Fuzzy Least Square Policy Iteration and Its Mathematical Analysis

Farzaneh Ghorbani, Vali Derhami & Mohsen Afsharchi

International Journal of Fuzzy Systems

ISSN 1562-2479

Int. J. Fuzzy Syst.
DOI 10.1007/s40815-016-0270-1

Springer

Int. J. Fuzzy Syst.
DOI 10.1007/s40815-016-0270-1

Fuzzy Least Square Policy Iteration and Its Mathematical Analysis

Farzaneh Ghorbani[1] · **Vali Derhami**[2] · **Mohsen Afsharchi**[1]

Received: 28 March 2015 / Revised: 20 September 2016 / Accepted: 18 October 2016

Abstract In this paper we present a novel approach to reinforcement learning for continuous state–action control problems. This approach is obtained by combining least square policy iteration (LSPI) with zero-order Takagi–Sugeno fuzzy system, which we call it, "fuzzy least square policy iteration (FLSPI)." FLSPI is a critic-only method and has advantages of both LSPI and fuzzy systems together. We define state–action basis functions based on a fuzzy system while LSPI theorem conditions are satisfied. Our aim is to find the most suitable continuous action in every state using fuzzy rules. This method is learning rate independent and has positive mathematical analysis that defines an error bound for it. We show by simulation that the learning is faster and the quality of results is better than the two previous fuzzy reinforcement learning approaches based on critic-only architecture, i.e., fuzzy Q-learning (FQL) and Fuzzy SARSA Learning (FSL). We test FLSPI on four well-known problems (i.e., boat problem, maze, inverted pendulum and cart–pole balancing) and show the FLSPI higher performance, function of its error bound, its convergence against FQL and FSL divergence and its excellence against the latest proposed methods, respectively.

✉ Mohsen Afsharchi
afsharchi@znu.ac.ir

Farzaneh Ghorbani
f.ghorbani@znu.ac.ir

Vali Derhami
vderhami@yazd.ac.ir

[1] University of Zanjan, Zanjan, Iran

[2] Yazd University, Yazd, Iran

Keywords Continuous state-action · Fuzzy systems · Least square policy iteration · Reinforcement learning · State-action function approximation

1 Introduction

Reinforcement learning (RL) is an algorithmic method for solving problems in which actions (decisions) are applied to a system over an extended period of time, in order to achieve a desired goal. The time variable is usually discrete and actions are taken at every discrete time step, leading to a sequential decision-making problem [4]. Most of the problems that RL tries to solve and deal with continuous state and action spaces but for large and continuous space problems standard RL methods can no longer be applied in their original form. Instead, approximate versions of solutions are introduced. Theoretical guarantees are provided on the performance of the algorithms, and numerical examples are used to illustrate their behavior. These methods have two major advantages which enable them to be applied to some successful applications [18, 21]. Firstly they do not need to keep tabular information about state–action value function, and hence, they do not need too much memory and second, they do not need exact information [16]. It is also necessary to notice that in discrete RL, state (or state–action) values are independent while in continuous RL, these values are approximated in every time step based on the approximator parameters. So parameter updating in every time step affects state (or state–action) values in the whole space.

Fuzzy systems are among efficient approximators [7]. Fuzzy reinforcement learning (FRL) methods have been proposed based on fuzzy systems to solve RL challenges in continuous spaces [2, 5, 9]. FRL methods often use two

Published online: 14 November 2016

well-known architectures: actor-critic and actor-only [9, 12]. Actor-critic architecture has two independent parts: actor and critic [22]. In the actor-critic method, actor produces output action and critic approximates value function. In contrast, actor-only methods have only one part, critic. Critic is used to approximate value function. In critic-only methods, final action is produced just by approximated values. FRL methods usually face two very important challenges. First, they either have no mathematical analysis [2] or their mathematical analysis are proposed for discrete spaces only [5, 9]. Second they are mostly dependent on learning rate parameter [2, 9] which is problem-specific.

In order to tackle these challenges, we use a combination of least square policy iteration (LSPI) method [5, 6, 16, 19, 23] and fuzzy system. LSPI is an approximate policy iteration (PI) method which has two phases: policy evaluation that evaluates the current policy by computing its approximate value function in every iteration and policy improvement that finds a new, improved policy using this value function. LSPI is an iterative algorithm that uses least squares techniques for policy evaluation phase. Least squares techniques have relaxed convergence requirements and approach their solution quickly as the number of samples increases. LSPI has positive mathematical analysis and learning rate independency and also high performance.

Most versions of LSPI employ discrete actions. Some methods apply LSPI on continuous action space and proposed the mathematical analysis for discrete action space [5, 6]. However, there exist important classes of control problems in which continuous actions are required. For instance, when a system must be stabilized around an unstable equilibrium, any discrete action policy will lead to undesirable chattering of the control action. A comprehensive review on LSPI is presented in [4].

In this paper we use fuzzy system as our general approximator to extend LSPI with continuous action space. Our method, which we call it fuzzy least square policy iteration (FLSPI), approximates state–action value function in LSPI. In another view, we use LSPI to adjust consequents of rules in the fuzzy system. The proposed Fuzzy LSPI is a learning rate-independent method, has positive mathematical analysis for continuous state–action spaces and convergence faster than other FRL methods. It also can be applied to both off-line and on-line LSPI (we use both form in our experiments). A very incomplete version of this work in persian language is appeared in [11].

The organization of this paper is as follows: In Sect. 2, we introduce the basic concepts and background knowledge for our method. In Sect. 3, we propose FLSPI. In Sect. 4, we present the theoretical analysis of FLSPI. Section 5 presents the simulation results, followed by Sect. 6.

2 Preliminary Concepts

In this section, we introduce tree main concepts that our work is based on.

2.1 Reinforcement Learning

Reinforcement learning is a type of learning in which agent learns something via trial and error [22]. In fact, agent is in contact with the environment by receiving two types of signals: The first one, *state* indicates in which state of the world agent is, and the second, *reward* shows the immediate desirability of each state. However, the agent's goal is to maximize its long-term utility rather than the immediate one. By taking actions, agent influences the environment and changes its state. Equation 1, which is called Bellman equation [1], is the basis of RL, where $V(s)$ is the value of state s (which shows the long-term usefulness of s), $R(s)$ is the immediate reward of state s, S and A are the set of states and actions, respectively, $\gamma(0 \leq \gamma < 1)$ is the discount factor, and $P_{sa}(s\prime)$ is the transition model which is the probability of reaching state s' after taking action a in state s.

$$\forall s \in S: \quad V(s) = R(s) + \gamma \max_{a \in A} \sum_{s' \in S} \mathcal{P}(s,a,s')V(s') \tag{1}$$

2.2 Least Square Policy Iteration (LSPI)

Policy iteration (PI) [13] is one of the RL methods which discover optimum policy for every MDP by producing a sequence of policies. PI is an iterative algorithm which has two phases in every iteration: Policy evaluation computes Q^{π_m} for current policy π_m:

$$Q^{\pi_m}(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s,a,s') \sum_{a' \in A} \pi(s',a') Q^{\pi_m}(s',a') \tag{2}$$

and policy improvement defines improved greedy policy π_{m+1} on Q^{π_m}:

$$\pi_{m+1}(s) = \text{argmax}_{a \in A} Q^{\pi_m}(s,a) \tag{3}$$

This approach has high performance in finite state and action space and also it has mathematical analysis [3]. We can compute exact value of Q^{π_m} by solving this equation:

$$T_\pi Q^\pi = Q^\pi \tag{4}$$

where T_π is the Bellman operator under policy π:

$$(T_\pi Q)(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s,a,s') \sum_{a' \in A} \pi(s',a') Q(s',a') \tag{5}$$

This approach has curse of dimensionality problem as the other RL approaches.

Least square policy iteration (LSPI) [16] is modified to use approximate policies and sample-based approximate policy improvements. This approach approximates state–action function instead of computing its exact value. LSPI is based on adapting the state–action function approximation (i.e., $\widehat{Q}$) with its image under Bellman operator.

Generally, the state–action function approximation can be defined as follows:

$$\widehat{Q}^{\pi}=\mathbf{\Phi W} \tag{6}$$

where W is weight matrix and $\mathbf{\Phi}$ is basis function matrix:

$$\mathbf{\Phi}=\begin{pmatrix} \phi(s_1,a_1)^T \\ \cdots \\ \phi(s_i,a_i)^T \\ \cdots \\ \phi\left(s_{|S|},a_{|A|}\right)^T \end{pmatrix} \tag{7}$$

$$\phi(s,a)=\begin{pmatrix} \phi_1(s,a) \\ \cdots \\ \phi_k(s,a) \end{pmatrix} \tag{8}$$

where $k \ll |S||A|$, and ϕ_j are basis functions. In LSPI, $\widehat{Q}^{\pi}$ is computed as follows:

$$\widehat{Q}^{\pi}=\mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}T_{\pi}\hat{Q}^{\pi} \tag{9}$$

By simplifying the previous equation (after expanded calculation that this paper is not suitable to explain them), one can obtain following results [16]:

$$AW = b \tag{10}$$

$$A = A + \phi(s,a)(\phi(s,a) - \gamma\phi(s',\pi(s')))^T \tag{11}$$

$$b = b + \phi(s,a)\mathcal{R} \tag{12}$$

where $\mathcal{R}$ is the reward of transition from state s to state s', while the action a is selected. γ is learning rate and matrix A and vector b are used to calculate weight matrix W and are computed iteratively.

2.3 The Takagi–Sugeno Fuzzy System

Generally, a fuzzy inference system (FIS) is a system with multiple inputs (associated with linguistic terms) and one or more output(s). FIS is based on if–then rules. Mamdani and Takagi–Sugeno are the major types of FIS . A sample rule for an n-input–m-output Mamdani FIS looks like as follows:

R : If x_1 is L_1 and. . . and If x_n is L_n Then
y_1 is $K_1, \ldots,$ y_m is K_m

where $x_i, i = 1, \ldots, n$ is input to fuzzy system, y_j, $j = 1, \ldots, m$ is the output of the system, and L_i, $i = 1, \ldots, n$ and K_j, $j = 1, \ldots, m$ are the linguistic terms. Takagi–Sugeno FIS is similar to Mamdani FIS with this difference that the output of Takagi–Sugeno FIS is crisp. A rule in Takagi–Sugeno FIS looks like as follows:

R : If x_1 is L_1 and. . . and if x_n is L_n then
$y_1 = f_1(x_1, \ldots, x_n) \ldots y_m = f_m(x_1, \ldots, x_n)$ is K_m

where f_j ,$j = 1, \ldots, m$ is a map from the input space into the output space.

3 Fuzzy Least Square Policy Iteration

RL methods are applicable and powerful, but they are weak in large and continuous space problems, while most of the real-world problems deal with continuous spaces. To overcome this weakness, many methods have been proposed, but they have shortcomings either in the continuous action spaces or in mathematical analysis foundations. In this paper, we try to tackle this weakness and in this direction, we gain some other advantages as we will explain later on.

LSPI is a flexible and powerful RL method with many advantages such as learning rate independence and fast convergence, but LSPI could not work on continuous action space. We take the advantages of LSPI and combine it with fuzzy system to solve RL continuity problems in state and action space. In this section, we explain our new FRL method which we call it, FLSPI.

Figure 1 shows the block diagram of FLSPI and the dependencies among its various components. At first, the state–action value function is approximated using the basis functions that are defined with the fuzzy system, and also the weight vector is obtained using LSPI. One of the advantages of approximation is computing policy on demand instead of storing it physically in a table [16]. In the next step, rule consequence weight parameters are adjusted using LSPI. Policy improvement, in any state s, is done by selecting action a that has maximum value of $\hat{Q}(s,a)$. In our method, based on our different definition of basis functions and $\hat{Q}$ formula, this problem will be changed into finding action with the maximum weight in every rule. The policy evaluation can be done in the same way as LSPI. This component computes approximated state–action value function using improved policy.

FLSPI can be viewed from two perspectives. From one, FLSPI uses fuzzy system to extend LSPI to apply to the problems with continuous action space. We use zero-order Takagi–Sugeno fuzzy system and define suitable basis functions to achieve our goal. These basis functions satisfy LSPI conditions. Basically, we partition problem state space and then we select m actions from action space while

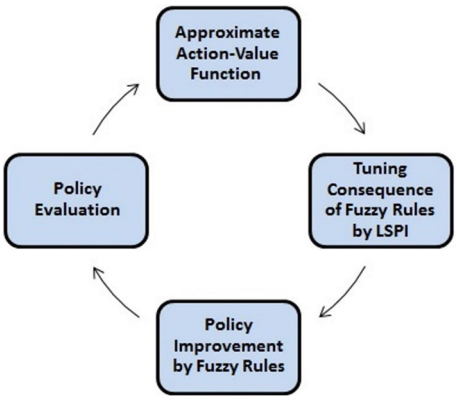

Fig. 1 Block diagram describing fuzzy least square policy iteration

paying attention to the problem. We define R rules as follows:

$$R_i: \textit{If } x_1 \textit{ is } L_{i1} \textit{ and } \ldots \textit{ and If } x_n \textit{ is } L_{in} \textit{ Then } (o_{i1} \textit{ with weight } w_{i1} \textit{ or } \ldots \textit{ or } o_{im} \textit{ with weight } w_{im}) \quad (13)$$

where $s = (x_1, \ldots, x_n)$ is the vector of n-dimensional vector space and $L_i = L_{i1} \times \ldots \times L_{in}$ is the n-dimensional strictly convex and normal fuzzy set of the ith rule with a unique center, m is the number of possible discrete actions for each rule, o_{ij} is the jth candidate action, and weight w_{ij} is the approximated value of the jth action in the ith rule. In ith rule, action o_{ii^+} (where i^+ is the index of the selected action) will be selected using $\varepsilon - greedy$ action selection as follows:

$$a_t(s_t) = \sum_{i=1}^{R} \mu_i(s_t) o_{ii^+} \quad (14)$$

where $\mu_i(s)$ is normalized firing strength of ith rule for state s.

The firing strength of each rule is obtained by the product of antecedents of fuzzy sets. Basis functions are defined by normalized firing strength functions of rules:

$$\phi(s,a) = \left[\overbrace{0 \ldots \mu_1(s) \ldots 0}^{m} \overbrace{0 \ldots \mu_2(s) \ldots 0}^{m} \ldots \overbrace{0 \ldots \mu_R(s) \ldots 0}^{m} \right]^T \quad (15)$$

so, this method produces continuous action values.

From another perspective, FLSPI uses LSPI to adjust the consequences of the defined rules in 13. FLSPI adjusts the values w_{ij} that are used to obtain the best policy. So this system receives a state from the continuous state space as an input and gives an action from the continuous action space as an output. The details of basis functions (i.e., number and the value of the actions that are used as rules consequent) are problem dependent and should be determined by user based on his/her experiences. The off-line FLSPI algorithm procedure is summarized in Algorithm 1.

A and b are defined in previous relations 12, 14, 15 and calculate weight vector W. Updating A and b is done by relation 16 and relation 17 in every time step, and updating of weight vector W is done by relation 18 at the end of every episode.

We can use on-line FLSPI for non-episodic problems or problems that need to make decision in every state based on past episodes observations. The on-line FLSPI algorithm procedure is summarized in Algorithm 2. In on-line FLSPI algorithm, weight vector is updated in every fix number of time steps. It is necessary to remember that updating weight vector actually is updating $\widehat{Q}$, which leads to the new policy.

We illustrate our on-line algorithm with a simple example. In this example, we consider a simple problem to avoid unnecessary complex calculations. Consider a maze problem without any obstacles. State space is a two-dimensional continuous space with size 10 in every dimension. We start from point (1,1) and the goal is a circle in (9,9) with a radius of 0.5. We partitioned both dimensions of state space into two equivalent partitions (i.e., low and high) and defined consequences of rules by a set of tree angles -180,0,180 to reduce matrix dimensions. Step size is equal to 1. The reward function is defined based on the distance between the agent and the goal and equals to the negative value of this distance. The fuzzy rules in this problem are as follows:

R_1 : *if x is low and y is low, then* o_1 *with weight* w_{11} or o_2 *with weight* w_{21} or o_3 *with weight* w_{31}
R_2 : *if x is low and y is high, then* o_1 *with weight* w_{12} or o_2 *with weight* w_{22} or o_3 *with weight* w_{32}
R_3 : *if x is high and y is low, then* o_1 *with weight* w_{13} or o_2 *with weight* w_{23} or o_3 *with weight* w_{33}
R_4 : *if x is high and y is high Then* o_1 *with weight* w_{14} or o_2 *with weight* w_{24} or o_3 *with weight* w_{34}

Learning should be done in several iterations while every iteration has several episodes, but the execution of the algorithm is very time-consuming. To make it brief, we will only present the first tree episodes of the first iteration. We update weight vector in every episode. We show agent state with X_i , weight vector with $W_i = [w_{11}, w_{21}, w_{31}, w_{21}, w_{22}, w_{23}, w_{31}, w_{32}, w_{33}, w_{41}, w_{42}, w_{43}]$. selected angle with $angle_i$, selected actions vector with $O_i = [o_{11^+}, o_{22^+}, o_{33^+}, o_{44^+}]$, normalized firing strengths vector with $M_i = [\mu_1, \mu_2, \mu_3, \mu_4]$ and reward with R_i in ith episode.

Algorithm 1 Off-line Fuzzy Least Square Policy Iteration

Require: p : Number of partitions (a vector with dimension of state space),R : Reward function, γ : Discount factor, π_0 : Initial policy, Initial weight vector W_1, $t = 1$, $k = 1$
Ensure: π : policy (w : weight vector)
repeat
  Observe initial state s_0
  Select a suitable action of each rule considering their weights and ε-greedy action selection
  repeat
    $a_t(s_t) = \sum_{i=1}^{R} \mu_i(s_t) o_{ii^+}$
    Apply a_t, observe s_{t+1} and receive reward r_{t+1}.

$$A_{t+1} = A_t + \phi(s_t, a_t)(\phi(s_t, a_t) - \gamma\phi(s_{t+1}, \pi(s_{t+1})))^T \tag{16}$$

$$b_{t+1} = b_t + \phi(s_t, a_t) r_{t+1} \tag{17}$$

    $t \leftarrow t + 1$
  until The end of the episode
  Solve

$$\frac{1}{k-1} A_t W_k = \frac{1}{k-1} b_t \tag{18}$$

  $t \leftarrow 1$
  $k \leftarrow k + 1$
until Adapt condition is meet.

Algorithm 2 On-line Fuzzy Least Square Policy Iteration

Require: p : Number of partitions (a vector with dimension of state space),R : Reward function, γ : Discount factor, π_0 : Initial policy, Initial weight vector W_1, $t = 1$, $k =$a fix number
Ensure: π : policy (w : weight vector)
repeat
  Observe initial state s_0
  Select a suitable action of each rule considering their weights and ε-greedy action selection
  repeat
    $a_t(s_t) = \sum_{i=1}^{R} \mu_i(s_t) o_{ii^+}$
    Apply a_t, observe s_{t+1} and receive reward r_{t+1}.

$$A_{t+1} = A_t + \phi(s_t, a_t)(\phi(s_t, a_t) - \gamma\phi(s_{t+1}, \pi(s_{t+1})))^T$$

$$b_{t+1} = b_t + \phi(s_t, a_t) r_{t+1}$$

    if $t = ck$ for some c
    Solve

$$\frac{1}{c-1} A_t W_c = \frac{1}{c-1} b_t$$

    end if
    $t \leftarrow t + 1$
  until The end of the episode
  $t \leftarrow 1$
until Adapt condition is meet.

We start from $X_0 = [1, 1]$,$W_1 = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$ and because of weight value equivalency, we add small random term to wights. After one episode, we have: $M_1 = [0.81, 0.09, 0.09, 0.01]$, $O_1 = [0, 180, 0, 0]$, $angle_1 = 16.2$, $R_1 = -12.27$, $X_1 = [0.12, 0.53]$, $W_2 = [0, 190.17, 0, 0, 0, 21.13, 0, 21.13, 0, 0, 2.35, 0]^T$. So after episode two: $M_2 = [0.94, 0.05, 0.01, 0]$, $O_2 = [0, 180, 0, 0]$, $angle_2 = 9.39$, $R_2 = -11.50$, $X_2 = [0.26, 1.52]$, $W_3 = [0, -187.90, 0, 0, 0, -138.99, 0, -232.11, 0, 0, -27.21, 0]^T$ and after episode tree, we have: $M_3 = [0.83, 0.15, 0.02, 0]$, $O_3 = [180, -180, -180, -180]$, $angle_3 = 117.39$, $R_3 = -10.54$, $X_3 = [0.84, 2.33]$, $W_4 = [0, -187.90, -70.64, -12.64, 0, -138.99, -1.90, -232.11, 0, -0.34, -27.21, 0]^T$. After 106 episodes, the agent reaches the goal. The parameters in this problem are not optimal and are chosen to decrease the example complexity.

4 Theoretical Analysis of FLSPI

In this section, we provide some theoretical results concerning the error bound for FLSPI. We use our definitions of FLSPI from section 2.2.

First, let us mention Stone–Weierstrass theorem [14] which is essential in our analysis.

Theorem 1 (Stone—Weierstrass theorem) *Let Z be a set of continuous function on the convex space X such that*

1. *Z is an algebra, i.e., Z is closed under sum, product and scalar product.*
2. *Z separates the points of X, i.e.,*

$$\forall x_1, x_2 \in X,\ x_1 \neq x_2;;\ \exists F \in Z\ s.t\ F(x_1) \neq F(x_2)$$

3. *Z is not zero in any point of X, i.e.,*

$$\forall x \in X;;\ \exists F \in Z\ s.t\ F(x) \neq 0$$

then for every continuous function $G(x)$ on X and every arbitrary $\varepsilon > 0$, there exists a function $F \in Z$ such that

$$\|F(x) - G(x)\|_\infty < \varepsilon.$$

We apply Theorem 1 to $\widehat{Q}$. Lemma 1 shows that defined fuzzy system in our work is a general approximator for the set of all continuous functions. Previously, it is proved in Ref. [24] *that fuzzy system is a general approximator, but its fuzzy system is different from our fuzzy system.*

Lemma 1 *For any continuous function Q and every arbitrary $\varepsilon > 0$, there exists function $\widehat{Q} = \sum_{i=1}^{R} \mu_i(s) w_{ii^+}$ such that*

$$\|\widehat{Q} - Q\|_\infty < \varepsilon.$$

Proof Let X be the space of the Cartesian product of the state space (the convex subspace of $\mathbb{R}^n$) and the action space (the convex subspace of $\mathbb{R}^m$). Then X is a convex subspace of $\mathbb{R}^{n+m}$, where n and m are dimensions of state and action space, respectively. We define Z as the space of all functions $\widehat{Q} = \mathbf{\Phi} W$, where W is vector and Φ is determined by relation 7.

$$\widehat{Q} : X \to Y \ , \ y = \widehat{Q}(s,a) \ , X \subseteq \mathbb{R}^{n+m} \ , Y \subseteq \mathbb{R}$$

Consider R rules as we defined before in relation 13:

R_i : *If* x_1 is L_{i1} and \ldots and if x_n is L_{in} then (o_{i1} with weight w_{i1} or … or o_{im} with weight w_{im})

and consider y as follows:

$$y = \widehat{Q}(s,a) = \sum_{i=1}^{R} \mu_i(s) w_{ii^+} \tag{19}$$

where w_{ii^+} is corresponding weight to selected action of ith rule and $\mu_i(s)$ is normalized firing strength of ith rule in order to input s:

$$\mu_i(s) = \frac{\alpha_i(s)}{\sum_{j=1}^{R} \alpha_j(s)} \tag{20}$$

where $\alpha_i(s)$ is firing strength of ith rule in order to input s:

$$\alpha_i(s) = \prod_{l=1}^{n} \mu_i^l(s_l) \tag{21}$$

$\mu_i^l(s_l)$ is the degree of membership for s_l in lth membership function. So we have:

$$\widehat{Q}(s,a) = \frac{\sum_{i=1}^{R} w_{ii^+} \prod_{l=1}^{n} \mu_i^l(s_l)}{\sum_{i=1}^{R} \prod_{l=1}^{n} \mu_i^l(s_l)} \tag{22}$$

Now consider the following equations:

$$\left(\sum_i p_i b_i\right)\left(\sum_j c_j\right) + \left(\sum_j q_j c_j\right)\left(\sum_i b_i\right) = \sum_i \sum_j (p_i + q_j)(b_i c_j) \tag{23}$$

$$\left(\sum_i a_i\right)\left(\sum_j b_j\right) = \sum_i \sum_j a_i b_j \tag{24}$$

We define functions $\widehat{Q}_1$ with R_1 rules and $\widehat{Q}_2$ with R_2 rules in Z,

$$\widehat{Q}_1(s,a) = \frac{\sum_{i=1}^{R_1} W_{ii^+} \prod_{l=1}^{n} \mu_i^l(s_l)}{\sum_{i=1}^{R_1} \prod_{l=1}^{n} \mu_i^l(s_l)}$$

$$\widehat{Q}_2(s,a) = \frac{\sum_{j=1}^{R_2} W_{jj^+} \prod_{l=1}^{n} \mu_j^l(s_l)}{\sum_{j=1}^{R_2} \prod_{l=1}^{n} \mu_j^l(s_l)}$$

By Equation 23 we have :

$$\widehat{Q}_1(s,a) + \widehat{Q}_2(s,a) = \frac{\sum_{i=1}^{R_1} \sum_{j=1}^{R_2} (w_{ii^+} + w_{jj^+}) \left(\prod_{l=1}^{n} \mu_i^l(s_l) \mu_j^l(s_l)\right)}{\sum_{i=1}^{R_1} \sum_{j=1}^{R_2} \left(\prod_{l=1}^{n} \mu_i^l(s_l) \mu_j^l(s_l)\right)}$$

$$\Rightarrow \widehat{Q}_1(s,a) + \widehat{Q}_2(s,a) \in Z$$

So Z is closed under summation. Now by Equation 24, we have:

$$\widehat{Q}_1(s,a).\widehat{Q}_2(s,a) = \frac{\sum_{i=1}^{R_1} \sum_{j=1}^{R_2} (w_{ii^+}.w_{jj^+}) \left(\prod_{l=1}^{n} \mu_i^l(s_l) \mu_j^l(s_l)\right)}{\sum_{i=1}^{R_1} \sum_{j=1}^{R_2} \left(\prod_{l=1}^{n} \mu_i^l(s_l) \mu_j^l(s_l)\right)}$$

This means that Z is closed under multiplication. Clearly Z is closed under scalar multiplication. So Z is an algebra and the first condition of Theorem 1 is satisfied.

Now, let $v = (s_1, a_1) \in X$ and $h = (s_2, a_2) \in X$ be two arbitrary vectors such that $v \neq h$. So $s_1 \neq s_2$ or ($a_1 \neq a_2$ and $s_1 = s_1$). Consider the fuzzy system with two rules that has two membership functions as follows:

$$\mu_1^l(s_l) = \exp\left(-\frac{1}{2}(s_l - s_{1l})^2\right)$$

$$\mu_2^l(s_l) = \left(-\frac{1}{2}(s_l - s_{2l})^2\right)$$

Let consequents of rules are o_{11} and o_{21}:

$$o_{11} = a_1/2, o_{12} = (a_2/2)\exp\left(\frac{1}{2}\|v - h\|^2\right)$$

$$o_{21} = (a_1/2)\exp\left(\frac{1}{2}\|v - h\|^2\right), o_{22} = a_2/2$$

Assume o_{11} and o_{21} are selected indexes for a_1, and o_{12} and o_{22} are selected indexes for a_2. In addition, if $s_1 \neq s_2$, define the weights:

$$w_{11} = w_{12} = 0, \ w_{21} = w_{22} = 1$$

and if $a_1 \neq a_2$ and $s_1 = s_1$, define the weights:

$$w_{11} = w_{21} = \frac{1}{2}, \ w_{12} = w_{22} = 1$$

then:

$$\alpha_1(s) = \prod_{l=1}^{n} \mu_1^l(s_l) = \exp\left(-\frac{1}{2}\|s - s_1\|_2^2\right)$$

$$\alpha_2(s) = \prod_{l=1}^{n} \mu_2^l(s_l) = \exp\left(-\frac{1}{2}\|s - s_2\|_2^2\right).$$

So, if $s_1 \neq s_2$ we have:

$$\widehat{Q}(s_1,a_1)=\frac{\exp\left(-\frac{1}{2}\|s_1-s_2\|_2^2\right)}{1+\exp\left(-\frac{1}{2}\|s_1-s_2\|_2^2\right)}$$
$$\widehat{Q}(s_2,a_2)=\frac{1}{1+\exp\left(-\frac{1}{2}\|s_1-s_2\|_2^2\right)}.$$

and if $a_1 \neq a_2$ and $s_1 = s_1$, we have:

$$\widehat{Q}(s_1,a_1)=\frac{1}{2},\widehat{Q}(s_2,a_2)=1$$
$$\Rightarrow \widehat{Q}(v)\neq\widehat{Q}(h)$$

So Z separates the points of X and second condition of Theorem 1 is satisfied.

Now let $w_{ij}=c\neq 0, i,j=1,2$. By previous definition, we have:

$$\widehat{Q}(x)=c\neq 0$$

for all $x\in X$. So Z is not zero in any point of X and this satisfies the third condition of Theorem 1 which completes the proof. □

In the following, we prove some lemmas to define error bound in Theorem 2.

Lemma 2 *Let π be a stationary policy and x be an arbitrary scalar. Then we have:*

$$T(Q+xe)(s,a)=TQ(s,a)+\gamma x \tag{25}$$

$$T_\pi(Q+xe)(s,a)=T_\pi Q(s,a)+\gamma x \tag{26}$$

where is identity.

Proof Since this is a discount problem with discount rate γ, then proof will be completed by Lemma 4.3 of Ref. [3]. □

Lemma 3 *Let π be a stationary policy. Then we have*:

$$\|TQ-T\widehat{Q}\|_\infty\leq\gamma\|Q-\widehat{Q}\|_\infty \tag{27}$$

$$\|T_\pi Q-T_\pi\widehat{Q}\|_\infty\leq\gamma\|Q-\widehat{Q}\|_\infty \tag{28}$$

Proof Since this is a discount problem with discount rate γ, then proof will be completed by Lemma 4.4 of Ref. [3]. □

Lemma 4 *Let π_k be kth policy introduced by kth iteration of FLSPI algorithm and $\varepsilon_k>0$ be an arbitrary real number such that*:

$$\|\widehat{Q}^{\pi_k}-Q^{\pi_k}\|_\infty\leq\varepsilon_k.$$

Then we have:

$$Q^{\pi_{k+1}}(s,a)\leq Q^{\pi_k}(s,a)+\frac{2\gamma\varepsilon_k}{1-\gamma}\quad,\forall(s,a). \tag{29}$$

Proof We define:

$$e_k=\sup_{(s,a)}(\,Q^{\pi_{k+1}}(s,a)-Q^{\pi_k}(s,a)).$$

So we have:

$$Q^{\pi_{k+1}}(s,a)=Q^{\pi_k}(s,a)+e_k\ ,\forall s,a.$$

But $T_\pi Q^\pi=Q^\pi$. By relation 26, we have:

$$Q^{\pi_{k+1}}(s,a)=T_{\pi_{k+1}}Q^{\pi_{k+1}}(s,a)$$
$$\leq T_{\pi_{k+1}}(Q^{\pi_k}(s,a)+e_k)=T_{\pi_{k+1}}Q^{\pi_k}(s,a)+\gamma e_k.$$

In addition, all errors of actor are zero in LSPI because LSPI does not need to represent policy approximation [16]. So we have: $T_{\pi_{k+1}}Q^{\pi_k}=T_{\pi_k}Q^{\pi_k}$, for every k. By relation 28, we can get:

$$Q^{\pi_{k+1}}(s,a)-Q^{\pi_k}(s,a)\leq T_{\pi_{k+1}}Q^{\pi_k}(s,a)+\gamma e_k$$
$$-Q^{\pi_k}(s,a)=T_{\pi_{k+1}}Q^{\pi_k}(s,a)-T_{\pi_{k+1}}\widehat{Q}^{\pi_k}(s,a)$$
$$+T_{\pi_{k+1}}\hat{Q}^{\pi_k}(s,a)-Q^{\pi_k}(s,a)+\gamma e_k\leq\gamma|\widehat{Q}^{\pi_k}(s,a)$$
$$-Q^{\pi_k}(s,a)|+\gamma|\widehat{Q}^{\pi_k}(s,a)-Q^{\pi_k}(s,a)|+\gamma e_k$$
$$=2\gamma\varepsilon_k+\gamma e_k.$$

So we have:

$$\sup_{(s,a)}(\,Q^{\pi_{k+1}}(s,a)-Q^{\pi_k}(s,a))\leq 2\gamma\varepsilon+\gamma e_k$$
$$\Rightarrow e_k\leq 2\gamma\varepsilon_k+\gamma e_k$$
$$\Rightarrow e_k\leq\frac{2\gamma}{1-\gamma}\varepsilon_k.$$

□

Lemma 5 *Let $\pi_{\mathbf{k}}$ be the kth policy introduced by the kth iteration of FLSPI algorithm and $\varepsilon_k\geq 0$ be an arbitrary real number such that*:

$$\|\widehat{Q}^{\pi_k}-Q^{\pi_k}\|_\infty\leq\varepsilon_k,$$

In addition assume:

$$f_k=\sup_{(s,a)}(\mathbf{Q}^{\pi_{\mathbf{k}}(s,a)-\mathbf{Q}^*(\mathbf{s},\mathbf{a})}), \tag{30}$$

Then we have:

$$f_{k+1}\leq\gamma f_k+\gamma e_k+2\gamma\varepsilon_k.$$

Proof By assumption, we have:

$$Q^{\pi_k}(s,a)\leq Q^*(s,a)+f_k,\qquad\forall(s,a).$$

But T is a non-descending operator [3] and also $TQ^*=Q^*$. So we have by relation 21:

$$TQ^{\pi_k}(s,a)\leq$$
$$T(Q^*(s,a)+f_k)=TQ^*(s,a)+\gamma f_k=Q^*(s,a)+\gamma f_k$$

Now by using relation 26 and this fact that if $|a-b|<\varepsilon$ then $a<b+\varepsilon$ and $b<a+\varepsilon$, we have:

$$
\begin{aligned}
T_{\pi_{k+1}}Q^{\pi_k}(s,a) &\le T_{\pi_{k+1}}(\widehat{Q}^{\pi_k}(s,a)+\varepsilon_k)\\
&= T_{\pi_{k+1}}\hat{Q}^{\pi_k}(s,a)+\gamma\varepsilon_k\\
&= T\widehat{Q}^{\pi_k}(s,a)+\gamma\varepsilon_k\\
&\le T(Q^{\pi_k}(s,a)+\varepsilon_k)+\gamma\varepsilon_k\\
&= TQ^{\pi_k}(s,a)+\gamma\varepsilon_k+\gamma\varepsilon_k\\
&\le Q^*(s,a)+\gamma f_k+2\gamma\varepsilon_k
\end{aligned}
$$

□

Theorem 2 *Let π_k kth policy be obtained from kth iteration of FLSPI algorithm, then we have:*

$$\mathbf{limsup}_{k\to\infty}\|\widehat{Q}^{\pi_k}-Q^*\|_\infty \le \frac{1+\gamma^2}{(1-\gamma)^2}\varepsilon \tag{31}$$

where $\varepsilon = \mathbf{limsup}\varepsilon_{k k\to\infty}$.

Proof We have by Lemma 4:

$$\mathop{\mathbf{limsup}}_{k\to\infty} f_k \le \gamma\,\mathbf{limsup}_{k\to\infty}f_k+\gamma\,\mathbf{limsup}_{k\to\infty}e_k+2\gamma\varepsilon$$

Now by Lemma 3 we have:

$$\mathop{\mathbf{limsup}}_{k\to\infty} e_k = \frac{2\gamma}{1-\gamma}\varepsilon$$

So we can deduce:

$$
\begin{aligned}
&(1-\gamma)\mathop{\mathbf{limsup}}_{k\to\infty} f_k \le \gamma\frac{2\gamma}{1-\gamma}\varepsilon+2\gamma\varepsilon=\frac{2\gamma}{1-\gamma}\varepsilon\\
&\Rightarrow \mathbf{limsup}_{k\to\infty}f_k=\frac{2\gamma}{(1-\gamma)^2}\varepsilon\\
&\mathbf{limsup}_{k\to\infty}\|Q^{\pi_k}-Q^*\|_\infty \le \frac{2\gamma}{(1-\gamma)^2}\varepsilon.
\end{aligned}
$$

Moreover, we have:

$$
\begin{aligned}
&\|\widehat{Q}^{\pi_k}-Q^*\|_\infty = \|\widehat{Q}^{\pi_k}-Q^{\pi_k}+Q^{\pi_k}-Q^*\|_\infty\\
&\le \|\widehat{Q}^{\pi_k}-Q^{\pi_k}\|_\infty+\|Q^{\pi_k}-Q^*\|_\infty\\
&\Rightarrow \mathbf{limsup}_{k\to\infty}\|\widehat{Q}^{\pi_k}-Q^*\|_\infty \le \mathbf{limsup}_{k\to\infty}\|\widehat{Q}^{\pi_k}-Q^{\pi_k}\|_\infty\\
&+\mathbf{limsup}_{k\to\infty}\|Q^{\pi_k}-Q^*\|_\infty \le \varepsilon+\frac{2\gamma}{(1-\gamma)^2}\varepsilon\\
&\Rightarrow \mathbf{limsup}_{k\to\infty}\|\hat{Q}^{\pi_k}-Q^*\|_\infty \le \frac{1+\gamma^2}{(1-\gamma)^2}\varepsilon.
\end{aligned}
$$

□

Briefly, two main objectives were analyzed in this section. The result of Lemma 1 shows that every arbitrary continuous function can be approximated by the defined function set of FLSPI fuzzy system. In other words, FLSPI is powerful to approximate any state–action value function with any arbitrary accuracy. Lemmas 2, 3, 4 and 5 are preliminaries for proving Theorem 2. This theorem defined the error bound for difference between approximated state–action value function and optimal state–action value function. This error bound depends on ε_k, i.e., the approximation accuracy of kth produced state–action value function by FLSPI. If this limit converges to zero, then the error converges to zero and approximated state–action value function converges to optimal state–action value function.

5 Simulation

In this section, we first demonstrate the performance of FLSPI versus FQL and FSL in the boat problem. We select these two methods for two reasons: First these methods are among FRL methods and second, their action space is continuous. We also show the function of error bound theorem (i.e., Theorem 2) for FLSPI in a single-goal environment. Then, we apply FLSPI, FQL and FSL to show FLSPI convergence versus divergence of FQL and FSL in the inverted pendulum problem. Finally, the well-known cart–pole balancing problem is selected with the aim of comparing performance of FLSPI against the latest proposed method with continuous state–action spaces.

5.1 Boat Problem

We implemented our off-line FLSPI on the well-known boat problem [15]. The goal is to tune a fuzzy controller using FLSPI to drive a boat from the left bank to the right bank in a river with strong nonlinear current. The goal of this problem is to reach the quay from any position on the left bank. The problem states are two-dimensional that have continuous variables, namely x and y which are the position of the boat bow ranging from 0 to 200. The quay center is located at (200,100) and its width is five. The action space of this problem is boat rudder angles. The learner agent's goal is to learn suitable angle (i.e., action) in any state (considering the related water force) by using reinforcement signal (i.e., reward).

To partition the input parameter x and y, five fuzzy sets is defined. So we have 25 fuzzy rules. The output of controller is boat angle. Twelve actions (boat angles) are determined for any rule consequences: $A = \{-100, -90, -75, -60, -45, -35, -15, 0, 15, 45, 75, 90\}$.

Although we could define different action set for every rule, we use same action sets for all rules. In addition, initial weight values are the same and equal to zero. Controller produces continuous output from combination of these discrete actions. We use ε-greedy action selection method with high initial exploration rate and decrease it along with algorithm execution. The learning aim is to find suitable actions for rule consequences.

Table 1 Simulation results

Method	Initial param.	Avg. DEI	Avg. LDI	SD (LDI)	Failure rate	Avg. time (s)
FQL	$\alpha = 0.1$	12.46	1865.9	1224.1	5.35	147.86
FSL	$\alpha = 0.1$	15.1	1839	1262.2	4.46	54.61
FLSPI		3.06	360.9	76.04	1.17	14.45

The results of 100 distinct runs of FLSPI are given in Table 1 and are compared to 100 distinct runs of FQL and FSL [9]. The average of learning duration index (Avg. LDI), standard deviation of learning duration index (std. LDI), average of distance error index (Avg. DEI), failure rate and average run time (avg. time) are depicted in Table 1. The experiment was based on a machine with Intel core i7 (2.20GH) processor and 8 gigabytes of memory.

As it is shown in Table 1, FLSPI achieved better results than FQL and FSL. For instance, average LDI for FLSPI is 5.17 and 5.1 times better than FQL and FSL, respectively. Generally all metrics show a major improvement. In addition, since the average time is remarkably decreased, FLSPI is suitable for real-time problems.

Figures 2, 3 and 4 show the histogram of LDI for FLSPI, FSL and FQL, respectively. One can see that in all cases the agent learns in less than 670 episodes with FLSPI while FSL and FQL cannot learn in the several iterations even after 5000 episodes. Also our experiment shows FLSPI does not have any divergence case in 100 distinct runs while two other methods have some divergence cases.

In Fig. 5, the episodic changes of the first rule weights in FLSPI are shown. Obviously in FLSPI, weights converge very fast. In fact, after almost 700 episodes the change of weights are very small and after 1000 episodes weights completely converge. Results of implementation of FLSPI on 40 test data are shown in Fig. 6. It can bee seen that the agent has learned the task very fine and satisfactorily.

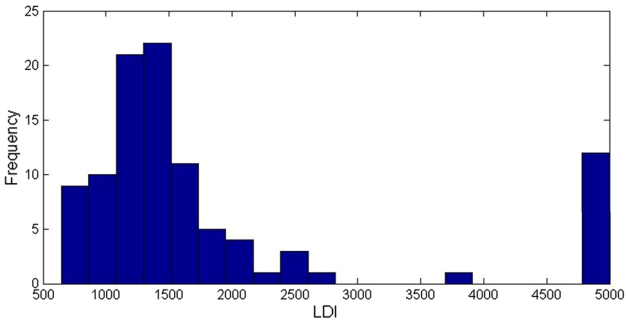

Fig. 3 Histogram of learning duration indexes for Fuzzy SARSA Learning

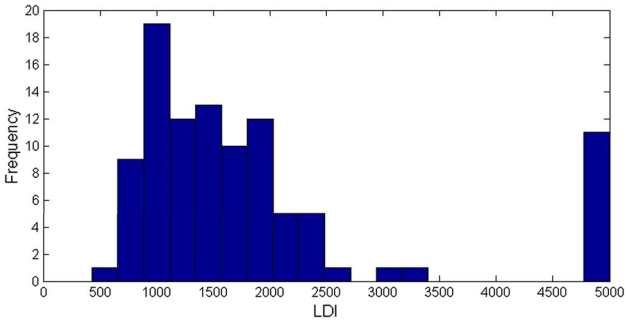

Fig. 4 Histogram of learning duration indexes for fuzzy Q-learning

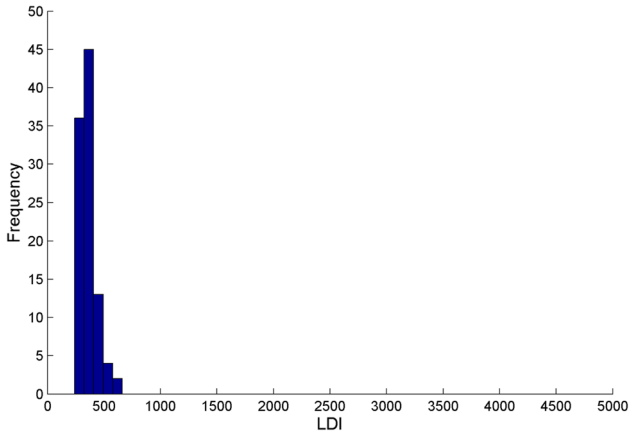

Fig. 2 Histogram of learning duration indexes for fuzzy least square policy iteration

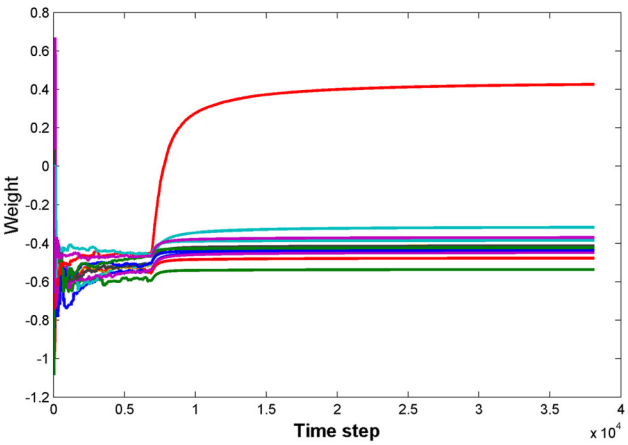

Fig. 5 Episodic changes of the first rule weights of fuzzy least square policy iteration

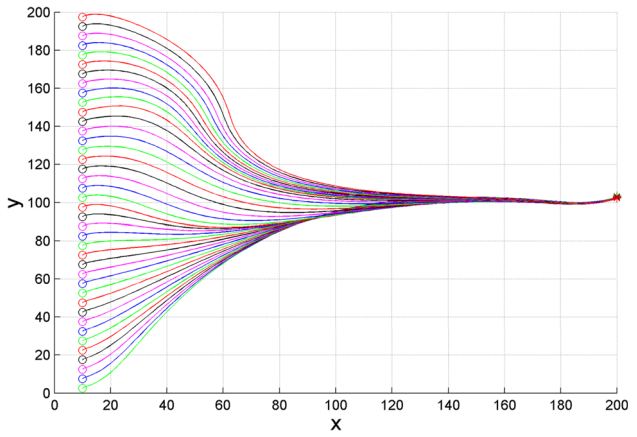

Fig. 6 Sample test of FLSPI after learning. Test is down on 40 discrete points

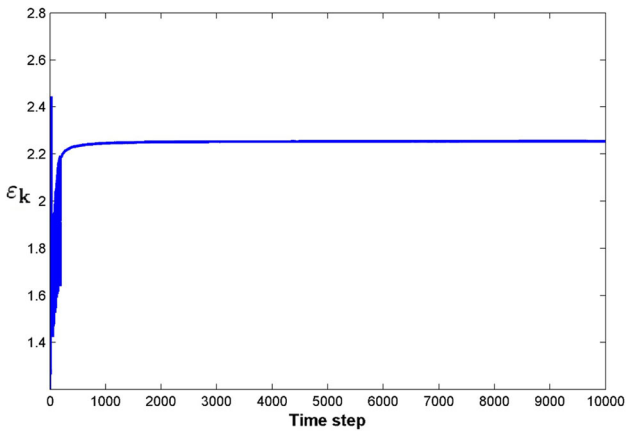

Fig. 7 $\varepsilon_{\mathbf{k}}$ changes for $\gamma = 0.5$

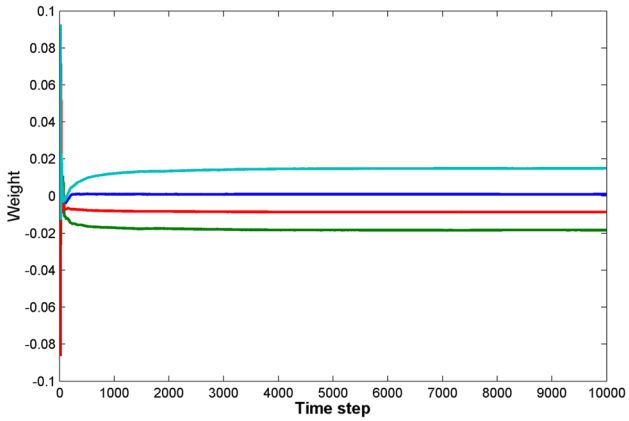

Fig. 8 Weight changes for $\gamma = 0.5$

5.2 Single-Goal Obstacle-Free Environment

We use a single-goal obstacle-free environment to show function of error bound theorem for FLSPI. This environment has two dimensions with 16 states and 4 partitions in every dimension. The start point is set to (1,1) and the goal to (4,4). Action set has four members: right, up, left and down with angles of $\{0^{\circ}, 90^{\circ}, 180^{\circ}, 270^{\circ}\}$. The final action is agent's rounded angle from horizontal axis. Step lengths are equal to 1. Agent receives reward $+1$ for goal state, -1 for edges and -0.01 for other states. Since we need to compare approximated state–action value function with actual state–action value function, we discretize the state space to use Theorem 2 .

The input of the system is state (x, y) and five triangular fuzzy sets are used to partition each dimension. So we have 25 fuzzy rules. 4 actions ({right, up, left, down}) are candidate for every rule. The ε-greedy action selection mechanism is also used. We examine this problem for two different values of discount factor (i.e., γ); 0.5 and 0.2. We execute 10 independent runs and each run has 10000 episodes. An episode finishes if the agent reaches the corners or if the number of steps exceeds 500.

Here, the obtained values for ε (that is defined in Theorem 2) are 2.2543 and 1.2334 for $\gamma = 0.5$ and $\gamma = 0.2$, respectively. We show the related diagram for $\gamma = 0.5$ in Fig. 7. Also after 10000 episodes, state–action value function converges to optimal value. So we can deduce:

$$\gamma = 0.5 \Rightarrow \mathbf{lim}sup_{k\to\infty}\|\widehat{Q}^{\pi_k} - Q^*\|_{\infty} = 2.2543\gamma = 0.2$$
$$\Rightarrow \mathbf{lim}sup_{k\to\infty}\|\widehat{Q}^{\pi_k} - Q^*\|_{\infty} = 1.2334$$

These values should be less than the defined error bound in Theorem 2:

$$\gamma = 0.5 \Rightarrow \frac{1+\gamma^2}{(1-\gamma)^2}\varepsilon = \frac{1+0.5^2}{(1-0.5)^2} * 2.2543 = 11.2715$$
$$\gamma = 0.2 \Rightarrow \frac{1+\gamma^2}{(1-\gamma)^2}\varepsilon = \frac{1+0.2^2}{(1-0.2)^2} * 1.2334 = 2.0042$$

So this experiment shows the correctness of Theorem 2 perceptively.

Figure 8 shows the fast weights convergence. Weights do not change after almost 3000 episodes.

5.3 Inverted Pendulum: Real-Time Control

We examine FLSPI as an on-line control mechanism on the inverted pendulum problem and compare results with FQL and FSL. This inverted pendulum [4] is a novel type of the classical well-known inverted pendulum. In this problem, a mass is placed off center on a disk and rotates in a vertical plan. This mass is driven by a DC motor that its voltage is limited so that the motor does not provide enough power to push the pendulum up in a single rotation. Figure 9 shows a

schematic of the inverted pendulum. The goal is to keep the pendulum up in a stable form. To reach to the goal, the pendulum needs to be swung back and forth to gather energy. This problem is a difficult and highly nonlinear control problem [4].

The dynamic of the inverted pendulum for continuous time is as follows:

$$\ddot{\theta} = \frac{1}{J}\left(mgl\sin(\theta) - b\dot{\theta} - \frac{K^2}{R}\dot{\theta} - \frac{K}{R}a\right) \tag{32}$$

The values of the parameters in this simulation are as follows [4]:

$$m = 0.055, g = 9.81, l = 0.042, J = 1.91 \times 10^{-4}$$
$$b = 3 \times 10^{-6}, K = 0.0536, R = 9.5$$

The state space has two dimensions and consists of the angle and the angular velocity of the pendulum, i.e., $s = [\theta, \dot{\theta}]^T$. The angle of the pendulum is in $[-\pi, \pi]$ rad (wraps around), where $\theta = -\pi$ points down and $\theta = 0$ points up. The velocity is in the interval $[-15\pi, 15\pi]$ rad/s, and the control action a (voltage) is limited to the interval $[-3, 3]$ V. We choose sample time (T_s) equal to 0.005s, and the dynamic of the system is calculated in consecutive time steps. We use an action for 10 consecutive time steps and calculate new action for the next 10 consecutive time steps. In other words, time step to calculate action and updating weight vector is 0.05s.

The goal is to stabilize the pendulum to point up (i.e., $\theta = 0$ and $\dot{\theta} = 0$). The reward function to reach the goal is as follows:

$$r = -s^T Q_{rew} s - R_{rew} a^2, \quad Q_{rew} = \begin{bmatrix} 5 & 0 \\ 0 & 0.1 \end{bmatrix}, R_{rew} = 1 \tag{33}$$

We set the discount factor to $\gamma = 0.98$. The exploration rate should be large enough at the beginning to visit the high rewards around the goal. Here we update the weight vector every five time steps (i.e., $k = 5$).

The state space is partitioned into 6 equidistant cores partition with triangular membership functions in the both dimensions. The discrete action sets are made of 7 voltages $A = \{-3, -2, -1, 0, 1, 2, 3\}$. We assume that there is a local controller that controls pendulum to reach the goal if $-0.05\pi \leq \theta \leq 0.05\pi$ and $-0.05 \times 15\pi \leq \dot{\theta} \leq 0.05 \times 15\pi$. We use the ε-greedy action selection method.

In this experiment, we determine 250 time step to reach the goal in every episode. The results of 100 distinct runs of FLSPI are given in Table 2. The used criterions in this problem are the same as those in boat problem. Here the maximum number of episodes that agent is allowed to learn the goal, is 3000 episodes.

The weights in FLSPI converge very fast. Figure 10 shows that the weights in FLSPI have been converged to their final amounts in less than 1000 time steps. But the weights in FQL and FSL that are shown in Figs. 11 and 12 have not been converged even after 10000 time steps. We repeated this experiment for several times and deduced that FQL and FSL cannot converge in this problem. So, the agent could not learn to reach the goal by using these algorithms, at all.

Histogram of learning duration indexes for FLSPI is depicted in Fig. 13. As one can see, FLSPI has fast convergence and generally learn in less than 300 time steps.

5.4 Cart–Pole Balancing

It is mentioned earlier, despite the fact that most of the real problems are in continuous state–action spaces, a few

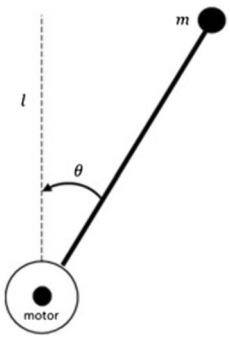

Fig. 9 A schematic representation of inverted pendulum

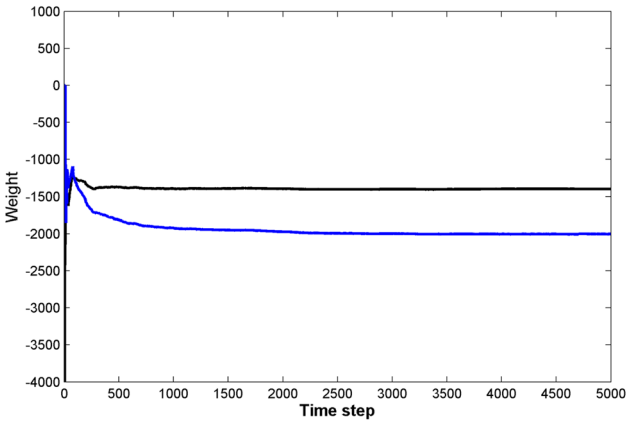

Fig. 10 Episodic changes of the first two actions of first rule of fuzzy least square policy iteration

Table 2 Simulation results

Method	Avg. DEI	Avg. LDI	SD (LDI)	Failure rate	Avg. time (s)
FLSPI	1.87	205.93	506.63	1.84	48.50

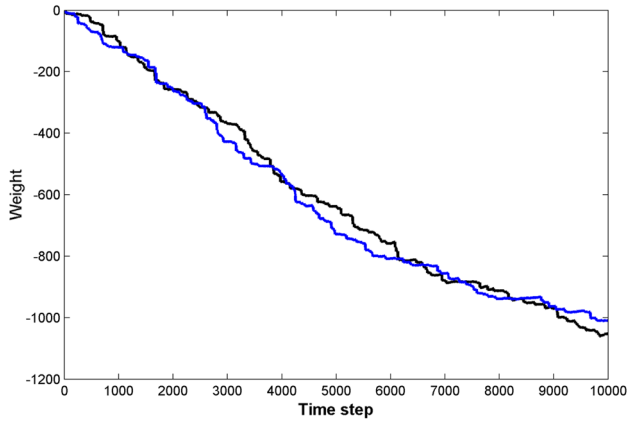

Fig. 11 Episodic changes of the first two actions of first rule of fuzzy Q-learning

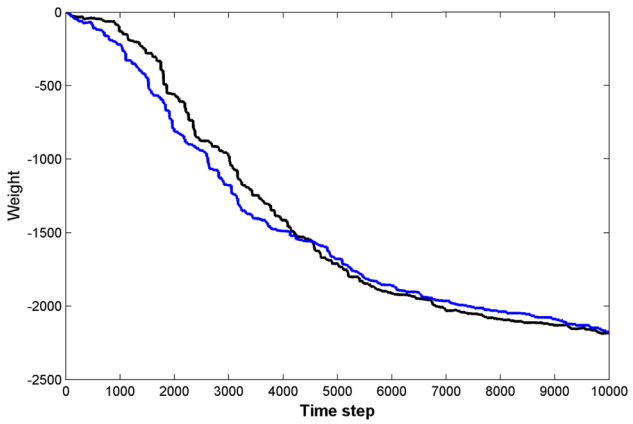

Fig. 12 Episodic changes of the first two actions of first rule of Fuzzy SARSA Learning

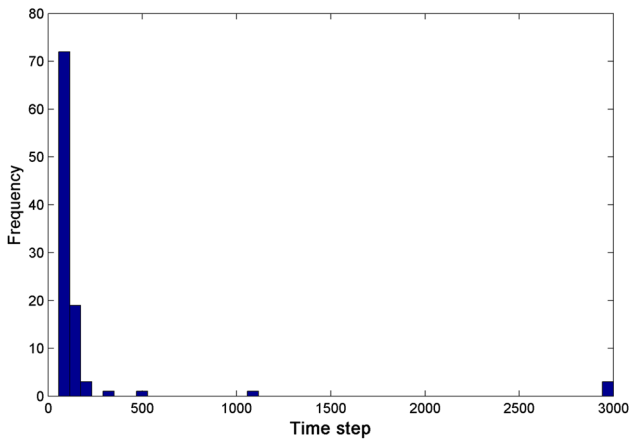

Fig. 13 Histogram of learning duration indexes for fuzzy least square policy iteration

Table 3 Simulation results

Method	FLSPI	DDPG	TRPO	TNPG	RWR
Mean return	24826	4634.4	4869.8	3986.4	4861.5

solution approaches are applicable for these problems. In this section we apply our method to the classic cart–pole balancing benchmark [17] as it is in continuous state–action spaces and compared its results with some new continuous RL methods (even though they are not in FRL field). In this problem, the agent should learn how to apply horizontal force to balance the pole on the cart. A four-dimensional state space $[x, x', \theta, \theta']$ is defined, where x is the cart horizontal coordinate, x' is its derivation (cart velocity), θ is the pole angle and θ' is its derivation (Angular velocity). State update is done by Equation 34:

$$\theta'' = \frac{1}{l(M + m\ \sin^2\theta)} \left[-f\cos\theta - m\ l\ \theta'^2 \cos\theta \sin\theta - (M+m)g \sin\theta\right] \tag{34}$$

where the parameters are as follows: $g = 9.8$ is the gravity, $M = 1$ is weight of the cart, $m = 0.1$ is the weight of the pole and $l = 0.1$ is the length of the pole. We partitioned state space into three parts in every dimension and used triangular fuzzy functions. The set $\{-10, -7, -4, 0, 4, 7, 10\}$ is defined as candidate action set (forces) . We used $r(s,f) = 10 - (1 - \cos(\theta)) - 10^{-5}||f||_2^2$, from [10] as reward function and initiate the variables from random points in every iteration. Time step is set to $Ts = 0.02s$ and an iteration will be terminated if $|x| \geq 2.4m$ or $|\theta| \geq 0.2rad$. We start with 0.05 as initial exploration rate in every iteration and decrease it with the rate of 0.98. Exploration rate could not be less than 0.0001 in our experiment.

Table 3 shows the average return (sum of rewards) over all training iterations of on-line FLSPI in comparison with Deep Deterministic Policy Gradient (DDPG) [10], Reward-Weighted Regression (RWR) [8], Trust Region Policy Optimization (TRPO) and Truncated Natural Policy Gradient (TNPG) [20], reported by Ref. [10], based on 2000 iterations training trials.

As it is seen, FLSPI could balance the pole on the cart in the longest time than other algorithms, on average, in the training phase. FLSPI could learn to balance the pole after some iteration and could balance the pole up to 99869 time step (i.e., 33.29 min) in the learning phase and could balance it entirely (infinite steps) in the testing phase (after learning).

Bringing all together, FLSPI has significant efficiency compared to the others. In addition, it has theoretical

analysis to prove its performance while most of the other proposed methods do not have such analysis.

6 Conclusion

In this paper, we presented a FRL method based on LSPI. The algorithm and mathematical analysis was presented. This algorithm is learning rate independent and has fast convergence. To evaluate this approach, we compared performance of FLSPI with two approaches in critic-only FRL, FQL and FSL, on the well-known boat problem (in off-line form). Results showed that FLSPI has higher performance and learns faster than FQL and FSL. The mathematical analysis defined an error bound for approximate state–action value function that was introduced by FLSPI algorithm. We used single-agent obstacle-free environment to show function of FLSPI error bound theorem. The results showed that error bound was true for these values and the weights converge very fast.

Also we apply on-line FLSPI and FSL and FQL to the inverted pendulum problem. Results show that FLSPI has fast convergence and high performance in this problem but FQL and FSL cannot converge even after 10000 episodes. So this problem is an example for diverging FSL and FQL. In addition, we compared FLSPI with some new continuous RL methods in the well-known cart–pole balancing benchmark. This experiment also proved FLSPI efficiency.

Generally, FLSPI is suitable for real-time problems and has high performance and fast convergence.

Acknowledgments This research was partially supported by Iran National Science Foundation (INSF). The authors would like to thank INSF for its support.

References

1. Bellman, R.: A markovian decision process. Technical report, DTIC Document (1957)
2. Berenji, H.R., Vengerov, D.: A convergent actor-critic-based FRL algorithm with application to power management of wireless transmitters. IEEE Trans. Fuzzy Syst. **11**(4), 478–485 (2003)
3. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
4. Buşoniu, L., Babuška, R., Schutter, B.D., Ernst, D.: Reinforcement Learning and Dynamic Programming Using Function Approximators. CRC Press, Boca Raton (2010)
5. Buşoniu, L., Ernst, D., De Schutter, B., Babuška, R.: Continuous-state reinforcement learning with fuzzy approximation. In: Kudenko, D., Kazakov, D., Alonso, E. (eds.) Adaptive Agents and Multi-Agent Systems III: Adaptation and Multi-Agent Learning, pp. 27–43. Springer, Berlin (2008)
6. Buşoniu, L., Ernst, D., Schutter, B.D.: Online least-squares policy iteration for reinforcement learning control. In: American Control Conference (ACC-10), (2010)
7. Castro, J.: Fuzzy logic controllers are universal approximators. IEEE Trans. Syst. Man Cybern. **25**, 629–635 (1995)
8. Deisenroth, M.P., Neumann, G., Peters, J., et al.: A survey on policy search for robotics. Found. Trends Robot. **2**(1–2), 1–142 (2013)
9. Derhami, V., Majd, V.J., Ahmadabadi, M.N.: Fuzzy sarsa learning and the proof of existence of its stationary points. Asian J. Control **10**, 535–549 (2008)
10. Duan, Y., Chen, X., Houthooft, R., Schulman, J., Abbeel, P.: Benchmarking deep reinforcement learning for continuous control. arXiv preprint arXiv:1604.06778 (2016)
11. Ghorbani, F., Derhami, V., Nezamabadipour, H.: A novel approach in fuzzy reinforcement learning. J. Control **8**, 11–20 (2014)
12. Glorennec, P.Y., Jouffe, L.: Fuzzy q-learning. In: IEEE International Conference on Fuzzy Systems (1997)
13. Howard, R.: Dynamic Programming and Markov Processes. MIT Press, Cambridge (1960)
14. Jang, J.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing. Prentice Hall, Englewood Cliffs (1997)
15. Jouffe, L.: Fuzzy inference system learning by reinforcement methods. IEEE Trans. Syst. Man Cybern. Part C **28**, 338–355 (1998)
16. Lagoudakis, M., Parr, R.: Least-squares policy iteration. J. Mach. Learn. Res. **4**(4), 1107–1249 (2003)
17. Michie, D., Chambers, R.A.: Boxes: an experiment in adaptive control. Mach. Intell. **2**(2), 137–152 (1968)
18. Panahi, F., Ohtsaki, T.: Optimal channel-sensing scheme for cognitive radio systems based on fuzzy q-learning. IEICE Trans. Commun. **97**(2), 283–294 (2014)
19. Rovcanin, M., Pooreter, E.D., Moerman, I., Demeester, P.: A reinforcement learning based solution for cognitive network cooperation between co-located, heterogeneous wireless sensor networks. Ad Hoc Networks **17**, 98–113 (2014)
20. Schulman, J., Levine, S., Moritz, P., Jordan, M.I., Abbeel, P.: Trust region policy optimization. CoRR. arXiv: abs/1502.05477 (2015)
21. Shamshirb, S., Patelc, A., Anuarb, N.B., Kiahb, M.L.M., Abrahame, A.: Cooperative game theoretic approach using fuzzy q-learning for detecting and preventing intrusions in wireless sensor networks. Eng. Appl. Artif. Intell. **32**, 228–241 (2014)
22. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
23. Thiery, C., Scherrer, B.L.: Least-squares policy iteration: bias-variance trade-off in control problems. In: International Conference on Machine Learning (2010)
24. Wang, L.-X., Mendel, J.M.: Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. IEEE Trans. Neural Netw. **3**(5), 807–814 (1992)

Farzaneh Ghorbani received M.S. degrees in Mathematics and Artificial Intelligence from Shahid Bahonar University of Kerman and Yazd University, Iran, in 2005 and 2013, respectively. Currently, she is working toward her Ph.D. degree in the Department of Computer Science, Zanjan University, Iran. Her research interests are fuzzy systems, reinforcement learning, robotics and multi-agent learning.

Vali Derhami received the B.Sc. degree from Esfahan University of Technology, Iran, in 1996. He received M.S. and Ph.D. degrees from Tarbiat Modares University, Iran, in 1998 and 2007, respectively. Currently, he is Associate Professor in Computer Engineering Department in Yazd University. His research interests are neural fuzzy systems, intelligent control, reinforcement learning and robotics.

Mohsen Afsharchi received his M.Sc. degree in Computer Engineering from the Iran University of Science and Technology in 1996, and Ph.D. in Artificial Intelligence from the University of Calgary, Canada, in 2006, respectively. From 1996 to 2002, he was a University Lecturer at the University of Zanjan. Since 2006, he has been with the Computer Engineering Department of the University of Zanjan where he leads the Multi-agent Systems Lab. He is also adjunct Researcher in the Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan, Iran. He is currently Associate Professor, and his research interests are in multi-agent learning, probabilistic reasoning and distributed constraint optimization.