Sz. Kovács: *SVD Reduction in Continuos Environment Reinforcement Learning*, Lecture Notes in Computer Science, Vol. 2206, Computational Intelligence, Theory and Applications, Bernard Reusch (Ed.), Springer, ISBN 3-540-42732-5, pp.719-738, Germany, (2001). – Draft version.

SVD Reduction in Continuos Environment Reinforcement Learning

Szilveszter Kovács

Research Institute of Manufacturing Information Technology Gifu Prefecture
4-179-19 Sue Kakamigahara Gifu 509-0108 Japan
On research leave from: Department of Information Technology,
University of Miskolc, Miskolc-Egyetemváros, Miskolc, H-3515, Hungary
E-mail: szkszilv@gold.uni-miskolc.hu,
http://www.iit.uni-miskolc.hu/~szkovacs

Abstract. Reinforcement learning methods, surviving the control difficulties of the unknown environment, are gaining more and more popularity recently in the autonomous robotics community. One of the possible difficulties of the reinforcement learning applications in complex situations is the huge size of the state-value- or action-value-function representation [2]. The case of continuous environment (continuous valued) reinforcement learning could be even complicated, as the state-value- or action-value-functions are turning into continuous functions. In this paper we suggest a way for tackling these difficulties by the application of SVD (Singular Value Decomposition) methods [3], [4], [15], [26].

1 Introduction

Reinforcement learning methods are trial-and-error stile learning methods adapting dynamic environment through incremental iteration. The principal ideas of reinforcement learning methods, the dynamical system state and the idea of "optimal return" or "value" function are inherited from optimal control and dynamic programming [1]. One common goal of the reinforcement learning strategies is to find an optimal policy by building the *state-value-* or *action-value-function* [2]. The *state-value-function* $V^{\pi}(s)$, is a function of the expected return (a function of the cumulative reinforcements), related to a given state $s \in S$ as a starting point, following a given policy π. Where the states of the learning agent are observable and the reinforcements (or rewards) are given by the environment. These rewards are the expression of the goal of the learning agent as a kind of evaluation follows the recent action (in spite of the instructive manner of error feedback based approximation techniques, like the gradient descent training). The policy is the description of the agent behavior, in the form of mapping between the agent states and the corresponding suitable actions. The *action-value function* $Q^{\pi}(s,a)$, is a function of the expected return, in case of taking action $a \in A_s$ in a given state *s*, and then following a given policy π. Having the action-value-function, the optimal (greedy) policy, which always take the optimal (the greatest estimated value) action in every states, can be constructed as [2]:

$$\pi(s) = \arg\max_{a \in A_s} Q^{\pi}(s,a). \qquad \textbf{(1)}$$

Namely for estimating the optimal policy, the action-value function $Q^{\pi}(s,a)$ is needed to be approximated. In discrete environment (discrete states and discrete actions) it means, that at least $\sum_{s \in S} \|A_s\|$ element must be handled. (Where $\|A_s\|$ is the cardinality of the set of possible actions in state *s*.) Having a complex task to adapt, both the number of possible states and the number of the possible actions could be an extremely high value.

1.1 Reinforcement Learning in Continuous Environment

To implement reinforcement learning in continuous environment (continuous valued states and actions), function approximation methods are widely used. Many of these methods are applying tailing or partitioning strategies to handle the continuous state and action spaces in the similar manner as it was done in the discrete case [2]. One of the difficulties of building an appropriate partition structure is the anonymity of the action-value-function structure. Applying fine resolution in the partition leads to high number of states, while coarse partitions could yield imprecise or unadaptable system. Handling high number of states also leads to high computational costs, which could be also unacceptable in many real time applications.

1.2 Fuzzy Techniques in Continuos Environment Reinforcement Learning

There are many methods in the literature for applying fuzzy techniques in reinforcement learning (e.g. for "Fuzzy Q-Learning" [9], [10], [11], [12], [6]). One of the main reason of their application beyond the simplicity of expressing priory knowledge in the form of fuzzy rules, is the universal approximator property [7], [8] of the fuzzy inference. It means that any kind of function can be approximated in an acceptable level, even if the analytic structure of the function is unknown. Despite of this useful property, the use of fuzzy inference could be strictly limited in time-consuming reinforcement learning by its complexity problems [16], because of the exponential complexity problem of fuzzy rule bases [13], [3], [4]. Fuzzy logic inference systems are suffering from exponentially growing computational complexity in respect to their approximation property. This difficulty comes from two inevitable facts. The first is that the most adopted fuzzy inference techniques do not hold the universal approximation property, if the numbers of antecedent sets are limited, as stated by Tikk in [17]. Furthermore, their explicit functions are sparse in the approximation function space. This fact inspires to increase the density, the number of antecedents in pursuit of gaining a good approximation, which, however, may soon lead to a conflict with the computational capacity available for the implementation, since the increasing number of antecedents explodes the computational requirement. The latter is the second fact and stated by Kóczy et al. in [16]. The effect of this contradiction is gained by the lack of a mathematical framework capable of estimating the necessary minimal number of antecedent sets. Therefore a heuristic setting of the number of antecedent sets is applied, which usually overestimates, in order to be on the safe side, the necessary number of antecedents resulting in an unnecessarily high computational cost. E.g. the structurally different Fuzzy Q-Learning method implementations introduced [9], [10], [11] and [12] are sharing the same concept of fixed, predefined fuzzy antecedent partitions, for state representation. One possible solution for this problem is suggested in [6]. By introducing "Adaptive State Partitions", an incremental fuzzy clustering of the observed state transitions. This method can lead to a better partition than the simple heuristic, by finding the best fitting one in respect to the minimal squared error, but still has the problem of limited approximation property inherited from the limited number of antecedent fuzzy sets.

Another promising solution, as a new topic in fuzzy theory, is the application of fuzzy rule base complexity reduction techniques.

1.3 Fuzzy rule base complexity reduction

The main idea of application fuzzy rule base complexity reduction techniques for reinforcement learning is enhancing the universal approximator property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low.

Some reduction techniques are classified regarding their concept in [14] and [4]. A fuzzy rule importance based technique is proposed by Song et al. in [20]. Another recent method proposed by Sudkamp et al. [22] combines rule learning with a region merging strategy.

Recently, several publications have applied orthogonal transformation methods for selecting important rules from a given rule base, for instance, in 1999 Yen and Wang investigated various techniques in [14] for possible fuzzy rule base simplification techniques such as orthogonal least-squares, eigenvalue decomposition, SVD-QR with column pivoting method, total least square method and direct SVD method. [21] also proposes an SVD based technique with examples.

SVD based fuzzy approximation technique was initialized in 1997 by Yam [15], which directly finds a minimal rule-base from sampled values. Shortly after, this concept was introduced as SVD fuzzy rule base

reduction and structure decomposition in [3], [24], [25]. Its key idea is conducting SVD of the consequents and generating proper linear combinations of the original membership functions to form new ones for the reduced set. [3], [15] characterizes fuzzy functions by the conditions of sum-normalization (SN), nonnegativenes (NN) and normality (NO), and extends SVD reduction with further tools to preserve SN and NN conditions of the new membership functions. It may have significant role if the purpose is not only saving computational cost, but maintaining the fuzzy concept and having a theoretical study of the reduced rule's features.

An extension of [14] to multi-dimensional cases may also be conducted in a similar fashion as the higher order SVD reduction technique proposed in [3], [13], [15]. Further developments of SVD based fuzzy reduction [3] [15] are proposed in [13], [18], [19], [23].

The key idea of using SVD in complexity reduction is that the singular values can be applied to decompose a given system and indicate the degree of significance of the decomposed parts. Reduction is conceptually obtained by the truncation of those parts, which have weak or no contribution at all to the output, according to the assigned singular values. This advantageous feature of SVD is used in this paper for enhancing the universal approximator property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low. The complexity and its reduction is discussed in regard of the number of rules, which result simplicity in operating with the rules, in reinforcement learning methods.

On the other hand, as one of the natural problems of any complexity reduction technique, the adaptivity property of the reduced approximation algorithm becomes highly restricted. Since the crucial concept of the Fuzzy Q-learning is based on the adaptivity of the action-value function this paper is aimed propose to adopt an algorithm [26] capable of embedding new approximation points into the reduced approximation while the calculation cost is kept.

This paper is organized as follows. Section 2 briefly summarizes the concept of Fuzzy Q-learning. Section 3 introduces the proposed Fuzzy Q-learning and dynamic partition allocation method. Section 4 examines various concepts of adaptation SVD based techniques, e.g. complexity reduction and approximation adaptation [26], for reinforcement learning. Section 5 gives two simple examples for the practical use of the proposed method.

2 Reinforcement Learning

For introducing a possible way of application of SVD complexity reduction techniques in Fuzzy Reinforcement Learning, a simple direct (model free) reinforcement learning method, the Q-Learning [5], was chosen.

The goal of the Q-learning is to find the fixed-point solution Q of the Bellman Equation [1] through iteration. In discrete environment *Q-Learning* [5], the action-value-function is approximated by the following iteration:

$$\begin{aligned} &Q_{i,u} \approx \tilde{Q}_{i,u}^{k+1} = \tilde{Q}_{i,u}^{k} + \Delta\tilde{Q}_{i,u}^{k+1} = \\ &\tilde{Q}_{i,u}^{k+1} = \tilde{Q}_{i,u}^{k} + \alpha_{i,u}^{k} \cdot \left(g_{i,u,j} + \gamma \cdot \max_{v \in U} \tilde{Q}_{j,v}^{k+1} - \tilde{Q}_{i,u}^{k} \right) \end{aligned} \qquad \forall i \in I, \forall u \in U \tag{2}$$

where $\tilde{Q}_{i,u}^{k+1}$ is the $k+1$ iteration of the action-value taking action A_u in state S_i, S_j is the new observed state, $g_{i,u,j}$ is the observed reward completing the $S_i \to S_j$ state-transition, γ is the discount factor and $\alpha_{i,u}^{k} \in [0,1]$ is the step size parameter (which can change during the iteration steps).

For applying this iteration to continuous environment by adopting fuzzy inference (Fuzzy Q-Learning), there are many solutions exist in the literature [6], [9], [10], [11], [12].

Having only demonstrational purposes, in this paper one of the simplest one, the order-0 Takagi-Sugeno Fuzzy Inference based Fuzzy Q-Learning is studied (a slightly modified, simplified version of the Fuzzy Q-Learning introduced in [9] and [6]). This case, for characterising the value function $Q(s,a)$ in continuous

state-action space, the order-0 Takagi-Sugeno Fuzzy Inference System approximation $\tilde{Q}(s,a)$ is adapted in the following manner:

If *s* **is** S_i **And** *a* **is** A_u **Then** $\tilde{Q}(s,a)=Q_{i,u}$, $i\in I, u\in U$, **(3)**

where S_i is the label of the i[th] membership function of the *n* dimensional state space, A_u is the label of the u[th] membership function of the one dimensional action space, $Q_{i,u}$ is the singleton conclusion and $\tilde{Q}(s,a)$ is the approximated continuous state-action-value function. Having the approximated state-action-value function $\tilde{Q}(s,a)$, the optimal policy can be constructed by function (1):

$$\tilde{\pi}(i)=\arg\max_{u\in U} Q(i,u), \quad \textbf{(4)}$$

Setting up the antecedent fuzzy partitions to be Ruspini partitions, the order-0 Takagi-Sugeno Fuzzy Inference forms the following approximation function:

$$f(x_1,x_2,...,x_N)=\sum_{j_1,j_2,\cdots,j_N}^{J_1,J_2,...,J_N}\prod_{n=1}^{N}\mu_{j_n,n}(x_n)b_{j_1j_2\ldots j_N}. \quad \textbf{(5)}$$

where $\mu_{j_n,n}(x_n)$ is the membership value of the j_n [th] antecedent fuzzy set at the n [th] dimension of the N dimensional antecedent universe X_n at the state-action observation x_n and $b_{j_1j_2\ldots j_N}$ is the value of the singleton conclusion of the $j_1j_2...j_N$ [th] fuzzy rule. In this notation all combination of the antecedents corresponds to one consequent fuzzy set defined these relations are expressed by rules as: IF $\mu_{j_1,1}(x_1)$ and $\mu_{j_2,n}(x_2)$ and … and $\mu_{j_N,N}(x_N)$ THEN $\beta_{j_1j_2\ldots j_N}$. Singleton consequent fuzzy sets $\beta_{j_1j_2\ldots j_N}$ are defined by their location $b_{j_1j_2\ldots j_N}$. on output universe Y.

Applying the notation introduced in (3), equation (5) turns to the following:

$$\tilde{Q}(s,a)=\sum_{i_1,i_2,\cdots,i_N,u}^{I_1,I_2,...,I_N,U}\prod_{n=1}^{N}\mu_{i_n,n}(s_n)\cdot\mu_u(a)\cdot q_{i_1i_2\ldots i_Nu} \quad \textbf{(6)}$$

where $\tilde{Q}(s,a)$ is the approximated state-action-value function $\mu_{i_n,n}(s_n)$ is the membership value of the i_n [th] state antecedent fuzzy set at the n [th] dimension of the N dimensional state antecedent universe at the state observation s_n, $\mu_u(a)$ is the membership value of the u [th] action antecedent fuzzy set of the one dimensional action antecedent universe at the action selection a and $q_{i_1i_2\ldots i_Nu}$ is the value of the singleton conclusion of the $i_1i_2...i_Nu$ [th] fuzzy rule.

Applying the approximation formula of the Q-learning (2) for adjusting the singleton conclusions in (5), leads to the following function:

$$q^{k+1}_{i_1i_2\ldots i_Nu}=q^{k}_{i_1i_2\ldots i_Nu}+\prod_{n=1}^{N}\mu_{i_n,n}(s_n)\cdot\mu_u(a)\cdot\Delta\tilde{Q}^{k+1}_{i,u} \quad \textbf{(7)}$$

$$q^{k+1}_{i_1i_2\ldots i_Nu}=q^{k}_{i_1i_2\ldots i_Nu}+\prod_{n=1}^{N}\mu_{i_n,n}(s_n)\cdot\mu_u(a)\cdot\alpha^{k}_{i,u}\cdot\left(g_{i,u,j}+\gamma\cdot\max_{v\in U}\tilde{Q}^{k+1}_{j,v}-\tilde{Q}^{k}_{i,u}\right)$$

where $q^{k+1}_{i_1i_2\ldots i_Nu}$ is the $k+1$ iteration of the singleton conclusion of the $i_1i_2...i_Nu$ [th] fuzzy rule taking action A_u in state S_i, S_j is the new observed state, $g_{i,u,j}$ is the observed reward completing the $S_i\rightarrow S_j$ state-

transition, γ is the discount factor and $\alpha_{i,u}^{k} \in [0,1]$ is the step size parameter. The $\max_{v \in U} \tilde{Q}_{j,v}^{k+1}$ and $\tilde{Q}_{i,u}^{k}$ action-values can be approximated by equation (6).

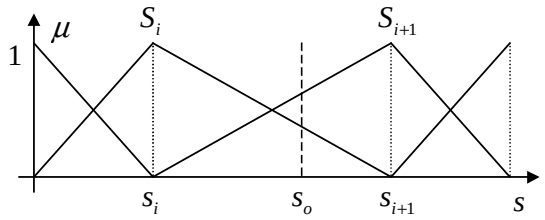
The next problematic question of the Fuzzy Reinforcement Learning, as it was introduced in section 1.2, is the proper way of building the fuzzy partitions. The methods sharing the concept of fixed, predefined fuzzy partitions, like [9], [10], [11] and [12] are facing the following question: More detailed partitions are yielding exponentially growing state spaces (rule base sizes), elongating the adaptation time, and dramatically increasing the computational resource demand, while sparse partitions could cause high approximation error, or unadaptable situation. One possible solution for this problem is suggested in [6]. By introducing "Adaptive State Partitions", an incremental fuzzy clustering of the observed state transitions. This method can lead to a better partition than the simple heuristic, by finding the best fitting one in respect to the minimal squared error, but still has the problem of limited approximation property inherited from the limited number of antecedent fuzzy sets.

In this paper another dynamic partition allocation method is suggested, which is instead of adjusting the sets of the fuzzy partition, simply increase the number of the fuzzy sets by inserting new sets in the required positions.

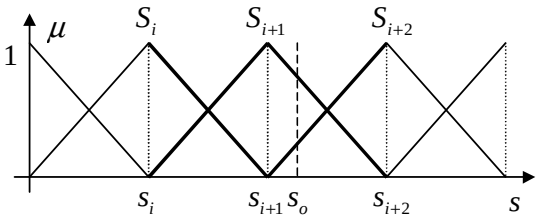
3 The Proposed Method

The reinforcement learning method proposed in this paper can be divided to two main parts. The first is the reinforcement method itself. It is the direct (model free) Fuzzy Q-Learning method as it was introduced in section 2 (order-0 Takagi-Sugeno Fuzzy Inference based modification of the Fuzzy Q-Learning introduced in [9] and [6]).

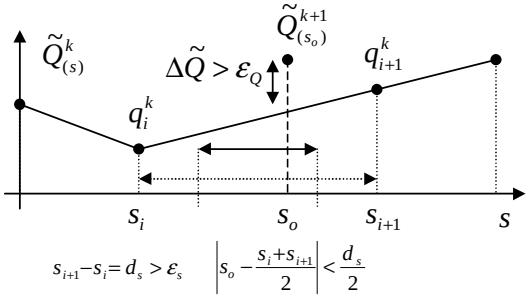
The second is the dynamic partition allocation method proposed in this paper. The main idea is very simple (see Fig.1. for an example). Initially a minimal sized (e.g. 2-3 sets only) Ruspini partition built up triangular shaped fuzzy sets on all the antecedent universes (see Fig.1.a.). In case if the action-value function update is high ($\Delta\tilde{Q} > \varepsilon_Q$), and the partition is not too dense already ($d_s > \varepsilon_s$), and the actual state-action point is far from the existing partition members (see Fig.1.b.), than a new fuzzy state is inserted to increase the resolution (see Fig.1.d.). If the update value is relatively low (see e.g. Fig.2.), or the actual state-action point is close to the existing partition members (Fig.3.), than the partition is staying unchanged. The state insertion is done in every state dimensions separately (in multidimensional case it means an insertion of a hyperplane), by interpolating the inserted values from the neighbouring ones (see Fig.1.e. and Fig.4. as a two dimensional example). Having the new state plane inserted in every required dimension, the value update is done regarding to the Fuzzy Q-Learning method as it was introduced in section 2, by the equation (7). (See e.g. on Fig.1.c,d, or Fig.4.d.)



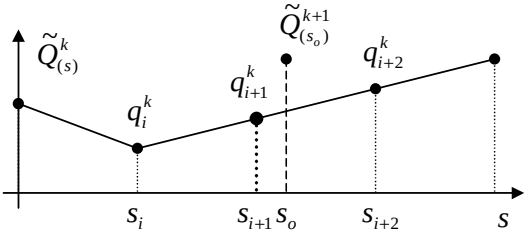
a., The original partition



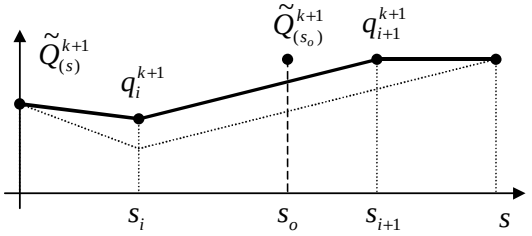
d., The modified partition (S_{i+1} is inserted)



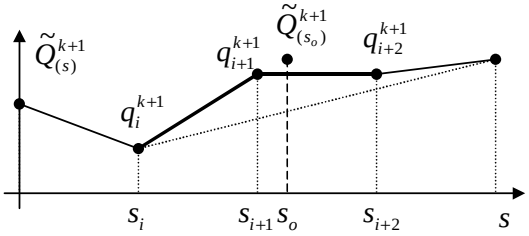
b., The next approximation of Q at s_o: $\tilde{Q}^{k+1}_{(s_o)}$



e., The inserted q^k_{i+1} values are interpolated

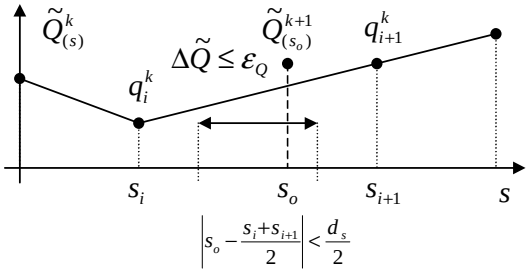


c., The next approximation, without partition modification



f., The next approximation, with partition modification

Fig. 1. The proposed dynamic partition allocation.



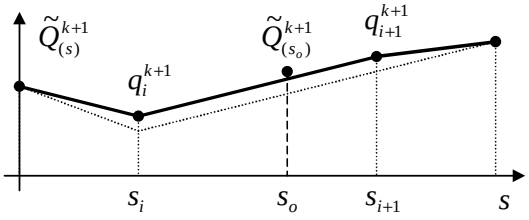
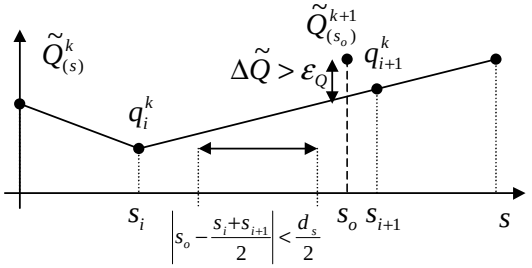


Fig. 2. The action-value function update is relatively low.



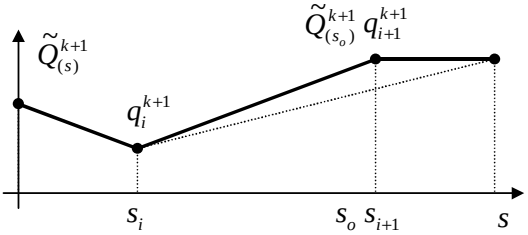
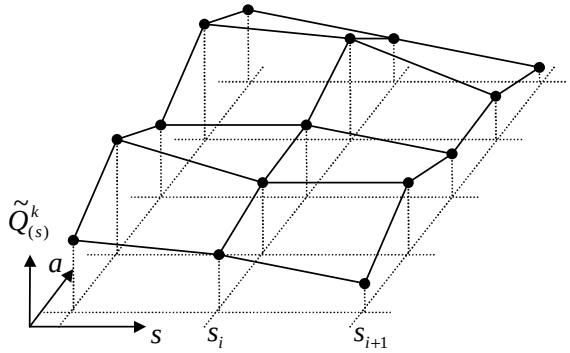
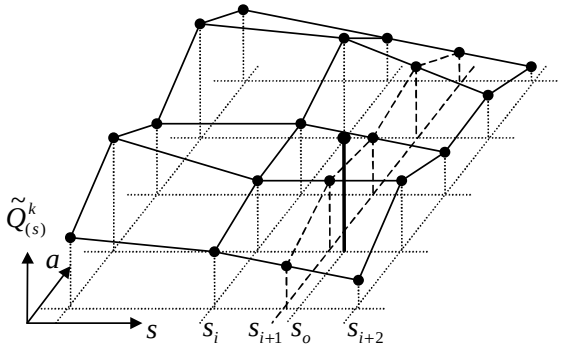


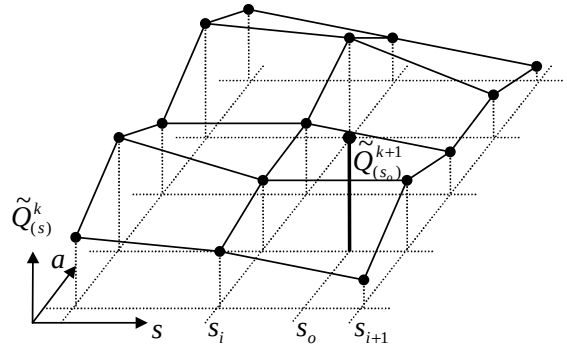
Fig. 3. The actual state-action point is close to the existing partition members.



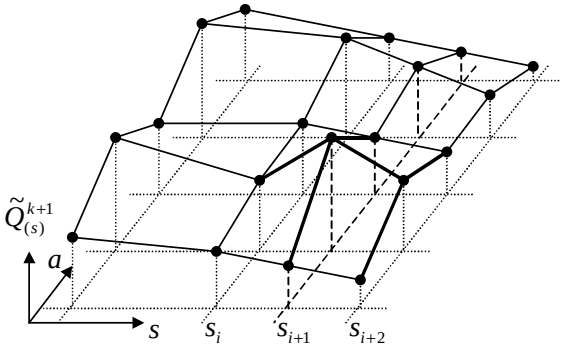
a., The original approximation of $\tilde{Q}^k_{(s)}$



c., State insertion s_{i+1}, by interpolating the inserted action values $q^k_{i+1,j}$, $\forall j \in [1, J]$, from the neighbouring $q^k_{i,j}$ and $q^k_{i+2,j}$ ones.



b., The next approximation of Q at s_o: $\tilde{Q}^{k+1}_{(s_o)}$



d., The next approximation $\tilde{Q}^{k+1}_{(s)}$, with state insertion s_{i+1}, and value update regarding to (7)

Fig. 4. The proposed dynamic partition allocation in two-dimensional (single state and action) antecedent case.

The proposed dynamic partition allocation method has the property of local step-by-step refinement in a manner very similar to the binary search. It can locate the radical positions of the value action function with the precision of $d_s^{i+k} = \frac{d_s^i}{2^k}$ in k steps (where d_s^i is the starting precision).

The main problem of the proposed simple dynamic partition allocation method is the nondecreasing adaptation manner of the antecedent fuzzy partitions. In some situation, it could mean rapidly increasing partition sizes. Moreover, these cases also lead rapidly growing, or at least nondecreasing computational resource demand.

For retaining the benefits of the dynamic partition allocation and maintaining the overall computational resource demand low, in this paper, the adoption of Higher Order SVD [13] based fuzzy rule base complexity reduction techniques and its fast adaptation method (see in section 4.2 and 4.3) is suggested. The application of the fast adaptation method [26] gives a simple way for increasing the rule density of a rule base stored in a compressed form directly (see section 4.2 and 4.3, equation (9) and [26]). Providing an economic sized structure for handling continuously increasing and varying rule bases, which is so typical in reinforcement learning.

4 SVD based Complexity Reduction

The essential idea of using SVD in complexity reduction is that the singular values can be applied to decompose a given system and indicate the degree of significance of the decomposed parts. Reduction is

conceptually obtained by the truncation of those parts, which have weak or no contribution at all to the output, according to the assigned singular values. This advantageous feature of SVD is used in this paper for enhancing the universal approximator property of the fuzzy inference by extending the number of antecedent sets while the computational complexity is kept relatively low. The complexity and its reduction is discussed in regard of the number of rules, which result simplicity in operating with the rules, in reinforcement learning methods.

4.1 Definitions

In this section some elementary definitions and concepts utilized in the further sections will be introduced. With respect to the notation, to facilitate the distinction between the types of given quantities, the notation will be reflected by their representation: scalar values are denoted by lower-case letters $\{a,b,...;\alpha,\beta,...\}$; column vectors and matrices are given by bold-face letters as $\{\mathbf{a},\mathbf{b},...\}$ and $\{\mathbf{A},\mathbf{B},...\}$ respectively, matrix $\mathbf{0}$ contains zero values only; tensors correspond to capital letters as $\{A,B,...\}$. The transpose of matrix $\mathbf{A}$ is denoted as $\mathbf{A}^{\mathrm{T}}$. Subscript is consistently used for a lower order of a given structure. E.g. and element of matrix $\mathbf{A}$ is defined by row-column number i,j symbolized as $(\mathbf{A})_{i,j}=a_{i,j}$. Systematically, the i-th column vector of $\mathbf{A}$ is denoted as $\mathbf{a}_i$, i.e. $\mathbf{A}=[\mathbf{a}_1 \quad \mathbf{a}_2 \quad \cdots]$. To enhance the overall readability characters $i,j,...$ in the meaning of indices (counters), $I,J,...$ are reserved to denote the index upper bounds, unless stated otherwise. $\Re^{I_1\times I_2\times\ldots\times I_N}$ is the vector space of real valued $(I_1\times I_2\times\ldots\times I_N)$-tensors. Letter N serves to denote the number of variables. Letter k has special role and it is: $k=1...N$, $k\neq n$.

Definition 1. (n-mode matrix of tensor A) Assume an N-th order tensor $A\in\Re^{I_1\times I_2\times\ldots\times I_N}$. The n-mode matrix $A_{(n)}\in\Re^{I_n\times J}$, $J=\prod_k I_l$ contains all the vectors in the n-th dimension of tensor A. The ordering of the vectors is arbitrary, this ordering shall, however, be consistently used later on. $(\mathbf{A}_{(n)})_j$ is called an j-th n-mode vector.

Note that any matrix of which the columns are given by n-mode vectors $(\mathbf{A}_{(n)})_j$ can evidently be restored to be tensor A.

Definition 2. (tensor interval) Assume N-th order tensors $A,B,C\in\Re^{I_1\times I_2\times\ldots\times I_N}$. $C\in_t[A,B]\Leftrightarrow A\leq_t C\leq_t B$, where $A\leq_t B\Leftrightarrow\forall i_1 i_2\ldots i_N : a_{i_1 i_2\ldots i_N}\leq b_{i_1 i_2\ldots i_N}$

Definition 3. (n-mode sub-tensor of tensor A) Assume an N-th order tensor $A\in\Re^{I_1\times I_2\times\ldots\times I_N}$. The n-mode sub-tensor $A_{i_n=\alpha}$ contains elements $a_{i_1,i_2,\ldots,i_{n-1},\alpha,i_{n+1},\ldots,i_N}$.

Definition 4. (n-mode matrix-tensor product) The n-mode product of tensor $A\in\Re^{I_1\times I_2\times\ldots\times I_N}$ by a matrix $\mathbf{U}\in\Re^{J\times I_n}$, denoted by $A\times_n\mathbf{U}$ is an $(I_1\times I_2\times\ldots\times I_{n-1}\times J\times I_{n+1}\times\ldots\times I_N)$-tensor of which the entries are given by $A\times_n\mathbf{U}=B$, where $B_{(n)}=\mathbf{U}\cdot A_{(n)}$. Let $A\times_1\mathbf{U}_1\times_2\mathbf{U}_2\ldots\times_N\mathbf{U}_N$ be noted for brevity as $A\bigotimes_{n=1}^{N}\mathbf{U}_n$.

There are major differences between matrices and higher-order tensors when rank properties are concerned. These differences directly affect the way an SVD generalization could look like. As a matter of fact, there is no unique way to generalize the rank concept. In this paper the description is restricted to n mode rank only.

Definition 5. (n mode rank of tensor) The n mode rank of A, denoted by $R_n = rank_n(A)$, is the dimension of the vector space spanned by the n mode vectors as $rank_n(A) = rank(A_{(n)})$.

Theorem (N-th Order SVD or HOSVD) Every tensor $A \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ can be written as the product $A = S \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$, in which $\mathbf{U}_n = [\mathbf{u}_{1,n} \quad \mathbf{u}_{2,n} \quad \ldots \quad \mathbf{u}_{I_N,n}]$ is a unitary $(I_N \times I_N)$-matrix called n-mode singular matrix. Tensor $S \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ of which the subtensors $S_{i_n=\alpha}$ have the properties of all-orthogonality (two subtensors $S_{i_n=\alpha}$ and $S_{i_n=\beta}$ are orthogonal for all possible values of n, α and β, when $\alpha \neq \beta$) and ordering: $\left\|S_{i_n=1}\right\| \geq \left\|S_{i_n=2}\right\| \geq \ldots \geq \left\|S_{i_n=I_n}\right\| \geq 0$ for all possible values of n.

See detailed discussion and notation of matrix SVD and Higher Order SVD (HOSVD) in [13].

4.2 SVD Based Fuzzy Rule Base Complexity Reduction

Since the state action value function is approximated by an order-0 Takagi-Sugeno Fuzzy Inference method this section is intended to provide a brief survey of the fundamentals in SVD based fuzzy rule base reduction techniques, which are proposed in [3], [4], [13], [15].

The calculation complexity of (5) explodes with values $J_1, J_2, \ldots J_N$, in this regards, for comprehensive analysis and exact theorems, see [16]. Decreasing the upper bound of the indices in the sum operator of (5), namely the number of antecedent sets, leads to the initial idea of calculation reduction. Formula (5) can be equivalently written in tensor product form as: $f(x_1, x_2, \ldots, x_N) = B \overset{N}{\underset{n=1}{\otimes}} \mathbf{m}_n$, where tensor $B \in \Re^{J_1 \times J_2 \ldots \times J_N}$ and vector $\mathbf{m}_n$ respectively contain elements $b_{j_1 j_2 \ldots j_N}$ and $\mu_{j_n,n}(x_n)$. This reduction can be conceptually obtained by reducing the size of tensor B via Higher Order SVD (HOSVD). According to the special terms in this topic, the following notation has emerged [3], [4], [13]:

Definition 6. (Exact / non-exact reduction) Assume an N-th order tensor $A \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$. **Exact** reduced form $A = A^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$, where "r" denotes "reduced", is defined by tensor $A^r \in \Re^{I_1^r \times I_2^r \times \ldots \times I_N^r}$ and basis matrices $\mathbf{U}_n \in \Re^{I_n \times I_n^r}$, $\forall n : I_n^r \leq I_n$ which are the result of HOSVD, where only zero singular values and the corresponding singular vectors are discarded. **Non-exact** reduced form $\hat{A} = A^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$, is obtained if not only zero singular values and the corresponding singular vectors are discarded.

The above properties directly lead to the following fundamental concept of.

Method 1. (exact SVD based fuzzy rule base reduction) The SVD based fuzzy rule base reduction transforms equation (5) to the form of:

$$f(x_1, x_2, \ldots, x_N) = \sum_{j_1, j_2, \ldots, j_N}^{J_1^r, J_2^r, \ldots, J_3^r} \prod_{n=1}^{N} \mu_{j_n,n}^r(x_n) b_{j_1 j_2 \ldots j_N}^r, \tag{8}$$

where $\forall n : J_n^r \leq J_n$ is obtained as the main essence of the reduction.

The reduced form is obtained via HOSVD capable of decomposing B into $B = B^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n$. Having $B^r \in \Re^{J_1^r \times J_2^r \ldots \times J_N^r}$ and its singular vectors the reduced form is determined as: $f(x_1, x_2, \ldots, x_N) = B^r \overset{N}{\underset{n=1}{\otimes}} \mathbf{m}_n^r$, where $\mathbf{m}_n^r = \mathbf{m}_n \mathbf{U}_n$. Equation (5) is an equivalent of (8) that is the starting point for theoretical developments of this topic.

Remark 1. Note that, the obtained functions may not be interpretable as antecedent fuzzy sets. In order to obtain functions which can be antecedent fuzzy stets, further to have *Ruspini* partition, sum-normalization (SN), nonnegativenes (NN) and normality (NO) transformation techniques are developed to HOSVD algorithm in [3], [4], [13].

Remark 2. The error controllable advantage of the reduction technique is conceptually obtained by the truncation of non-zero singular values. The error bound of $\hat{f}(x_1, x_2, \ldots, x_N)$ can be estimated during the execution of the SVD reduction algorithm. Note that, the final error of $\hat{f}(x_1, x_2, \ldots, x_N)$ depends on the type of the antecedent functions applied. Typical practical cases are analysed in [4].

4.3 Adaptation of SVD based Approximation

One of the natural problems of any complexity reduction technique is that the adaptivity property of the reduced approximation algorithm becomes highly restricted. Since the crucial concept of the reinforcement learning is based on the adaptivity of the action-value function, in this paper the "fast adaptation of SVD based approximation" (introduced in [26]) is suggested to adopt for reinforcement learning. This fast adaptation method, directly adapts the reduced approximation by replacing, or embedding new approximation points. The ability of embedding new approximation points provides the practical applicability of the proposed dynamic partition allocation method.

Therefore, the application of the fast adaptation method in the proposed reinforcement learning structure is twofold. On one side, it helps the dynamic partition allocation by increasing the rule density. On the other side, by the replacement of the previously fetched and modified values serves the adaptation of the approximated action-value function.

Consequently the key idea of the proposed reinforcement algorithm is to insert a set of new rules, for instance A, into the existing rule base B. As it is already discussed, in order to avoid complexity problems, the reduced form of B (namely B^r) is utilised. This, hence, means that the embedding of the new rules contained in A should directly performed on B^r. One more important constraint should be emphasised here. In order to fix the complexity of the rule base only those in formation of the new rules should be inserted into B^r which do not increase the size of B^r. Actually this is equivalent to the key idea of the fast adaptation introduced in [26]. More precisely in regard of the following algorithm, only those sub-tensors of A are embedded into B^r, which are linearly dependent from B^r [26]. Since the elements in B^r are fixed, no SVD is needed during embedding, which offers a chance to develop a fast algorithm to adapt HOSVD. Here an elementary step of the idea is discussed when the rule base is being increased in an arbitrary dimension n.

Method 2. (n mode fast adaptation [26]) Assume a reduced rule base defined by tensor $B^r \in \Re^{J_1^r \times J_2^r \times \ldots \times J_N^r}$ and its corresponding matrices $\mathbf{Z}_n \in \Re^{J_n \times J_n^r}$ resulted from B by HOSVD. Furthermore, let $A \in \Re^{J_1 \times J_2 \times \ldots J_{n-1} \times I \times J_{n+1} \times \ldots \times J_N}$ be given, that has the same size as B except in the n-th dimension where I may differ from J_n. The localized error threshold of the adaptation is defined by ∇.

The goal is to determine the reduced form E^r of extended rule base E defined by tensor $E' = \left[B \quad A' \right]_n$, where E' contains the selected n mode sub-tensors of E according to the given error threshold ∇ as

$$\hat{E}' = \left(B^r \overset{N}{\underset{k=1}{\otimes}} \mathbf{Z}_k \right) \times_n \mathbf{U}, \tag{9}$$

and $A' \in \Re^{J_1 \times J_2 \times \ldots J_{n-1} \times I' \times J_{n+1} \times \ldots \times J_N}$ contains the selected n mode sub-tensors of A and lets the corresponding sub-tensors $T'_{\min/\max}$ selected from the corresponding $T_{\min/\max}$. For brevity let $\nabla' = [T'_{\min} \quad T'_{\max}]$. $\mathbf{U} = [\mathbf{Z}_n \quad \mathbf{V}] \in \Re^{(J_n + I') \times J_n^r}$, $I' \le I$, where $\mathbf{V}$ is determined to fulfil (9) subject to $\hat{E}' - E' \in_t \nabla'$.

Method 2 has built-up from the following two algorithms:

Algorithm: *(n-mode high order d way defective projection to a given basis [26])* This algorithm defectively projects, according to threshold ∇, a given tensor $A \in \Re^{J_1 \times J_2 \times \ldots J_{n-1} \times I \times J_{n+1} \times \ldots \times J_N}$ in dimension n to basis $\mathbf{Z}_n \in \Re^{I \times J_n^r}$. The result is $\hat{A}' = A^p \times_n \mathbf{Z}_n$, where $A' \in \Re^{J_1 \times J_2 \times \ldots J_{d-1} \times J'_d \times J_{d+1} \times \ldots \times J_N}$, which may be defective in the specified dimension d, $J'_d \le J_d$, and consists of selected d-mode sub-tensors of A. The projection is done by the above defined defective matrix projection to yield: $\mathbf{Z}_n A^p_{(n)} - A'_{(n)} \in_t \nabla'$ under the condition of projection as $\mathbf{P}_n A'_{(n)} - A'_{(n)} \in_t \nabla'_{(n)}$, where $\mathbf{P}_n = \mathbf{Z}_n \cdot \mathbf{Z}_n^+$. From the point of calculation, the condition is actually checked vector by vector as: $\mathbf{P}_n (A'_{(n)})_{i'} - (A'_{(n)})_{i'} \in_t (\nabla'_{(n)})_{i'}$. Important step is here, in high order case, that not only those vectors are ignored which do not satisfy the above condition of projection, but all vectors contained in a d-mode sub-tensor of A, where at least one vector exists not holding the condition of projection. The resulted $A^p_{(n)}$ projected from the remaining vectors can be restored to be tensor A^p. The size of A' may differ from A only in dimension d. The size of A' can be defined by removing the cancelled sub-tensors from A. $\mathbf{P}_n (A'_{(n)})_{i'} - (A'_{(n)})_{i'} = (H'_{(n)})_{i'}$ can be restored to be tensor H', like in the case of A'. Let tensor H_n with the size of A be generated via extending H' with zeros and be called as n mode projection error.

Having the above Algorithm, the first step of Method 2 is the following: Repeating the k-mode high order n-way defective projection to basis $\mathbf{Z}_k \in \Re^{J_k \times J_k^r}$ which results in $\hat{A}' = A^p \overset{N}{\underset{k=1}{\otimes}} \mathbf{Z}_k$, where $A' \in \Re^{J_1 \times J_2 \times \ldots J_{n-1} \times I' \times J_{n+1} \times \ldots \times J_N}$ and n-mode sub-tensors of A' are selected from among the n-mode sub-tensors of A. The cumulated error obtained by the defective matrix projection in each k-th step is $S'_k = \sum_k \left(H'_k \overset{k-1}{\underset{h=1, h \ne n}{\otimes}} \mathbf{Z}_k \right)$. As a matter of fact, the tolerance criteria of the condition of projection should be corrected in each step by S_k. The last step of Method 2 is the inverse of the previous step:

Algorithm: *(n-mode high order defective basis [26])* Assume given tensor B^r with $\mathbf{Z}_k \in \Re^{J_k \times J_k^r}$ as the reduced form of B by HOSVD; corrected error threshold ∇ and $A^p \in \Re^{J_1^r \times J_2^r \times \ldots J_{n-1}^r \times I' \times J_{n+1}^r \times \ldots \times J_N^r}$ as a result of the first step of Method 2. This algorithm determines defective basis $\mathbf{U} \in \Re^{(J_n + I'') \times J_n^r}$ in such a way that $\left[\left(B^r \overset{N}{\underset{k=1}{\otimes}} \mathbf{Z}_k \right) \times_n \mathbf{U} - E'' \right] \in_t \nabla''$, where $E'' = [B \quad A'']_n$. In other words $\mathbf{V} \in \Re^{I'' \times J_n^r}$ is determined

here which fulfills: $\left(\left(B^r \overset{N}{\underset{k=1}{\otimes}} \mathbf{Z}_k\right)\times_n \mathbf{V} - A''\right) \in_t \nabla''$, where A'' is created from A' according to the condition of projection like A' is created from A. Applying n-order defective projection of $(A^r_{(n)})^{\mathrm{T}}$ to basis $(B^r_{(n)})^{\mathrm{T}}$ accordingly to error of $T'-S'_n$ to find $(\hat{A}^r_{(n)})^{\mathrm{T}} = (B^r_{(n)})^{\mathrm{T}} \cdot \mathbf{V}^{\mathrm{T}}$ leads to the solution where the rows of $A^r_{(n)}$ are selected from among the rows of $A^P_{(n)}$. Consequently, $\mathbf{V}$ is obtained that shows the linear dependence between the rest of A^p and B^r. The final step is, hence, to fit the new basis as: $\mathbf{U} = \begin{bmatrix} \mathbf{Z}_n \\ \mathbf{V} \end{bmatrix}$. The required antecedent sets are defined as: $\mathbf{s}^r_k = \mathbf{s}_k \mathbf{Z}_k$ and the new basis is $\mathbf{s}^r_n = \mathbf{s}_n \mathbf{U}$.

The more detailed description of the fast adaptation algorithm is given in [26]

5 Practical use of the Proposed Reinforcement Technique

For introducing the proposed application way of SVD based fuzzy rule based approximation techniques in reinforcement learning, two simple application examples were chosen.

The first is a simple, for the sake of visualization of the action value (Q) function, a one dimensional state-space system characterized by the following state-transition function (10):

$$s^{k+1} = 2 \cdot \left(s^k + a^k\right), \tag{10}$$

where $s \in S = [-1,1]$ is the one dimensional state and $a \in A = [-0.2,0.2]$ is the action.

The reward is calculated in the following manner:

$$r = 1 \ \textit{iff} \ \ s \in [-0.1,0.1] \ \textit{else} \ \ r = 0 \tag{11}$$

The second example is the well known cart-pole balancing application characterized by the following state-transition functions (12,13):

$$\ddot{x} = \frac{f - b_C \cdot \dot{x} + m \cdot l \cdot \sin(\varphi) \cdot \dot{\varphi}^2 - m \cdot g \cdot \cos(\varphi) \cdot \sin(\varphi) + b_P \cdot \dot{\varphi} \cdot \cos(\varphi)/l}{M + m - m \cdot \cos(\varphi)^2} \tag{12}$$

$$\ddot{\varphi} = \frac{(f - b_C \cdot \dot{x}) \cdot \cos(\varphi) - (M+m) \cdot (g \cdot \sin(\varphi) - b_P \cdot \dot{\varphi}/(l \cdot m)) + m \cdot l \cdot \cos(\varphi) \cdot \sin(\varphi) \cdot \dot{\varphi}^2}{m \cdot l \cdot \cos(\varphi)^2 - (M+m) \cdot l} \tag{13}$$

where x is the position of the cart, φ is the angle of the pole, f is the actuating force, m is the mass of the pole, l is the length of the pole, M is the mass of the cart, g is the gravity acceleration, b_C is the friction coefficient of the cart, b_P is the friction coefficient of the pole.

The reward in the cart-pole balancing example is calculated in the following manner:

$$r = 1 \ \textit{iff} \ \ \varphi \in [-0.0025 \cdot \pi, 0.0025 \cdot \pi] \ \textit{else} \ \ r = 0 \tag{14}$$

The first experiment is related to the efficiency of the proposed dynamic partition allocation method, and based on the first application example (see results on fig.5). Fig.5.b and Fig.5.c are introducing the two basic problems of fixed partition. The lack of universal approximator property in case of rough partition (e.g. on Fig.5.b) and the difficulties of adaptivity (e.g. on Fig.5.c).

The second experiment is related to the efficiency of the proposed SVD based complexity reduction and approximation adaptation (fast adaptation method). This experiment is based on the first and second application example (see results on fig.6. and fig.7.). In case of the first example application fig.6.a.

introduces five stages of a 20000 step iteration. On fig.6.b. the same iteration process turns the action-value rule base to reduced form at the iteration step 1000, by applying the *Method 1* of section 4.2. From this step the iteration is continuing up to 20000 iterations using the fast adaptation method (*Method 2* of section 4.3.). Fig.6.c. is the same experiment as fig.6.b., except the turning the reduction is done at the step 5000.

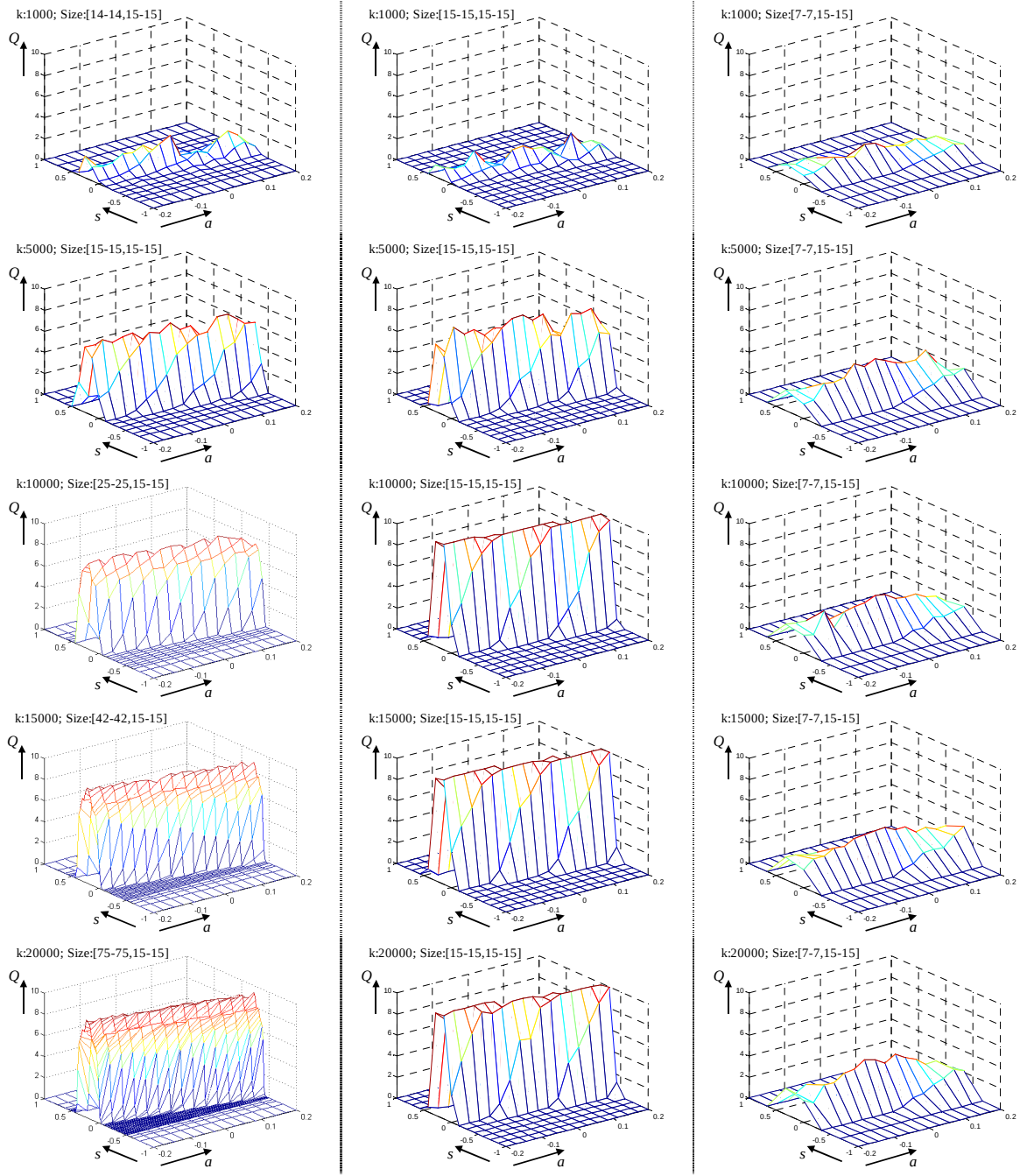
Performing the second experiment on the second example application gives very similar results. The main difference is the poor scaling of the universes. Most of the reinforcements are gained in a very small area of the state space. Fixed partition state descriptions could have difficulties in case of inappropriate universe scaling, while the proposed dynamic partition allocation method simply overcome the situation.

As the main conclusion of the second experiment, it seems that in many cases the action-value function is considerably reducible. Moreover due to the fast adaptation method this reduction can be performed in an early stage of the adaptation and the iteration steps can be continued on an economic sized structure.

6 Conclusions

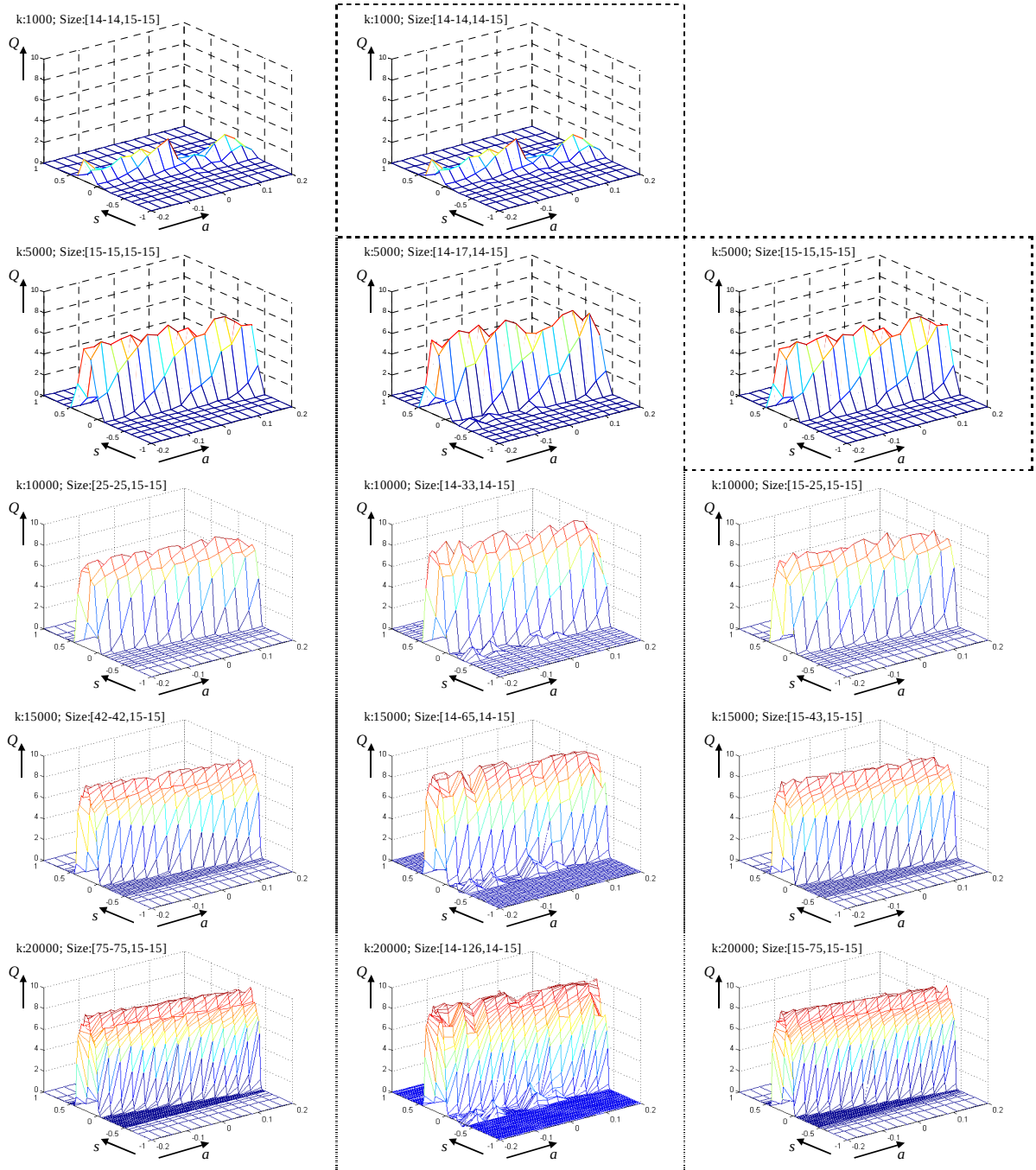
One of the possible difficulties of the reinforcement learning applications in complex situations, is the huge size of the state-value- or action-value-function representation [2]. The case of continuous environment reinforcement learning could be even complicated, in case of applying dense partitions to describe the continuous universes, to achieve precise approximation of the basically unknown state-value- or action-value-function. The fine resolution of the partitions leads to high number of states, and handling high number of states usually leads to high computational costs, which could be unacceptable not only in many real time applications, but in case of any real (limited) computational resource. As a simple solution of these problems, in this paper the adoption of Higher Order SVD [13] based fuzzy rule base complexity reduction techniques and its fast adaptation method [26] is suggested. The application of the fast adaptation method [26] gives a simple way for increasing the rule density of a rule base stored in a compressed form directly. To fully exploit this feature, a dynamic partition allocation method is also suggested.

Based on the application examples, the main conclusion of this paper is the reducibility of action-value function. It seems that in many cases the representation of the action-value function is considerably reducible. Moreover due to the fast adaptation method this reduction can be performed in an early stage of the adaptation and the iteration steps can be continued on an economic sized action-value function representation.



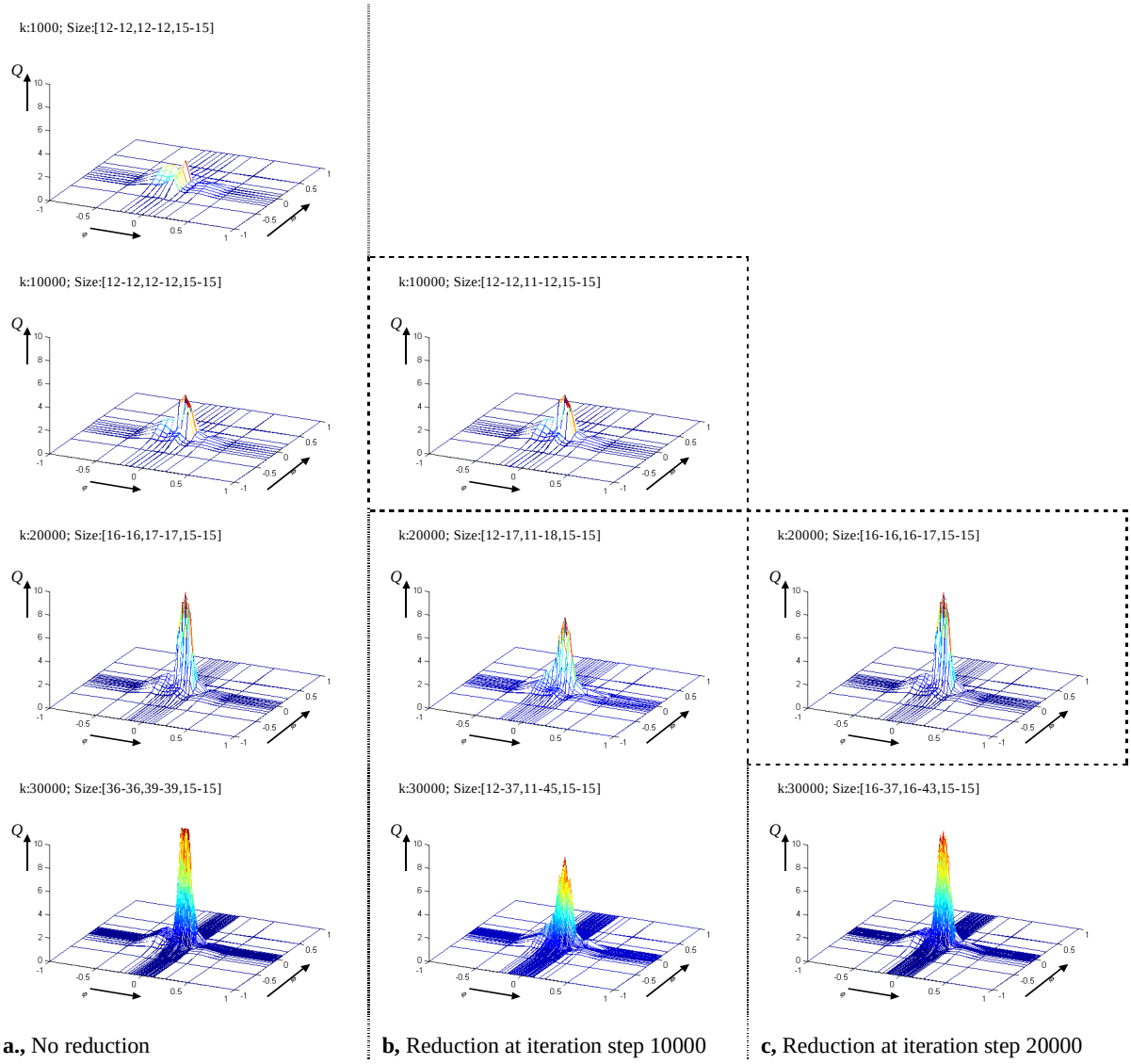
a., Dynamic partition allocation **b.,** Fixed, 15 equidistant set partition **c.,** Fixed, 7 equidistant set partition

Fig. 5. First application; dynamic and fixed partition allocation (k is the iteration number, Size: [*S*,*A*] set numbers)



a., No reduction **b,** Reduction at iteration step 1000 **c,** Reduction at iteration step 5000

Fig. 6. First application; the effect of SVD based complexity reduction and approximation adaptation, where k is the iteration number and Size is the size of the reduced (B^r as it is stored) and the extended (B as its used) action-value rule base (e.g. Size:[14-126,14-15] means, that the original 126x15 sized action value rule base is stored and adapted in a 14x14 reduced format).



a., No reduction **b,** Reduction at iteration step 10000 **c,** Reduction at iteration step 20000

Fig. 7. Second application; the effect of SVD based complexity reduction and approximation adaptation, where k is the iteration number and Size is the size of the reduced (B^r as it is stored) and the extended (B as its used) action-value rule base (e.g. Size:[12-37,11-45,15-15] means, that the original 37x45x15 sized action value rule base is stored and adapted in a 12x11x15 reduced format).

Acknowledgement

This research was partly supported by the Hungarian National Scientific Research Fund grant no: F 029904.

References

1. Bellman, R. E.: Dynamic Programming. Princeton University Press, Princeton, NJ (1957)
2. Sutton, R. S., A. G. Barto: Reinforcement Learning: An Introduction, MIT Press, Cambridge (1998)
3. Yam, Y., Baranyi, P., Yang, C. T.: Reduction of Fuzzy Rule Base Via Singular Value Decomposition. IEEE Transaction on Fuzzy Systems. Vol.: 7, No. 2 (1999) 120-131.

4. Baranyi, P., Yam, Y.: Fuzzy rule base reduction. Chapter 7 of Fuzzy IF-THEN Rules in Computational Intelligence: Theory and Applications Eds., D. Ruan and E.E. Kerre, Kluwer (2000) 135-160
5. Watkins, C. J. C. H.: Learning from Delayed Rewards. Ph.D. thesis, Cambridge University, Cambridge, England (1989)
6. Appl, M.: Model-based Reinforcement Learning in Continuous Environments. Ph.D. thesis, Technical University of München, München, Germany, dissertation.de, Verlag im Internet (2000)
7. Wang, L.X.: Fuzzy Systems are Universal Approximators. Proceedings of the First IEEE Conference on Fuzzy Systems, San Diego (1992) 1163-1169
8. Castro, J.L.: Fuzzy Logic Controllers are Universal Approximators. IEEE Transaction on SMC, Vol.25, 4 (1995)
9. Horiuchi, T., Fujino, A., Katai, O., Sawaragi, T.: Fuzzy Interpolation-Based Q-learning with Continuous States and Actions. Proc. of the 5th IEEE International Conference on Fuzzy Systems, Vol.1 (1996) 594-600
10. Glorennec, P.Y., Jouffe, L.: Fuzzy Q-Learning. Proc. of the 6th IEEE International Conference on Fuzzy Systems (1997) 659-662
11. Berenji, H.R.: Fuzzy Q-Learning for Generalization of Reinforcement Learning. Proc. of the 5th IEEE International Conference on Fuzzy Systems (1996) 2208-2214
12. Bonarini, A.: Delayed Reinforcement, Fuzzy Q-Learning and Fuzzy Logic Controllers. In Herrera, F., Verdegay, J. L. (Eds.) Genetic Algorithms and Soft Computing, (Studies in Fuzziness, 8), Physica-Verlag, Berlin, D, (1996) 447-466
13. Baranyi, P., Várkonyi-Kóczy, A., Yam, Y., Patton, R.J., Michelberger, P., Sugiyama M.:SVD Based Reduction of TS Models. IEEE Trans. Industrial Electronics (accepted with minor revision)
14. Yen, J., Wang, L.: Simplifying Fuzzy Rule-based Models Using Orthogonal Transformation Methods. IEEE Trans. SMC, Vol 29: Part B, No. 1 (1999) 13-24
15. Yam, Y.: Fuzzy approximation via grid point sampling and singular value decomposition. IEEE Trans. SMC, Vol. 27 (1997) 933-951
16. Kóczy, L.T., Hirota, K.: Size Reduction by Interpolation in Fuzzy Rule Bases. IEEE Trans. SMC, vol. 27 (1997) 14-25
17. Tikk, D.: On nowhere denseness of certain fuzzy controllers containing prerestricted number of rules. Tatra Mountains Mathematical Publications vol. 16. (1999) 369-377
18. Lei, K., Baranyi, P., Yam, Y.: Complexity Minimalisation of Non-singleton Based Fuzzy-Neural Network. International Journal of Advanced Computational Intelligence, vol.4, no.4 (2000) 1-8
19. Baranyi, P., Yam, Y., Várlaki, P., Michelberger, P.: Singular Value Decomposition of Linguistically Defined Relations. Int. Jour. Fuzzy Systems, Vol. 2, No. 2, June (2000) 108-116
20. Song, F., Smith, S.M.: A Simple Based Fuzzy Logic Controller Rule Base Reduction Method. IEEE Int. Conf. System Man and Cybernetics (IEEE SMC' 2000), Nashville, Tennessee, USA (2000) 3794-3798
21. Setnes, M., Hellendoorn, H.: Orthogonal Transforms for Ordering and Reduction of Fuzzy Rules. 9th IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE 2000), San Antonio, Texas (2000) 700-705
22. Sudkamp, T., Knapp, A., Knapp J.: A Greedy Approach to Rule Reduction in Fuzzy Models. IEEE Int. Conf. System Man and Cybernetics (IEEE SMC'2000), Nashville, Tennessee, USA (2000) 3716-3721
23. Baranyi, P., Yam, Y., Yang, C.T., Várlaki, P., Michelberger, P.: Generalised SVD Fuzzy Rule Base Complexity Reduction. International Journal of Advanced Computational Intelligence (accepted, to be printed in 2001)
24. Baranyi, P., Yam, Y.: Singular Value-Based Approximation with Non-Singleton Fuzzy Rule Base. 7th Int. Fuzzy Systems Association World Congress (IFSA'97) Prague (1997) 127-132
25. Baranyi, P., Yam, Y.: Singular Value-Based Approximation with Takagi-Sugeno Type Fuzzy Rule Base. 6th IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE'97) Barcelona, Spain (1997) 265-270
26. Baranyi, P., Várkonyi-Kóczy, A.R., Yam, Y., Várlaki, P., Michelberger, P.: An Adaption Technique to SVD Reduced Rule Bases. IFSA 2001, Vancouver (accepted for presentation)