6**(37):770–784.
9. Saridis GN, Lee CG. An approximation theory of optimal control for trainable manipulators. *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics* 1979; **9**(3):152–159.
10. Beard R, Saridis GN, Wen J. Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica* 1997; **33**(12):2159–2177.
11. Beard R, Saridis GN, Wen J. Approximate solutions to the time-invariant Hamilton-Jacobi-Bellman equation. *Journal of Optimization Theory and Applications* 1998; **96**(3):589–626.
12. Vamvoudakis KG, Lewis FL. Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 2010; **46**(5):878–888.
13. Beard RW, Mclain TW. Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *International Journal of Control* 1998; **71**(5):717–743.
14. Vamvoudakis KG, Lewis FL. Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton-Jacobi equations. *Automatica* 2011; **47**(8):1556–1569.
15. Abu-Khalaf M, Lewis FL, Huang J. Policy iterations and the Hamilton-Jacobi-Isaacs equation for $H_\infty$ state feedback control with input saturation. *IEEE Transactions on Automatic Control* 2006; **51**(12):1989–1995.



DOI: 10.1002/rnc

16. Abu-Khalaf M, Lewis FL, Huang J. Neurodynamic programming and zero-sum games for constrained control systems. *IEEE Transactions on Neural Networks* 2008; **19**(7):1243–1252.
17. Zhang H, Wei Q, Liu D. An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. *Automatica* 2011; **47**(1):207–214.
18. Feng Y, Anderson B, Rotkowitz M. A game theoretic algorithm to compute local stabilizing solutions to HJBI equations in nonlinear $H_\infty$ control. *Automatica* 2009; **45**(4):881–888.
19. Lanzon A, Feng Y, Anderson BDO, Rotkowitz M. Computing the positive stabilizing solution to algebraic Riccati equations with an indefinite quadratic term via a recursive method. *IEEE Transactions on Automatic Control* 2008; **53**(10):2280–2291.
20. Albrecht EG. On the optimal stabilization of nonlinear systems. *PMM-Journal of Applied Mathematics Mechanics* 1961; **25**:1254–1266.
21. Navasca CL, Krener A. Solutions of Hamilton-Jacobi-Bellman equations. *Proceedings of the IEEE Conference on Decision and Control*, Sydney Convention and Exhibition Centre Sydney, NSW Australia, 2000.
22. Zeidler E. *Nonlinear Functional Analysis vol.1: Fixed Point Theorems*. Springer-Verlag: New York, 1985.
23. Kantorovitch L. The method of successive approximation for functional equations. *Acta Mathematica* 1939; **71**(1):63–97.
24. Tapia RA. The Kantorovich theorem for Newton's method. *The American Mathematical Monthly* 1971; **78**(4): 389–392.
25. Rall LB. A note on the convergence of Newton's method. *SIAM Journal on Numerical Analysis* 1974; **11**(1):34–36.
26. Finlayson BA. *The Method of Weighted Residuals and Variational Principles*. Academic Press: New York, 1972.
27. McKay MD, Beckman RJ, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 1979; **21**(2):239–245.
28. Vrabie D, Lewis FL. Adaptive dynamic programming for online solution of a zero-sum differential game. *Journal of Control Theory and Applications* 2011; **9**(3):353–360.
29. Vrabie D, Lewis FL. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Networks* 2009; **22**(3):237–246.
30. Tsiotras P, Corless M, Rotea MA. An $L_2$ disturbance attenuation solution to the nonlinear benchmark problem. *International Journal of Robust and Nonlinear Control* 1998; **8**(4-5):311–330.
31. Cheng T, Lewis FL. Neural network solution for finite-horizon H-infinity constrained optimal control of nonlinear systems. *Journal of Control Theory and Applications* 2007; **5**(1):1–11.



DOI: 10.1002/rnc