Chapter 2
Optimal State Feedback Control for Discrete-Time Systems

2.1 Introduction

The optimal control problem of nonlinear systems has always been the key focus of control fields in the past several decades. Traditional optimal control approaches are mostly based on linearization methods or numerical computation methods. However, closed-loop optimal feedback control is desired for most researchers in practice. Therefore, in this chapter, several near-optimal control scheme will be developed for different nonlinear discrete-time systems by introducing the different iterative ADP algorithms.

First, an infinite-horizon optimal state feedback controller is developed for a class of discrete-time systems based on DHP. Then, due to the special advantages of GDHP algorithm, a new optimal control scheme is developed with discounted cost functional. Moreover, based on GHJB algorithm, an infinite-horizon optimal state feedback stabilizing controller is designed. Further, most existing controllers are implemented in infinite time horizon. However, many real-world systems need to be effectively controlled within a finite time horizon. Therefore, we further propose a finite-horizon optimal controllers with ε-error bound, where the number of optimal control steps can be determined definitely.

2.2 Infinite-Horizon Optimal State Feedback Control Based on DHP

Saturation, dead-zone, backlash, and hysteresis are the most common actuator nonlinearities in practical control system applications. Due to the nonanalytic nature of the actuator nonlinear dynamics and the fact that the exact actuator nonlinear functions are unknown, the systems with saturation present a challenge to control engineers. In this section, we study this problem in the framework of the HJB equation from optimal control theory. First, based on nonquadratic functionals, the HJB equation is formulated, whose solution results in a smooth saturated controller. Then

H. Zhang et al., *Adaptive Dynamic Programming for Control*,
Communications and Control Engineering, DOI 10.1007/978-1-4471-4757-2_2,

a new iterative ADP algorithm is presented with convergence proof to solve the HJB equation derived.

2.2.1 Problem Formulation

Consider a class of discrete-time affine nonlinear systems as follows:

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{2.1}$$

where $x(k) \in \mathbb{R}^n$ is the state vector, and $f: \mathbb{R}^n \to \mathbb{R}^n$ and $g: \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are differentiable in their arguments with $f(0) = 0$. Assume that $f + gu$ is Lipschitz continuous on a set Ω in $\mathbb{R}^n$ containing the origin, and that the system (2.1) is controllable in the sense that there exists at least a continuous control law on Ω that asymptotically stabilizes the system. We denote $\Omega_u = \{u(k) \mid u(k) = [u_1(k), u_2(k), \ldots, u_m(k)]^{\mathrm{T}} \in \mathbb{R}^m,\ |u_i(k)| \le \bar{u}_i,\ i = 1, \ldots, m\}$, where $\bar{u}_i$ is the saturating bound for the ith actuator. Let $\bar{U} \in \mathbb{R}^{m\times m}$ be the constant diagonal matrix given by $\bar{U} = \mathrm{diag}\{\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m\}$.

In this subsection, we mainly discuss how to design an optimal state feedback controller for this class of constrained discrete-time systems. It is desired to find the optimal control law $v(x)$ so that the control sequence $u(\cdot) = (u(i), u(i+1), \ldots)$ with each $u(i) \in \Omega_u$ minimizes the generalized *cost functional* as follows:

$$J(x(k), u(\cdot)) = \sum_{i=k}^{\infty} \left\{x^{\mathrm{T}}(i)Qx(i) + W(u(i))\right\}, \tag{2.2}$$

where $u(i) = v(x(i))$, $W(u(i)) \in \mathbb{R}$ is positive definite, and the weight matrix Q is also positive definite.

For optimal control problems, the state feedback control law $v(x)$ must not only stabilize the system on Ω but also guarantee that (2.2) is finite. Such a control law is said to be admissible.

Definition 2.1 A control law $v(x)$ is said to be admissible with respect to (2.2) on Ω if $v(x)$ is continuous with $v(x(k)) \in \Omega_u$ for $\forall x(k) \in \Omega$ and stabilizes (2.1) on Ω, $v(0) = 0$, and for $\forall x(0) \in \Omega$, $J(x(0), u(\cdot))$ is finite, where $u(\cdot) = (u(0), u(1), \ldots)$ and $u(k) = v(x(k))$, $k = 0, 1, \ldots$.

Based on the above definition, we are ready to explain the *admissible control law* sequence. A control law sequence $\{\eta_i\} = (\eta_0, \eta_1, \ldots, \eta_\infty)$ is called admissible if the resultant control sequence $(u(0), u(1), \ldots, u(\infty))$ stabilizes the system (2.1) with any initial state $x(0)$ and guarantees that $J(x(0), u(\cdot))$ is finite. It should be mentioned that, in this case, each control action obeys a different control law, i.e., $u(i)$ is produced by a control law η_i for $i = 0, 1, \ldots$. The control law sequence $\{\eta_i\} = (\eta_0, \eta_1, \ldots, \eta_\infty)$ is also called a nonstationary policy in the literature [2].

For convenience, in the sequel $J^*(x(k))$ is used to denote the optimal value function which is defined as $J^*(x(k)) = \min_{u(\cdot)} J(x(k), u(\cdot))$, and $u^*(x)$ is used to denote the corresponding optimal control law.

For the unconstrained control problem, $W(u(i))$ in the performance functional (2.2) is commonly chosen as the quadratic form of the control input $u(i)$. However, in this subsection, to confront the bounded control problem, we employ a *non-quadratic functional* as follows:

$$W(u(i)) = 2\int_0^{u(i)} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds, \tag{2.3}$$

$$\varphi^{-1}(u(i)) = [\varphi^{-1}(u_1(i)), \varphi^{-1}(u_2(i)), \ldots, \varphi^{-1}(u_m(i))]^{\mathrm{T}},$$

where R is positive definite and assumed to be diagonal for simplicity of analysis, $s \in \mathbb{R}^m$, $\varphi \in \mathbb{R}^m$, $\varphi(\cdot)$ is a bounded one-to-one function satisfying $|\varphi(\cdot)| \leq 1$ and belonging to C^p $(p \geq 1)$ and $L_2(\Omega)$. Moreover, it is a monotonic increasing odd function with its first derivative bounded by a constant M. Such a function is easy to find, and one example is the hyperbolic tangent function $\varphi(\cdot) = \tanh(\cdot)$. It should be noticed that, by the definition above, $W(u(i))$ is ensured to be positive definite because $\varphi^{-1}(\cdot)$ is a monotonic odd function and R is positive definite.

According to Bellman's principle of optimality, the *optimal value function* $J^*(x)$ should satisfy the following HJB equation:

$$\begin{aligned} J^*(x(k)) &= \min_{u(\cdot)} \sum_{i=k}^{\infty} \left\{ x^{\mathrm{T}}(i)Qx(i) + 2\int_0^{u(i)} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds \right\} \\ &= \min_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + 2\int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds \right. \\ &\qquad \left. + J^*(x(k+1)) \right\}. \end{aligned} \tag{2.4}$$

The optimal control law $u^*(x)$ should satisfy

$$\begin{aligned} u^*(x(k)) = \arg\min_{u(k)} &\left\{ x^{\mathrm{T}}(k)Qx(k) + 2\int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds \right. \\ &\left. + J^*(x(k+1)) \right\}. \end{aligned} \tag{2.5}$$

The optimal control problem can be solved if the optimal value function $J^*(x)$ can be obtained from (2.4). However, there is currently no method for solving this value function of the constrained optimal control problem. Therefore, in the next subsection we will discuss how to utilize the iterative ADP algorithm to seek the near-optimal control solution.

2.2.2 Infinite-Horizon Optimal State Feedback Control via DHP

Since direct solution of the HJB equation is computationally intensive, we develop in this subsection an iterative ADP algorithm, based on Bellman's principle of optimality and the greedy iteration principle.

First, we start with initial value function $V_0(\cdot) = 0$ which is not necessarily the optimal value function. Then, we find the law of single control vector $v_0(x)$ as follows:

$$v_0(x(k)) = \arg\min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + 2 \int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds + V_0(x(k+1)) \right\}, \tag{2.6}$$

and we update the value function by

$$V_1(x(k)) = x^{\mathrm{T}}(k) Q x(k) + 2 \int_0^{v_0(x(k))} \varphi^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds. \tag{2.7}$$

Therefore, for $i = 1, 2, \ldots$, the iterative ADP algorithm iterates between

$$v_i(x(k)) = \arg\min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + 2 \int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds + V_i(x(k+1)) \right\} \tag{2.8}$$

and

$$V_{i+1}(x(k)) = \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + 2 \int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds + V_i(x(k+1)) \right\}. \tag{2.9}$$

It can be seen that, based on (2.8), (2.9) can further be written as

$$V_{i+1}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + 2 \int_0^{v_i(x(k))} \varphi^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds + V_i\left(x(k+1)\right), \tag{2.10}$$

where $x(k+1) = f(x(k)) + g(x(k)) v_i(x(k))$.

In summary, in this iterative algorithm, the value function sequence $\{V_i\}$ and control law sequence $\{v_i\}$ are updated by implementing the recurrent iteration between (2.8) and (2.10) with the iteration number i increasing from 0 to ∞.

To further explain the iteration process, next we are ready to analyze this iterative algorithm. First, based on (2.10) we obtain

$$V_i(x(k+1)) = x^{\mathrm{T}}(k+1)Qx(k+1) + 2\int_0^{v_{i-1}(x(k+1))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds + V_{i-1}(x(k+2)), \tag{2.11}$$

where $x(k+2) = f(x(k+1)) + g(x(k+1))v_{i-1}(x(k+1))$. Then, by further expanding (2.10), we have

$$\begin{aligned} V_{i+1}(x(k)) = {} & x^{\mathrm{T}}(k)Qx(k) + 2\int_0^{v_i(x(k))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds \\ & + x^{\mathrm{T}}(k+1)Qx(k+1) + 2\int_0^{v_{i-1}(x(k+1))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds \\ & + \cdots + x^{\mathrm{T}}(k+i)Qx(k+i) \\ & + 2\int_0^{v_0(x(k+i))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds + V_0(x(k+i+1)), \end{aligned} \tag{2.12}$$

where $V_0(x(k+i+1)) = 0$.

From (2.12), it can be seen that during the iteration process, the control actions for different control steps obey different control laws. After the iteration number $i+1$, the obtained control law sequence is $(v_i, v_{i-1}, \ldots, v_0)$. With the iteration number i increasing to ∞, the obtained control law sequence has a length of ∞. For the infinite-horizon problem, both the optimal value function and the optimal control law are unique. Therefore, it is desired that the control law sequence will converge when the iteration number $i \to \infty$. In the following, we will prove that both the value function sequence $\{V_i\}$ and the control law sequence $\{v_i\}$ are convergent.

In this subsection, in order to prove the convergence characteristics of the iterative ADP algorithm for the constrained nonlinear system, we first present two lemmas before presenting our theorems. For convenience, the nonquadratic functional $2\int_0^{u(k)} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds$ will be written as $W(u(k))$ in the sequel.

Lemma 2.2 *Let $\{\mu_i\}$ be an arbitrary sequence of control laws, and $\{v_i\}$ be the control law sequence as in (2.8). Let V_i be as in (2.9) and Λ_i be*

$$\Lambda_{i+1}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + W(\mu_i(x(k))) + \Lambda_i(x(k+1)). \tag{2.13}$$

If $V_0(\cdot) = \Lambda_0(\cdot) = 0$, then $V_i(x) \le \Lambda_i(x),\ \forall i$.

Proof It is clear from the fact that V_{i+1} is the result of minimizing the right hand side of (2.9) with respect to the control input $u(k)$, while Λ_{i+1} is a result of arbitrary control input. □

Lemma 2.3 *Let the sequence $\{V_i\}$ be defined as in (2.9). If the system is controllable, then there is an upper bound Y such that $0 \le V_i(x(k)) \le Y,\ \forall i$.*

Proof Let $\{\eta_i(x)\}$ be a sequence of stabilizing and admissible control laws, and let $V_0(\cdot) = P_0(\cdot) = 0$, where V_i is updated by (2.9) and P_i is updated by

$$P_{i+1}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + W(\eta_i(x(k))) + P_i(x(k+1)). \tag{2.14}$$

From (2.14), we further obtain

$$\begin{aligned} P_i(x(k+1)) = {} & x^{\mathrm{T}}(k+1)Qx(k+1) + W(\eta_{i-1}(x(k+1))) \\ & + P_{i-1}(x(k+2)). \end{aligned} \tag{2.15}$$

Thus, the following relation can be obtained:

$$\begin{aligned} P_{i+1}(x(k)) = {} & x^{\mathrm{T}}(k)Qx(k) + W(\eta_i(x(k))) \\ & + x^{\mathrm{T}}(k+1)Qx(k+1) + W(\eta_{i-1}(x(k+1))) \\ & + P_{i-1}(x(k+2)) \\ = {} & x^{\mathrm{T}}(k)Qx(k) + W(\eta_i(x(k))) \\ & + x^{\mathrm{T}}(k+1)Qx(k+1) + W(\eta_{i-1}(x(k+1))) \\ & + x^{\mathrm{T}}(k+2)Qx(k+2) + W(\eta_{i-2}(x(k+2))) \\ & + P_{i-2}(x(k+3)) \\ & \vdots \\ = {} & x^{\mathrm{T}}(k)Qx(k) + W(\eta_i(x(k))) \\ & + x^{\mathrm{T}}(k+1)Qx(k+1) + W(\eta_{i-1}(x(k+1))) \\ & + x^{\mathrm{T}}(k+2)Qx(k+2) + W(\eta_{i-2}(x(k+2))) \\ & + \cdots \\ & + x^{\mathrm{T}}(k+i)Qx(k+i) + W(\eta_0(x(k+i))) \\ & + P_0(x(k+i+1)), \end{aligned} \tag{2.16}$$

where $P_0(x(k+i+1)) = 0$.

Let $l_i(x(k)) = x^{\mathrm{T}}(k)Qx(k) + W(\eta_i(x(k)))$, and then (2.16) can further be written as

$$\begin{aligned} P_{i+1}(x(k)) &= \sum_{j=0}^{i} l_{i-j}(x(k+j)) \\ &= \sum_{j=0}^{i} \left\{ x^{\mathrm{T}}(k+j)Qx(k+j) + W(\eta_{i-j}(x(k+j))) \right\} \end{aligned}$$

$$\leq \lim_{i\to\infty} \sum_{j=0}^{i} \left\{ x^{\mathrm{T}}(k+j) Q x(k+j) + W\left(\eta_{i-j}(x(k+j))\right) \right\}. \tag{2.17}$$

Note that $\{\eta_i(x)\}$ is an admissible control law sequence, i.e., $x(k) \to 0$ as $k \to \infty$. Therefore there exists an upper bound Y such that

$$\forall i: \quad P_{i+1}(x(k)) \leq \lim_{i\to\infty} \sum_{j=0}^{i} l_{i-j}(x(k+j)) \leq Y. \tag{2.18}$$

Combining with Lemma 2.2, we obtain

$$\forall i: \quad V_{i+1}(x(k)) \leq P_{i+1}(x(k)) \leq Y. \tag{2.19}$$

This completes the proof. □

Next, Lemmas 2.2 and 2.3 will be used in the proof of our main theorems.

Theorem 2.4 (cf. [17]) *Define the value function sequence* $\{V_i\}$ *as in* (2.10) *with* $V_0(\cdot) = 0$, *and the control law sequence* $\{v_i\}$ *as in* (2.8). *Then, we can conclude that* $\{V_i\}$ *is a nondecreasing sequence satisfying* $V_{i+1}(x(k)) \geq V_i(x(k)),\ \forall i$.

Proof For convenience of analysis, define a new sequence $\{\Phi_i\}$ as follows:

$$\Phi_{i+1}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + W(v_{i+1}(x(k))) + \Phi_i(x(k+1)), \tag{2.20}$$

where $\Phi_0(\cdot) = V_0(\cdot) = 0$. The control law sequence $\{v_i\}$ is updated by (2.8) and the value function sequence $\{V_i\}$ is updated by (2.10).

In the following, we prove that $\Phi_i(x(k)) \leq V_{i+1}(x(k))$ by mathematical induction.

First, we prove that it holds for $i = 0$. Noticing that

$$V_1(x(k)) - \Phi_0(x(k)) = x^{\mathrm{T}}(k) Q x(k) + W(v_0(x(k))) \geq 0, \tag{2.21}$$

thus for $i = 0$, we have

$$V_1(x(k)) \geq \Phi_0(x(k)). \tag{2.22}$$

Second, we assume that it holds for $i-1$. That is to say, for any $x(k)$, we have $V_i(x(k)) \geq \Phi_{i-1}(x(k))$. Then, for i, since

$$\Phi_i(x(k)) = x^{\mathrm{T}}(k) Q x(k) + W(v_i(x(k))) + \Phi_{i-1}(x(k+1)) \tag{2.23}$$

and

$$V_{i+1}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + W(v_i(x(k))) + V_i(x(k+1)) \tag{2.24}$$

hold, we obtain

$$V_{i+1}(x(k)) - \Phi_i(x(k)) = V_i(x(k+1)) - \Phi_{i-1}(x(k+1)) \geq 0, \tag{2.25}$$

i.e., the following equation holds:

$$\Phi_i(x(k)) \leq V_{i+1}(x(k)). \tag{2.26}$$

Therefore, (2.26) is proved for any i by mathematical induction.

Furthermore, from Lemma 2.2 we know that $V_i(x(k)) \leq \Phi_i(x(k))$. Therefore we have

$$V_i(x(k)) \leq \Phi_i(x(k)) \leq V_{i+1}(x(k)). \tag{2.27}$$

The proof is completed. □

Next, we are ready to exploit the limit of the value function sequence $\{V_i\}$ when $i \to \infty$.

Let $\{\eta_i^{(l)}\}$ be the lth admissible control law sequence, similar to the proof of Lemma 2.3, we can construct the associated sequence $P_i^{(l)}(x)$ as follows:

$$P_{i+1}^{(l)}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + W(\eta_i^{(l)}(x(k))) + P_i^{(l)}(x(k+1)), \tag{2.28}$$

with $P_0^{(l)}(\cdot) = 0$.

Let $l_i^{(l)}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + W(\eta_i^{(l)}(x(k)))$. Then, the following relation can be obtained similarly:

$$P_{i+1}^{(l)}(x(k)) = \sum_{j=0}^{i} l_{i-j}^{(l)}(x(k+j)). \tag{2.29}$$

Let $i \to \infty$; we have

$$P_{\infty}^{(l)}(x(k)) = \lim_{i\to\infty} \sum_{j=0}^{i} l_{i-j}^{(l)}(x(k+j)). \tag{2.30}$$

Combining (2.29) with (2.30), we obtain

$$P_{i+1}^{(l)}(x(k)) \leq P_{\infty}^{(l)}(x(k)). \tag{2.31}$$

Theorem 2.5 (cf. [17]) *Define $P_{\infty}^{(l)}(x(k))$ as in (2.30), and the value function sequence $\{V_i\}$ as in (2.10) with $V_0(\cdot) = 0$. For any state vector $x(k)$, define $J^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, which can be considered as the "optimal" value function starting from $x(k)$ under all admissible control law sequences with length of ∞. Then, we can conclude that J^* is the limit of the value function sequence $\{V_i\}$.*

Proof According to the definition of $P_{\infty}^{(l)}(x(k))$, the associated control law sequence $\{\eta_i^{(l)}(x)\}$ is admissible. Thus, it is guaranteed that $\lim_{i\to\infty} \sum_{j=0}^{i} l_{i-j}^{(l)}(x(k+j))$ is

finite, i.e., $P_{\infty}^{(l)}(x(k))$ is finite. Hence for any l, there exists an upper bound Y_l such that

$$P_{i+1}^{(l)}(x(k)) \leq P_{\infty}^{(l)}(x(k)) \leq Y_l. \tag{2.32}$$

Combining with Lemma 2.2, we further obtain

$$\forall l, i: \ V_{i+1}(x(k)) \leq P_{i+1}^{(l)}(x(k)) \leq Y_l. \tag{2.33}$$

Since $J^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, for any $\epsilon > 0$, there exists a sequence of admissible control laws $\{\eta_i^{(K)}\}$ such that the associated value function satisfies $P_{\infty}^{(K)}(x(k)) \leq J^*(x(k)) + \epsilon$. According to (2.33), we have $V_i(x(k)) \leq P_i^{(l)}(x(k))$ for any l and i. Thus, we obtain $\lim_{i\to\infty} V_i(x(k)) \leq P_{\infty}^{(K)}(x(k)) \leq J^*(x(k)) + \epsilon$. Noting that ϵ is chosen arbitrarily, we have

$$\lim_{i\to\infty} V_i(x(k)) \leq J^*(x(k)). \tag{2.34}$$

On the other hand, since $V_{i+1}(x(k)) \leq P_{i+1}^{(l)}(x(k)) \leq Y_l, \forall l, i$, we have $\lim_{i\to\infty} V_i(x(k)) \leq \inf_l\{Y_l\}$. According to the definition of admissible control law sequence, the control law sequence associated with the value function $\lim_{i\to\infty} V_i(x(k))$ must be an admissible control law sequence, i.e., there exists a sequence of admissible control laws $\{\eta_i^{(N)}\}$ such that $\lim_{i\to\infty} V_i(x(k)) = P_{\infty}^{(N)}(x(k))$. Combining with the definition $J^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, we can obtain

$$\lim_{i\to\infty} V_i(x(k)) \geq J^*(x(k)). \tag{2.35}$$

Therefore, combining (2.34) with (2.35), we can conclude that $\lim_{i\to\infty} V_i(x(k)) = J^*(x(k))$, i.e., J^* is the limit of the value function sequence $\{V_i\}$.

The proof is completed. □

Next, let us consider what will happen when we make $i \to \infty$ in (2.9). The left hand side is simply $V_{\infty}(x)$. But for the right hand side, it is not obvious to see since the minimum will reach at different $u(k)$ for different i. However, the following result can be proved.

Theorem 2.6 *For any state vector $x(k)$, the "optimal" value function $J^*(x)$ satisfies the HJB equation*

$$J^*(x(k)) = \inf_{u(k)}\left\{x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1))\right\}.$$

Proof For any $u(k)$ and i, according to (2.9), we have

$$V_i(x(k)) \leq x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + V_{i-1}(x(k+1)). \tag{2.36}$$

According to Theorems 2.4 and 2.5, the value function sequence $\{V_i\}$ is a nondecreasing sequence satisfying $\lim_{i\to\infty} V_i(x(k)) = J^*(x(k))$, hence the relation $V_{i-1}(x(k+1)) \le J^*(x(k+1))$ holds for any i. Thus, we obtain

$$V_i(x(k)) \le x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1)). \tag{2.37}$$

Let $i \to \infty$; we have

$$J^*(x(k)) \le x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1)). \tag{2.38}$$

Since $u(k)$ in the above equation is chosen arbitrarily, the following equation holds:

$$J^*(x(k)) \le \inf_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1)) \right\}. \tag{2.39}$$

On the other hand, for any i the value function sequence satisfies

$$V_i(x(k)) = \min_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + V_{i-1}(x(k+1)) \right\}. \tag{2.40}$$

Combining with $V_i(x(k)) \le J^*(x(k)), \forall i$, we have

$$J^*(x(k)) \ge \inf_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + V_{i-1}(x(k+1)) \right\}. \tag{2.41}$$

Let $i \to \infty$; then we obtain

$$J^*(x(k)) \ge \inf_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1)) \right\}. \tag{2.42}$$

Combining (2.39) and (2.42), we have

$$J^*(x(k)) = \inf_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1)) \right\}. \tag{2.43}$$

The proof is completed. □

According to Theorems 2.4 and 2.5, we can conclude that $V_i(x(k)) \le V_{i+1}(x(k))$, $\forall i$ and $\lim_{i\to\infty} V_i(x(k)) = J^*(x(k))$. Furthermore, according to Theorem 2.6, we have $J^*(x(k)) = \inf_{u(k)}\{x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1))\}$. Therefore, we can conclude that the value function sequence $\{V_i\}$ converges to the optimal value function of the discrete-time HJB equation, i.e., $V_i \to J^*$ as $i \to \infty$. Since the value function sequence is convergent, according to (2.5) and (2.8), we can conclude that the corresponding control law sequence $\{v_i\}$ converges to the optimal control law u^* as $i \to \infty$.

It should be mentioned that the value function $V_i(x)$ we constructed is a new function that is different from ordinary cost function. Via Lemma 2.3 and Theorem 2.4, we have showed that for any $x(k) \in \Omega$, the function sequence $\{V_i(x(k))\}$ is a nondecreasing sequence, which will increase its value with an upper bound. This

is in contrast to other work in the literature, e.g., [5], where the value functions are constructed as a nonincreasing sequence with lower bound. Moreover, it should be noted that we do not require every control law in the sequence $\{v_i\}$ to be admissible. What we need is a control law sequence to be admissible, i.e., the resultant sequence of control vectors can stabilize the system.

Next, we are ready to discuss the implementation of the iterative ADP algorithm.

(1) *Derivation of the iterative DHP algorithm*. First, we assume that the value function $V_i(x)$ is smooth. In order to implement the iteration between (2.8) and (2.10), for $i = 0, 1, \ldots$, we further assume that the minimum of the right hand side of (2.8) can be exactly solved by letting the gradient of the right hand side of (2.8) with respect to $u(k)$ equal to zero, i.e.,

$$\frac{\partial \left(x^{\mathrm{T}}(k)Qx(k) + W(u(k))\right)}{\partial u(k)} + \left(\frac{\partial x(k+1)}{\partial u(k)}\right)^{\mathrm{T}} \frac{\partial V_i(x(k+1))}{\partial x(k+1)} = 0. \tag{2.44}$$

Therefore, for $i = 0, 1, \ldots$, the corresponding control law $v_i(x)$ can be obtained by solving the above equation, i.e.,

$$v_i(x(k)) = \bar{U}\varphi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}(x(k))\frac{\partial V_i(x(k+1))}{\partial x(k+1)}\right). \tag{2.45}$$

From (2.45), we find that the control law $v_i(x)$ at each step of iteration has to be computed by $\partial V_i(x(k+1))/\partial x(k+1)$, which is not an easy task. Furthermore, at each iteration step of value function $V_{i+1}(x(k))$ in (2.10), there exists an integral term $2\int_0^{v_i(x(k))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds$ to compute, which is a large computing burden. Therefore, in the following we will present another method called iterative DHP algorithm to implement the iterative ADP algorithm.

Define the costate function $\lambda(x) = \partial V(x)/\partial x$. Here, we assume that the value function $V(x)$ is smooth so that $\lambda(x)$ exists. Then, the recurrent iteration between (2.8) and (2.10) can be implemented as follows.

First, we start with an initial costate function $\lambda_0(\cdot) = 0$. Then, for $i = 0, 1, \ldots$, by substituting $\lambda_i(x) = \partial V_i(x)/\partial x$ into (2.45), we obtain the corresponding control law $v_i(x)$ as

$$v_i(x(k)) = \bar{U}\varphi\Big(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}(x(k))\lambda_i(x(k+1))\Big). \tag{2.46}$$

For $\lambda_{i+1}(x(k)) = \dfrac{\partial V_{i+1}(x(k))}{\partial x(k)}$, according to (2.10) we can obtain

$$\begin{aligned}\lambda_{i+1}(x(k)) = {} & \frac{\partial \left(x^{\mathrm{T}}(k)Qx(k) + W\left(v_i(x(k))\right)\right)}{\partial x(k)} \\ & + \left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}} \frac{\partial \left(x^{\mathrm{T}}(k)Qx(k) + W(v_i(x(k)))\right)}{\partial v_i(x(k))} \\ & + \left(\frac{\partial x(k+1)}{\partial x(k)}\right)^{\mathrm{T}} \frac{\partial V_i(x(k+1))}{\partial x(k+1)}\end{aligned}$$

$$
\begin{aligned}
&+\left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}}\left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^{\mathrm{T}}\frac{\partial V_i(x(k+1))}{\partial x(k+1)}\\
&=\frac{\partial\left(x^{\mathrm{T}}(k)Qx(k)+W(v_i(x(k)))\right)}{\partial x(k)}\\
&\quad+\left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}}\left[\frac{\partial\left(x^{\mathrm{T}}(k)Qx(k)+W(v_i(x(k)))\right)}{\partial v_i(x(k))}\right.\\
&\qquad\left.+\left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^{\mathrm{T}}\frac{\partial V_i(x(k+1))}{\partial x(k+1)}\right]\\
&\quad+\left(\frac{\partial x(k+1)}{\partial x(k)}\right)^{\mathrm{T}}\frac{\partial V_i(x(k+1))}{\partial x(k+1)}.
\end{aligned}
\tag{2.47}
$$

According to (2.44) and (2.45), we have

$$
\frac{\partial\left(x^{\mathrm{T}}(k)Qx(k)+W(v_i(x(k)))\right)}{\partial v_i(x(k))}+\left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^{\mathrm{T}}\frac{\partial V_i(x(k+1))}{\partial x(k+1)}=0. \tag{2.48}
$$

Therefore (2.47) can further be written as

$$
\begin{aligned}
\lambda_{i+1}(x(k))=&\frac{\partial\left(x^{\mathrm{T}}(k)Qx(k)+W(v_i(x(k)))\right)}{\partial x(k)}\\
&+\left(\frac{\partial x(k+1)}{\partial x(k)}\right)^{\mathrm{T}}\frac{\partial V_i(x(k+1))}{\partial x(k+1)},
\end{aligned}
\tag{2.49}
$$

i.e.,

$$
\lambda_{i+1}(x(k))=2Qx(k)+\left(\frac{\partial x(k+1)}{\partial x(k)}\right)^{\mathrm{T}}\lambda_i(x(k+1)). \tag{2.50}
$$

Therefore, the iteration between (2.46) and (2.50) is an implementation of the iteration between (2.8) and (2.10). From (2.46) the control law v_i can directly be obtained by the costate function. Hence the iteration of value function in (2.10) can be omitted in the implementation of this iterative algorithm. Considering the principle of DHP algorithm in Chap. 1, we call such iterative algorithm as iterative DHP algorithm.

Next, we present a *convergence analysis* of the iteration between (2.46) and (2.50).

Theorem 2.7 *Define the control law sequence* $\{v_i\}$ *as in* (2.8), *and update the value function sequence* $\{V_i\}$ *by* (2.10) *with* $V_0(\cdot)=0$. *Define the costate function sequence* $\{\lambda_i\}$ *as in* (2.50) *with* $\lambda_0(\cdot)=0$. *Then, the costate function sequence* $\{\lambda_i\}$ *and the control law sequence* $\{v_i\}$ *are convergent as* $i\to\infty$. *The optimal value* λ^* *is defined as the limit of the costate function* λ_i *when* v_i *approaches the optimal value* u^*.

Proof According to Theorems 2.4–2.6, we have proved that $\lim_{i\to\infty} V_i(x(k)) = J^*(x(k))$, and $J^*(x(k))$ satisfies the corresponding HJB equation, i.e.,

$$J^*(x(k)) = \inf_{u(k)}\{x^{\mathrm{T}}(k)Qx(k) + W(u(k)) + J^*(x(k+1))\}.$$

Therefore, we conclude that the value function sequence $\{V_i\}$ converges to the optimal value function of the DTHJB equation, i.e., $V_i \to J^*$ as $i \to \infty$. With $\lambda_i(x(k)) = \partial V_i(x(k))/\partial x(k)$, we conclude that the corresponding costate function sequence $\{\lambda_i\}$ is also convergent with $\lambda_i \to \lambda^*$ as $i \to \infty$. Since the costate function is convergent, we can conclude that the corresponding control law sequence $\{v_i\}$ converges to the optimal control law u^* as $i \to \infty$. □

Remark 2.8 In the iterative DHP algorithm, via the costate sequence (2.50), the corresponding control law sequence can be directly obtained by (2.46), which does not require the computation of $\partial V_i(x(k+1))/\partial x(k+1)$. Furthermore, in (2.10) there is an integral term $2\int_0^{v_i(x(k))} \varphi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds$ to compute at each iteration step, which is not an easy task. However, in (2.50) the integral term has been removed, which greatly reduces the computational burden. On the other hand, in order to compute the costate function by (2.50), the internal dynamics $f(x(k))$ and $g(x(k))$ of the system are needed. In the implementation part of the algorithm, a model network is constructed to approximate the nonlinear dynamics of the system, which avoids the requirement of known $f(x(k))$ and $g(x(k))$.

(2) *RBFNN implementation of the iterative DHP algorithm*. In the iterative DHP algorithm, the optimal control is difficult to solve analytically. For example, in (2.46), the control at step k is a function of costate at step $k+1$. A closed-form explicit solution is difficult to solve, if not impossible. Therefore we need to use parametric structures, such as fuzzy models [15] or neural networks, to approximate the costate function and the corresponding control law in the iterative DHP algorithm. In this subsection, we choose radial basis function (RBF) NNs to approximate the nonlinear functions.

An RBFNN consists of three-layers (input, hidden and output). Each input value is assigned to a node in the input layer and passed directly to the hidden layer without weights. Nodes at the hidden layer are called RBF units, determined by a vector called center and a scalar called width. The Gaussian density function is used as an activation function for the hidden neurons. Then, linear output weights connect the hidden and output layers. The overall input–output equation of the RBFNN is given as

$$y_i = b_i + \sum_{j=1}^{h} w_{ji}\phi_j(X), \tag{2.51}$$

where X is the input vector, $\phi_j(X) = \exp(-\|X - C_j\|^2/{\sigma_j}^2)$ is the activation function of the jth RBF unit in the hidden layer, $C_j \in \mathbb{R}^n$ is the center of the jth RBF

unit, h is the number of RBF units, b_i and w_{ji} are the bias term and the weight between hidden and output layer, and y_i is the ith output in the m-dimensional space. Once the optimal RBF centers are established over a wide range of operating points of the plant, the width of the ith center in the hidden layer is calculated by the following formula:

$$\sigma_i = \sqrt{\frac{1}{h}\sum_{j=1}^{h}\sum_{k=1}^{n}(\|c_{ki} - c_{kj}\|)}, \tag{2.52}$$

where c_{ki} and c_{kj} are the kth value of the center of the ith and jth RBF units, respectively. In (2.51) and (2.52), $\|\cdot\|$ represents the Euclidean norm. To avoid the extensive computational complexity during training, the batch mode k-means clustering algorithm is used to calculate the centers of the RBF units.

In order to implement the iterative ADP algorithm, i.e., implement the iteration between (2.46) and (2.50), we employ RBFNNs to approximate the costate function $\lambda_i(x)$ and the corresponding control law $v_i(x)$ at each iteration step i. In the implementation of the iterative DHP algorithm, there are three networks, which are model network, critic network and action network, respectively. All the neural networks are chosen as RBF networks. The inputs of the model network are $x(k)$ and $v_i(x(k))$ and the inputs of the critic network and action network are $x(k+1)$ and $x(k)$, respectively. The diagram of the whole structure is shown in Fig. 2.1.

For unknown plants, before carrying out the iterative DHP algorithm, we first train the model network. For any given $x(k)$ and $\hat{v}_i(x(k))$, we obtain $\hat{x}(k+1)$, and the output of the model network is denoted

$$\hat{x}(k+1) = w_m^{\mathrm{T}}\phi(I_m(k)), \tag{2.53}$$

where $I_m(k) = [x^{\mathrm{T}}(k)\hat{v}_i^{\mathrm{T}}(x(k))]^{\mathrm{T}}$ is the input vector of the model network.

We define the error function of the model network as

$$e_m(k) = \hat{x}(k+1) - x(k+1). \tag{2.54}$$

The weights in the model network are updated to minimize the following performance measure:

$$E_m(k) = \frac{1}{2}e_m^{\mathrm{T}}(k)e_m(k). \tag{2.55}$$

The weight updating rule for model network is chosen as a gradient-based adaptation rule

$$w_m(k+1) = w_m(k) - \alpha_m\left[\frac{\partial E_m(k)}{\partial w_m(k)}\right], \tag{2.56}$$

where α_m is the learning rate of the model network.

After the model network is trained, its weights are kept unchanged.

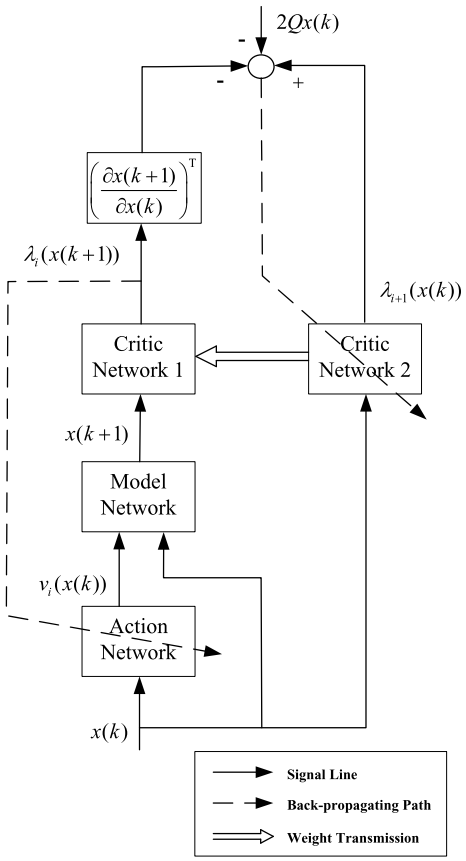

Fig. 2.1 The structure diagram of the iterative DHP algorithm

The critic network is used to approximate the costate function $\lambda_{i+1}(x)$. The output of the critic network is denoted

$$\hat{\lambda}_{i+1}(x(k)) = w_{c(i+1)}^{\mathrm{T}} \phi(x(k)). \tag{2.57}$$

The target costate function is given as in (2.50). Define the error function for the critic network as

$$e_{c(i+1)}(k) = \hat{\lambda}_{i+1}(x(k)) - \lambda_{i+1}(x(k)). \tag{2.58}$$

The objective function to be minimized for the critic network is

$$E_{c(i+1)}(k) = \frac{1}{2} e_{c(i+1)}^{\mathrm{T}}(k) e_{c(i+1)}(k). \tag{2.59}$$

The weight updating rule for the critic network is a gradient-based adaptation given by

$$w_{c(i+1)}(j+1) = w_{c(i+1)}(j) - \alpha_c \left[\frac{\partial E_{c(i+1)}(k)}{\partial w_{c(i+1)}(j)} \right], \tag{2.60}$$

where $\alpha_c > 0$ is the learning rate of the critic network, and j is the inner-loop iteration step for updating the weight parameters.

In the action network, the state $x(k)$ is used as the input of the network and the output can be formulated as

$$\hat{v}_i(x(k)) = w_{ai}^{\mathrm{T}} \phi(x(k)). \tag{2.61}$$

The target value of the control $v_i(x(k))$ is obtained by (2.46). So we can define the error function of the action network as

$$e_{ai}(k) = \hat{v}_i(x(k)) - v_i(x(k)). \tag{2.62}$$

The weights of the action network are updated to minimize the following performance error measure:

$$E_{ai}(k) = \frac{1}{2} e_{ai}^{\mathrm{T}}(k) e_{ai}(k). \tag{2.63}$$

The updating algorithm is then similar to the one for the critic network. By the gradient descent rule, we obtain

$$w_{ai}(j+1) = w_{ai}(j) - \alpha_a \left[\frac{\partial E_{ai}(k)}{\partial w_{ai}(j)} \right], \tag{2.64}$$

where $\alpha_a > 0$ is the learning rate of the action network, and j is the inner-loop iteration step for updating the weight parameters.

From the neural-network implementation, we can find that in this iterative DHP algorithm, $\partial V_i(x(k+1))/\partial x(k+1)$ is replaced by $\hat{\lambda}_i(x(k+1))$, which is just the output of the critic network. Therefore, it is more accurate than computing by back-propagation through the critic network as in [1].

(3) *Design procedure of the approximate optimal controller*. Based on the iterative DHP algorithm, the design procedure of the optimal control scheme is summarized as follows:

1. Choose $i_{\max}$, $j_{\max}^a$, $j_{\max}^c$, ε_m, ε_0, $\bar{U}$, α_m, α_c, α_a and the weight matrices Q and R.
2. Construct the model network $\hat{x}(k+1) = w_m^{\mathrm{T}} \phi(I_m(k))$ with the initial weight parameters w_{m0} chosen randomly from $[-0.1, 0.1]$ and train the model network with a random input vector uniformly distributed in the interval $[-1, 1]$ and arbitrary initial state vector in $[-1, 1]$ till the given accuracy ε_m is reached.
3. Set the iteration step $i = 0$. Set the initial weight parameters of critic network w_{c0} as zero so that the initial value of the costate function $\lambda_0(\cdot) = 0$, and initialize the action network with the weight parameters w_{a0} chosen randomly in $[-0.1, 0.1]$.
4. Choose an array of state vector $x(k) = (x^{(1)}(k), x^{(2)}(k), \ldots, x^{(p)}(k))$ randomly from the operation region and compute the corresponding output target $v_i(x(k)) = (v_i(x^{(1)}(k)), v_i(x^{(2)}(k)), \ldots, v_i(x^{(p)}(k)))$ by (2.46), where the state

vector at the next time instant

$$x(k+1) = \left(x^{(1)}(k+1), x^{(2)}(k+1), \ldots, x^{(p)}(k+1)\right)$$

is computed by the model network (2.53). With the same state vector $x(k) = (x^{(1)}(k), x^{(2)}(k), \ldots, x^{(p)}(k))$ and

$$x(k+1) = \left(x^{(1)}(k+1), x^{(2)}(k+1), \ldots, x^{(p)}(k+1)\right),$$

compute the resultant output target

$$\lambda_{i+1}(x(k)) = \left(\lambda_{i+1}(x^{(1)}(k)), \lambda_{i+1}(x^{(2)}(k)), \ldots, \lambda_{i+1}(x^{(p)}(k))\right)$$

by (2.50).

5. Set $w_{c(i+1)} = w_{ci}$. With the data set $(x^{(j)}(k), \lambda_{i+1}(x^{(j)}(k)))$, $j = 1, 2, \ldots, p$, update the weight parameters of the critic network $w_{c(i+1)}$ by (2.60) for $j^c_{\max}$ steps to get the approximate costate function $\hat{\lambda}_{i+1}$.
6. With the data set $(x^{(j)}(k), v_i(x^{(j)}(k)))$, $j = 1, 2, \ldots, p$, update the weight parameters of the action network w_{ai} by (2.64) for $j^a_{\max}$ steps to get the approximate control law $\hat{v}_i$.
7. If

$$\|\lambda_{i+1}(x(k)) - \lambda_i(x(k))\|^2 < \varepsilon_0,$$

go to Step 9; otherwise, go to Step 8.
8. If $i > i_{\max}$, go to Step 9; otherwise, set $i = i + 1$ and go to Step 4.
9. Set the final approximate optimal control law $\hat{u}^*(x) = \hat{v}_i(x)$.
10. Stop.

As stated in the last subsection, the iterative algorithm will be convergent with $\lambda_i(x) \to \lambda^*(x)$ and the control sequence $v_i(x) \to u^*(x)$ as $i \to \infty$. However, in practical applications, we cannot implement the iteration till $i \to \infty$. Actually, we iterate the algorithm for a max number $i_{\max}$ or with a pre-specified accuracy ε_0 to test the convergence of the algorithm. In the above procedure, there are two levels of loops. The outer loop starts from Step 3 and ends at Step 8. There are two inner loops in Steps 5 and 6, respectively. The inner loop of Step 5 includes $j^c_{\max}$ iterative steps, and the inner loop of Step 6 includes $j^a_{\max}$ iterative steps. The state vector $x(k)$ is chosen randomly at Step 4. Suppose that the associated random probability density function is nonvanishing everywhere. Then we can assume that all the states will be explored. So we know that the resulting networks tend to satisfy the formulas (2.46) and (2.50) for all state vectors $x(k)$. The limits of $\hat{\lambda}_i$ and $\hat{v}_i$ will approximate the optimal ones λ^* and u^*, respectively. The parameters ε_0 and $i_{\max}$ are chosen by the designer. The smaller the value of ε_0 is set, the more accurate the costate function and the optimal control law will be. If the condition set in Step 7 is satisfied, it implies that the costate function sequence is convergent with the pre-specified

accuracy. The larger the value of $i_{\max}$ in Step 8 is set, the more accurate the obtained control law $\hat{v}(x)$ will be at the price of increased computational burden.

2.2.3 Simulations

In this section, two examples are provided to demonstrate the effectiveness of the control scheme developed in this subsection.

Example 2.9 (Nonlinear Discrete-Time System) Consider the following nonlinear system [5]:

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{2.65}$$

where $f(x(k)) = \begin{bmatrix} -0.8x_2(k) \\ \sin(0.8x_1(k) - x_2(k)) + 1.8x_2(k) \end{bmatrix}$, $g(x(k)) = \begin{bmatrix} 0 \\ -x_2(k) \end{bmatrix}$, and assume that the control constraint is set to $|u| \leq 0.3$.

Define the cost functional as

$$J(x(k), u(\cdot)) = \sum_{i=k}^{\infty} \left\{ x^{\mathrm{T}}(i) Q x(i) + 2 \int_0^{u(i)} \tanh^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds \right\}, \tag{2.66}$$

where $\bar{U} = 0.3$, and the weight matrices are chosen as $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $R = [0.5]$.

First, we perform the simulation of iterative ADP algorithm. In this iterative algorithm, we choose RBFNNs as the critic network, the action network and the model network with the structure 2–9–2, 2–9–1 and 3–9–2, respectively. The training sets are selected as $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$, which is the operation region of the system. It should be mentioned that the model network should be trained first. The initial state vectors are chosen randomly from $[-1, 1]$. Under the learning rate of $\alpha_m = 0.1$, the model network is trained until the given accuracy $\varepsilon_m = 10^{-6}$ is reached. After the training of the model network is completed, the weights are kept unchanged. Then, the critic network and the action network are trained with the learning rates $\alpha_a = \alpha_c = 0.1$ and the inner-loop iteration number $j^c_{\max} = j^a_{\max} = 2000$. Meanwhile the pre-specified accuracy ε_0 is set to 10^{-20}. Denote the outer loop iteration number as L. After implementing the outer loop iteration for $L = i_{\max} = 100$, the convergence curves of the costate function are shown in Fig. 2.2. It can be seen that the costate function is basically convergent with the outer loop iteration $L > 15$. In order to compare the different actions of the control laws obtained under different outer loop iteration numbers, for the same initial state vector $x_1(0) = 0.5$ and $x_2(0) = 0.5$, we apply different control laws to the plant for 30 time steps and obtain the simulation results as follows. The state curves are shown in Figs. 2.3 and 2.4, and the corresponding control inputs are shown in Fig. 2.5. It can be seen that the system responses are improved when the outer loop iteration number L is increased. When $L > 80$, the system responses only improve slightly in performance.

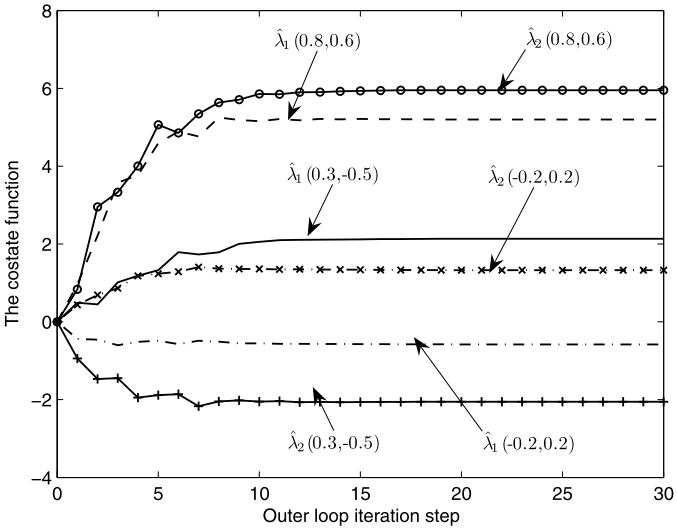

Fig. 2.2 The convergence process of the costate function at $x = (0.3, -0.5)$, $x = (-0.2, 0.2)$, $x = (0.8, 0.6)$

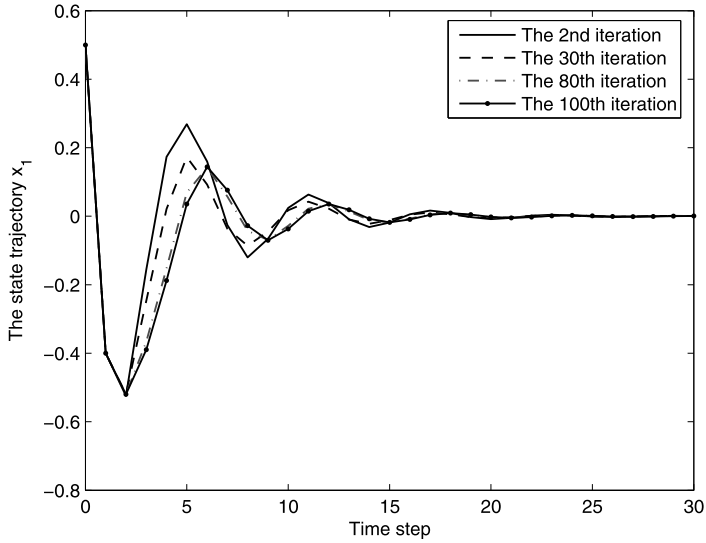

Fig. 2.3 The state trajectory x_1 for $L = 2, 30, 80, 100$

It should be mentioned that in order to show the convergence characteristics of the iterative process more clearly, we set the required accuracy ε_0 to a very small number 10^{-20} and we set the max iteration number to twice of what is needed.

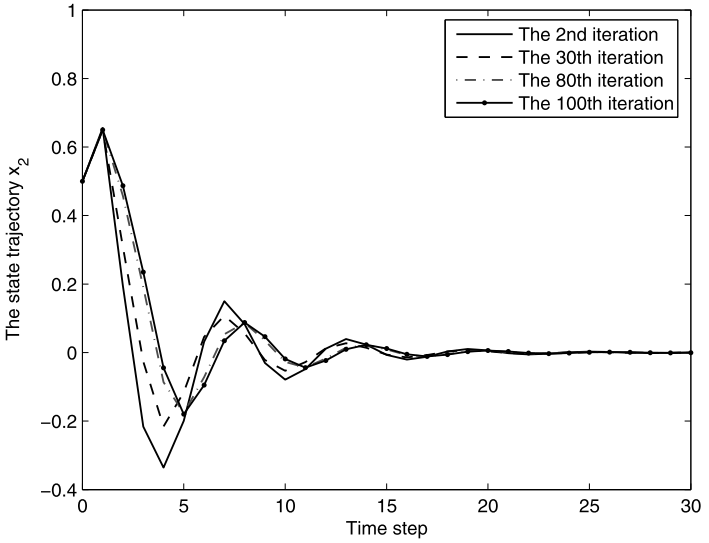

Fig. 2.4 The state trajectory x_2 for $L = 2, 30, 80, 100$

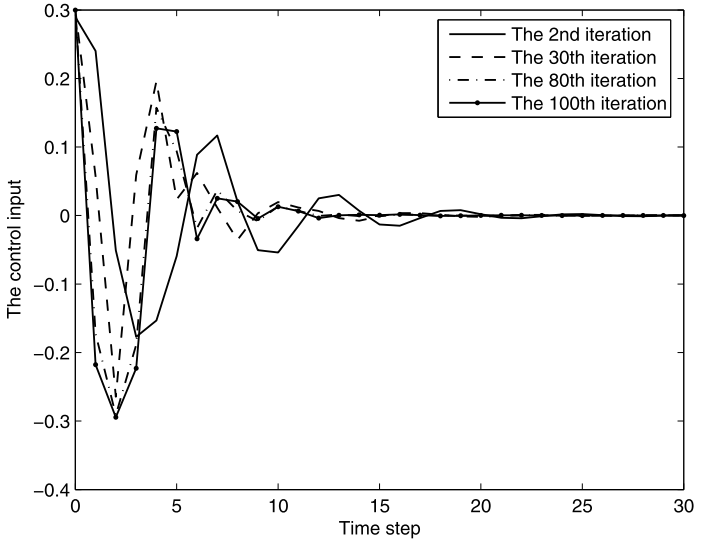

Fig. 2.5 The control input u for $L = 2, 30, 80, 100$

In this way, the given accuracy ε_0 did not take effect even when the max iteration number is reached. Therefore, it seems that the max iteration number $i_{\max}$ becomes the stopping criterion in this case. If the designer wants to save the running time, the

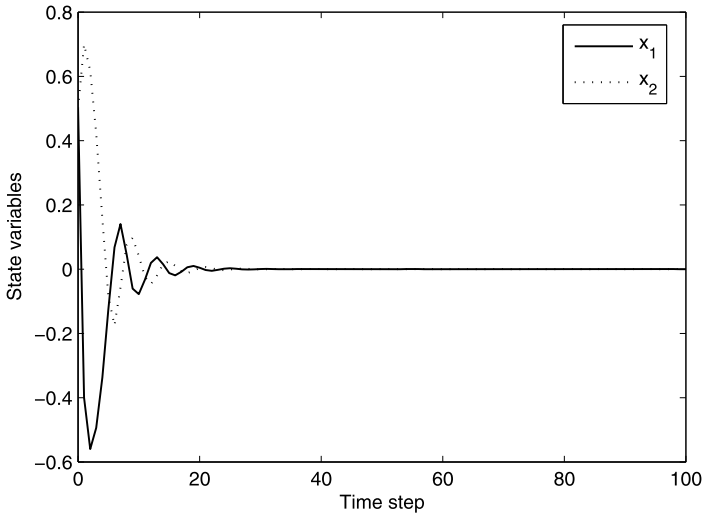

Fig. 2.6 The state variables curves without considering the actuator saturation in the controller design

pre-specified accuracy ε_0 can be set to a normal value so that the iterative process will be stopped once the accuracy ε_0 is reached.

Moreover, in order to make comparison with the controller designed without considering the actuator saturation, we also present the system responses obtained by the controller designed regardless of the actuator saturation. However, the actuator saturation is actually existing, therefore in the simulation if the control input overrun the saturation bound, it is limited to the bound value. After simulation, the state curves are as shown in Fig. 2.6, and the control curve is shown in Fig. 2.7.

From the simulation results, we can see that the iterative costate function sequences do converge to the optimal ones with very fast speed, which also indicates the validity of the iterative ADP algorithm for dealing with constrained nonlinear systems. Comparing Fig. 2.5 with Fig. 2.7, we can see that in Fig. 2.5 the restriction of actuator saturation has been overcome successfully, but in Fig. 2.7 the control input has overrun the saturation bound and therefore be limited to the bound value. From this point, we can conclude that the present iterative ADP algorithm is effective in dealing with the constrained optimal control problem.

Example 2.10 (Mass–Spring System) Consider the following discrete-time nonlinear mass–spring system:

$$\begin{cases} x_1(k+1) = 0.05x_2(k) + x_1(k), \\ x_2(k+1) = -0.0005x_1(k) - 0.0335x_1^3(k) + 0.05u(k) + x_2(k), \end{cases} \tag{2.67}$$

where $x(k)$ is the state vector, and $u(k)$ is the control input.

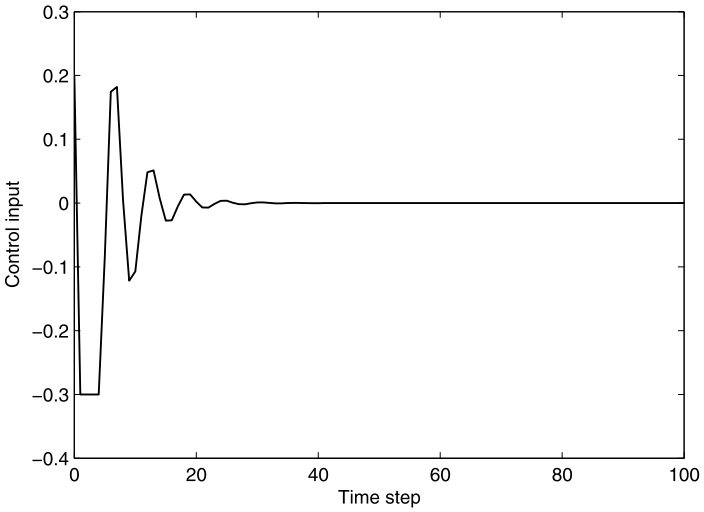

Fig. 2.7 The control input curve without considering the actuator saturation in the controller design

Define the cost functional as

$$J(x(k), u(\cdot)) = \sum_{i=k}^{\infty} \left\{ x^{\mathrm{T}}(i) Q x(i) + 2 \int_0^{u(i)} \tanh^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds \right\}, \tag{2.68}$$

where the control constraint is set to $\bar{U} = 0.6$, and the weight matrices are chosen as $Q = \left[\begin{smallmatrix} 0.5 & 0 \\ 0 & 0.5 \end{smallmatrix}\right]$ and $R = [1]$. The training sets are $-1 \le x_1 \le 1$ and $-1 \le x_2 \le 1$. The critic network, the action network and the model network are chosen as RBF neural networks with the structure of 2–16–2, 2–16–1 and 3–16–2, respectively. In the training process, the learning rates are set to $\alpha_a = \alpha_c = 0.1$. The other parameters are set the same as those in Example 2.9. After implementing the outer loop iteration for $L = i_{\max} = 300$, the convergence curves of the costate function are shown in Fig. 2.8. It can be seen that the costate function is basically convergent with the outer loop iteration $L > 200$. In order to compare the different actions of the control laws obtained under different outer loop iteration numbers, for the same initial state vector $x_1(0) = -1$ and $x_2(0) = 1$, we apply different control laws to the plant for 300 time steps and obtain the simulation results as follows. The state curves are shown in Figs. 2.9, 2.10, and the corresponding control inputs are shown in Fig. 2.11. It can be seen that the closed-loop system is divergent when using the control law obtained by $L = 2$, and the system's responses are improved when the outer loop iteration number L is increased. When $L > 200$, the system

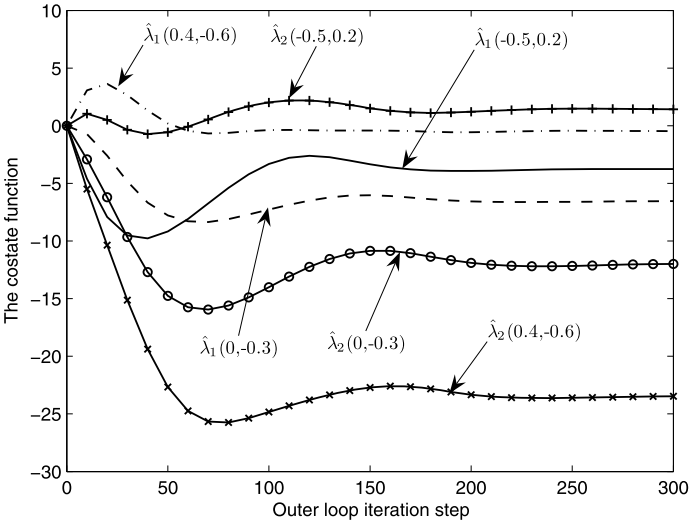

Fig. 2.8 The convergence process of the costate function at $x = (-0.5, 0.2)$, $x = (0.4, -0.6)$, $x = (0, -0.3)$

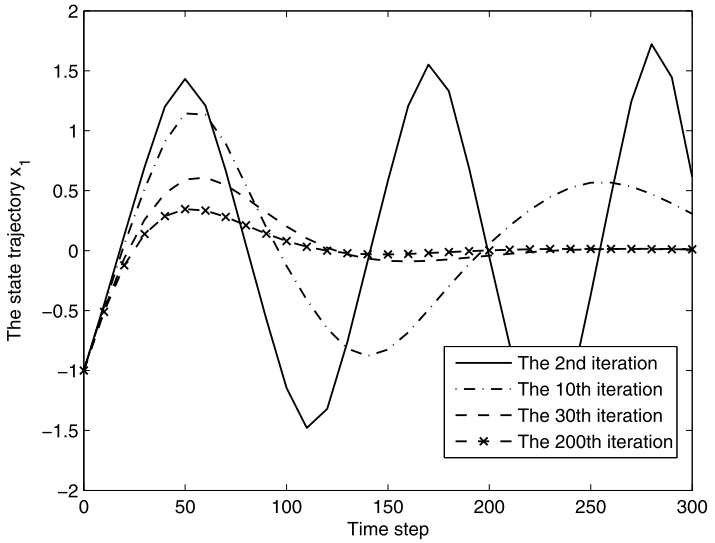

Fig. 2.9 The state trajectory x_1 for $L = 2, 10, 30, 200$

responses basically remain unchanged with no significant improvement in performance.

In order to make comparison with the controller without considering the actuator saturation, we also present the controller designed by iterative ADP algorithm

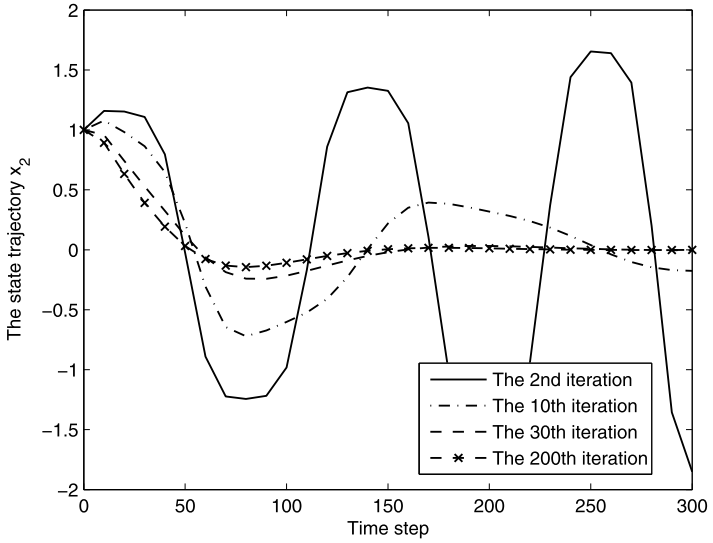

Fig. 2.10 The state trajectory x_2 for $L = 2, 10, 30, 200$

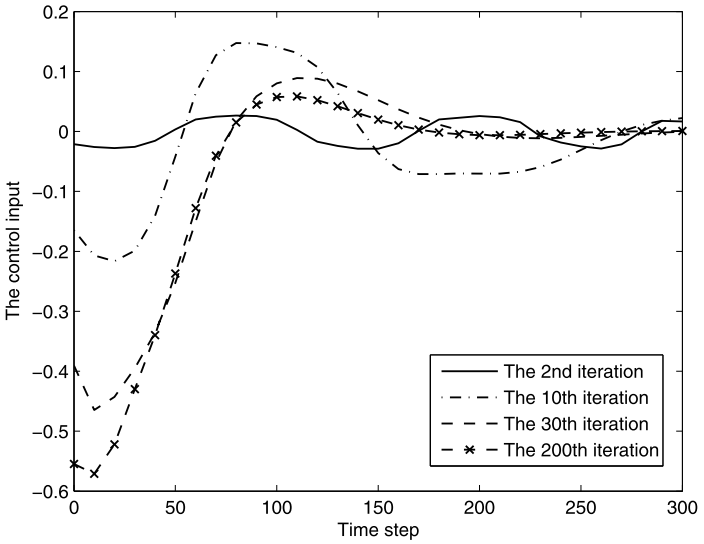

Fig. 2.11 The control input u for $L = 2, 10, 30, 200$

regardless of the actuator saturation. The state curves are shown in Fig. 2.12 and the control curve is shown in Fig. 2.13.

From the simulation results, we can see that the iterative costate function sequence does converge to the optimal one very fast. Comparing Fig. 2.11 with

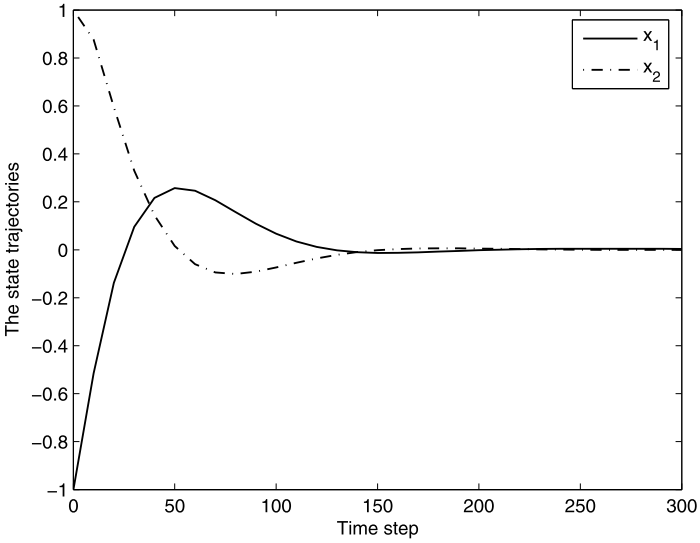

Fig. 2.12 The state curves without considering the actuator saturation in controller design

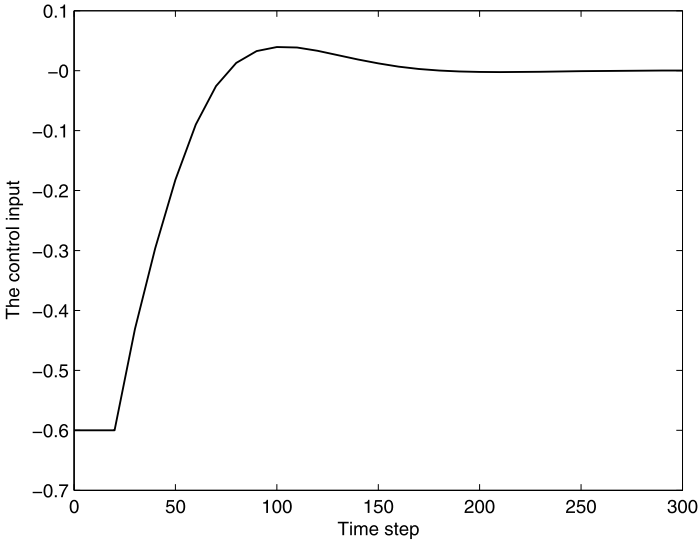

Fig. 2.13 The control curves without considering the actuator saturation in controller design

Fig. 2.13, we can find that in Fig. 2.11 the restriction of actuator saturation has been overcome successfully, which further verifies the effectiveness of the present iterative ADP algorithm.

2.3 Infinite-Horizon Optimal State Feedback Control Based on GDHP

2.3.1 Problem Formulation

In this section, we will study the discrete-time nonlinear systems described by

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{2.69}$$

where $x(k) \in \mathbb{R}^n$ is the state vector and $u(k) \in \mathbb{R}^m$ is the control vector, $f(\cdot)$ and $g(\cdot)$ are differentiable in their arguments with $f(0) = 0$. Assume that $f + gu$ is Lipschitz continuous on a set Ω in $\mathbb{R}^n$ containing the origin, and that the system (2.69) is controllable in the sense that there exists a continuous control law on Ω that asymptotically stabilizes the system.

Let $x(0)$ be an initial state and define $\underline{u}_0^{N-1} = (u(0), u(1), u(N-1))$ be a control sequence with which the system (2.69) gives a trajectory starting from $x(0)$: $x(1) = f(x(0)) + g(x(0))u(0), x(2) = f(x(1)) + g(x(1))u(1), \ldots, x(N) = f(x(N-1)) + g(x(N-1))u(N-1)$. We call the number of elements in the control sequence $\underline{u}_0^{N-1}$ the length of $\underline{u}_0^{N-1}$ and denote it as $|\underline{u}_0^{N-1}|$. Then, $|\underline{u}_0^{N-1}| = N$. The final state under the control sequence $\underline{u}_0^{N-1}$ can be denoted $x^{(f)}(x(0), \underline{u}_0^{N-1}) = x(N)$. When the control sequence starting from $u(0)$ has infinite length, we denote it as $\underline{u}_0^{\infty} = (u(0), u(1), \ldots)$ and then the correspondingly final state can be written as $x^{(f)}(x(0), \underline{u}_0^{\infty}) = \lim_{k\to\infty} x(k)$.

Definition 2.11 A nonlinear dynamical system is said to be stabilizable on a compact set $\Omega \in \mathbb{R}^n$, if for all initial conditions $x(0) \in \Omega$, there exists a control sequence $\underline{u}_0^{\infty} = (u(0), u(1), \ldots)$, $u(i) \in \mathbb{R}^m$, $i = 0, 1, \ldots$, such that the state $x^{(f)}(x(0), \underline{u}_0^{\infty}) = 0$.

Let $\underline{u}_k^{\infty} = (u(k), u(k+1), \ldots)$ be the control sequence starting at k. It is desired to find the control sequence $\underline{u}_k^{\infty}$ which minimizes the infinite-horizon cost functional given by

$$J(x(k), \underline{u}_k^{\infty}) = \sum_{i=k}^{\infty} \gamma^{i-k} l(x(i), u(i)), \tag{2.70}$$

where l is the utility function, $l(0, 0) = 0$, $l(x(i), u(i)) \geq 0$ for $\forall x(i), u(i)$, and γ is the discount factor with $0 < \gamma \leq 1$. Generally speaking, the utility function can be chosen as the quadratic form as follows:

$$l(x(i), u(i)) = x^{\mathrm{T}}(i) Q x(i) + u^{\mathrm{T}}(i) R u(i).$$

For optimal control problems, the designed feedback control must not only stabilize the system on Ω but also guarantee that (2.70) is finite, i.e., the control must be admissible.

It is noted that a control law sequence $\{\eta_i\} = (\eta_N, \ldots, \eta_1, \eta_0)$, $N \to \infty$, is called admissible if the resultant control sequence $(u(0), u(1), \ldots, u(N))$ stabilizes system (2.69) with any initial state $x(0)$ and guarantees that $J(x(0), \underline{u}_0^N)$ is finite. In this case, it should be mentioned that each control action obeys a different control law, i.e., the control action $u(i)$ is produced by the control law η_{N-i} or $u(i) = \eta_{N-i}(x(i))$, for $i = 0, 1, \ldots, N$, $N \to \infty$.

Let

$$\mathfrak{A}_{x(k)} = \left\{ \underline{u}_k^\infty : x^{(f)}(x(k), \underline{u}_k^\infty) = 0 \right\}$$

be the set of all *infinite-horizon* admissible control sequences of $x(k)$. Define the optimal value function as

$$J^*(x(k)) = \inf_{\underline{u}_k^\infty} \left\{ J(x(k), \underline{u}_k^\infty) : \underline{u}_k^\infty \in \mathfrak{A}_{x(k)} \right\}. \tag{2.71}$$

Note that (2.70) can be written as

$$\begin{aligned} J(x(k), \underline{u}_k^\infty) &= x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma \sum_{i=k+1}^{\infty} \gamma^{i-k-1} l(x(i), u(i)) \\ &= x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma J(x(k+1), \underline{u}_{k+1}^\infty). \end{aligned} \tag{2.72}$$

According to Bellman's optimality principle, it is known that, for the case of infinite-horizon optimization, the optimal value function $J^*(x(k))$ is time invariant and satisfies the DTHJB equation

$$J^*(x(k)) = \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma J^*(x(k+1)) \right\}. \tag{2.73}$$

The optimal control u^* satisfies the first-order necessary condition, which is given by the gradient of the right hand side of (2.73) with respect to $u(k)$ as

$$\frac{\partial \left(x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) \right)}{\partial u(k)} + \gamma \left(\frac{\partial x(k+1)}{\partial u(k)} \right)^{\mathrm{T}} \frac{\partial J^*(x(k+1))}{\partial x(k+1)} = 0.$$

Then, we obtain

$$u^*(x(k)) = -\frac{\gamma}{2} R^{-1} g^{\mathrm{T}}(x(k)) \frac{\partial J^*(x(k+1))}{\partial x(k+1)}. \tag{2.74}$$

By substituting (2.74) into (2.73), the DTHJB equation becomes

$$\begin{aligned} J^*(x(k)) = x^{\mathrm{T}}(k) Q x(k) &+ \frac{\gamma^2}{4} \left(\frac{\partial J^*(x(k+1))}{\partial x(k+1)} \right)^{\mathrm{T}} g(x(k)) R^{-1} \\ &\times g^{\mathrm{T}}(x(k)) \frac{\partial J^*(x(k+1))}{\partial x(k+1)} + \gamma J^*(x(k+1)) \end{aligned} \tag{2.75}$$

where $J^*(x(k))$ is the optimal value function corresponding to the optimal control law $u^*(x(k))$. When dealing with the linear quadratic regulator (LQR) optimal control problems, this equation reduces to the Riccati equation which can be efficiently solved. In the general nonlinear case, however, the HJB equation cannot be solved exactly.

2.3.2 Infinite-Horizon Optimal State Feedback Control Based on GDHP

Four parts are included in this subsection. In the first part, the unknown nonlinear system is identified via an NN system identification scheme with stability proof. The iterative ADP algorithm is introduced in the second part, while in the third part, the corresponding convergence proof is developed. Then, in the fourth part, the implementation of the iterative ADP algorithm based on NN is described in detail.

2.3.2.1 NN Identification of the Unknown Nonlinear System

For the design of the NN identifier, a three-layer NN is considered as the function approximation structure. Let the number of hidden-layer neurons be denoted by l, the ideal weight matrix between the input layer and hidden layer be denoted by ν_m^*, and the ideal weight matrix between the hidden layer and output layer be denoted by ω_m^*. According to the universal approximation property [8] of NN, the system dynamics (2.69) has a NN representation on a compact set S, which can be written as

$$x(k+1) = \omega_m^{*\mathrm{T}} \sigma\left(\nu_m^{*\mathrm{T}} z(k)\right) + \theta(k). \tag{2.76}$$

In (2.76), $z(k) = [x^{\mathrm{T}}(k)\ u^{\mathrm{T}}(k)]^{\mathrm{T}}$ is the NN input, $\theta(k)$ is the bounded NN functional approximation error according to the universal approximation property, and $[\sigma(\bar{z})]_i = (e^{\bar{z}_i} - e^{-\bar{z}_i})/(e^{\bar{z}_i} + e^{-\bar{z}_i})$, $i = 1, 2, \ldots, l$, are the activation functions selected in this work, where $\bar{z}(k) = \nu_m^{*\mathrm{T}} z(k)$, $\bar{z}(k) \in \mathbb{R}^l$. Additionally, the NN activation functions are bounded such that $\|\sigma(\bar{z}(k))\| \leq \sigma_M$ for a constant σ_M.

In the system identification process, we keep the weight matrix between the input layer and the hidden layer as constant while only tune the weight matrix between the hidden layer and the output layer. So, we define the NN system identification scheme as

$$\hat{x}(k+1) = \omega_m^{\mathrm{T}}(k) \sigma(\bar{z}(k)), \tag{2.77}$$

where $\hat{x}(k)$ is the estimated system state vector, and $\omega_m(k)$ is the estimation of the constant ideal weight matrix ω_m^*.

Denote $\tilde{x}(k) = \hat{x}(k) - x(k)$ as the system identification error. Combining (2.76) and (2.77), we can obtain the identification error dynamics as

$$\tilde{x}(k+1) = \tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}(k)) - \theta(k), \tag{2.78}$$

where $\tilde{\omega}_m(k) = \omega_m(k) - \omega_m^*$. Let $\psi(k) = \tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}(k))$. Then, (2.78) can be rewritten as

$$\tilde{x}(k+1) = \psi(k) - \theta(k). \tag{2.79}$$

The weights in the system identification process are updated to minimize the following performance measure:

$$E(k+1) = \frac{1}{2}\tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k+1). \tag{2.80}$$

Using the gradient-based adaptation rule, the weights can be updated as

$$\begin{aligned}\omega_m(k+1) &= \omega_m(k) - \alpha_m\left[\frac{\partial E(k+1)}{\partial \omega_m(k)}\right] \\ &= \omega_m(k) - \alpha_m\sigma(\bar{z}(k))\tilde{x}^{\mathrm{T}}(k+1),\end{aligned} \tag{2.81}$$

where $\alpha_m > 0$ is the NN learning rate.

We now give the following assumption before presenting the asymptotic stability proof of the state estimation error $\tilde{x}(k)$.

Assumption 2.12 The NN approximation error term $\theta(k)$ is assumed to be upper bounded by a function of the state estimation error $\tilde{x}(k)$ such that

$$\theta^{\mathrm{T}}(k)\theta(k) \le \theta_{Mk} = \delta\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k), \tag{2.82}$$

where δ is the constant target value with δ_M as its upper bound, i.e., $\|\delta\| \le \delta_M$.

Next, the stability analysis of the present NN-based system identification scheme is presented by using the Lyapunov theory.

Theorem 2.13 (cf. [10]) *Let the identification scheme* (2.77) *be used to identify the nonlinear system* (2.69), *and let the parameter update law given in* (2.81) *be used for tuning the NN weights. Then, the state estimation error dynamics* $\tilde{x}(k)$ *is asymptotically stable while the parameter estimation error* $\tilde{\omega}_m(k)$ *is bounded.*

Proof Consider the following positive definite Lyapunov function candidate:

$$L_k = L_{1k} + L_{2k}, \tag{2.83}$$

where

$$L_{1k} = \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k),$$

$$L_{2k} = \frac{1}{\alpha_m}\mathrm{tr}\{\tilde{\omega}_m^{\mathrm{T}}(k)\tilde{\omega}_m(k)\}.$$

Taking the first difference of the Lyapunov function (2.83) and substituting the identification error dynamics (2.79) and the NN weight update law (2.81) reveal that

$$\begin{aligned}
\Delta L_{1k} &= \tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k+1) - \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) \\
&= \psi^{\mathrm{T}}(k)\psi(k) - 2\psi^{\mathrm{T}}(k)\theta(k) + \theta^{\mathrm{T}}(k)\theta(k) - \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) \\
\Delta L_{2k} &= \frac{1}{\alpha_m}\mathrm{tr}\{\tilde{\omega}_m^{\mathrm{T}}(k+1)\tilde{\omega}_m(k+1) - \tilde{\omega}_m^{\mathrm{T}}(k)\tilde{\omega}_m(k)\} \\
&= \frac{1}{\alpha_m}\mathrm{tr}\{-2\alpha_m\psi(k)\tilde{x}^{\mathrm{T}}(k+1) \\
&\quad + \alpha_m^2\tilde{x}(k+1)\sigma^{\mathrm{T}}(\bar{z}(k))\sigma(\bar{z}(k))\tilde{x}^{\mathrm{T}}(k+1)\} \\
&= -2\psi^{\mathrm{T}}(k)\tilde{x}(k+1) + \alpha_m\sigma^{\mathrm{T}}(\bar{z}(k))\sigma(\bar{z}(k))\tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k+1).
\end{aligned}$$

After applying the Cauchy–Schwarz inequality $((a_1 + a_2 + \cdots + a_n)^{\mathrm{T}}(a_1 + a_2 + \cdots + a_n) \le n(a_1^{\mathrm{T}}a_1 + a_2^{\mathrm{T}}a_2 + \cdots + a_n^{\mathrm{T}}a_n))$ to ΔL_{2k}, we have

$$\begin{aligned}
\Delta L_{2k} \le &-2\psi^{\mathrm{T}}(k)(\psi(k) - \theta(k)) \\
&+ 2\alpha_m\sigma^{\mathrm{T}}(\bar{z}(k))\sigma(\bar{z}(k))\big(\psi^{\mathrm{T}}(k)\psi(k) + \theta^{\mathrm{T}}(k)\theta(k)\big).
\end{aligned}$$

Therefore, we can find that

$$\begin{aligned}
\Delta L_k \le &-\psi^{\mathrm{T}}(k)\psi(k) + \theta^{\mathrm{T}}(k)\theta(k) - \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) \\
&+ 2\alpha_m\sigma^{\mathrm{T}}(\bar{z}(k))\sigma(\bar{z}(k))\big(\psi^{\mathrm{T}}(k)\psi(k) + \theta^{\mathrm{T}}(k)\theta(k)\big).
\end{aligned}$$

Considering $\|\sigma(\bar{z}(k))\| \le \sigma_M$ and (2.82), we obtain

$$\begin{aligned}
\Delta L_k \le &-\big(1 - 2\alpha_m\sigma_M^2\big)\|\psi(k)\|^2 \\
&- \big(1 - \delta_M - 2\alpha_m\delta_M\sigma_M^2\big)\|\tilde{x}(k)\|^2.
\end{aligned} \tag{2.84}$$

Define $\alpha_m \le \rho^2/(2\sigma_M^2)$; then (2.84) becomes

$$\begin{aligned}
\Delta L_k &\le -\big(1 - \rho^2\big)\|\psi(k)\|^2 - \big(1 - \delta_M - \delta_M\rho^2\big)\|\tilde{x}(k)\|^2 \\
&= -\big(1 - \rho^2\big)\big\|\tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}(k))\big\|^2 \\
&\quad - \big(1 - \delta_M - \delta_M\rho^2\big)\|\tilde{x}(k)\|^2.
\end{aligned} \tag{2.85}$$

From (2.85), we can conclude that $\Delta L_k \le 0$ provided $0 < \delta_M < 1$ and

$$\max\left\{-\sqrt{\frac{1-\delta_M}{\delta_M}}, -1\right\} \le \rho \le \min\left\{\sqrt{\frac{1-\delta_M}{\delta_M}}, 1\right\},$$

where $\rho \neq 0$. As long as the parameters are selected as discussed above, $\Delta L_k \leq 0$ in (2.85), which shows stability in the sense of Lyapunov. Therefore, $\tilde{x}(k)$ and $\tilde{\omega}_m(k)$ are bounded, provided $\tilde{x}_0$ and $\tilde{\omega}_m(0)$ are bounded in the compact set S. Furthermore, by summing both sides of (2.85) to infinity and taking account of $\Delta L_k \leq 0$, we have

$$\left| \sum_{k=0}^{\infty} \Delta L_k \right| = \left| \lim_{k\to\infty} L_k - L_0 \right| < \infty.$$

This implies that

$$\sum_{k=0}^{\infty} \left\{ \left(1-\rho^2\right) \left\| \tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}(k)) \right\|^2 + \left(1-\delta_M-\delta_M\rho^2\right) \|\tilde{x}(k)\|^2 \right\} < \infty.$$

Hence, it can be concluded that the estimation error approaches zero, i.e., $\|\tilde{x}(k)\| \to 0$ as $k \to \infty$. □

Remark 2.14 According to Theorem 2.13, after a sufficient learning session, the NN system identification error converges to zero, i.e., we have

$$f(x(k)) + \hat{g}(x(k))u(k) = \omega_m^{\mathrm{T}}(k)\sigma(\bar{z}(k)), \tag{2.86}$$

where $\hat{g}(x(k))$ denotes the estimated value of the control coefficient matrix $g(x(k))$. Taking the partial derivative of both sides of (2.86) with respect to $u(k)$ yields

$$\begin{aligned} \hat{g}(x(k)) &= \frac{\partial\left(\omega_m^{\mathrm{T}}(k)\sigma(\bar{z}(k))\right)}{\partial u(k)} \\ &= \omega_m^{\mathrm{T}}(k)\frac{\partial\sigma(\bar{z}(k))}{\partial\bar{z}(k)}\nu_m^{*\mathrm{T}}\frac{\partial z(k)}{\partial u(k)}, \end{aligned} \tag{2.87}$$

where

$$\frac{\partial z(k)}{\partial u(k)} = \begin{bmatrix} 0_{n\times m} \\ \hline I_m \end{bmatrix},$$

and I_m is the $m \times m$ identity matrix.

Next, this result will be used in the derivation and implementation of the iterative ADP algorithm for the optimal control of unknown discrete-time nonlinear systems.

2.3.2.2 Derivation of the Iterative ADP Algorithm

In this part, we mainly present the iterative ADP algorithm. First, we start with the initial value function $V_0(\cdot) = 0$, and then solve for the law of single control vector $v_0(x(k))$ as follows:

$$v_0(x(k)) = \arg\min_{u(k)} \left\{ x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_0(x(k+1)) \right\}. \tag{2.88}$$

Once the control law $v_0(x(k))$ is determined, we update the cost function as

$$\begin{aligned} V_1(x(k)) &= \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_0(x(k+1)) \right\} \\ &= x^{\mathrm{T}}(k) Q x(k) + v_0^{\mathrm{T}}(x(k)) R v_0(x(k)). \end{aligned} \tag{2.89}$$

Therefore, for $i = 1, 2, \ldots$, the iterative ADP algorithm can be used to implement the iteration between the control law

$$\begin{aligned} v_i(x(k)) &= \arg\min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_i(x(k+1)) \right\} \\ &= -\frac{\gamma}{2} R^{-1} \hat{g}^{\mathrm{T}}(x(k)) \frac{\partial V_i(x(k+1))}{\partial x(k+1)} \end{aligned} \tag{2.90}$$

and the value function

$$\begin{aligned} V_{i+1}(x(k)) &= \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_i(x(k+1)) \right\} \\ &= x^{\mathrm{T}}(k) Q x(k) + v_i^{\mathrm{T}}(x(k)) R v_i(x(k)) + \gamma V_i(x(k+1)). \end{aligned} \tag{2.91}$$

In the above recurrent iteration, i is the iteration index of the control law and value function, while k is the time index of the system's control and state trajectories. The value function and control law are updated until they converge to the optimal ones. In the following part, we will present a proof of convergence of the iteration between (2.90) and (2.91) with the value function $V_i \to J^*$ and the control law $v_i \to u^*$ as $i \to \infty$.

2.3.2.3 Convergence Analysis of the Iterative ADP Algorithm

Lemma 2.15 *Let $\{\mu_i\}$ be an arbitrary sequence of control laws and $\{v_i\}$ be the control law sequence described in (2.90). Define V_i as in (2.91) and Λ_i as*

$$\Lambda_{i+1}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + \mu_i^{\mathrm{T}}(x(k)) R \mu_i(x(k)) + \gamma \Lambda_i(x(k+1)). \tag{2.92}$$

If $V_0(x(k)) = \Lambda_0(x(k)) = 0$, then $V_i(x(k)) \le \Lambda_i(x(k))$, $\forall i$.

Proof It can easily be derived noticing that V_{i+1} is the result of minimizing the right hand side of (2.91) with respect to the control input $u(k)$, while Λ_{i+1} is a result of arbitrary control input. □

Lemma 2.16 *Let the value function sequence $\{V_i\}$ be defined as in (2.91). If the system is controllable, then there is an upper bound Y such that $0 \le V_i(x(k)) \le Y$, $\forall i$.*

Proof Let $\{\eta_i(x)\}$ be a sequence of admissible control laws, and let $V_0(\cdot) = Z_0(\cdot) = 0$, where V_i is updated as in (2.91) and Z_i is updated by

$$Z_{i+1}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + \eta_i^{\mathrm{T}}(x(k))R\eta_i(x(k)) + \gamma Z_i(x(k+1)). \tag{2.93}$$

It is clear that

$$\begin{aligned} Z_i(x(k+1)) = {} & x^{\mathrm{T}}(k+1)Qx(k+1) + \eta_{i-1}^{\mathrm{T}}(x(k+1))R\eta_{i-1}(x(k+1)) \\ & + \gamma Z_{i-1}(x(k+2)). \end{aligned} \tag{2.94}$$

Noticing that $l(x(k), \eta_i(x(k))) = x^{\mathrm{T}}(k)Qx(k) + \eta_i^{\mathrm{T}}(x(k))R\eta_i(x(k))$, we can further obtain

$$\begin{aligned} Z_{i+1}(x(k)) = {} & l(x(k), \eta_i(x(k))) + \gamma l(x(k+1), \eta_{i-1}(x(k+1))) \\ & + \gamma^2 Z_{i-1}(x(k+2)) \\ = {} & l(x(k), \eta_i(x(k))) + \gamma l(x(k+1), \eta_{i-1}(x(k+1))) \\ & + \gamma^2 l(x(k+2), \eta_{i-2}(x(k+2))) + \gamma^3 Z_{i-2}(x(k+3)) \\ & \vdots \\ = {} & l(x(k), \eta_i(x(k))) + \gamma l(x(k+1), \eta_{i-1}(x(k+1))) \\ & + \gamma^2 l(x(k+2), \eta_{i-2}(x(k+2))) \\ & + \cdots + \gamma^i l(x(k+i), \eta_0(x(k+i))) \\ & + \gamma^{i+1} Z_0(x(k+i+1)), \end{aligned} \tag{2.95}$$

where $Z_0(x(k+i+1)) = 0$. Then, (2.95) can be written as

$$\begin{aligned} Z_{i+1}(x(k)) = {} & \sum_{j=0}^{i} \gamma^j l(x(k+j), \eta_{i-j}(x(k+j))) \\ = {} & \sum_{j=0}^{i} \gamma^j \left(x^{\mathrm{T}}(k+j)Qx(k+j) + \eta_{i-j}^{\mathrm{T}}(x(k+j))R\eta_{i-j}(x(k+j))\right) \\ \le {} & \lim_{i\to\infty} \sum_{j=0}^{i} \gamma^j \big(x^{\mathrm{T}}(k+j)Qx(k+j) \\ & + \eta_{i-j}^{\mathrm{T}}(x(k+j))R\eta_{i-j}(x(k+j))\big). \end{aligned} \tag{2.96}$$

Since $\{\eta_i(x)\}$ is an admissible control law sequence, we have $x(k) \to 0$ as $k \to \infty$, and there exists an upper bound Y such that

$$Z_{i+1}(x(k)) \le \lim_{i\to\infty} \sum_{j=0}^{i} \gamma^j l(x(k+j), \eta_{i-j}(x(k+j))) \le Y, \ \forall i. \tag{2.97}$$

By using Lemma 2.15, we obtain

$$V_{i+1}(x(k)) \le Z_{i+1}(x(k)) \le Y, \ \forall i. \tag{2.98}$$

□

Based on Lemmas 2.15 and 2.16, we now present our main theorems.

Theorem 2.17 *Define the value function sequence* $\{V_i\}$ *as in* (2.91) *with* $V_0(\cdot) = 0$, *and the control law sequence* $\{v_i\}$ *as in* (2.90). *Then,* $\{V_i\}$ *is a monotonically nondecreasing sequence satisfying* $V_{i+1} \ge V_i$, $\forall i$.

Proof Define a new sequence

$$\Phi_{i+1}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + v_{i+1}^{\mathrm{T}}(x(k))Rv_{i+1}(x(k)) + \gamma\Phi_i(x(k+1)) \tag{2.99}$$

with $\Phi_0(\cdot) = V_0(\cdot) = 0$. Let the control law sequence $\{v_i\}$ and the value function sequence $\{V_i\}$ be updated as in (2.90) and (2.91), respectively.

In the following part, we prove that $\Phi_i(x(k)) \le V_{i+1}(x(k))$ by mathematical induction.

First, we prove that it holds for $i = 0$. Considering

$$V_1(x(k)) - \Phi_0(x(k)) = x^{\mathrm{T}}(k)Qx(k) + v_0^{\mathrm{T}}(x(k))Rv_0(x(k)) \ge 0$$

then, for $i = 0$, we get

$$V_1(x(k)) \ge \Phi_0(x(k)). \tag{2.100}$$

Second, we assume that it holds for $i-1$, i.e., $V_i(x(k)) \ge \Phi_{i-1}(x(k))$, $\forall x(k)$. Then, for i, noticing that

$$V_{i+1}(x(k)) = x^{\mathrm{T}}(k)Qx(k) + v_i^{\mathrm{T}}(x(k))Rv_i(x(k)) + \gamma V_i(x(k+1))$$

and

$$\Phi_i(x(k)) = x^{\mathrm{T}}(k)Qx(k) + v_i^{\mathrm{T}}(x(k))Rv_i(x(k)) + \gamma\Phi_{i-1}(x(k+1)),$$

we get

$$V_{i+1}(x(k)) - \Phi_i(x(k)) = \gamma(V_i(x(k+1)) - \Phi_{i-1}(x(k+1))) \ge 0$$

i.e.,

$$V_{i+1}(x(k)) \geq \Phi_i(x(k)). \tag{2.101}$$

Thus, we complete the proof through mathematical induction.

Furthermore, from Lemma 2.15 we know that $V_i(x(k)) \leq \Phi_i(x(k))$, therefore, we have

$$V_{i+1}(x(k)) \geq \Phi_i(x(k)) \geq V_i(x(k)). \tag{2.102}$$

□

We have reached the conclusion that the value function sequence $\{V_i\}$ is a monotonically nondecreasing sequence with an upper bound, and therefore, its limit exists. Now, we can derive the following theorem.

Theorem 2.18 *For any state vector $x(k)$, define*

$$\lim_{i\to\infty} V_i(x(k)) = V_\infty(x(k))$$

as the limit of the value function sequence $\{V_i\}$. Then, the following equation holds:

$$V_\infty(x(k)) = \min_{u(k)} \left\{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_\infty(x(k+1))\right\}.$$

Proof For any $u(k)$ and i, according to (2.91), we can derive

$$V_i(x(k)) \leq x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_{i-1}(x(k+1)).$$

Combining with

$$V_i(x(k)) \leq V_\infty(x(k)),\ \forall i \tag{2.103}$$

which is obtained from Theorem 2.17, we have

$$V_i(x(k)) \leq x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_\infty(x(k+1)),\ \forall i.$$

Let $i \to \infty$, we can acquire

$$V_\infty(x(k)) \leq x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_\infty(x(k+1)).$$

Note that in the above equation, $u(k)$ is chosen arbitrarily; thus, we obtain

$$V_\infty(x(k)) \leq \min_{u(k)} \left\{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_\infty(x(k+1))\right\}. \tag{2.104}$$

On the other hand, since the value function sequence satisfies

$$V_i(x(k)) = \min_{u(k)} \left\{x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k) + \gamma V_{i-1}(x(k+1))\right\}$$

for any i, considering (2.103), we have

$$V_\infty(x(k)) \geq \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_{i-1}(x(k+1)) \right\}, \ \forall i.$$

Let $i \to \infty$; we get

$$V_\infty(x(k)) \geq \min_{u(k)} \left\{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_\infty(x(k+1)) \right\}. \tag{2.105}$$

Based on (2.104) and (2.105), we can acquire the conclusion that $V_\infty(x(k)) = \min_{u(k)} \{ x^{\mathrm{T}}(k) Q x(k) + u^{\mathrm{T}}(k) R u(k) + \gamma V_\infty(x(k+1)) \}$. □

Next, we will prove that the value function sequence $\{V_i\}$ converges to the optimal value function $J^*(x(k))$ as $i \to \infty$.

Theorem 2.19 (cf. [10]) *Define the value function sequence $\{V_i\}$ as in (2.91) with $V_0(\cdot) = 0$. If the system state $x(k)$ is controllable, then J^* is the limit of the value function sequence $\{V_i\}$, i.e.,*

$$\lim_{i \to \infty} V_i(x(k)) = J^*(x(k)).$$

Proof Let $\{\eta_i^{(l)}\}$ be the lth admissible control law sequence. We construct the associated sequence $\{P_i^{(l)}(x)\}$ as follows:

$$P_{i+1}^{(l)}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + \eta_i^{(l)\mathrm{T}}(x(k)) R \eta_i^{(l)}(x(k)) + \gamma P_i^{(l)}(x(k+1)) \tag{2.106}$$

with $P_0^{(l)}(\cdot) = 0$. Similar to the derivation of (2.95), we get

$$P_{i+1}^{(l)}(x(k)) = \sum_{j=0}^{i} \gamma^j l\big(x(k+j), \eta_{i-j}^{(l)}(x(k+j))\big). \tag{2.107}$$

Using Lemmas 2.15 and 2.16, we have

$$V_{i+1}(x(k)) \leq P_{i+1}^{(l)}(x(k)) \leq Y_l, \ \forall l, i \tag{2.108}$$

where Y_l is the upper bound associated with the sequence $\{P_{i+1}^{(l)}(x(k))\}$. Denote

$$\lim_{i \to \infty} P_i^{(l)}(x(k)) = P_\infty^{(l)}(x(k));$$

then, we obtain

$$V_\infty(x(k)) \leq P_\infty^{(l)}(x(k)) \leq Y_l, \ \forall l. \tag{2.109}$$

Let the corresponding control sequence associated with (2.107) be

$$\begin{aligned}{}^{(l)}\underline{\hat{u}}_k^{k+i} &= \big({}^{(l)}\hat{u}(k), {}^{(l)}\hat{u}(k+1), \ldots, {}^{(l)}\hat{u}_{k+i}\big) \\ &= \big(\eta_i^{(l)}(x(k)), \eta_{i-1}^{(l)}(x(k+1)), \ldots, \eta_0^{(l)}(x(k+i))\big);\end{aligned}$$

then we have

$$J\big(x(k), {}^{(l)}\underline{\hat{u}}_k^{k+i}\big) = \sum_{j=0}^{i} \gamma^j l\big(x(k+j), \eta_{i-j}^{(l)}(x(k+j))\big) = P_{i+1}^{(l)}(x(k)). \tag{2.110}$$

Letting $i \to \infty$, and denoting the admissible control sequence related to $P_\infty^{(l)}(x(k))$ with length ∞ as ${}^{(l)}\underline{\hat{u}}_k^\infty$, we get

$$J\big(x(k), {}^{(l)}\underline{\hat{u}}_k^\infty\big) = \sum_{j=0}^{\infty} \gamma^j l\big(x(k+j), {}^{(l)}\hat{u}(k+j)\big) = P_\infty^{(l)}(x(k)). \tag{2.111}$$

Then, according to the definition of $J^*(x(k))$ in (2.71), for any $\varepsilon > 0$, there exists a sequence of admissible control laws $\{\eta_i^{(M)}\}$ such that the associated cost function

$$J\big(x(k), {}^{(M)}\underline{\hat{u}}_k^\infty\big) = \sum_{j=0}^{\infty} \gamma^j l\big(x(k+j), {}^{(M)}\hat{u}(k+j)\big) = P_\infty^{(M)}(x(k)) \tag{2.112}$$

satisfies $J(x(k), {}^{(M)}\underline{\hat{u}}_k^\infty) \le J^*(x(k)) + \varepsilon$. Combining with (2.109), we have

$$V_\infty(x(k)) \le P_\infty^{(M)}(x(k)) \le J^*(x(k)) + \varepsilon. \tag{2.113}$$

Since ε is chosen arbitrarily, we get

$$V_\infty(x(k)) \le J^*(x(k)). \tag{2.114}$$

On the other hand, because $V_{i+1}(x(k)) \le P_{i+1}^{(l)}(x(k)) \le Y_l, \forall l, i$, we can get $V_\infty(x(k)) \le \inf_l\{Y_l\}$. According to the definition of admissible control law sequence, the control law sequence associated with the cost function $V_\infty(x(k))$ must be an admissible control law sequence. We can see that there exists a sequence of admissible control laws $\{\eta_i^{(N)}\}$ such that $V_\infty(x(k)) = P_\infty^{(N)}(x(k))$. Combining with (2.111), we get $V_\infty(x(k)) = J(x(k), {}^{(N)}\underline{\hat{u}}_k^\infty)$. Sine $J^*(x(k))$ is the infimum of all admissible control sequences starting at k with length ∞, we obtain

$$V_\infty(x(k)) \ge J^*(x(k)). \tag{2.115}$$

Based on (2.114) and (2.115), we can conclude that J^* is the limit of the value function sequence $\{V_i\}$, i.e., $V_\infty(x(k)) = J^*(x(k))$. □

From Theorems 2.17 and 2.18, we can derive that the limit of the value function sequence $\{V_i\}$ satisfies the DTHJB equation, i.e., $V_\infty(x(k)) = \min_{u(k)}\{x^{\mathrm{T}}(k)Qx(k)$

$+ u^{\mathrm{T}}(k)Ru(k) + \gamma V_\infty(x(k+1))\}$. Besides, from Theorem 2.19, we can get the result that $V_\infty(x(k)) = J^*(x(k))$. Therefore, we can find that the cost function sequence $\{V_i(x(k))\}$ converges to the optimal value function $J^*(x(k))$ of the DTHJB equation, i.e., $V_i \to J^*$ as $i \to \infty$. Then, according to (2.74) and (2.90), we can conclude the convergence of the corresponding control law sequence. Now, we present the following corollary.

Corollary 2.20 *Define the value function sequence* $\{V_i\}$ *as in* (2.91) *with* $V_0(\cdot) = 0$, *and the control law sequence* $\{v_i\}$ *as in* (2.90). *If the system state* $x(k)$ *is controllable, then the sequence* $\{v_i\}$ *converges to the optimal control law* u^* *as* $i \to \infty$, *i.e.*,

$$\lim_{i\to\infty} v_i(x(k)) = u^*(x(k)).$$

Remark 2.21 Like (2.95), when we further expand (2.91), we obtain a control law sequence $(v_i, v_{i-1}, \ldots, v_0)$ and the resultant control sequence $(v_i(x(0)), v_{i-1}(x(1)), \ldots, v_0(x(i)))$. With the iteration number increasing to ∞, the derived control law sequence has the length of ∞. Then, using the corresponding control sequence, we obtain a state trajectory. However, it is not derived from a single control law. For infinite-horizon optimal control problem, what we should get is a unique optimal control law under which we can obtain the optimal state trajectory. Therefore, we only use the optimal control law u^* obtained in Corollary 2.20 to produce a control sequence when we apply the algorithm to practical systems.

2.3.2.4 NN Implementation of the Iterative ADP Algorithm Using GDHP Technique

When the controlled system is linear and the cost function is quadratic, we can obtain a linear control law. In the nonlinear case, however, this is not necessarily true. Therefore, we need to use function approximation structure, such as NN, to approximate both $v_i(x(k))$ and $V_i(x(k))$.

Now, we implement the iterative GDHP algorithm in (2.90) and (2.91). In the iterative GDHP algorithm, there are three networks, which are model network, critic network and action network. All the networks are chosen as three-layer feedforward NNs. The input of the critic network and action network is $x(k)$, while the input of the model network is $x(k)$ and $\hat{v}_i(x(k))$. The diagram of the whole structure is shown in Fig. 2.14, where

$$\mathrm{DER} = \left(\frac{\partial \hat{x}(k+1)}{\partial x(k)} + \frac{\partial \hat{x}(k+1)}{\partial \hat{v}_i(x(k))}\frac{\partial \hat{v}_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}}.$$

The training of the model network is completed after the system identification process and its weights are kept unchanged. Then, according to Theorem 2.13, when given $x(k)$ and $\hat{v}_i(x(k))$, we can compute $\hat{x}(k+1)$ by (2.77), i.e.,

$$\hat{x}(k+1) = \omega_m^{\mathrm{T}}(k)\sigma\left(v_m^{*\mathrm{T}}[x^{\mathrm{T}}(k)\ \hat{v}_i^{\mathrm{T}}(x(k))]^{\mathrm{T}}\right).$$

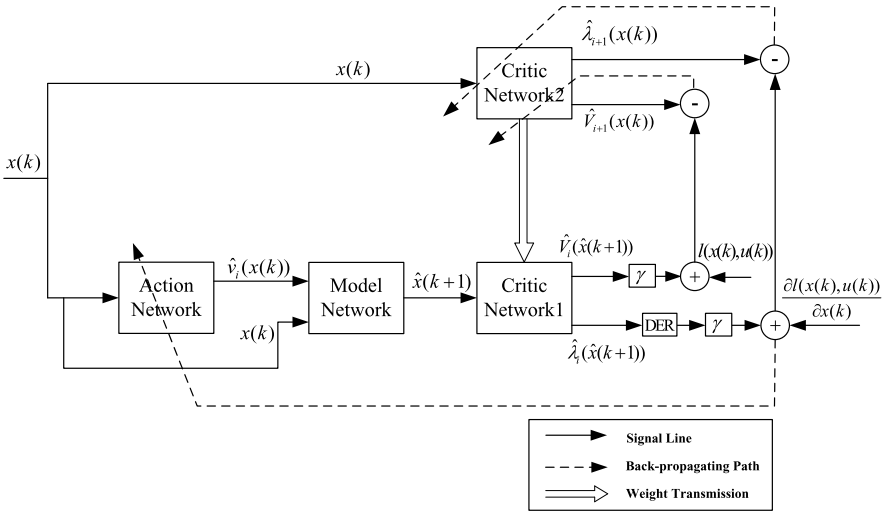

Fig. 2.14 The structure diagram of the iterative GDHP algorithm

As a result, we avoid the requirement of knowing $f(x(k))$ and $g(x(k))$ during the implementation of the iterative GDHP algorithm.

Next, the learned NN system model will be used in the process of training critic network and action network.

The critic network is used to approximate both $V_i(x(k))$ and its derivative $\partial V_i(x(k))/\partial x(k)$, which is named the costate function and denoted $\lambda_i(x(k))$. The output of the critic network is denoted

$$\begin{bmatrix} \hat{V}_i(x(k)) \\ \hat{\lambda}_i(x(k)) \end{bmatrix} = \begin{bmatrix} \omega_{ci}^{1\mathrm{T}} \\ \omega_{ci}^{2\mathrm{T}} \end{bmatrix} \sigma\left(v_{ci}^{\mathrm{T}} x(k)\right) = \omega_{ci}^{\mathrm{T}} \sigma\left(v_{ci}^{\mathrm{T}} x(k)\right), \tag{2.116}$$

where

$$\omega_{ci} = \left[\omega_{ci}^{1} \;\; \omega_{ci}^{2}\right],$$

i.e.,

$$\hat{V}_i(x(k)) = \omega_{ci}^{1\mathrm{T}} \sigma\left(v_{ci}^{\mathrm{T}} x(k)\right) \tag{2.117}$$

and

$$\hat{\lambda}_i(x(k)) = \omega_{ci}^{2\mathrm{T}} \sigma\left(v_{ci}^{\mathrm{T}} x(k)\right). \tag{2.118}$$

The target function can be written as

$$V_{i+1}(x(k)) = x^{\mathrm{T}}(k) Q x(k) + v_i^{\mathrm{T}}(x(k)) R v_i(x(k)) + \gamma \hat{V}_i(\hat{x}(k+1)) \tag{2.119}$$

and

$$\begin{aligned}\lambda_{i+1}(x(k)) &= \frac{\partial\left(x^{\mathrm{T}}(k)Qx(k)+v_i^{\mathrm{T}}(x(k))Rv_i(x(k))\right)}{\partial x(k)}+\gamma\frac{\partial\hat{V}_i(\hat{x}(k+1))}{\partial x(k)}\\ &= 2Qx(k)+2\left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}}Rv_i(x(k))\\ &\quad+\gamma\left(\frac{\partial\hat{x}(k+1)}{\partial x(k)}+\frac{\partial\hat{x}(k+1)}{\partial\hat{v}_i(x(k))}\frac{\partial\hat{v}_i(x(k))}{\partial x(k)}\right)^{\mathrm{T}}\hat{\lambda}_i(\hat{x}(k+1)).\end{aligned}\tag{2.120}$$

Then, we define the error function for training the critic network as

$$e_{cik}^1=\hat{V}_i(x(k))-V_{i+1}(x(k))\tag{2.121}$$

and

$$e_{cik}^2=\hat{\lambda}_i(x(k))-\lambda_{i+1}(x(k)).\tag{2.122}$$

The objective function to be minimized in the critic network training is

$$E_{cik}=(1-\beta)E_{cik}^1+\beta E_{cik}^2,\tag{2.123}$$

where

$$E_{cik}^1=\frac{1}{2}e_{cik}^{1\mathrm{T}}e_{cik}^1\tag{2.124}$$

and

$$E_{cik}^2=\frac{1}{2}e_{cik}^{2\mathrm{T}}e_{cik}^2.\tag{2.125}$$

The weight updating rule for training the critic network is also gradient-based adaptation given by

$$\omega_{ci}(j+1)=\omega_{ci}(j)-\alpha_c\left[(1-\beta)\frac{\partial E_{cik}^1}{\partial\omega_{ci}(j)}+\beta\frac{\partial E_{cik}^2}{\partial\omega_{ci}(j)}\right]\tag{2.126}$$

$$v_{ci}(j+1)=v_{ci}(j)-\alpha_c\left[(1-\beta)\frac{\partial E_{cik}^1}{\partial v_{ci}(j)}+\beta\frac{\partial E_{cik}^2}{\partial v_{ci}(j)}\right]\tag{2.127}$$

where $\alpha_c>0$ is the learning rate of the critic network, j is the inner-loop iteration step for updating the weight parameters, and $0\le\beta\le 1$ is a parameter that adjusts how HDP and DHP are combined in GDHP. For $\beta=0$, the training of the critic network reduces to a pure HDP, while $\beta=1$ does the same for DHP.

In the action network, the state $x(k)$ is used as input to obtain the optimal control. The output can be formulated as

$$\hat{v}_i(x(k))=\omega_{ai}^{\mathrm{T}}\sigma\left(v_{ai}^{\mathrm{T}}x(k)\right).\tag{2.128}$$

The target control input is given as

$$v_i(x(k)) = -\frac{\gamma}{2} R^{-1} \hat{g}^{\mathrm{T}}(x(k)) \frac{\partial \hat{V}_i(\hat{x}(k+1))}{\partial \hat{x}(k+1)}. \tag{2.129}$$

The error function of the action network can be defined as

$$e_{aik} = \hat{v}_i(x(k)) - v_i(x(k)). \tag{2.130}$$

The weights of the action network are updated to minimize the following performance error measure:

$$E_{aik} = \frac{1}{2} e_{aik}^{\mathrm{T}} e_{aik}. \tag{2.131}$$

Similarly, the weight updating algorithm is

$$\omega_{ai}(j+1) = \omega_{ai}(j) - \alpha_a \left[\frac{\partial E_{aik}}{\partial \omega_{ai}(j)} \right], \tag{2.132}$$

$$v_{ai}(j+1) = v_{ai}(j) - \alpha_a \left[\frac{\partial E_{aik}}{\partial v_{ai}(j)} \right] \tag{2.133}$$

where $\alpha_a > 0$ is the learning rate of the action network, and j is the inner-loop iteration step for updating the weight parameters.

Remark 2.22 According to Theorem 2.19, $V_i(x(k)) \to J^*(x(k))$ as $i \to \infty$. Since $\lambda_i(x(k)) = \partial V_i(x(k))/\partial x(k)$, we can conclude that the costate function sequence $\{\lambda_i(x(k))\}$ is also convergent with $\lambda_i(x(k)) \to \lambda^*(x(k))$ as $i \to \infty$.

Remark 2.23 From Fig. 2.14, we can see that the outputs of the critic network of the iterative GDHP algorithm contain not only the cost function but also its derivative. This is important because the information associated with the cost function is as useful as the knowledge of its derivative. Besides, as is shown in (2.126) and (2.127), training the critic network of the iterative GDHP algorithm utilizes an error measure which is a combination of the error measures of HDP and DHP. Though it is more complicated to do this, the resulting behavior is expected to be superior to simple ADP methods.

2.3.3 Simulations

An example is provided in this subsection to demonstrate the effectiveness of the control scheme derived by the iterative GDHP algorithm.

Example 2.24 Consider the following nonlinear system:

$$x(k+1) = f(x(k)) + g(x(k))u(k),$$

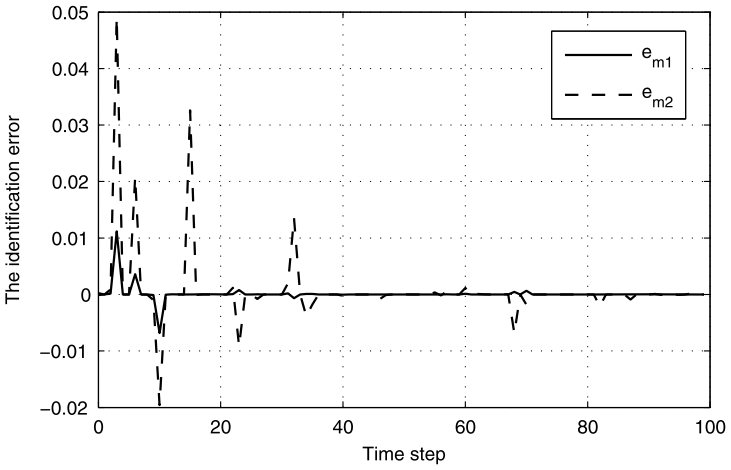

Fig. 2.15 The system identification error ($\tilde{x}_{k1}$ and $\tilde{x}_{k2}$ are denoted e_{m1} and e_{m2}, respectively)

where $x(k) = [x_1(k)\ x_2(k)]^{\mathrm{T}} \in \mathbb{R}^2$ and $u(k) \in \mathbb{R}$ are the state and control variables, respectively. The cost function is chosen as $l(x(k), u(k)) = x^{\mathrm{T}}(k)x(k) + u^{\mathrm{T}}(k)Ru(k)$. The system functions are given as

$$f(x(k)) = \begin{bmatrix} -\sin(0.5x_2(k)) \\ -\cos(1.4x_2(k))\sin(0.9x_1(k)) \end{bmatrix}$$

$$g(x(k)) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We choose three-layer feedforward NNs as model network, critic network and action network with the structures 3–8–2, 2–8–3, 2–8–1, respectively. In the system identification process, the initial weights between the input layer and the hidden layer, and the hidden layer and the output layer are chosen randomly in $[-0.5, 0.5]$ and $[-0.1, 0.1]$, respectively. We apply the NN identification scheme for 100 steps under the learning rate $\alpha_m = 0.05$ and obtain the result as shown in Fig. 2.15. It is clearly observed that the NN identifier successfully learns the unknown nonlinear system. Then, we finish the training of the model network and keep its weights unchanged.

The initial weights of the critic network and action network are all set to be random in $[-0.1, 0.1]$. Then, let the discount factor $\gamma = 1$ and the adjusting parameter $\beta = 0.5$, we train the critic network and action network for 10 training cycles with each cycle of 2000 steps. In the training process, the learning rate $\alpha_c = \alpha_a = 0.05$. The convergence process of the value function and its derivative of the iterative GDHP algorithm at time instant $k = 0$ are shown in Fig. 2.16. We can see that the iterative value function sequence does converge to the optimal value function quite rapidly, which also indicates the effectiveness of the iterative GDHP algorithm.

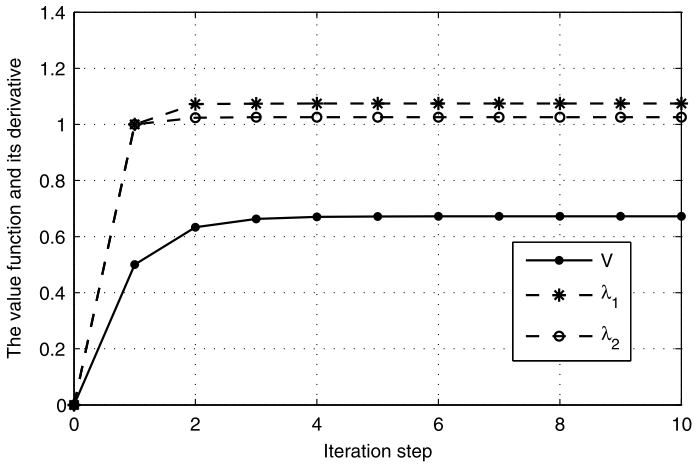

Fig. 2.16 The convergence process of the value function and its derivative of the iterative GDHP algorithm

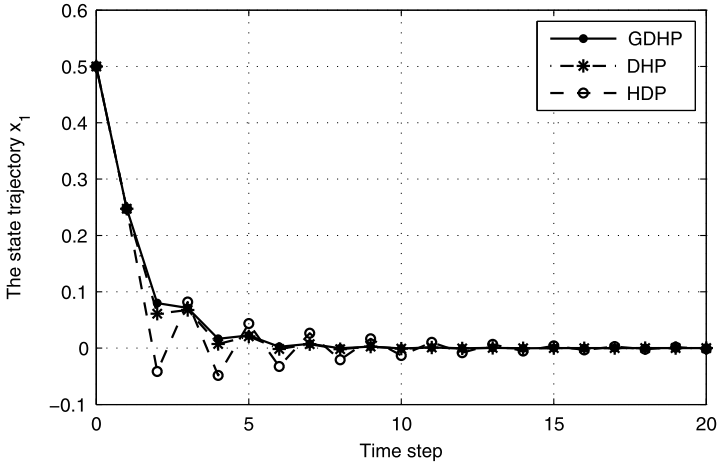

Fig. 2.17 The state trajectory x_1

Moreover, in order to make a comparison with the iterative ADP algorithm using HDP and DHP technique (iterative HDP algorithm and iterative DHP algorithm for brief), we also present the controllers designed by iterative HDP algorithm and iterative DHP algorithm, respectively. Then, for the given initial state $x_1(0) = 0.5$ and $x_2(0) = 0.5$, we apply the optimal control laws designed by iterative GDHP, HDP and DHP algorithm to the controlled system for 20 time steps, respectively, and obtain the state curves as shown in Figs. 2.17 and 2.18. The corresponding control curves are shown in Fig. 2.19. It can be seen from the simulation results that the

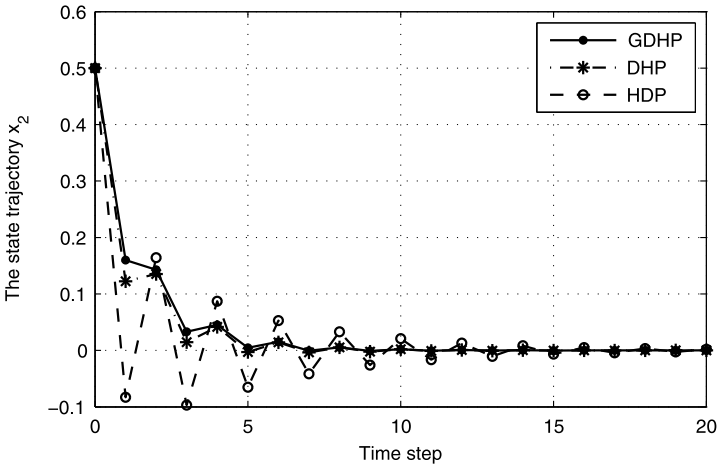

Fig. 2.18 The state trajectory x_2

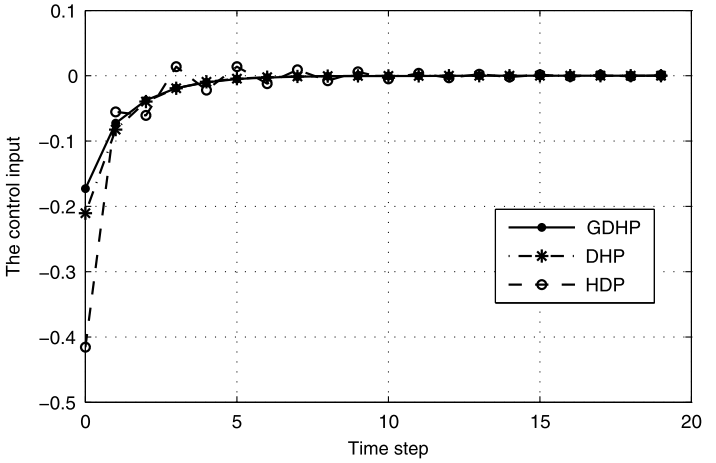

Fig. 2.19 The control input u

controller designed by the iterative GDHP algorithm has better performance than iterative HDP algorithm and iterative DHP algorithm. The most important property that the iterative GDHP algorithm superior to the iterative DHP algorithm is the former can show us the convergence process of the value function sequence. Besides, the time that the iterative GDHP algorithm takes in the entire computation process is much less than that of HDP. For the same problem, the iterative GDHP algorithm takes about 26.6 seconds while the iterative HDP algorithm takes about 61.3 seconds before satisfactory results are obtained.

2.4 Infinite-Horizon Optimal State Feedback Control Based on GHJB Algorithm

2.4.1 Problem Formulation

Consider an affine nonlinear discrete-time system of the form

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{2.134}$$

where $x(k) \in \Omega \in \mathbb{R}^n$, $f: \mathbb{R}^n \to \mathbb{R}^n$, $g: \mathbb{R}^n \to \mathbb{R}^{n\times m}$. The input satisfies $u(k) \in \Omega_u$, $\Omega_u = \{u(k) = [u_1(k), u_2(k), \ldots, u_m(k)]^{\mathrm{T}} \in \mathbb{R}^m: |u_i(k)| < \bar{u}_i(k), i = 1, 2, \ldots, m\}$, where $\bar{u}_i(k)$ is the saturating bound of the ith actuator. Let $\bar{U} = \mathrm{diag}\{\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m\}$. Assuming that $f + gu$ is continuous on a set $\Omega \subseteq \mathbb{R}^n$ containing the origin, and system (2.134) is controllable in the sense that there exists a continuous control on Ω that asymptotically stabilizes the system. In this subsection, the infinite-horizon optimal control problem for nonlinear discrete-time systems with actuator saturation is investigated. It is desired to find the constrained state feedback input $u(x(k))$ which minimizes a generalized cost functional as follows:

$$J(x(0), u) = \sum_{k=0}^{\infty} Q(x(k)) + W(u(x(k))), \tag{2.135}$$

where $Q(x(k))$ and $W(u(x(k)))$ are positive definite on Ω. For optimal control problem, it is worthy to note that $u(x(k))$ must both stabilize the system and make the cost functional finite, i.e., it must be an admissible control.

For system (2.134), the nonlinear discrete-time GHJB equation without considering saturation is given as follows:

$$\nabla V^{\mathrm{T}}(x)(f(x) + g(x)u(x) - x) + Q(x) + W(u(x)) = 0, \tag{2.136}$$

$$V(x)|_{x=0} = 0. \tag{2.137}$$

For a given admissible control u, there exists a positive definite continuously differentiable value function $V(x)$ whose initial value $V(x(0))$ equals $J(x(0), u)$.

For unconstrained control problem, a common choice of function $W(u(x))$ is $u^{\mathrm{T}}(x)Ru(x)$, where $R \in \mathbb{R}^{m\times m}$ is positive definite. On substitution of the optimal control $u^*(x) = -R^{-1}g^{\mathrm{T}}(x)\nabla J^*(x)/2$, where $J^*(x)$ is the optimal value function corresponding to optimal control $u^*(x)$, the GHJB equation (2.136) with the boundary condition (2.137) becomes the well-known HJB equation as follows:

$$\begin{aligned}&\nabla J^{*\mathrm{T}}(x)(f(x) + g(x)u(x) - x) + Q(x)\\ &\quad + \frac{1}{4}\nabla J^{*\mathrm{T}}(x)g(x)R^{-1}g^{\mathrm{T}}(x)\nabla J^*(x) = 0,\end{aligned} \tag{2.138}$$

$$J^*(x)|_{x=0} = 0. \tag{2.139}$$

However, the HJB equation above is not suitable for constrained optimal control problem. To guarantee bounded controls, we introduce a generalized nonquadratic functional as follows:

$$W(u(x)) = 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds, \tag{2.140}$$

where $W(u(x))$ is a scalar, $\Phi(v) = [\varphi(v_1), \ldots, \varphi(v_m)]^{\mathrm{T}}$ and

$$\Phi^{-1}(u) = [\varphi(u_1)^{-1}, \ldots, \varphi(u_m)^{-1}]^{\mathrm{T}}$$

are bounded one-to-one functions that belong to $C^p(p) \geq 1$ and $L_2(\Omega)$, satisfying $|\varphi(\cdot)| \leq 1$. Moreover, $\varphi(\cdot)$ is a monotonic odd function with its first derivative bounded by a constant M. It is not difficult to find such functions, such as the hyperbolic tangent function $\varphi(\cdot) = \tanh(\cdot)$. R is positive definite and assumed to be symmetric for simplicity of analysis. Substituting (2.140) into (2.136), the constrained discrete-time GHJB equation with boundary condition is derived as follows:

$$\nabla V^{\mathrm{T}}(x)(f(x) + g(x)u(x) - x) + Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds = 0, \tag{2.141}$$

$$V(x)|_{x=0} = 0. \tag{2.142}$$

According to the first-order necessary condition of the optimal control, the constrained optimal state feedback control law can be obtained as follows:

$$u^*(x) = \bar{U}\Phi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}(x)\nabla J^*(x)\right). \tag{2.143}$$

Substitute (2.143) into (2.141), and the constrained discrete-time HJB equation can be derived as follows:

$$\begin{aligned}&\nabla J^{*\mathrm{T}}(x)\left(f(x) + g\bar{U}\Phi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla J^*(x)\right) - x\right)\\ &\quad + Q(x) + 2\int_0^{\bar{U}\Phi(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}(x)\nabla J^*(x))} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds = 0,\end{aligned}$$

$$J^*(x)|_{x=0} = 0. \tag{2.144}$$

If this HJB equation can be solved for the optimal value function $J^*(x)$, then (2.143) gives the optimal constrained control. However, this equation is generally impossible to solve.

2.4.2 Constrained Optimal Control Based on GHJB Equation

Contrast to HJB equation (2.144) being nonlinear difference equation about $\nabla J^*(x)$, the GHJB equation (2.141) is linear in $\nabla V(x)$. So it is easier to solve the GHJB equation than the HJB equation from a theoretical viewpoint. That is the reason why a successive approximation based on GHJB equation is introduced to solve the HJB equation. Via solving a sequence of $\mathrm{GHJB}(V^{[i]}, u^{[i]}) = 0$ we can obtain a sequence for $V^{[i]}$ and prove $V^{[i]} \to J^*$.

Theorem 2.25 (cf. [6]) *If $u^{[i]}(x) \in \Psi(\Omega)$, $x(0) \in \Omega$, the value function $V^{[i]}$ is positive definite and continuously differentiable on Ω, and satisfies* $\mathrm{GHJB}(V^{[i]}, u^{[i]}) = 0$ *with the boundary condition $V^{[i]}(0) = 0$, then*

$$u^{[i+1]}(x) = \bar{U}\Phi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla V^{[i]}(x)\right) \tag{2.145}$$

is an admissible control for (2.134) on Ω. Moreover, if $\varphi(\cdot)$ is monotone odd, and $V^{[i+1]}$ is the unique positive definite function, which satisfies $\mathrm{GHJB}(V^{[i+1]}, u^{[i+1]}) = 0$ *with the boundary condition $V^{[i+1]}(0) = 0$, then we can conclude that $V^{(i+1)}(x(0)) \le V^{[i]}(x(0))$.*

Proof First, we should prove that $u^{[i+1]}$ is admissible.

For simplicity, in the following we write $f(x)$ as f, $g(x)$ as g. Taking the difference of the system with the control $u^{[i+1]}$ along the system $(f, g, u^{[i+1]})$, we have

$$\begin{aligned}\Delta V^{[i]}(x(k)) &= V^{[i]}(x(k+1)) - V^{[i]}(x(k)) \\ &\approx \nabla V_k^{[i]\mathrm{T}}\left(f_k + g_k u_k^{[i+1]} - x(k)\right),\end{aligned} \tag{2.146}$$

where

$$\begin{aligned}\nabla V_k &= \left.\frac{\partial V(x)}{\partial x}\right|_{x=x(k)} \\ &= \left.\left[\frac{\partial}{\partial x_1}V(x), \frac{\partial}{\partial x_2}V(x), \ldots, \frac{\partial}{\partial x_n}V(x)\right]^{\mathrm{T}}\right|_{x=x(k)},\end{aligned}$$

$f_k = f(x(k))$, $g_k = g(x(k))$, and $u_k = u(x(k))$. For any $x(k) \in \Omega$, since $\mathrm{GHJB}(V^{[i]}, u^{[i]}) = 0$, we have

$$\nabla V_k^{[i]\mathrm{T}}\left(f_k + g_k u_k^{[i]} - x(k)\right) + Q(x(k)) + 2\int_0^{u_k^{[i]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}R\,ds = 0. \tag{2.147}$$

Substituting (2.147) into (2.146), we have

$$\Delta V^{[i]}(x(k)) = \nabla V_k^{[i]\mathrm{T}} g_k \left(u_k^{[i+1]} - u_k^{[i]}\right) - Q(x(k)) - 2\int_0^{u_k^{[i]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds. \tag{2.148}$$

Since

$$\nabla V_k^{[i]\mathrm{T}}(x) g_k = -2\Phi^{-\mathrm{T}}\left(\bar{U}^{-1}u_k^{[i+1]}\right)\bar{U}R, \tag{2.149}$$

we get

$$\Delta V^{[i]}(x(k)) = -Q(x(k)) + 2\left[\Phi^{-\mathrm{T}}\left(\bar{U}^{-1}u_k^{[i+1]}\right)\bar{U}R\left(u_k^{[i]} - u_k^{[i+1]}\right) - \int_0^{u_k^{[i]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right]. \tag{2.150}$$

Because φ and φ^{-1} are monotone odd, the second term of (2.150) is negative. This implies that $\Delta V^{[i]}(x(k)) < 0$ for $x(k) \neq 0$. Thus, $V^{[i]}$ is a Lyapunov function for $u^{[i+1]}$ on Ω and the system (2.134) is asymptotically stable.

Next, we are ready to show that the value function of the system with the updated control $u^{[i+1]}$ is finite.

Since $u^{[i]}$ is admissible, from Definition 2.1, we get

$$V^{[i]}(x(0)) = J(x(0), u^{[i]}) < \infty, x(0) \in \Omega. \tag{2.151}$$

The value function for $u^{[i+1]}$ is

$$V(x(0), u^{[i+1]}) = \sum_{k=0}^{\infty}\left\{Q(x(k) + 2\int_0^{u_k^{[i+1]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right\}, \tag{2.152}$$

where $x(k)$ is the state trajectory of system with admissible control $u^{[i+1]}$.

From (2.150) and (2.152), we have

$$\begin{aligned} V^{[i]}(x(\infty)) - V^{[i]}(x(0)) &= \sum_{k=0}^{\infty} \Delta V^{[i]}(k) \\ &= \sum_{k=0}^{\infty}\Big\{ - Q(x(k)) + 2[\Phi^{-\mathrm{T}}(\bar{U}^{-1}u_k^{[i+1]})\bar{U}R(u_k^{[i]} - u_k^{[i+1]}) \\ &\quad - \int_0^{u_k^{[i]}} \Phi^{-1}(\bar{U}^{-1}s)\bar{U}Rds]\Big\} \end{aligned}$$

$$= - J(x(0), u^{[i+1]}) + 2\sum_{k=0}^{\infty}\left\{\Phi^{-\mathrm{T}}(\bar{U}^{-1}u_k^{[i+1]})\bar{U}R(u_k^{[i]} - u_k^{[i+1]}) + \int_{u_k^{[i]}}^{u_k^{[i+1]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right\}. \tag{2.153}$$

Since $x(\infty) = 0$ and $V^{[i]}(x)|_{x=0} = 0$, we get $V^{[i]}(x(\infty)) = 0$. By rewriting (2.152), we have

$$J(x(0), u^{[i+1]}) = V^{[i]}(x(0)) + 2\sum_{k=0}^{\infty}\left\{\Phi^{-\mathrm{T}}(\bar{U}^{-1}u_k^{[i+1]}\bar{U}R(u_k^{[i]} - u_k^{[i+1]}) + \int_{u_k^{[i]}}^{u_k^{[i+1]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right\}. \tag{2.154}$$

Since φ and φ^{-1} are monotone odd, and the second term of (2.154) is less than 0, we have

$$J(x(0), u^{[i+1]}) < V^{[i]}(x(0)) = J(x(0), u^{[i]}) < \infty. \tag{2.155}$$

Because $V^{[i]}$ is continuously differentiable, and $g : R^n \to R^{n\times m}$ is a Lipschitz continuous function, $u^{[i+1]}$ is continuous. Since $V^{[i]}$ is positive definite and attains its minimum at the origin, and $\Delta V^{[i]}$ must approach 0 at the origin, from (2.145) we have $u^{[i+1]}(x)|_{x=0} = 0$.

From Definition 2.1, we know that $u^{[i+1]}$ is an admissible control on Ω. Since $u^{[i+1]}$ is admissible, there exists a $V^{[i+1]}$ satisfying GHJB$(V^{[i+1]}, u^{[i+1]}) = 0$, and

$$V^{[i+1]}(x(0)) = J(x(0), u^{[i+1]}). \tag{2.156}$$

From (2.154) and (2.156), we get

$$V^{[i+1]}(x(0)) - V^{[i]}(x(0)) = -2\sum_{k=0}^{\infty}\left\{\Phi^{-\mathrm{T}}(\bar{U}^{-1}u_k^{[i+1]})\bar{U}R(u_k^{[i]} - u_k^{[i+1]}) + \int_{u_k^{[i]}}^{u_k^{[i+1]}} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right\} \le 0. \tag{2.157}$$

The proof is completed. □

Corollary 2.26 *Given $u^{[0]}(x) \in \Psi(\Omega)$, if one iteratively solve the GHJB equation GHJB $(V^{[i]}, u^{[i]}) = 0$ and for $i = 0, 1, 2, \ldots$, update the control as $u^{[i+1]}(x) = \bar{U}\Phi(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla V^{[i]}(x))$, then it can be concluded that $V^{[i]}(x) \to J^*(x)$.*

Proof According to Theorem 2.25, $V^{[i]}$ is a decreasing sequence with a lower bound. Since $V^{[i]} > 0$, $V^{[i+1]} - V^{[i]} < 0$, $V^{[i]}$ will converge to a positive definite function $V^{[i+1]} = V^{[i]} = V^d$ when $i \to \infty$. Due to the HJB equation having a unique solution, we just need to prove $V^d = J^*$. When $V^{[i]} = V^{[i+1]} = V^d$, from (2.145) we have

$$u^{[i]}(x) = u^{[i+1]}(x) = \bar{U}\Phi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\Delta V^{[i]}(x)\right). \tag{2.158}$$

The GHJB equation with input $u^{[i]}$ can be written as

$$\nabla V^{[i]}(x)^{\mathrm{T}}\left(f(x) + g\bar{U}\Phi\left(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla V^{[i]}(x)\right) - x\right) + Q(x)$$
$$+ 2\int_0^{\bar{U}\Phi(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla V^{[i]}(x))} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds = 0, \tag{2.159}$$

$$V^{[i]}(x)|_{x=0} = 0. \tag{2.160}$$

From (2.144), the conclusion can be drawn that (2.159) with boundary condition (2.160) is the HJB equation. This implies that $V^{[i]}(x) \to J^*(x)$, $u^{[i]}(x) \to u^*(x)$. □

In the next part, we are ready to discuss how to design the nearly optimal saturated controller using NNs. In general, the closed-form solution of GHJB equation (2.141) cannot be obtained even though solving GHJB equation (2.141) is easier than solving HJB equation (2.144). In this section, a neural network is used to approximate the solution $V(x)$ of constrained nonlinear discrete-time GHJB equation. Finally, the nearly optimal state feedback control is obtained according to (2.143).

$V(x)$ is approximated by a neural network as follows:

$$V_L(x) = \sum_{j=1}^{L} w_j\sigma_j(x) = W_L^{\mathrm{T}}\bar{\sigma}_L(x), \tag{2.161}$$

where w_j are the weights of the neural network, $\sigma_j(x)$ are the activation functions, $\sigma_j(x)$ are continuous and satisfy $\sigma_j(x)|_{x=0} = 0$. L is the number of hidden-layer neurons. $\bar{\sigma}_L(x) \equiv [\sigma_1(x), \sigma_2(x), \ldots, \sigma_L(x)]^{\mathrm{T}}$ is the vector activation function, $W_L(x) \equiv [w_1(x), w_2(x), \ldots, w_L(x)]^{\mathrm{T}}$ is the vector weight. The control objective is to make the residual error minimum in a least-square sense by tuning the weights.

Substituting (2.161) into (2.141), we have

$$\mathrm{GHJB}\left(V_L = \sum_{j=1}^{L} w_j\sigma_j, u\right) = e_L(x). \tag{2.162}$$

The method of weighted residuals is used to find the least-square solution, i.e.,

$$\left\langle \frac{\partial(e_L(x))}{\partial W_L(x)}, e_L(x) \right\rangle = 0, \tag{2.163}$$

where $\langle f, g \rangle = \int_\Omega fg dx$ is a *Lebesgue integral*.

By expanding (2.163), we get

$$\begin{aligned} &\langle \nabla\bar{\sigma}_L(x)\Delta x, \nabla\bar{\sigma}_L(x)\Delta x \rangle \cdot W_L \\ &+ \left\langle Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds, \nabla\bar{\sigma}_L(x)\Delta x \right\rangle = 0. \end{aligned} \tag{2.164}$$

Lemma 2.27 *If the set* $\{\sigma_j(x)\}_1^L$ *is linearly independent and* $u \in \Omega_u$, *then the set* $\{\nabla\sigma_j^{\mathrm{T}}\Delta x\}_1^L$ *is also linearly independent*.

From Lemma 2.27, $\langle \nabla\bar{\sigma}_L(x)\Delta x, \nabla\bar{\sigma}_L(x)\Delta x \rangle$ is invertible. Therefore there exists a unique solution as follows:

$$\begin{aligned} W_L = &- \langle \nabla\bar{\sigma}_L(x)\Delta x, \nabla\bar{\sigma}_L(x)\Delta x \rangle^{-1} \\ &\times \left\langle Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds, \nabla\bar{\sigma}_L(x)\Delta x \right\rangle, \end{aligned} \tag{2.165}$$

and the control can be derived as

$$u = \bar{U}\Phi\Big(-\frac{1}{2}(\bar{U}R)^{-1} g^{\mathrm{T}} \nabla\sigma_L^{\mathrm{T}} W_L \Big). \tag{2.166}$$

For reducing computation, the integration in (2.165) is approximated by the definition of Riemann integration [4].

A mesh of points over the integral region can be introduced on Ω, with the size δx chosen as small as possible. Moreover, p is required to be larger than L, and the activation functions are linearly independent to guarantee $(A^{\mathrm{T}}A)$ invertible. The specific expressions are given as follows:

$$A = [\, \nabla\bar{\sigma}_L(x)\Delta x|_{x=x_1}, \ldots, \nabla\bar{\sigma}_L(x)\Delta x|_{x=x_p}]^{\mathrm{T}}, \tag{2.167}$$

$$B = \begin{bmatrix} Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\Big|_{x=x_1} \\ \vdots \\ Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\Big|_{x=x_p} \end{bmatrix}, \tag{2.168}$$

$$\langle \nabla\bar{\sigma}_L(x)\Delta x, \nabla\bar{\sigma}_L(x)\Delta x \rangle = \lim_{\|\delta x\| \to 0} (A^{\mathrm{T}}A) \cdot \delta x, \tag{2.169}$$

$$\left\langle Q(x) + 2\int_0^{u(x)} \Phi^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds, \nabla\bar{\sigma}_L(x)\Delta x \right\rangle = \lim_{\|\delta x\|\to 0}(A^{\mathrm{T}}B)\cdot\delta x. \quad (2.170)$$

Therefore, we get

$$W_L = -(A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}B. \quad (2.171)$$

The design procedure of the optimal constrained controller of nonlinear discrete-time systems with actuator saturation is given below:

1. Using a neural network to approximate $V(x)$, i.e., we have $V(x) = \sum_{j=1}^{L} \times w_j\sigma_j(x)$.
2. Select an initial admissible control $u^{[0]}$, then solve GHJB$(V^{[0]}, u^{[0]}) = 0$ by applying the least-square method to obtain $\mathrm{W}^{[0]}$, and accordingly $V^{[0]}$ is computed.
3. For $i = 0, 1, 2, \ldots$, update the control $u^{[i+1]} = \bar{U}\Phi(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla V^{[i]})$.
4. For $i = 0, 1, 2, \ldots$, solve GHJB$(V^{[i+1]}, u^{[i+1]}) = 0$ by the least-square method to obtain W^{i+1}, and then we can get $V^{[i+1]}$.
5. If $V^{[i]}(0) - V^{[i+1]}(0) \le \varepsilon$, where ε is a small positive constant, then $J^* = V^{[i]}$, stop; else $i = i + 1$, go back to step 3 and go on.
6. After J^* being solved off-line, the optimal state feedback control $u^* = \bar{U}\Phi(-\frac{1}{2}(\bar{U}R)^{-1}g^{\mathrm{T}}\nabla J^*)$ will be implemented on-line.

2.4.3 Simulations

In order to demonstrate the effectiveness of the method developed in this section, an example is presented in this subsection.

Example 2.28 Consider the following affine nonlinear discrete-time system with actuator saturation:

$$x(k+1) = f(x(k)) + g(x(k))u(k), \quad (2.172)$$

where

$$f(x(k)) = \begin{bmatrix} -0.8x_2(k) \\ \sin(0.8x_1(k) - x_2(k)) + 1.8x_2(k) \end{bmatrix},$$

$$g(x(k)) = \begin{bmatrix} 0 \\ -x_2(k) \end{bmatrix},$$

and the upper bound $\bar{U}$ of actuator saturation is 0.35.

The control objective is to design an optimal controller with bound less than 0.35. Define the cost functional as

$$J(x(0), u) = \sum_{k=0}^{\infty}\left\{x^{\mathrm{T}}(k)Qx(k) + 2\int_0^{u(x(k))} \tanh^{-\mathrm{T}}(\bar{U}^{-1}s)\bar{U}Rds\right\}, \quad (2.173)$$

where the weight matrices are chosen as $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ and $R = 1$.

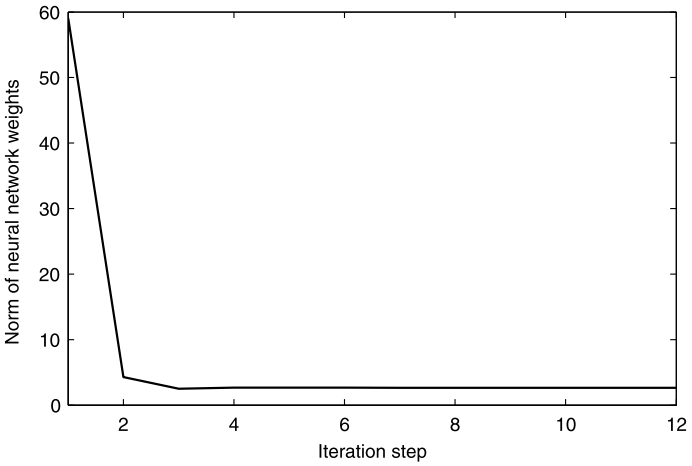

Fig. 2.20 Norm of neural-network weights at each step

To find a nearly optimal controller, a Volterra neural network is used to approximate the value function of the system as follows:

$$\begin{aligned} V(x) = {} & w_1 x_1^2 + w_2 x_2^2 + w_3 x_1 x_2 + w_4 x_1^4 + w_5 x_2^4 + w_6 x_1^3 x_2 \\ & + w_7 x_1^2 x_2^2 + w_8 x_1 x_2^3 + w_9 x_1^6 + w_{10} x_2^6 + w_{11} x_1^5 x_2 \\ & + w_{12} x_1^4 x_2^2 + w_{13} x_1^3 x_2^3 + w_{14} x_1^2 x_2^4 + w_{15} x_1 x_2^5. \end{aligned} \tag{2.174}$$

The algorithm is implemented over the region Ω defined by $|x_1| \leq 0.5$, $|x_2| \leq 0.5$. Select the initial control $u_0(k) = x_1(k) + 1.5x_2(k)$, which is admissible, and then update the control by $u^{[i+1]} = \bar{U} \tanh(-(\bar{U}R)^{-1} g^{\mathrm{T}} \nabla V^{[i]}/2)$, where $u^{[i]}$ and $V^{[i]}$ satisfy the following GHJB equation:

$$\begin{aligned} & \nabla V^{[i]\mathrm{T}}(x)(f(x) + g(x)u^{[i]}(x) - x) + x^{\mathrm{T}} Q x \\ & \quad + 2 \int_0^{u^{[i]}(x)} \tanh^{-\mathrm{T}}(\bar{U}^{-1} s) \bar{U} R ds = 0. \end{aligned} \tag{2.175}$$

In the simulation, the parameters are chosen as follows: the mesh size $\delta x = 0.01$, the small positive constant $\varepsilon = 0.01$, and the initial states $x_1(0) = x_2(0) = 0.5$. Figure 2.20 shows the trajectory of the norm of neural-network weights at each iteration step. Figure 2.21 shows that the value function converges to a constant very rapidly. After 12 successive iterative steps, the nearly optimal saturated control can be obtained off-line. Then, the controller is applied to the system with given initial states for 100 time steps. Figure 2.22 shows the control trajectory and Fig. 2.23 shows the state trajectories, whereas Figs. 2.24 and 2.25 illustrate the control trajectory and state trajectories without considering actuator saturation, respectively. By comparison, we can see that saturated control and corresponding state trajectories have fewer oscillations, and saturation has been overcome successfully.

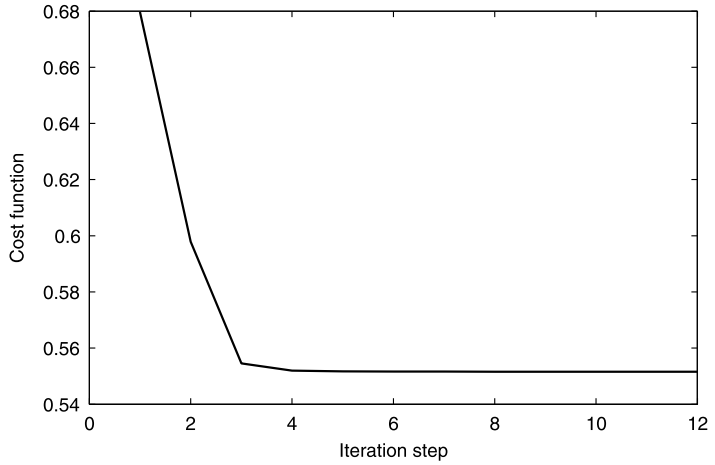

Fig. 2.21 The value function at each iteration step

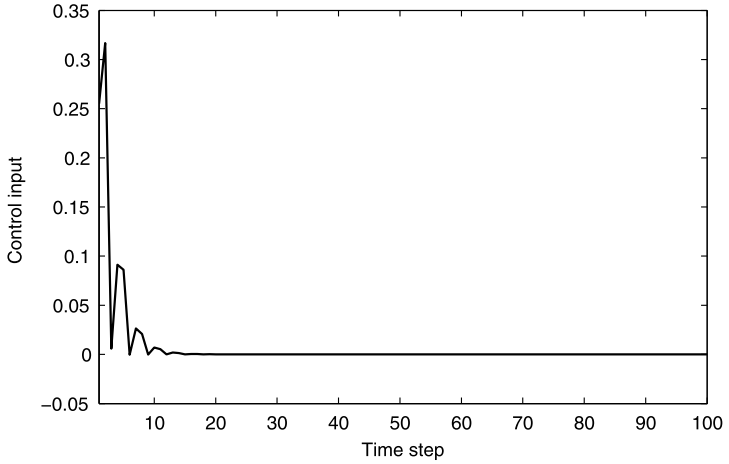

Fig. 2.22 The control trajectory

2.5 Finite-Horizon Optimal State Feedback Control Based on HDP

In this section, we will develop a new ADP scheme for the finite-horizon optimal control problem. We will study the optimal control problem with an ε-error bound using ADP algorithms. First, the HJB equation for finite-horizon optimal control of discrete-time systems is derived. In order to solve this HJB equation, a new iterative ADP algorithm is developed with convergence and optimality proofs. Second, the difficulties of obtaining the optimal solution using the iterative ADP algorithm is presented and then the *ε-optimal* control algorithm is derived based on the iterative

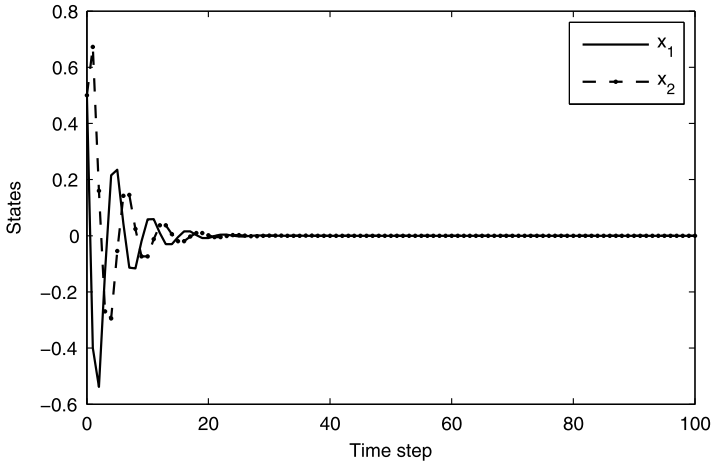

Fig. 2.23 The state trajectories

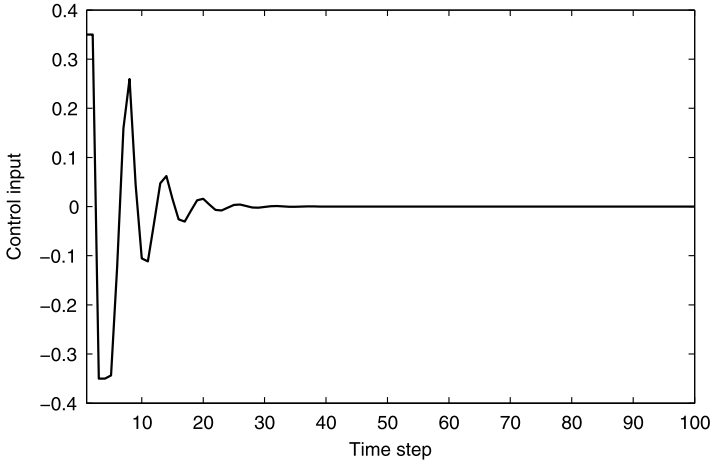

Fig. 2.24 The control trajectory without considering the actuator saturation in the controller design

ADP algorithm. Next, it will be shown that the ε-optimal control algorithm can obtain suboptimal control solutions within a fixed finite number of control steps that make the value function converge to its optimal value with an ε-error. Furthermore, in order to facilitate the implementation of the iterative ADP algorithms, we use NNs to obtain the iterative value function and the optimal control policy. Finally, an ε-optimal state feedback controller is obtained for the finite-horizon optimal control problem.

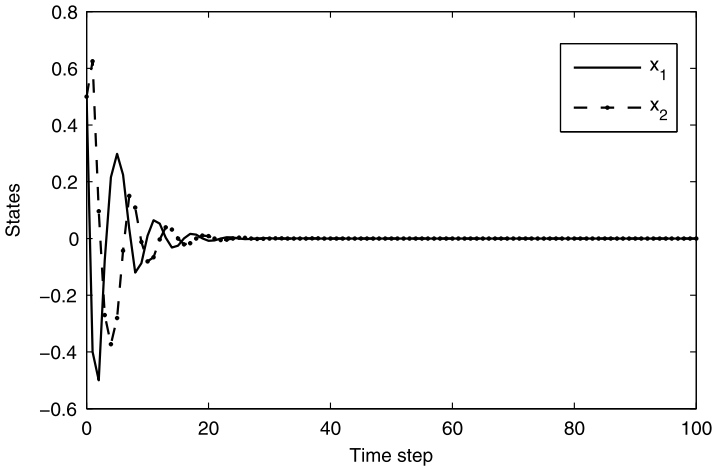

Fig. 2.25 The state trajectories without considering the actuator saturation in the controller design

2.5.1 Problem Formulation

In this section, we will study the following deterministic discrete-time systems:

$$x(k+1) = F(x(k), u(k)),\ k = 0, 1, 2, \ldots, \tag{2.176}$$

where $x(k) \in \mathbb{R}^n$ is the state and $u(k) \in \mathbb{R}^m$ is the control vector. Let $x(0)$ be the initial state. The system function $F(x(k), u(k))$ is continuous for $\forall x(k), u(k)$ and $F(0, 0) = 0$. Hence, $x = 0$ is an equilibrium state of system (2.176) under the control $u = 0$. The cost function for state $x(0)$ under the control sequence $\underline{u}_0^{N-1} = (u(0), u(1), \ldots, u(N-1))$ is defined as

$$J\left(x(0), \underline{u}_0^{N-1}\right) = \sum_{i=0}^{N-1} l(x(i), u(i)), \tag{2.177}$$

where l is the utility function, $l(0, 0) = 0$, and $l(x(i), u(i)) \geq 0$ for $\forall x(i), u(i)$.

The sequence $\underline{u}_0^{N-1}$ defined above is a finite sequence of controls. Using this sequence of controls, system (2.176) gives a trajectory starting from $x(0)$: $x(1) = F(x(0), u(0)), x(2) = F(x(1), u(1)), \ldots, x(N) = F(x(N-1), u(N-1))$. We call the number of elements in the control sequence $\underline{u}_0^{N-1}$ the length of $\underline{u}_0^{N-1}$ and denote it as $|\underline{u}_0^{N-1}|$. Then, $|\underline{u}_0^{N-1}| = N$. The length of the associated trajectory $\underline{x}_0^N = (x(0), x(1), \ldots, x(N))$ is $N+1$. We denote the final state of the trajectory as $x^{(f)}(x(0), \underline{u}_0^{N-1})$, i.e., $x^{(f)}(x(0), \underline{u}_0^{N-1}) = x_N$. Then, for $\forall k \geq 0$, the finite control sequence starting at k can be written as $\underline{u}_k^{k+i-1} = (u(k), u(k+1), \ldots, u(k+i-1))$, where $i \geq 1$ is the length of the control sequence. The final state can be written as $x^{(f)}(x(k), \underline{u}_k^{k+i-1}) = x(k+i)$.

We note that the cost function defined in (2.177) does not have the term associated with the final state since in the present study we specify the final state $x(N) = F(x(N-1), u(N-1))$ to be at the origin, i.e., $x(N) = x^{(f)} = 0$. For the present finite-horizon optimal control problem, the feedback controller $u(k) = u(x(k))$ must not only drive the system state to zero within a finite number of time steps but also guarantee the cost function (2.177) to be finite, i.e., $\underline{u}_k^{N-1} = (u(x(k)), u(x(k+1)), \ldots, u(x(N-1)))$ must be a finite-horizon admissible control sequence, where $N > k$ is a finite integer.

Definition 2.29 A control sequence $\underline{u}_k^{N-1}$ is said to be finite-horizon admissible for a state $x(k) \in \mathbb{R}^n$, if $x^{(f)}(x(k), \underline{u}_k^{N-1}) = 0$ and $J(x(k), \underline{u}_k^{N-1})$ is finite, where $N > k$ is a finite integer.

A state $x(k)$ is said to be finite-horizon controllable (controllable for brief) if there is a finite-horizon admissible control sequence associated with this state.

Let $\underline{u}_k$ be an arbitrary finite-horizon admissible control sequence starting at k and let

$$\mathfrak{A}_{x(k)} = \left\{\underline{u}_k : x^{(f)}\left(x(k), \underline{u}_k\right) = 0\right\}$$

be the set of all finite-horizon admissible control sequences of $x(k)$. Let

$$\mathfrak{A}_{x(k)}^{(i)} = \left\{\underline{u}_k^{k+i-1} : x^{(f)}\left(x(k), \underline{u}_k^{k+i-1}\right) = 0,\ \left|\underline{u}_k^{k+i-1}\right| = i\right\}$$

be the set of all finite-horizon admissible control sequences of $x(k)$ with length i. Then, $\mathfrak{A}_{x(k)} = \bigcup_{1 \le i < \infty} \mathfrak{A}_{x(k)}^{(i)}$. In this notation, a state $x(k)$ is controllable if and only if $\mathfrak{A}_{x(k)} \neq \emptyset$.

For any given system state $x(k)$, the objective of the present finite-horizon optimal control problem is to find a finite-horizon admissible control sequence $\underline{u}_k^{N-1} \in \mathfrak{A}_{x(k)}^{(N-k)} \subseteq \mathfrak{A}_{x(k)}$ to minimize the cost $J(x(k), \underline{u}_k^{N-1})$. The control sequence $\underline{u}_k^{N-1}$ has finite length. However, before it is determined, we do not know its length which means that the length of the control sequence $|\underline{u}_k^{N-1}| = N - k$ is unspecified. This kind of optimal control problems have been called finite-horizon problems with unspecified terminal time [3] (but in the present case, with fixed terminal state $x^{(f)} = 0$).

Define the optimal value function as

$$J^*(x(k)) = \inf_{\underline{u}_k} \left\{J(x(k), \underline{u}_k) : \underline{u}_k \in \mathfrak{A}_{x(k)}\right\}. \tag{2.178}$$

Then, according to Bellman's principle of optimality, $J^*(x(k))$ satisfies the discrete-time HJB equation

$$J^*(x(k)) = \min_{u_k} \left\{l(x(k), u(k)) + J^*(F(x(k), u(k)))\right\}. \tag{2.179}$$

Now, define the law of optimal control sequence starting at k by

$$\underline{u}^*(x(k)) = \arg\inf_{\underline{u}_k} \left\{ J(x(k), \underline{u}_k) \colon \underline{u}_k \in \mathfrak{A}_{x(k)} \right\},$$

and define the law of optimal control vector by

$$u^*(x(k)) = \arg\min_{u(k)} \left\{ l(x(k), u(k)) + J^*(F(x(k), u(k))) \right\}.$$

In other words, $\underline{u}^*(x(k)) = \underline{u}_k^*$ and $u^*(x(k)) = u_k^*$. Hence, we have

$$J^*(x(k)) = l(x(k), u_k^*) + J^*(F(x(k), u_k^*)).$$

2.5.2 *Finite-Horizon Optimal State Feedback Control Based on HDP*

In this subsection, a new iterative ADP algorithm is developed to obtain the finite-horizon optimal controller for nonlinear systems. The goal of the present iterative ADP algorithm is to construct an optimal control policy $u^*(x(k))$, $k = 0, 1, \ldots$, which drives the system from an arbitrary initial state $x(0)$ to the singularity 0 within a finite time, and simultaneously minimizes the performance index function. Convergence proofs will also be given to show that the performance index function will indeed converge to the optimum.

2.5.2.1 Derivation and Properties of the Iterative ADP Algorithm

We first consider the case where for any state $x(k)$, there exists a control vector $u(k)$ such that $F(x(k), u(k)) = 0$, i.e., we can control the state of system (2.176) to zero in one step from any initial state. For the case where $F(x(k), u(k)) = 0$ does not hold, we will discuss and solve the problem later in the subsection.

In the iterative ADP algorithm, the value function and control policy are updated by recursive iterations, with the iteration index number i increasing from 0 and with the initial performance index function $V_0(x) = 0$ for $\forall\, x \in \mathbb{R}^n$.

The value function for $i = 1$ is computed as

$$\begin{aligned} V_1(x(k)) &= \min_{u(k)} \{ l(x(k), u(k)) + V_0(F(x(k), u(k))) \} \\ &\qquad \text{subject to } \ F(x(k), u(k)) = 0 \\ &= \min_{u(k)} l(x(k), u(k)) \ \text{ subject to } \ F(x(k), u(k)) = 0 \\ &= l(x(k), u_k^*(x(k))), \end{aligned} \tag{2.180}$$

where $V_0(F(x(k),u(k))) = 0$ and $F(x(k),u_k^*(x(k))) = 0$. The control vector $v_1(x(k))$ for $i = 1$ is chosen as $v_1(x(k)) = u_k^*(x(k))$. Therefore, (2.180) can also be written as

$$\begin{aligned} V_1(x(k)) &= \min_{u(k)} l(x(k),u(k)) \ \text{ subject to } \ F(x(k),u(k)) = 0 \\ &= l(x(k), v_1(x(k))), \end{aligned} \tag{2.181}$$

where

$$v_1(x(k)) = \arg\min_{u(k)} l(x(k),u(k)) \ \text{ subject to } \ F(x(k),u(k)) = 0. \tag{2.182}$$

For $i = 2, 3, 4, \ldots$, the iterative ADP algorithm will be implemented as follows:

$$\begin{aligned} V_i(x(k)) &= \min_{u(k)} \{l(x(k),u(k)) + V_{i-1}(F(x(k),u(k)))\} \\ &= l(x(k), v_i(x(k))) + V_{i-1}(F(x(k), v_i(x(k)))), \end{aligned} \tag{2.183}$$

where

$$\begin{aligned} v_i(x(k)) &= \arg\min_{u(k)} \{l(x(k),u(k)) + V_{i-1}(x(k+1))\} \\ &= \arg\min_{u(k)} \{l(x(k),u(k)) + V_{i-1}(F(x(k),u(k)))\}. \end{aligned} \tag{2.184}$$

Equations (2.181)–(2.184) form the iterative ADP algorithm.

Remark 2.30 Equations (2.181)–(2.184) in the iterative ADP algorithm are similar to the HJB equation (2.179), but they are not the same. There are at least two obvious differences:

1. For any finite time k, if $x(k)$ is the state at k, then the optimal value function in HJB equation (2.179) is unique, i.e., $J^*(x(k))$, while in the iterative ADP equations (2.181)–(2.184), the value function is different for each iteration index i, i.e., $V_i(x(k)) \neq V_j(x(k))$ for $\forall\, i \neq j$ in general.
2. For any finite time k, if $x(k)$ is the state at k, then the optimal control law obtained by HJB equation (2.179) possesses the unique optimal control expression, i.e., $u_k^* = u^*(x(k))$, while the control law solved by the iterative ADP algorithm (2.181)–(2.184) is different from each other for each iteration index i, i.e., $v_i(x(k)) \neq v_j(x(k))$ for $\forall\, i \neq j$ in general.

Remark 2.31 According to (2.177) and (2.183), we have

$$V_{i+1}(x(k)) = \min_{\underline{u}_k^{k+i}} \left\{ J\left(x(k), \underline{u}_k^{k+i}\right) : \underline{u}_k^{k+i} \in \mathfrak{A}_{x(k)}^{(i+1)} \right\}. \tag{2.185}$$

Since

$$\begin{aligned} V_{i+1}(x(k)) &= \min_{u(k)} \{l(x(k), u(k)) + V_i(x(k+1))\} \\ &= \min_{u(k)} \Bigg\{ l(x(k), u(k)) + \min_{u(k+1)} \Big\{ l(x(k+1), u(k+1)) \\ &\quad + \min_{u(k+2)} \big\{ l(x(k+2), u(k+2)) + \cdots \\ &\quad + \min_{u(k+i-1)} \{ l(x(k+i-1), u(k+i-1)) \\ &\quad + V_1(x(k+i)) \} \cdots \big\} \Big\} \Bigg\}, \end{aligned}$$

where

$$\begin{aligned} V_1(x(k+i)) &= \min_{u(k+i)} l(x(k+i), u(k+i)) \\ &\quad \text{subject to } F(x(k+i), u(k+i)) = 0, \end{aligned}$$

we obtain

$$\begin{aligned} V_{i+1}(x(k)) &= \min_{\underline{u}_k^{k+i}} \{ l(x(k), u(k)) + l(x(k+1), u(k+1)) \\ &\quad + \cdots + l(x(k+i), u(k+i)) \} \\ &\quad \text{subject to } F(x(k+i), u(k+i)) = 0, \\ &= \min_{\underline{u}_k^{k+i}} \left\{ J\big(x(k), \underline{u}_k^{k+i}\big) : \underline{u}_k^{k+i} \in \mathfrak{A}_{x(k)}^{(i+1)} \right\}. \end{aligned}$$

Using the notation in (2.184), we can also write

$$V_{i+1}(x(k)) = \sum_{j=0}^{i} l\big(x(k+j), v_{i+1-j}(x(k+j))\big). \tag{2.186}$$

In the above, we can see that the value function $J^*(x(k))$ solved by HJB equation (2.179) is replaced by a sequence of iterative value functions $V_i(x(k))$ and the optimal control law $u^*(x(k))$ is replaced by a sequence of iterative control law $v_i(x(k))$, where $i \geq 1$ is the index of iteration. We can prove that $J^*(x(k))$ defined in (2.178) is the limit of $V_i(x(k))$ as $i \to \infty$.

Theorem 2.32 *Let $x(k)$ be an arbitrary state vector. Suppose that $\mathfrak{A}_{x(k)}^{(1)} \neq \emptyset$. Then, the value function $V_i(x(k))$ obtained by (2.181)–(2.184) is a monotonically nonincreasing sequence for $\forall\, i \geq 1$, i.e., $V_{i+1}(x(k)) \leq V_i(x(k))$ for $\forall\, i \geq 1$.*

Proof We prove this by mathematical induction. First, we let $i = 1$. Then, we have $V_1(x(k))$ given as in (2.181) and the finite-horizon admissible control sequence is $\hat{\underline{u}}_k^k = (v_1(x(k)))$.

Next, we show that there exists a finite-horizon admissible control sequence $\hat{\underline{u}}_k^{k+1}$ with length 2 such that $J(x(k), \hat{\underline{u}}_k^{k+1}) = V_1(x(k))$. The trajectory starting from $x(k)$ under the control of $\hat{\underline{u}}_k^k = (v_1(x(k)))$ is $x(k+1) = F(x(k), v_1(x(k))) = 0$. Then, we create a new control sequence $\hat{\underline{u}}_k^{k+1}$ by adding a 0 to the end of sequence $\hat{\underline{u}}_k^k$ to obtain the control sequence $\hat{\underline{u}}_k^{k+1} = (\hat{\underline{u}}_k^k, 0)$. Obviously, $|\hat{\underline{u}}_k^{k+1}| = 2$. The state trajectory under the control of $\hat{\underline{u}}_k^{k+1}$ is $x(k+1) = F(x(k), v_1(x(k))) = 0$ and $x(k+2) = F(x(k+1), \hat{u}(k+1))$, where $\hat{u}(k+1) = 0$. Since $x(k+1) = 0$ and $F(0,0) = 0$, we have $x(k+2) = 0$. So, $\hat{\underline{u}}_k^{k+1}$ is a finite-horizon admissible control sequence. Furthermore,

$$\begin{aligned} J(x(k), \hat{\underline{u}}_k^{k+1}) &= l(x(k), v_1(x(k))) + l(x(k+1), \hat{u}(k+1)) \\ &= l(x(k), v_1(x(k))) \\ &= V_1(x(k)) \end{aligned}$$

since $l(x(k+1), \hat{u}(k+1)) = l(0,0) = 0$. On the other hand, according to Remark 2.31, we have

$$V_2(x(k)) = \min_{\underline{u}_k^{k+1}} \left\{ J\left(x(k), \underline{u}_k^{k+1}\right) : \underline{u}_k^{k+1} \in \mathfrak{A}_{x(k)}^{(2)} \right\}.$$

Then, we obtain

$$\begin{aligned} V_2(x(k)) &= \min_{\underline{u}_k^{k+1}} \left\{ J\left(x(k), \underline{u}_k^{k+1}\right) : \underline{u}_k^{k+1} \in \mathfrak{A}_{x(k)}^{(2)} \right\} \\ &\le J\left(x(k), \hat{\underline{u}}_k^{k+1}\right) \\ &= V_1(x(k)). \end{aligned} \tag{2.187}$$

Therefore, the theorem holds for $i = 1$.

Assume that the theorem holds for any $i = q$, where $q > 1$. From (2.186), we have

$$V_q(x(k)) = \sum_{j=0}^{q-1} l\left(x(k+j), v_{q-j}(x(k+j))\right).$$

The corresponding finite-horizon admissible control sequence is $\hat{\underline{u}}_k^{k+q-1} = (v_q(x(k)), v_{q-1}(x(k+1)), \ldots, v_1(x(k+q-1)))$.

For $i = q+1$, we create a control sequence

$$\hat{\underline{u}}_k^{k+q} = \left(v_q(x(k)), v_{q-1}(x(k+1)), \ldots, v_1(x(k+q-1)), 0\right)$$

with length $q+1$. Then, the state trajectory under the control of $\underline{\hat{u}}_k^{k+q}$ is $x(k)$, $x(k+1)=F(x(k),v_q(x(k)))$, $x(k+2)=F(x(k+1),v_{q-1}(x(k+1)))$, …, $x(k+q)=F(x(k+q-1),v_1(x(k+q-1)))=0$, $x(k+q+1)=F(x(k+q),0)=0$. So, $\underline{\hat{u}}_k^{k+q}$ is a finite-horizon admissible control sequence. The value function under this control sequence is

$$\begin{aligned}J(x(k),\underline{\hat{u}}_k^{k+q}) =& l(x(k),v_q(x(k)))+l(x(k+1),v_{q-1}(x(k+1)))\\ &+\cdots+l(x(k+q-1),v_1(x(k+q-1)))+l(x(k+q),0)\\ =&\sum_{j=0}^{q-1} l\big(x(k+j),v_{q-j}(x(k+j))\big)\\ =& V_q(x(k))\end{aligned}$$

since $l(x(k+q),0)=l(0,0)=0$.

On the other hand, we have

$$V_{q+1}(x(k))=\min_{\underline{u}_k^{k+q}}\left\{J\big(x(k),\underline{u}_k^{k+q}\big):\underline{u}_k^{k+q}\in\mathfrak{U}_{x(k)}^{(q+1)}\right\}.$$

Thus, we obtain

$$\begin{aligned}V_{q+1}(x(k))=&\min_{\underline{u}_k^{k+q}}\left\{J\big(x(k),\underline{u}_k^{k+q}\big):\underline{u}_k^{k+q}\in\mathfrak{U}_{x(k)}^{(q+1)}\right\}\\ \le& J\big(x(k),\underline{\hat{u}}_k^{k+q}\big)\\ =& V_q(x(k)),\end{aligned}$$

which completes the proof. □

From Theorem 2.32, we know that the value function $V_i(x(k))\ge 0$ is a monotonically nonincreasing sequence and is bounded below for iteration index $i=1,2,\ldots$. Now, we can derive the following theorem.

Theorem 2.33 *Let $x(k)$ be an arbitrary state vector. Define the performance index function $V_\infty(x(k))$ as the limit of the iterative function $V_i(x(k))$, i.e.,*

$$V_\infty(x(k))=\lim_{i\to\infty}V_i(x(k)). \tag{2.188}$$

Then, we have

$$V_\infty(x(k))=\min_{u(k)}\{l(x(k),u(k))+V_\infty(x(k+1))\}.$$

Proof Let $\eta_k=\eta(x(k))$ be any admissible control vector. According to Theorem 2.32, for $\forall\, i$, we have

$$V_\infty(x(k))\le V_{i+1}(x(k))\le l(x(k),\eta_k)+V_i(x(k+1)).$$

Let $i \to \infty$, we have

$$V_\infty(x(k)) \le l(x(k), \eta_k) + V_\infty(x(k+1)),$$

which is true for $\forall\, \eta_k$. Therefore,

$$V_\infty(x(k)) \le \min_{u(k)}\{l(x(k), u(k)) + V_\infty(x(k+1))\}. \tag{2.189}$$

Let $\varepsilon > 0$ be an arbitrary positive number. Since $V_i(x(k))$ is nonincreasing for $i \ge 1$ and $\lim_{i\to\infty} V_i(x(k)) = V_\infty(x(k))$, there exists a positive integer p such that

$$V_p(x(k)) - \varepsilon \le V_\infty(x(k)) \le V_p(x(k)).$$

From (2.183), we have

$$\begin{aligned} V_p(x(k)) &= \min_{u(k)}\{l(x(k), u(k)) + V_{p-1}(F(x(k), u(k)))\} \\ &= l(x(k), v_p(x(k))) + V_{p-1}(F(x(k), v_p(x(k)))). \end{aligned}$$

Hence,

$$\begin{aligned} V_\infty(x(k)) &\ge l(x(k), v_p(x(k))) + V_{p-1}(F(x(k), v_p(x(k)))) - \varepsilon \\ &\ge l(x(k), v_p(x(k))) + V_\infty(F(x(k), v_p(x(k)))) - \varepsilon \\ &\ge \min_{u(k)}\{l(x(k), u(k)) + V_\infty(x(k+1))\} - \varepsilon. \end{aligned}$$

Since ε is arbitrary, we have

$$V_\infty(x(k)) \ge \min_{u(k)}\{l(x(k), u(k)) + V_\infty(x(k+1))\}. \tag{2.190}$$

Combining (2.189) and (2.190), we prove the theorem. □

Next, we will prove that the iterative value function $V_i(x(k))$ converges to the optimal value function $J^*(x(k))$ as $i \to \infty$.

Theorem 2.34 (cf. [14]) *Let $V_\infty(x(k))$ be defined in (2.188). If the system state $x(k)$ is controllable, then we have the value function $V_\infty(x(k))$ equal to the optimal value function $J^*(x(k))$, i.e.,*

$$\lim_{i\to\infty} V_i(x(k)) = J^*(x(k)),$$

where $V_i(x(k))$ is defined in (2.183).

Proof According to (2.178) and (2.185), we have

$$J^*(x(k)) \le \min_{\underline{u}_k^{k+i-1}} \left\{ J(x(k), \underline{u}_k^{k+i-1}) : \underline{u}_k^{k+i-1} \in \mathfrak{A}_{x(k)}^{(i)} \right\} = V_i(x(k)).$$

Then, let $i \to \infty$, and we obtain

$$J^*(x(k)) \leq V_\infty(x(k)). \tag{2.191}$$

Next, we show that

$$V_\infty(x(k)) \leq J^*(x(k)). \tag{2.192}$$

For any $\omega > 0$, by the definition of $J^*(x(k))$ in (2.178), there exists $\underline{\eta}_k \in \mathfrak{A}_{x(k)}$ such that

$$J(x(k), \underline{\eta}_k) \leq J^*(x(k)) + \omega. \tag{2.193}$$

Suppose that $|\underline{\eta}_k| = p$. Then, $\underline{\eta}_k \in \mathfrak{A}^{(p)}_{x(k)}$. So, by Theorem 2.32 and (2.185), we have

$$\begin{aligned} V_\infty(x(k)) &\leq V_p(x(k)) \\ &= \min_{\underline{u}_k^{k+p-1}} \left\{ J(x(k), \underline{u}_k^{k+p-1}) : \underline{u}_k^{k+p-1} \in \mathfrak{A}^{(p)}_{x(k)} \right\} \\ &\leq J(x(k), \underline{\eta}_k) \\ &\leq J^*(x(k)) + \omega. \end{aligned}$$

Since ω is chosen arbitrarily, we know that (2.192) is true. Therefore, from (2.191) and (2.192), we have proven the theorem. □

We can now present the following corollary.

Corollary 2.35 *Let the value function $V_i(x(k))$ be defined by (2.183). If the system state $x(k)$ is controllable, then the iterative control law $v_i(x(k))$ converges to the optimal control law $u^*(x(k))$, i.e.,*

$$\lim_{i \to \infty} v_i(x(k)) = u^*(x(k))$$

Remark 2.36 Generally speaking, for the *finite-horizon* optimal control problem, the optimal value function depends not only on state $x(k)$ but also on the time left (see [7] and [11]). For the finite-horizon optimal control problem with unspecified terminal time, we have proved that the iterative value functions converge to the optimal as the iterative index i reaches infinity. Then, the time left is negligible and we say that the optimal value function $J^*(x(k))$ is only a function of the state $x(k)$ which is like the case of infinite-horizon optimal control problems.

According to Theorem 2.34 and Corollary 2.35, we know that if $x(k)$ is controllable, then, as $i \to \infty$, the iterative value function $V_i(x(k))$ converges to the optimal value function $J^*(x(k))$ and the iterative control law $v_i(x(k))$ also converges to the optimal control law $u^*(x(k))$. So, it is important to note that for controllable state

$x(k)$, the iterative value functions $V_i(x(k))$ are well defined for all i under the iterative control law $v_i(x(k))$.

Let $\mathcal{T}_0 = \{0\}$. For $i = 1, 2, \ldots$, define

$$\mathcal{T}_i = \{x(k) \in \mathbb{R}^n | \exists u(k) \in \mathbb{R}^m \text{ s.t. } F(x(k), u(k)) \in \mathcal{T}_{i-1}\}. \tag{2.194}$$

Next, we prove the following theorem.

Theorem 2.37 *Let $\mathcal{T}_0 = \{0\}$ and $\mathcal{T}_i$ be defined in (2.194). Then, for $i = 0, 1, \ldots$, we have $\mathcal{T}_i \subseteq \mathcal{T}_{i+1}$.*

Proof We prove the theorem by mathematical induction. First, let $i = 0$. Since $\mathcal{T}_0 = \{0\}$ and $F(0, 0) = 0$, we know that $0 \in \mathcal{T}_1$. Hence, $\mathcal{T}_0 \subseteq \mathcal{T}_1$.

Next, assume that $\mathcal{T}_{i-1} \subseteq \mathcal{T}_i$ holds. Now, if $x(k) \in \mathcal{T}_i$, we have $F(x(k), \eta_{i-1}(x(k))) \in \mathcal{T}_{i-1}$ for some $\eta_{i-1}(x(k))$. Hence, $F(x(k), \eta_{i-1}(x(k))) \in \mathcal{T}_i$ by the assumption of $\mathcal{T}_{i-1} \subseteq \mathcal{T}_i$. So, $x(k) \in \mathcal{T}_{i+1}$ by (2.194). Thus, $\mathcal{T}_i \subseteq \mathcal{T}_{i+1}$, which proves the theorem. □

According to Theorem 2.37, we have

$$\{0\} = \mathcal{T}_0 \subseteq \mathcal{T}_1 \subseteq \cdots \subseteq \mathcal{T}_{i-1} \subseteq \mathcal{T}_i \subseteq \cdots.$$

We can see that by introducing the sets $\mathcal{T}_i$, $i = 0, 1, \ldots$, the state $x(k)$ can be classified correspondingly. According to Theorem 2.37, the properties of the ADP algorithm can be derived in the following theorem.

Theorem 2.38

(i) *For any i, $x(k) \in \mathcal{T}_i \Leftrightarrow \mathfrak{A}_{x(k)}^{(i)} \neq \emptyset \Leftrightarrow V_i(x(k))$ is defined at $x(k)$.*
(ii) *Let $\mathcal{T}_\infty = \bigcup_{i=1}^{\infty} \mathcal{T}_i$. Then, $x(k) \in \mathcal{T}_\infty \Leftrightarrow \mathfrak{A}_{x(k)} \neq \emptyset \Leftrightarrow J^*(x(k))$ is defined at $x(k) \Leftrightarrow x(k)$ is controllable.*
(iii) *If $V_i(x(k))$ is defined at $x(k)$, then $V_j(x(k))$ is defined at $x(k)$ for every $j \geq i$.*
(iv) *$J^*(x(k))$ is defined at $x(k)$ if and only if there exists an i such that $V_i(x(k))$ is defined at $x(k)$.*

2.5.2.2 The ε-Optimal Control Algorithm

In the previous subsection, we have proved that the iterative value function $V_i(x(k))$ converges to the optimal value function $J^*(x(k))$ and $J^*(x(k)) = \min_{\underline{u}_k}\{J(x(k), \underline{u}_k),\ \underline{u} \in \mathfrak{A}_{x(k)}\}$ satisfies the Bellman's equation (2.179) for any controllable state $x(k) \in \mathcal{T}_\infty$.

To obtain the optimal value function $J^*(x(k))$, a natural strategy is to run the iterative ADP algorithm (2.181)–(2.184) until $i \to \infty$. But unfortunately, it is not practical to do so. In many cases, we cannot find the equality $J^*(x(k)) = V_i(x(k))$ for any finite i. That is, for any admissible control sequence $\underline{u}_k$ with finite length,

the cost starting from $x(k)$ under the control of $\underline{u}_k$ will be larger than, not equal to, $J^*(x(k))$. On the other hand, by running the iterative ADP algorithm (2.181)–(2.184), we can obtain a control vector $v_\infty(x(k))$ and then construct a control sequence $\underline{u}_\infty(x(k)) = (v_\infty(x(k)), v_\infty(x(k+1)), \ldots, v_\infty(x(k+i)), \ldots)$, where $x(k+1) = F(x(k), v_\infty(x(k))), \ldots, x(k+i) = F(x(k+i-1), v_\infty(x(k+i-1)))$, ... In general, $\underline{u}_\infty(x(k))$ has infinite length. That is, the controller $v_\infty(x(k))$ cannot control the state to reach the target in finite number of steps. To overcome this difficulty, a new ε-optimal control method using iterative ADP algorithm will be developed.

First, we will introduce our method of iterative ADP with the consideration of the length of control sequences. For different $x(k)$, we will consider different length i for the optimal control sequence. For a given error bound $\varepsilon > 0$, the number i will be chosen so that the error between $J^*(x(k))$ and $V_i(x(k))$ is within the bound.

Let $\varepsilon > 0$ be any small number and $x(k) \in \mathcal{T}_\infty$ be any controllable state. Let the value function $V_i(x(k))$ be defined by (2.183) and $J^*(x(k))$ be the optimal value function. According to Theorem 2.34, given $\varepsilon > 0$, there exists a finite i such that

$$|V_i(x(k)) - J^*(x(k))| \leq \varepsilon. \tag{2.195}$$

Definition 2.39 (cf. [14]) Let $x(k) \in \mathcal{T}_\infty$ be a controllable state vector. Let $\varepsilon > 0$ be a small positive number. The approximate length of optimal control sequence with respect to ε is defined as

$$K_\varepsilon(x(k)) = \min\{i : |V_i(x(k)) - J^*(x(k))| \leq \varepsilon\}. \tag{2.196}$$

Given a small positive number ε, for any state vector $x(k)$, the number $K_\varepsilon(x(k))$ gives a suitable length of control sequence for optimal control starting from $x(k)$. For $x(k) \in \mathcal{T}_\infty$, since $\lim_{i\to\infty} V_i(x(k)) = J^*(x(k))$, we can always find i such that (2.195) is satisfied. Therefore, $\{i : |V_i(x(k)) - J^*(x(k))| \leq \varepsilon\} \neq \emptyset$ and $K_\varepsilon(x(k))$ is well defined.

We can see that an error ε between $V_i(x(k))$ and $J^*(x(k))$ is introduced into the iterative ADP algorithm which makes the value function $V_i(x(k))$ converge within a finite number of iteration steps. In this part, we will show that the corresponding control is also an effective control that drives the value function to within error bound ε from its optimal.

From Definition 2.39, we can see that all the states $x(k)$ that satisfy (2.196) can be classified into one set. Motivated by the definition in (2.194), we can further classify this set using the following definition.

Definition 2.40 (cf. [14]) Let ε be a positive number. Define $\mathcal{T}_0^{(\varepsilon)} = \{0\}$ and for $i = 1, 2, \ldots$, define

$$\mathcal{T}_i^{(\varepsilon)} = \{x(k) \in \mathcal{T}_\infty : K_\varepsilon(x(k)) \leq i\}.$$

Accordingly, when $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, to find the optimal control sequence which has value less than or equal to $J^*(x(k)) + \varepsilon$, one only needs to consider the control sequences $\underline{u}_k$ with length $|\underline{u}_k| \leq i$. The sets $\mathcal{T}_i^{(\varepsilon)}$ have the following properties.

Theorem 2.41 (cf. [14]) *Let $\varepsilon > 0$ and $i = 0, 1, \ldots$. Then*:

(i) $x(k) \in \mathcal{T}_i^{(\varepsilon)}$ *if and only if* $V_i(x(k)) \leq J^*(x(k)) + \varepsilon$
(ii) $\mathcal{T}_i^{(\varepsilon)} \subseteq \mathcal{T}_i$
(iii) $\mathcal{T}_i^{(\varepsilon)} \subseteq \mathcal{T}_{i+1}^{(\varepsilon)}$
(iv) $\bigcup_i \mathcal{T}_i^{(\varepsilon)} = \mathcal{T}_\infty$
(v) *If $\varepsilon > \delta > 0$, then* $\mathcal{T}_i^{(\varepsilon)} \supseteq \mathcal{T}_i^{(\delta)}$

Proof (i) Let $x(k) \in \mathcal{T}_i^{(\varepsilon)}$. By Definition 2.40, $K_\varepsilon(x(k)) \leq i$. Let $j = K_\varepsilon(x(k))$. Then, $j \leq i$ and by Definition 2.39, $|V_j(x(k)) - J^*(x(k))| \leq \varepsilon$. So, $V_j(x(k)) \leq J^*(x(k)) + \varepsilon$. By Theorem 2.32, $V_i(x(k)) \leq V_j(x(k)) \leq J^*(x(k)) + \varepsilon$. On the other hand, if $V_i(x(k)) \leq J^*(x(k)) + \varepsilon$, then $|V_i(x(k)) - J^*(x(k))| \leq \varepsilon$. So, $K_\varepsilon(x(k)) = \min\{j : |V_j(x(k)) - J^*(x(k))| \leq \varepsilon\} \leq i$, which implies that $x(k) \in \mathcal{T}_i^{(\varepsilon)}$.

(ii) If $x(k) \in \mathcal{T}_i^{(\epsilon)}$, $K_\epsilon(x(k)) \leq i$ and $|V_i(x(k)) - J^*(x(k))| \leq \varepsilon$. So, $V_i(x(k))$ is defined at $x(k)$. According to Theorem 2.38 (i), we have $x(k) \in \mathcal{T}_i$. Hence, $\mathcal{T}_i^{(\epsilon)} \subseteq \mathcal{T}_i$.

(iii) If $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, $K_\varepsilon(x(k)) \leq i < i + 1$. So, $x(k) \in \mathcal{T}_{i+1}^{(\varepsilon)}$. Thus, $\mathcal{T}_i^{(\varepsilon)} \subseteq \mathcal{T}_{i+1}^{(\varepsilon)}$.

(iv) Obviously, $\bigcup_i \mathcal{T}_i^{(\varepsilon)} \subseteq \mathcal{T}_\infty$ since $\mathcal{T}_i^{(\varepsilon)}$ are subsets of $\mathcal{T}_\infty$. For any $x(k) \in \mathcal{T}_\infty$, let $p = K_\varepsilon(x(k))$. Then, $x(k) \in \mathcal{T}_p^{(\varepsilon)}$. So, $x(k) \in \bigcup_i \mathcal{T}_i^{(\varepsilon)}$. Hence, $\mathcal{T}_\infty \subseteq \bigcup_i \mathcal{T}_i^{(\varepsilon)} \subseteq \mathcal{T}_\infty$, and we obtain, $\bigcup_i \mathcal{T}_i^{(\varepsilon)} = \mathcal{T}_\infty$.

(v) If $x(k) \in \mathcal{T}_i^{(\delta)}$, $V_i(x(k)) \leq J^*(x(k)) + \delta$ by part (i) of this theorem. Clearly, $V_i(x(k)) \leq J^*(x(k)) + \varepsilon$ since $\delta < \varepsilon$. This implies that $x(k) \in \mathcal{T}_i^{(\varepsilon)}$. Therefore, $\mathcal{T}_i^{(\varepsilon)} \supseteq \mathcal{T}_i^{(\delta)}$. □

According to Theorem 2.41(i), $\mathcal{T}_i^{(\varepsilon)}$ is just the region where $V_i(x(k))$ is close to $J^*(x(k))$ with error less than ε. This region is a subset of $\mathcal{T}_i$ according to Theorem 2.41(ii). As stated in Theorem 2.41(iii), when i is large, the set $\mathcal{T}_i^{(\varepsilon)}$ is also large. That means that, when i is large, we have a large region where we can use $V_i(x(k))$ as the approximation of $J^*(x(k))$ under certain error. On the other hand, we claim that if $x(k)$ is far away from the origin, we have to choose long control sequence to approximate the optimal control sequence. Theorem 2.41(iv) means that for every controllable state $x(k) \in \mathcal{T}_\infty$, we can always find a suitable control sequence with length i to approximate the optimal control. The size of the set $\mathcal{T}_i^{(\varepsilon)}$ depends on the value of ε. Smaller value of ε gives smaller set $\mathcal{T}_i^{(\varepsilon)}$ which is indicated by Theorem 2.41(v).

Let $x(k) \in \mathcal{T}_\infty$ be an arbitrary controllable state. If $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, the iterative value function satisfies (2.195) under the control $v_i(x(k))$, we call this control the

ε-optimal control and denote it as $\mu_\varepsilon^*(x(k))$, i.e.,

$$\mu_\varepsilon^*(x(k)) = v_i(x(k)) = \arg\min_{u(k)} \{l(x(k), u(k)) + V_{i-1}(F(x(k), u(k)))\}. \quad (2.197)$$

We have the following corollary.

Corollary 2.42 (cf. [14]) *Let $\mu_\varepsilon^*(x(k))$ be expressed in (2.197) that makes the value function satisfy (2.195) for $x(k) \in \mathcal{T}_i^{(\varepsilon)}$. Then, for any $x_k' \in \mathcal{T}_i^{(\varepsilon)}$, $\mu_\varepsilon^*(x_k')$ guarantees*

$$|V_i(x_k') - J^*(x_k')| \leq \varepsilon. \quad (2.198)$$

Proof The corollary can be proved by contradiction. Assume that the conclusion is not true. Then, the inequality (2.198) is false under the control $\mu_\varepsilon^*(\cdot)$ for some $x_k'' \in \mathcal{T}_i^{(\varepsilon)}$.

As $\mu_\varepsilon^*(x(k))$ makes the value function satisfy (2.195) for $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, we have $K_\varepsilon(x(k)) \leq i$. Using the ε-optimal control law $\mu_\varepsilon^*(\cdot)$ at the state x_k'', according to the assumption, we have $|V_i(x_k'') - J^*(x_k'')| > \varepsilon$. Then, $K_\varepsilon(x_k'') > i$ and $x_k'' \notin \mathcal{T}_i^{(\varepsilon)}$. It is in contradiction with the assumption $x_k'' \in \mathcal{T}_i^{(\varepsilon)}$. Therefore, the assumption is false and (2.198) holds for any $x_k' \in \mathcal{T}_i^{(\varepsilon)}$. □

Remark 2.43 Corollary 2.42 is very important for neural-network implementation of the iterative ADP algorithm. It shows that we do not need to obtain the optimal control law by searching the entire subset $\mathcal{T}_i^{(\varepsilon)}$. Instead, we can just find one point of $\mathcal{T}_i^{(\varepsilon)}$, i.e., $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, to obtain the ε-optimal control $\mu_\varepsilon^*(x(k))$ which will be effective for any other state $x_k' \in \mathcal{T}_i^{(\varepsilon)}$. This property not only makes the computational complexity much reduced but also makes the optimal control law easily obtained using neural networks.

Theorem 2.44 (cf. [14]) *Let $x(k) \in \mathcal{T}_i^{(\varepsilon)}$ and let $\mu_\varepsilon^*(x(k))$ be expressed in (2.197). Then, $F(x(k), \mu_\varepsilon^*(x(k))) \in \mathcal{T}_{i-1}^{(\varepsilon)}$. In other words, if $K_\varepsilon(x(k)) = i$, then $K_\varepsilon(F(x(k), \mu_\varepsilon^*(x(k)))) \leq i - 1$.*

Proof Since $x(k) \in \mathcal{T}_i^{(\varepsilon)}$, by Theorem 2.41 (i) we know that

$$V_i(x(k)) \leq J^*(x(k)) + \varepsilon. \quad (2.199)$$

According to (2.183) and (2.197), we have

$$V_i(x(k)) = l(x(k), \mu_\varepsilon^*(x(k))) + V_{i-1}(F(x(k), \mu_\varepsilon^*(x(k)))). \quad (2.200)$$

Combining (2.199) and (2.200), we have

$$\begin{aligned} V_{i-1}(F(x(k), \mu_\varepsilon^*(x(k)))) &= V_i(x(k)) - l(x(k), \mu_\varepsilon^*(x(k))) \\ &\leq J^*(x(k)) + \varepsilon - l(x(k), \mu_\varepsilon^*(x(k))). \end{aligned} \quad (2.201)$$

On the other hand, we have

$$J^*(x(k)) \leq l(x(k), \mu_\varepsilon^*(x(k))) + J^*(F(x, \mu_\varepsilon^*(x(k)))). \tag{2.202}$$

Putting (2.202) into (2.201), we obtain

$$V_{i-1}(F(x(k), \mu_\varepsilon^*(x(k)))) \leq J^*(F(x(k), \mu_\varepsilon^*(x(k)))) + \varepsilon.$$

By Theorem 2.41 (i), we have

$$F(x(k), \mu_\varepsilon^*(x(k))) \in \mathcal{T}_{i-1}^{(\varepsilon)}. \tag{2.203}$$

So, if $K_\varepsilon(x(k)) = i$, we know that $x(k) \in \mathcal{T}_i^{(\varepsilon)}$ and $F(x, \mu_\varepsilon^*(x(k))) \in \mathcal{T}_{i-1}^{(\varepsilon)}$ according to (2.203). Therefore, we have

$$K_\varepsilon(F(x(k), \mu_\varepsilon^*(x(k)))) \leq i - 1,$$

which proves the theorem. □

Remark 2.45 From Theorem 2.44 we can see that the parameter $K_\varepsilon(x(k))$ gives an important property of the finite-horizon ADP algorithm. It not only gives an optimal condition of the iterative process, but also gives an optimal number of control steps for the finite-horizon ADP algorithm. For example, if $|V_i(x(k)) - J^*(x(k))| \leq \varepsilon$ for small ε, then we have $V_i(x(k)) \approx J^*(x(k))$. According to Theorem 2.44, we can get $N = k + i$, where N is the number of control steps to drive the system to zero. The whole control sequence $\underline{u}_0^{N-1}$ may not be ε-optimal but the control sequence $\underline{u}_k^{N-1}$ is ε-optimal control sequence. If $k = 0$, we have $N = K_\varepsilon(x(0)) = i$. Under this condition, we say that the iteration index $K_\varepsilon(x(0))$ denotes the number of ε-optimal control steps.

Corollary 2.46 *Let $\mu_\varepsilon^*(x(k))$ be expressed in (2.197) that makes the value function satisfy (2.195) for $x(k) \in \mathcal{T}_i^{(\varepsilon)}$. Then, for any $x_k' \in \mathcal{T}_j^{(\varepsilon)}$, where $0 \leq j \leq i$, $\mu_\varepsilon^*(x_k')$ guarantees*

$$|V_i(x_k') - J^*(x_k')| \leq \varepsilon. \tag{2.204}$$

Proof The proof is similar to Corollary 2.42 and is omitted here. □

Remark 2.47 Corollary 2.46 shows that the ε-optimal control $\mu_\varepsilon^*(x(k))$ obtained for $\forall\, x(k) \in \mathcal{T}_i^{(\varepsilon)}$ is effective for any state $x_k' \in \mathcal{T}_j^{(\epsilon)}$, where $0 \leq j \leq i$. This means that for $\forall\, x_k' \in \mathcal{T}_j^{(\varepsilon)}$, $0 \leq j \leq i$, we can use a same ε-optimal control $\mu_\varepsilon^*(x_k')$ to control the system.

According to Theorem 2.41(iii) and Corollary 2.42, the ε-optimal control $\mu_\varepsilon^*(x(k))$ obtained for an $x(k) \in \mathcal{T}_i^{(\varepsilon)}$ is effective for any state $x_k' \in \mathcal{T}_{i-1}^{(\varepsilon)}$ (which

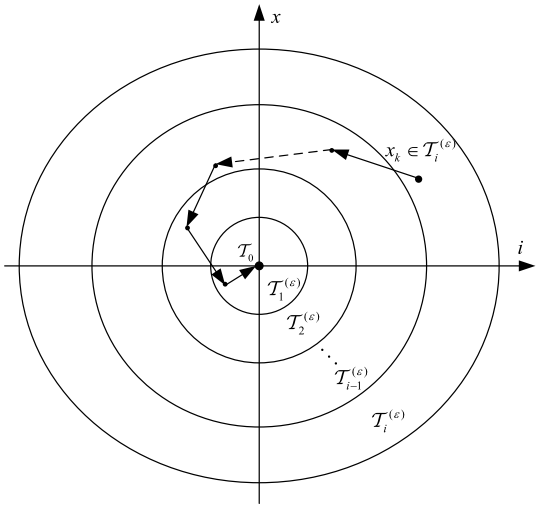

Fig. 2.26 The control process of the controllable sate $x(k) \in \mathcal{T}_i^{(\varepsilon)}$ using the iterative ADP algorithm

is also stated in Corollary 2.46). That is to say: in order to obtain effective ε-optimal control, the iterative ADP algorithm only needs to run at some state $x(k) \in \mathcal{T}_\infty$. In order to obtain an effective ε-optimal control law $\mu_\varepsilon^*(x(k))$, we should choose the state $x(k) \in \mathcal{T}_i^{(\varepsilon)} \backslash \mathcal{T}_{i-1}^{(\varepsilon)}$ for each i to run the iterative ADP algorithm. The control process using the iterative ADP algorithm is illustrated in Fig. 2.26.

From the iterative ADP algorithm (2.181)–(2.184), we can see that for any state $x(k) \in \mathbb{R}^n$, there exists a control $u(k) \in \mathbb{R}^m$ that drives the system to zero in one step. In other words, for $\forall x(k) \in \mathbb{R}^n$, there exists a control $u(k) \in \mathbb{R}^m$ such that $x(k+1) = F(x(k), u(k)) = 0$ holds. A large class of systems possesses this property; for example, all linear systems of the type $x(k+1) = Ax(k) + Bu(k)$ when B is invertible and the affine nonlinear systems with the type $x(k+1) = f(x(k)) + g(x(k))u(k)$ when the inverse of $g(x(k))$ exists. But there are also other classes of systems for which there does not exist any control $u(k) \in \mathbb{R}^m$ that drives the state to zero in one step for some $x(k) \in \mathbb{R}^n$, i.e., $\exists\, x(k) \in \mathbb{R}^n$ such that $F(x(k), u(k)) = 0$ is not possible for $\forall u(k) \in \mathbb{R}^m$. In the following part, we will discuss the situation where $F(x(k), u(k)) \neq 0$ for some $x(k) \in \mathbb{R}^m$.

Since $x(k)$ is controllable, there exists a finite-horizon admissible control sequence $\underline{u}_k^{k+i-1} = (u_k, u(k+1), \ldots, u_{k+i-1}) \in \mathfrak{A}_{x(k)}^{(i)}$ that makes $x^{(f)}(x(k), \underline{u}_k^{k+i-1}) = x(k+i) = 0$. Let $N = k+i$ be the terminal time. Assume that for $k+1, k+2, \ldots, N-1$, the optimal control sequence $\underline{u}_{k+1}^{(N-1)*} = (u^*(k+1), u_{k+2}^*, \ldots, u_{N-1}^*) \in \mathfrak{A}_{x(k+1)}^{(N-k-1)}$ has been determined. Denote the value function for $x(k+1)$ as $J(x(k+1), \underline{u}_{k+1}^{(N-1)*}) = V_0(x(k+1))$. Now, we use the iterative ADP algorithm to determine the optimal control sequence for the state $x(k)$.

The value function for $i = 1$ is computed as

$$V_1(x(k)) = l(x(k), v_1(x(k))) + V_0(F(x(k), v_1(x(k)))), \tag{2.205}$$

where

$$v_1(x(k)) = \arg\min_{u(k)}\{l(x(k), u(k)) + V_0(F(x(k), u(k)))\}. \tag{2.206}$$

Note that the initial condition used in the above expression is the value function V_0 which is obtained previously for $x(k+1)$ and now applied at $F(x(k), u(k))$. For $i = 2, 3, 4, \ldots$, the iterative ADP algorithm will be implemented as follows:

$$V_i(x(k)) = l(x(k), v_i(x(k))) + V_{i-1}(F(x(k), v_i(x(k)))), \tag{2.207}$$

where

$$v_i(x(k)) = \arg\min_{u(k)}\{l(x(k), u(k)) + V_{i-1}(F(x(k), u(k)))\}. \tag{2.208}$$

Theorem 2.48 *Let $x(k)$ be an arbitrary controllable state vector. Then, the value function $V_i(x(k))$ obtained by (2.205)–(2.208) is a monotonically nonincreasing sequence for $\forall\, i \geq 0$, i.e., $V_{i+1}(x(k)) \leq V_i(x(k))$ for $\forall\, i \geq 0$.*

Proof It can easily be proved by following the proof of Theorem 2.32, and the proof is omitted here. □

Theorem 2.49 *Let the value function $V_i(x(k))$ be defined by (2.207). If the system state $x(k)$ is controllable, then the value function $V_i(x(k))$ obtained by (2.205)–(2.208) converges to the optimal value function $J^*(x(k))$ as $i \to \infty$, i.e.,*

$$\lim_{i\to\infty} V_i(x(k)) = J^*(x(k)).$$

Proof This theorem can be proved following similar steps to the proof of Theorem 2.34 and the proof is omitted here. □

We can see that the iterative ADP algorithm (2.205)–(2.208) is an expansion from of the previous one (2.181)–(2.184). So, the properties of the iterative ADP algorithm (2.181)–(2.184) is also effective for the current one (2.205)–(2.208). But there also exist differences. From Theorem 2.32, we can see that $V_{i+1}(x(k)) \leq V_i(x(k))$ for all $i \geq 1$, which means that $V_1(x(k)) = \max\{V_i(x(k)) : i = 0, 1, \ldots\}$. While Theorem 2.48 shows that $V_{i+1}(x(k)) \leq V_i(x(k))$ for all $i \geq 0$ which means that $V_0(x(k)) = \max\{V_i(x(k)) : i = 0, 1, \ldots\}$. This difference is caused by the difference of the initial conditions of the two iterative ADP algorithms.

In the previous iterative ADP algorithm (2.181)–(2.184), it begins with the initial value function $V_0(x(k)) = 0$ since $F(x(k), u(k)) = 0$ can be solved. While in the current iterative ADP algorithm (2.205)–(2.208), it begins with the value function V_0 for the state $x(k+1)$ which is determined previously. This also causes the difference between the proofs of Theorems 2.32 and 2.34 and the corresponding results in Theorems 2.48 and 2.49. But the difference of the initial conditions of the iterative performance index function does not affect the convergence property of the two iterative ADP algorithms.

For the iterative ADP algorithm, the optimal criterion (2.195) is very difficult to verify because the optimal value function $J^*(x(k))$ is unknown in general. So, an equivalent criterion is established to replace (2.195).

If $|V_i(x(k)) - J^*(x(k))| \leq \varepsilon$ holds, we have $V_i(x(k)) \leq J^*(x(k)) + \varepsilon$ and $J^*(x(k)) \leq V_{i+1}(x(k)) \leq V_i(x(k))$. These imply that

$$0 \leq V_i(x(k)) - V_{i+1}(x(k)) \leq \varepsilon, \tag{2.209}$$

or

$$|V_i(x(k)) - V_{i+1}(x(k))| \leq \varepsilon.$$

On the other hand, according to Theorem 2.49, $|V_i(x(k)) - V_{i+1}(x(k))| \to 0$ implies that $V_i(x(k)) \to J^*(x(k))$. Therefore, for any given small ε, if $|V_i(x(k)) - V_{i+1}(x(k))| \leq \varepsilon$ holds, we have $|V_i(x(k)) - J^*(x(k))| \leq \varepsilon$ if i is sufficiently large.

We will use inequality (2.209) as the optimal criterion instead of the optimal criterion (2.195).

Let $\underline{\widehat{u}}_0^{K-1} = (u(0), u(1), \ldots, u(K-1))$ be an arbitrary finite-horizon admissible control sequence and the corresponding state sequence be $\underline{\widehat{x}}_0^{K} = (x(0), x(1), \ldots, x(K))$ where $x(K) = 0$.

We can see that the initial control sequence $\underline{\widehat{u}}_0^{K-1} = (u(0), u(1), \ldots, u(K-1))$ may not be optimal, which means that the initial number of control steps K may not be optimal. So, the iterative ADP algorithm must complete two kinds of optimization. One is to optimize the number of control steps. The other is to optimize the control law. In the following, we will show how the number of control steps and the control law are optimized simultaneously in the iterative ADP algorithm.

For the state $x(K-1)$, we have $F(x(K-1), u(K-1)) = 0$. Then, we run the iterative ADP algorithm (2.181)–(2.184) at $x(K-1)$ as follows. The value function for $i = 1$ is computed as

$$\begin{aligned} V_1^1(x(K-1)) &= \min_{u(K-1)} \{l(x(K-1), u(K-1)) + V_0(F(x(K-1), u(K-1)))\} \\ &\quad \text{subject to } F(x(K-1), u(K-1)) = 0 \\ &= l(x(K-1), v_1^1(x(K-1))), \end{aligned} \tag{2.210}$$

where

$$\begin{aligned} v_1^1(x(K-1)) &= \arg \min_{u(K-1)} l(x(K-1), u(K-1)) \\ &\quad \text{subject to } F(x(K-1), u(K-1)) = 0, \end{aligned} \tag{2.211}$$

and $V_0(F(x(K-1), u(K-1))) = 0$. The iterative ADP algorithm will be implemented as follows for $i = 2, 3, 4, \ldots$:

$$V_i^1(x(K-1)) = l(x(K-1), v_i^1(x(K-1)))$$

$$+ V_{i-1}^1(F(x(K-1), v_i^1(x(K-1)))), \tag{2.212}$$

where

$$v_i^1(x(K-1)) = \arg\min_{u(K-1)} \left\{ l(x(K-1), u(K-1)) + V_{i-1}^1(F(x(K-1), u(K-1))) \right\}, \tag{2.213}$$

until the inequality

$$\left| V_{l_1}^1(x(K-1)) - V_{l_1+1}^1(x(K-1)) \right| \leq \varepsilon \tag{2.214}$$

is satisfied for $l_1 > 0$. This means that $x(K-1) \in \mathcal{T}_{l_1}^{(\varepsilon)}$ and the optimal number of control steps is $K_\varepsilon(x(K-1)) = l_1$.

Considering $x(K-2)$, we have $F(x(K-2), u(K-2)) = x(K-1)$. Put $x(K-2)$ into (2.214). If $|V_{l_1}^1(x(K-2)) - V_{l_1+1}^1(x(K-2))| \leq \varepsilon$ holds, then according to Theorem 2.41(i), we know that $x(K-2) \in \mathcal{T}_{l_1}^{(\varepsilon)}$. Otherwise, if $x(K-2) \notin \mathcal{T}_{l_1}^{(\varepsilon)}$, we will run the iterative ADP algorithm as follows. Using the value function $V_{l_1}^1$ as the initial condition, we compute, for $i = 1$,

$$V_1^2(x(K-2)) = l(x(K-2), v_1^2(x(K-2))) + V_{l_1}^1(F(x(K-2), v_1^2(x(K-2)))), \tag{2.215}$$

where

$$v_1^2(x(K-2)) = \arg\min_{u(K-2)} \{ l(x(K-2), u(K-2)) + V_{l_1}^1(F(x(K-2), u(K-2))) \}. \tag{2.216}$$

For $i = 2, 3, 4, \ldots$, the iterative ADP algorithm will be implemented as follows:

$$V_i^2(x(K-2)) = l(x(K-2), v_i^2(x(K-2))) + V_{i-1}^2(F(x(K-2), v_i^2(x(K-2)))), \tag{2.217}$$

where

$$v_i^2(x(K-2)) = \arg\min_{u(K-2)} \left\{ l(x(K-2), u_{K-2}) + V_{i-1}^2(F(x(K-2), u(K-2))) \right\}, \tag{2.218}$$

until the inequality

$$\left| V_{l_2}^2(x(K-2)) - V_{l_2+1}^2(x(K-2)) \right| \leq \varepsilon \tag{2.219}$$

is satisfied for $l_2 > 0$. We then obtain $x(K-2) \in \mathcal{T}_{l_2}^{(\varepsilon)}$, and the optimal number of control steps is $K_\varepsilon(x(K-2)) = l_2$.

Next, assume that $j \geq 2$ and $x(K-j+1) \in \mathcal{T}_{l_{j-1}}^{(\varepsilon)}$, i.e.,

$$\left| V_{l_{j-1}}^{j-1}(x(K-j+1)) - V_{l_{j-1}+1}^{j-1}(x(K-j+1)) \right| \leq \varepsilon \tag{2.220}$$

holds. Considering $x(K-j)$, we have $F(x(K-j), u(K-j)) = x(K-j+1)$. Putting $x(K-j)$ into (2.220) and if

$$\left| V_{l_{j-1}}^{j-1}(x(K-j)) - V_{l_{j-1}+1}^{j-1}(x(K-j)) \right| \leq \varepsilon \tag{2.221}$$

holds, then we know that $x(K-j) \in \mathcal{T}_{l_{j-1}}^{(\varepsilon)}$. Otherwise, if $x(K-j) \notin \mathcal{T}_{l_{j-1}}^{(\varepsilon)}$, then we run the iterative ADP algorithm as follows. Using the performance index function $V_{l_{j-1}}^{j-1}$ as the initial condition, we compute for $i = 1$,

$$\begin{aligned} V_1^j(x(K-j)) = {} & l(x(K-j), v_1^j(x(K-j))) \\ & + V_{l_{j-1}}^{j-1}(F(x(K-j), v_1^j(x(K-j)))), \end{aligned} \tag{2.222}$$

where

$$\begin{aligned} v_1^j(x(K-j)) = \arg \min_{u(K-j)} \{ & l(x(K-j), u(K-j)) \\ & + V_{l_{j-1}}^{j-1}(F(x(K-j), u(K-j)))\}. \end{aligned} \tag{2.223}$$

For $i = 2, 3, 4, \ldots$, the iterative ADP algorithm will be implemented as follows:

$$\begin{aligned} V_i^j(x(K-j)) = {} & l(x(K-j), v_i^j(x(K-j))) \\ & + V_{i-1}^j(F(x(K-j), v_i^j(x(K-j)))), \end{aligned} \tag{2.224}$$

where

$$\begin{aligned} v_i^j(x(K-j)) = \arg \min_{u(K-j)} \big\{ & l(x(K-j), u(K-j)) \\ & + V_{i-1}^j(F(x(K-j), u(K-j)))\big\}, \end{aligned} \tag{2.225}$$

until the inequality

$$\left| V_{l_j}^j(x(K-j)) - V_{l_j+1}^j(x(K-j)) \right| \leq \varepsilon \tag{2.226}$$

is satisfied for $l_j > 0$. We then obtain $x(K-j) \in \mathcal{T}_{l_j}^{(\varepsilon)}$, and the optimal number of control steps is $K_\varepsilon(x(K-j)) = l_j$.

Finally, considering $x(0)$, we have $F(x(0), u(0)) = x(1)$. If

$$\left| V_{l_{K-1}}^{K-1}(x(0)) - V_{l_{K-1}+1}^{K-1}(x(0)) \right| \leq \varepsilon$$

holds, then we know that $x(0) \in \mathcal{T}_{l_{K-1}}^{(\varepsilon)}$. Otherwise, if $x(0) \notin \mathcal{T}_{l_{K-1}}^{(\varepsilon)}$, then we run the iterative ADP algorithm as follows. Using the performance index function $V_{l_{K-1}}^{K-1}$ as the initial condition, we compute, for $i = 1$,

$$V_1^K(x(0)) = l(x(0), v_1^K(x(0))) + V_{l_{K-1}}^{K-1}(F(x(0), v_1^K(x(0)))), \tag{2.227}$$

where

$$v_1^K(x(0)) = \arg\min_{u(0)} \{l(x(0), u(0)) + V_{l_{K-1}}^{K-1}(F(x(0), u(0)))\}. \tag{2.228}$$

For $i = 2, 3, 4, \ldots$, the iterative ADP algorithm will be implemented as follows:

$$V_i^K(x(0)) = l(x(0), v_i^K(x(0))) + V_{i-1}^K(F(x(0), v_i^K(x(0)))), \tag{2.229}$$

where

$$v_i^K(x(0)) = \arg\min_{u(0)} \left\{ l(x(0), u(0)) + V_{i-1}^K(F(x(0), u(0))) \right\}, \tag{2.230}$$

until the inequality

$$\left| V_{l_K}^K(x(0)) - V_{l_K+1}^K(x(0)) \right| \leq \varepsilon \tag{2.231}$$

is satisfied for $l_K > 0$. Therefore, we obtain $x(0) \in \mathcal{T}_{l_K}^{(\varepsilon)}$, and the optimal number of control steps is $K_\varepsilon(x(0)) = l_K$.

Starting from the initial state $x(0)$, the optimal number of control steps is l_K according to our ADP algorithm.

Remark 2.50 For the case where there are some $x(k) \in \mathbb{R}^n$, there does not exist a control $u(k) \in \mathbb{R}^m$ that drives the system to zero in one step; the computational complexity of the iterative ADP algorithm is strongly related to the original finite-horizon admissible control sequence $\underline{\widehat{u}}_0^{K-1}$. First, we repeat the iterative ADP algorithm at $x(K-1), x(K-2), \ldots, x(1), x(0)$, respectively. It is related to the control steps K of $\underline{\widehat{u}}_0^{K-1}$. If K is large, it means that $\underline{\widehat{u}}_0^{K-1}$ takes large number of control steps to drive the initial state $x(0)$ to zero and then the number of times needed to repeat the iterative ADP algorithm will be large. Second, the computational complexity is also related to the quality of control results of $\underline{\widehat{u}}_0^{K-1}$. If $\underline{\widehat{u}}_0^{K-1}$ is close to the optimal control sequence $\underline{u}_0^{(N-1)*}$, then it will take less computation to make (2.226) hold for each j.

Now, we summarize the iterative ADP algorithm as follows:

Step 1. Choose an error bound ε and choose randomly an array of initial states $x(0)$.

Step 2. Obtain an initial finite-horizon admissible control sequence $\widehat{\underline{u}}_0^{K-1} = (u(0), u(1), \ldots, u(K-1))$ and obtain the corresponding state sequence

$$\widehat{\underline{x}}_0^K = (x(0), x(1), \ldots, x(K)),$$

where $x(K) = 0$.

Step 3. For the state $x(K-1)$ with $F(x(K-1), u(K-1)) = 0$, run the iterative ADP algorithm (2.210)–(2.213) at $x(K-1)$ until (2.214) holds.

Step 4. Record $V_{l_1}^1(x(K-1))$, $v_{l_1}^1(x_{K-1})$ and $K_\varepsilon(x(K-1)) = l_1$.

Step 5. For $j = 2, 3, \ldots, K$, if for $x(K-j)$ the inequality (2.221) holds, go to Step 7; otherwise, go to Step 6.

Step 6. Using the value function $V_{l_{j-1}}^{j-1}$ as the initial condition, run the iterative ADP algorithm (2.222)–(2.225) until (2.226) is satisfied.

Step 7. If $j = K$, then we have obtained the optimal value function $V^*(x(0)) = V_{l_K}^K(x(0))$, the law of the optimal control sequence $u^*(x(0)) = v_{l_K}^K(x(0))$ and the number of optimal control steps $K_\varepsilon(x(0)) = l_K$; otherwise, set $j = j + 1$, and go to Step 5.

Step 8. Stop.

2.5.3 Simulations

To evaluate the performance of our iterative ADP algorithm, we choose two examples with quadratic utility functions for numerical experiments.

Example 2.51 Our first example is chosen from [16]. We consider the following nonlinear system:

$$x(k+1) = f(x(k)) + g(x(k))u(k),$$

where $x(k) = [x_1(k) \;\; x_2(k)]^{\mathrm{T}}$ and $u(k) = [u_1(k) \;\; u_2(k)]^{\mathrm{T}}$ are the state and control variables, respectively. The system functions are given as

$$f(x(k)) = \begin{bmatrix} 0.2x_1(k)\exp(x_2^2(k)) \\ 0.3x_2^3(k) \end{bmatrix}, \quad g(x(k)) = \begin{bmatrix} -0.2 & 0 \\ 0 & -0.2 \end{bmatrix}.$$

The initial state is $x(0) = [1 \;\; -1]^{\mathrm{T}}$. The value function is in quadratic form with finite time horizon expressed as

$$J\big(x(0), \underline{u}_0^{N-1}\big) = \sum_{k=0}^{N-1} \big(x^{\mathrm{T}}(k)Qx(k) + u^{\mathrm{T}}(k)Ru(k)\big),$$

where the matrix $Q = R = I$, and I denotes the identity matrix with suitable dimensions.

The error bound of the iterative ADP is chosen as $\varepsilon = 10^{-5}$. Neural networks are used to implement the iterative ADP algorithm and the neural-network structure can

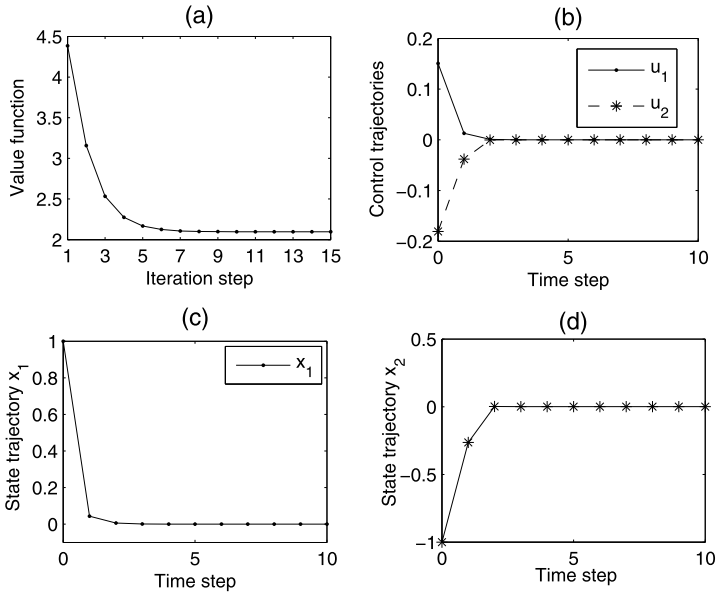

Fig. 2.27 Simulation results for Example 2.51. (**a**) The convergence of value function. (**b**) The ε-optimal control vectors. (**c**) and (**d**) The corresponding state trajectories

be seen in [13, 16]. The critic network and the action network are chosen as three-layer BP neural networks with the structures of 2–8–1 and 2–8–2, respectively. The model network is also chosen as a three-layer BP neural network with the structure of 4–8–2. The critic network is used to approximate the iterative value functions which are expressed by (2.210), (2.212), (2.215), (2.217), (2.222), (2.224), (2.227), and (2.229). The action network is used to approximate the optimal control laws which are expressed by (2.211), (2.213), (2.216), (2.218), (2.223), (2.225), (2.228), and (2.230). The training rules of the neural networks can be seen in [12]. For each iteration step, the critic network and the action network are trained for 1000 iteration steps using the learning rate of $\alpha = 0.05$, so that the neural-network training error becomes less than 10^{-8}. Enough iteration steps should be implemented to guarantee the iterative value functions and the control law to converge sufficiently. We let the algorithm run for 15 iterative steps to obtain the optimal value function and optimal control law. The convergence curve of the value function is shown in Fig. 2.27(a). Then, we apply the optimal control law to the system for $T_f = 10$ time steps and obtain the following results. The ε-optimal control trajectories are shown in Fig. 2.27(b) and the corresponding state curves are shown in Figs. 2.27(c) and (d).

After seven iteration steps, we have $|V_6(x(0)) - V_7(x(0))| \leq 10^{-5} = \varepsilon$. Then, we obtain the optimal number of control steps $K_\varepsilon(x(0)) = 6$. We can see that after six time steps, the state variable becomes $x_6 = [0.912 \times 10^{-6},\ 0.903 \times 10^{-7}]^{\mathrm{T}}$. The entire computation process takes about 10 seconds before satisfactory results are obtained.

Example 2.52 The second example is chosen from [9] with some modifications. We consider the following system:

$$x(k+1) = F(x(k), u(k)) = x(k) + \sin(0.1x(k)^2 + u(k)), \tag{2.232}$$

where $x(k), u(k) \in \mathbb{R}$, and $k = 0, 1, 2, \ldots$. The cost functional is defined as in Example 2.51 with $Q = R = 1$. The initial state is $x(0) = 1.5$. Since $F(0, 0) = 0$, $x(k) = 0$ is an equilibrium state of system (2.232). But since $(\partial F(x(k), u(k))/\partial x(k))(0, 0) = 1$, system (2.232) is marginally stable at $x(k) = 0$ and the equilibrium $x(k) = 0$ is not attractive.

We can see that, for the fixed initial state $x(0)$, there does not exist a control $u(0) \in \mathbb{R}$ that makes $x(1) = F(x(0), u(0)) = 0$. The error bound of the iterative ADP algorithm is chosen as $\varepsilon = 10^{-4}$. The critic network, the action network, and the model network are chosen as three-layer BP neural networks with the structures of 1–3–1, 1–3–1, and 2–4–1, respectively. According to (2.232), the control can be expressed by

$$u(k) = -0.1x(k)^2 + \sin^{-1}(x(k+1) - x(k)) + 2\lambda\pi, \tag{2.233}$$

where $\lambda = 0, \pm 1, \pm 2, \ldots$.

To show the effectiveness of our algorithm, we choose two initial finite-horizon admissible control sequences.

Case 1. The control sequence is $\underline{\hat{u}}_0^1 = (-0.225 - \sin^{-1}(0.7),\ -0.064 - \sin^{-1}(0.8))$ and the corresponding state sequence is $\underline{\hat{x}}_0^2 = (1.5, 0.8, 0)$.

For the initial finite-horizon admissible control sequences in this case, run the iterative ADP algorithm at the states $x(k) = 0.8$ and 1.5, respectively. For each iterative step, the critic network and the action network are trained for 1000 iteration steps using the learning rate of $\alpha = 0.05$ so that the neural-network training accuracy of 10^{-8} is reached. After the algorithm has run for 15 iterative steps, we obtain the performance index function trajectories shown in Figs. 2.28(a) and (b), respectively. The ε-optimal control and state trajectories are shown in Figs. 2.28(c) and (d), respectively, for 10 time steps. We obtain $K_\varepsilon(0.8) = 5$ and $K_\varepsilon(1.5) = 8$.

Case 2. The control sequence is $\underline{\hat{u}}_0^3 = (-0.225 - \sin^{-1}(0.01),\ 2\pi - 2.2201 - \sin^{-1}(0.29),\ -0.144 - \sin^{-1}(0.5),\ -\sin^{-1}(0.7))$ and the corresponding state sequence is $\underline{\hat{x}}_0^4 = (1.5, 1.49, 1.2, 0.7, 0)$.

For the initial finite-horizon admissible control sequence in this case, run the iterative ADP algorithm at the states $x(k) = 0.7$, 1.2, and 1.49, respectively. For each iteration step, the critic network and the action network are also trained for 1000 iteration steps using the learning rate of $\alpha = 0.05$, so that the neural-network training accuracy of 10^{-8} is reached. Then we obtain the value function trajectories shown in Figs. 2.29(a)–(c), respectively. We have $K_\varepsilon(0.7) = 4$, $K_\varepsilon(1.2) = 6$, and $K_\varepsilon(1.49) = 8$.

After 25 iteration steps, the value function $V_i(x(k))$ is sufficiently convergent at $x(k) = 1.49$, with $V_8^3(1.49)$ as the value function. For the state $x(k) = 1.5$, we have $|V_8^3(1.5) - V_9^3(1.5)| = 0.52424 \times 10^{-7} < \varepsilon$. Therefore, the optimal value function

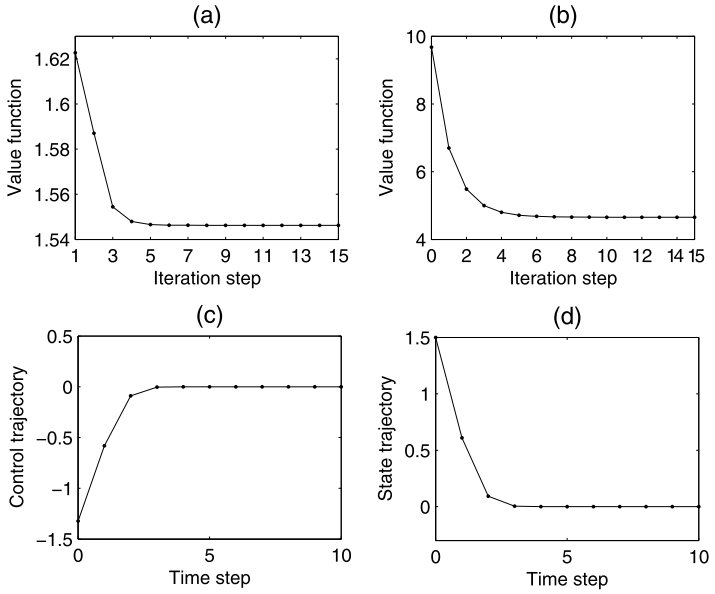

Fig. 2.28 Simulation results for Case 1 of Example 2.52. (**a**) The convergence of value function at $x(k) = 0.8$. (**b**) The convergence of value function at $x(k) = 1.5$. (**c**) The ε-optimal control trajectory. (**d**) The corresponding state trajectory

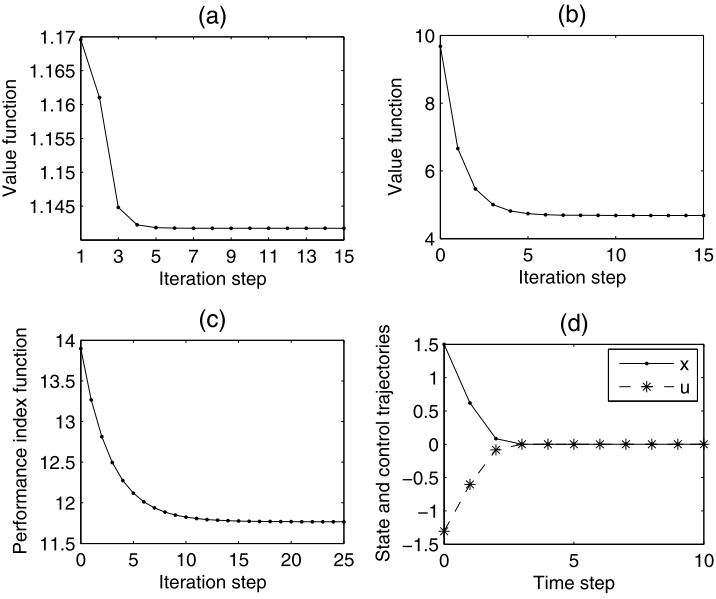

Fig. 2.29 Simulation results for Case 2 of Example 2.52. (**a**) The convergence of value function at $x(k) = 0.7$. (**b**) The convergence of value function at $x(k) = 1.2$. (**c**) The convergence of performance index function at $x(k) = 1.49$. (**d**) The ε-optimal control trajectory and the corresponding state trajectory

at $x(k) = 1.5$ is $V_8^3(1.5)$ and, thus, we have $x(k) = 1.5 \in \mathcal{T}_8^{(\varepsilon)}$ and $K_\varepsilon(1.5) = 8$. The whole computation process takes about 20 seconds and then satisfactory results are obtained.

Then we apply the optimal control law to the system for $T_f = 10$ time steps. The ε-optimal control and state trajectories are shown in Fig. 2.29(d).

We can see that the ε-optimal control trajectory in Fig. 2.29(d) is the same as the one in Fig. 2.28(c). The corresponding state trajectory in Fig. 2.29(d) is the same as the one in Fig. 2.28(d). Therefore, the optimal control law is not dependent on the initial control law. The initial control sequence $\underline{\hat{u}}_0^{K-1}$ can arbitrarily be chosen as long as it is finite-horizon admissible.

Remark 2.53 If the number of control steps of the initial admissible control sequence is larger than the number of control steps of the optimal control sequence, then we will find some of the states in the initial sequence to possess the same number of optimal control steps. For example, in Case 2 of Example 2.52, we see that the two states $x = 1.49$ and $x = 1.5$ possess the same number of optimal control steps, i.e., $K_\varepsilon(1.49) = K_\varepsilon(1.5) = 8$. Thus, we say that the control $u = -0.225 - \sin^{-1}(0.01)$ that makes $x = 1.5$ run to $x = 1.49$ is an unnecessary control step. After the unnecessary control steps are identified and removed, the number of control steps will reduce to the optimal number of control steps and, thus, the initial admissible control sequence does not affect the final optimal control results.

2.6 Summary

In this chapter, several infinite-time and finite-time optimal control schemes have been developed to solve the corresponding control problems for several kinds of nonlinear system. In Sects. 2.2, 2.3, and 2.4, the presented optimal controllers were all infinite-time optimal state feedback controllers though the developed ADP algorithms. The optimal control objective can be achieved when the number of control steps tends to infinity. In Sect. 2.5, an effective iterative ADP algorithm has been developed for the ε-optimal controller for a class of discrete-time nonlinear systems, where the optimal control objective can be achieved with a finite number of control steps.

References

1. Al-Tamimi A, Lewis FL (2007) Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. In: Proceedings of IEEE international symposium on approximate dynamic programming and reinforcement learning, Honolulu, HI, pp 38–43
2. Bagnell J, Kakade S, Ng A, Schneider J (2003) Policy search by dynamic programming. In: Proceedings of 17th annual conference on neural information processing systems, Vancouver, Canada, vol 16, pp 831–838

3. Bryson AE, Ho YC (1975) Applied optimal control: optimization, estimation, and control. Hemisphere–Wiley, New York
4. Burk F (1998) Lebesgue measure and integration. Wiley, New York
5. Chen Z, Jagannathan S (2008) Generalized Hamilton–Jacobi–Bellman formulation-based neural network control of affine nonlinear discrete-time systems. IEEE Trans Neural Netw 19(1):90–106
6. Cui LL, Zhang HG, Liu D, Kim YS (2009) Constrained optimal control of affine nonlinear discrete-time systems using GHJB method. In: Proceedings of IEEE international symposium on adaptive dynamic programming and reinforcement learning, Nashville, USA, pp 16–21
7. Han D, Balakrishnan SN (2002) State-constrained agile missile control with adaptive-critic-based neural networks. IEEE Trans Control Syst Technol 10(4):481–489
8. Haykin S (1999) Neural networks: a comprehensive foundation. Prentice Hall, Upper Saddle River
9. Jin N, Liu D, Huang T, Pang Z (2007) Discrete-time adaptive dynamic programming using wavelet basis function neural networks. In: Proceedings of the IEEE symposium on approximate dynamic programming and reinforcement learning, Honolulu, HI, pp 135–142
10. Liu D, Wang D, Zhao D, Wei Q, Jin N (2012) Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming. IEEE Trans Autom Sci Eng 9(3):628–634
11. Plumer ES (1996) Optimal control of terminal processes using neural networks. IEEE Trans Neural Netw 7(2):408–418
12. Si J, Wang YT (2001) On-line learning control by association and reinforcement. IEEE Trans Neural Netw 12(2):264–276
13. Wang FY, Zhang HG, Liu D (2009) Adaptive dynamic programming: an introduction. IEEE Comput Intell Mag 4(2):39–47
14. Wang FY, Jin N, Liu D, Wei Q (2011) Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with ε-error bound. IEEE Trans Neural Netw 22(1):24–36
15. Zhang HG, Xie X (2011) Relaxed stability conditions for continuous-time T-S fuzzy-control systems via augmented multi-indexed matrix approach. IEEE Trans Fuzzy Syst 19(3):478–492
16. Zhang HG, Wei QL, Luo YH (2008) A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm. IEEE Trans Syst Man Cybern, Part B, Cybern 38(4):937–942
17. Zhang HG, Luo YH, Liu D (2009) Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints. IEEE Trans Neural Netw 20(9):1490–1503

http://www.springer.com/978-1-4471-4756-5

Adaptive Dynamic Programming for Control
Algorithms and Stability
Zhang, H.; Liu, D.; Luo, Y.; Wang, D.
2013, XVI, 424 p., Hardcover
ISBN: 978-1-4471-4756-5