Proceedings of the 2007 IEEE Symposium on Approximate
Dynamic Programming and Reinforcement Learning (ADPRL 2007)

Continuous-Time ADP for Linear Systems with Partially Unknown Dynamics

Draguna Vrabie, Murad Abu-Khalaf, Frank L. Lewis, Youyi Wang

Abstract—**Approximate Dynamic Programming has been formulated and applied mainly to discrete-time systems. Expressing the ADP concept for continuous-time systems raises difficult issues related to sampling time and system model knowledge requirements. In this paper is presented a novel online adaptive critic (AC) scheme, based on approximate dynamic programming (ADP), to solve the infinite horizon optimal control problem for continuous-time dynamical systems; thus bringing together concepts from the fields of computational intelligence and control theory. Only partial knowledge about the system model is used, as knowledge about the plant internal dynamics is not needed. The method is thus useful to determine the optimal controller for plants with partially unknown dynamics. It is shown that the proposed iterative ADP algorithm is in fact a Quasi-Newton method to solve the underlying Algebraic Riccati Equation (ARE) of the optimal control problem. An initial gain that determines a stabilizing control policy is not required. In control theory terms, in this paper is developed a direct adaptive control algorithm for obtaining the *optimal* control solution *without knowing the system A matrix.***

Index Terms—**Approximate Dynamic Programming, Adaptive Critics, Policy iterations, V-learning.**

I. INTRODUCTION

THE research in ADP, having its roots in computational intelligence, has traditionally been focused mainly to discrete-time systems. For continuous-time systems there are difficult issues related to sampling times, and requirements for knowing the system dynamical equations. It is known, for instance, that as sampling times become small, traditional discrete-time ADP does not lead to the optimal control solution for continuous-time systems [1].

In this paper, bringing together concepts from ADP and control systems theory, we develop a new ADP technique which offers solution, obtained in a forward-in-time fashion, to the continuous-time infinite horizon optimal control problem for linear systems with partially unknown dynamics (i.e. the internal dynamics, specified by the system matrix *A*).

ADP combines reinforcement learning designs with dynamic programming to determine the solution of the optimal control problem using a forward-in-time computation. Reinforcement learning techniques, namely Adaptive Critics, were first proposed by Werbos [15]. These methods consist in an iterative process of updating the control policy and value function estimate in order to bring them closer to the optimal control policy and the corresponding optimal value function. Each iteration step consists of an update of the value function estimate based on the current control policy, followed by a greedy update of the control policy based on the new value function estimation.

Initially developed for systems with finite state and action spaces these methods were based on Sutton's temporal difference method [13], Watkins's Q-learning [14] and Werbos's Heuristic Dynamic Programming (HDP) [16]. For the case of discrete-time systems with continuous state and action spaces different adaptive critic architectures were reported, with successful implementations and rigorous proofs; an incomplete list being [6],[8],[10],[11].

However, for the continuous-time case only little progress has been achieved. A dynamic programming-based reinforcement learning scheme, formulated using a so-called Advantage function, was introduced by Baird in [1]. Doya proposed in [5] reinforcement learning techniques based on the temporal difference method. The current status of work in ADP is given in [12]. However the equivalent ADP formulation, for continuous time and continuous state control systems, to the existent discrete time techniques is not straight forward. The main reason is that the optimal control solution for continuous-time systems cannot be obtained by simply reducing the value of the sampling time in the discrete-time formulation of ADP [1]. Secondly, unlike the discrete-time Hamiltonian, the Hamiltonian for continuous-time systems immediately involves the system dynamics, [9], which must therefore be known. This is making much more difficult the formulation of a mathematical approach which will not require the system model knowledge. Consequently, ADP is direct to formulate for discrete-time systems, but not straightforward to formulate for continuous-time systems with unknown dynamics.

Further investigation needs to be done in order to complete the ADP framework for continuous-time systems. This is even more important since it is known that it is impossible to exactly discretize continuous-time nonlinear systems; Euler approximations for discretization are not accurate enough in real-world control system applications. And because of this the

This work was supported by the National Science Foundation ECS-0501451 and the Army Research Office W91NF-05-1-0314.

D. Vrabie, M. Abu-Khalaf, F. L. Lewis are with the Automation and Robotics Research Institute at the University of Texas at Arlington, 7300 Jack Newell Blvd. S., Fort Worth, TX 76118-7115 USA (e-mail: {dvrabie, abukhalaf}@ arri.uta.edu, lewis@ uta.edu).

Y. Wang is with the School of Electrical and Electronic Engineering, Nanyang Technological University, No. 5 Engineering Drive 1, 117608, Singapore, Republic of Singapore (e-mail: EYYWANG@ntu.edu.sg).

system models or Jacobians that are required for the discrete-time ADP techniques, with the exception of Q-learning, cannot be computed.

In this paper we use concepts from both ADP and control systems theory to present a Continuous-Time approach to HDP (CT-HDP) for linear systems. That is, the adaptive critic is solving for the continuous-time version of the optimal value function - an approach also known as *V-learning*. The investigation of linear systems is relevant for real world control applications since, even though generally the system is nonlinear, the requirement is often that it has to be controlled for best performance around a specific operating point. At such point a linear model is regularly a good approximate description of the system. However, the exact values of the model parameters (i.e. matrix *A*) are not easy to find, requiring a tedious identification procedure, and also these values can drift over time. Systems that fit this description could be chemical plants, airplanes and power systems.

The CT-HDP algorithm is presented in the next section. A mathematical formulation of the algorithm is given proving the equivalence of the ADP iteration with a Quasi-Newton method. The algorithm is then tested in simulation and the optimal controller for a linear power system model is obtained without making use of any knowledge regarding the system matrix *A*.

II. CONTINUOUS-TIME ADAPTIVE CRITIC SOLUTION FOR THE INFINITE-HORIZON OPTIMAL CONTROL PROBLEM

A. Dynamic Programming and LQR

Consider the linear, time-invariant dynamical system given by

$$\dot{x} = Ax + Bu \tag{1}$$

with $x(t) \in R^n$, subject to the infinite-horizon optimal control problem

$$V(x_0) = \min_{u(t)} \int_0^{\infty} (x^T Q x + u^T R u) d\tau \tag{2}$$

with $Q \geq 0, R > 0$ and (A, B) controllable (i.e. the existence of a control signal that will determine a state transition path between any two points in the state space is guaranteed [4]).

It is known that the control solution of this problem, determined by Bellman's optimality principle, is given by $u = -Kx$ with

$$K = R^{-1} B^T P \tag{3}$$

where the matrix P is obtained by solving the Algebraic Riccati Equation (ARE)

$$A^T P + PA - PBR^{-1} B^T P + Q = 0\,. \tag{4}$$

The solution of the infinite horizon optimization problem can be obtained using the Dynamic Programming approach by solving a finite horizon optimization problem backwards in time and extend the horizon to infinity. In this case the following Riccati differential equation has to be solved

$$\begin{aligned} -\dot{P} &= A^T P + PA - PBR^{-1} B^T P + Q \\ P(t_f) &= P_{t_f} \end{aligned} \tag{5}$$

the solution of which will converge to the solution of the ARE for $t_f \to \infty$.

It should be noted that obtaining the Dynamic Programming solution of equation (4) requires complete knowledge of the model of the system, i.e. both A and B must be known.

Equation (4) can also be solved using policy iterations, i.e. Newton's method, provided that the iteration is initialized by a stable policy [6]. The resulting iterative algorithm is the following:

$$\begin{aligned} 0 &= (A - BK_i)^T P_{i+1} + P_{i+1}(A - BK_i) + Q + K_i^T R K_i \\ K_i &= R^{-1} B^T P_i \end{aligned} \tag{6}$$

The algorithm (6) can be compactly written as

$$\begin{aligned} 0 = {} & (A - BR^{-1} B^T P_i)^T P_{i+1} + P_{i+1}(A - BR^{-1} B^T P_i) + \\ & + Q + P_i B R^{-1} B^T P_i \end{aligned} \tag{7}$$

such that P_{i+1} is the solution of the Lyapunov equation

$$A_i^T P_{i+1} + P_{i+1} A_i = -Q - P_i B R^{-1} B^T P_i \tag{8}$$

where $A_i = A - BR^{-1} B^T P_i$.

Equation (7) can be expressed in a Newton's method-like setting as

$$P_{i+1} = P_i - (Ric'_{P_i})^{-1} Ric(P_i) \tag{9}$$

where

$$Ric(P_i) = A^T P_i + P_i A + Q - P_i B R^{-1} B^T P_i \tag{10}$$

and Ric'_{P_i} denotes the Frechet derivative of $Ric(P_i)$ taken with respect to P_i. The matrix function Ric'_{P_i} evaluated at a given matrix *M* will thus be $Ric'_{P_i}(M) = A_i^T M + M A_i$.

Kleinman showed in [6] that if the initial policy is stabilizing then all the subsequent updated policies will be stabilizing and Newton's method will solve for the root of the quadratic matrix equation that will result in the stabilizing optimal policy.

B. Continuous-time ADP formulation

In the following we present a new ADP approach that allows one to recursively calculate the infinite horizon optimal cost, and solve the infinite horizon linear quadratic regulator problem (2), without having any knowledge about the plant internal dynamics, i.e. the *A* matrix need not be known (matrix *B* is required), and without starting with an initial stabilizing policy.

The infinite horizon cost of a policy is heuristically approximated as the summation of the observed reward for using a given control policy over a finite interval $[t, t+T]$ and an approximation of the cost from $t+T$ to ∞ which we denote with $W(t+T)$, the latter depending only on the observed new state $x(t+T)$:

$$V(x(t)) = \int_t^{t+T} (x^T Qx + u^T Ru) d\tau + W(x(t+T)) \tag{11}$$

Note that for any stable policy the associated infinite horizon cost is given as

$$\begin{aligned} V(x(t)) &= \int_t^{\infty} (x^T Qx + u^T Ru) d\tau \\ &= \int_t^{t+T} (x^T Qx + u^T Ru) d\tau + V(x(t+T)) \end{aligned} \tag{12}$$

where $V(x(t))$ serves as a Lyapunov function. If $u = -Kx$ and K is given by (3), then $V(x(t)) = x^T Px$ is the optimal cost and P satisfies (4).

Based on equation (11) the following Greedy iteration scheme may be implemented online

$$V_{i+1}(x(t)) = \int_t^{t+T} (x^T Qx + u_i^T Ru_i) d\tau + V_i(x(t+T)) \tag{13}$$

$$u_i = -R^{-1}B^T P_i x = K_i x \tag{14}$$

with the V-function parameterized as $V_i(x) = x^T P_i x$. The approximated value of the cost from $t+T$ to ∞ (denoted in (11) with $W(t+T)$) at the i-th iteration step can be calculated as $V_i(x(t+T)) = x^T(t+T)P_i x(t+T)$, since the matrix P_i gives the most recent characterization of the infinite horizon cost. Thus, (13) can be explicitly rewritten in parametric form as

$$x^T(t)P_{i+1}x(t) = \int_t^{t+T} (x^T Qx + u_i^T Ru_i) d\tau + x^T(t+T)P_i x(t+T) \tag{15}$$

and the ADP value function update amounts to the update of the kernel matrix P_i.

A restriction on the initial matrix P_0 such that the corresponding K_0 be a stabilizing controller is not required. Equations (15) and (14) formulate a new ADP scheme for continuous-time systems, motivated by the work of Murray et al. in [11]. The algorithm presented in [11] is iterating on Lyapunov equations using the measurement of the infinite horizon control cost at each iteration step, this requiring stabilizing control policy at all times. This algorithm avoids the use of the A matrix by measuring not only the system states but also their derivative.

C. Online tuning based on V-learning algorithm for partially unknown systems

For the implementation of the iteration scheme given by (15) and (14) one only needs to have knowledge of the B matrix. The information on the A matrix of the system is embedded in the states $x(t)$ and $x(t+T)$ which are observed online.

To find the parameters of V_{i+1} in (15), the left-hand side of (15) is written as

$$V_{i+1}(\bar{x}(t), \bar{p}_{i+1}) = x^T(t)P_{i+1}x(t) = \bar{p}_{i+1}{}^T \bar{x}(t) \tag{16}$$

where $\bar{x}(t)$ is the Kronecker product quadratic polynomial basis vector with the elements $\{x_i(t)x_j(t)\}_{i=1,n;\, j=i,n}$ and $\bar{p} = \nu(P)$ with $\nu(.)$ a vector valued matrix function that acts on $n \times n$ matrices and gives a column vector by stacking the elements of the symmetric matrix into a vector with the off-diagonal elements summed as $P_{ij} + P_{ji}$, [3].

The right-hand side of (15), using (14), is

$$d(x(t), P_i) = \int_t^{t+T} x(\tau)^T (Q + P_i BR^{-1}B^T P_i) x(\tau) d\tau + \bar{p}_i{}^T \bar{x}(t+T) \tag{17}$$

Equating (16) and (17), (15) and (14) can be written as

$$\bar{p}_{i+1}{}^T \bar{x}(t) = \int_t^{t+T} x(\tau)^T (Q + P_i BR^{-1}B^T P_i) x(\tau) d\tau + \bar{p}_i{}^T \bar{x}(t+T) \tag{18}$$

At each iteration step, after a sufficient number of state-trajectory points are collected using the same control policy K_i, a least-squares method is employed to solve for the V-function parameters, $\bar{p}_{i+1}$, which will then yield P_{i+1}. The parameter vector $\bar{p}_{i+1}$ is found by minimizing, in the least-squares sense, the error between the target function given by (17) and the parameterized relation (16) over a compact set $\Omega \subset R^n$. Evaluating (18) at $N \ge n(n+1)/2$ points $\bar{x}^i$ in the state space, the least-squares solution is obtained as

$$\bar{p}_{i+1} = (XX^T)^{-1} XY \tag{19}$$

where

$$X = [\bar{x}^1 \quad \bar{x}^2 \quad \ldots \quad \bar{x}^N]$$

$$Y = [d(\bar{x}^1, P_i) \quad d(\bar{x}^2, P_i) \quad \ldots \quad d(\bar{x}^N, P_i)]^T .$$

To obtain a solution for the least-squares problem (19) one requires at least $N = n(n+1)/2$ points, which is the number of independent elements in the matrix P. The least-squares problem can be solved in real-time after a sufficient number of data points are collected along a single state trajectory. In practice, the matrix inversion in (19) is not performed, the solution of the equation being obtained using algorithms that involve techniques such as Gaussian elimination, backsubstitution, and Householder reflections. The solution of equation (18) can also be obtained using the Recursive Least Squares algorithm (RLS) in which case a persistence of excitation condition is required.

This procedure requires only measurements of the states at discrete moments in time, t and $t+T$, as well as knowledge of the observed reward over the sample time interval $[t, t+T]$

$$r(x, P_i) = \int_t^{t+T} x^T(\tau)(Q + P_i BR^{-1}B^T P_i) x(\tau) d\tau .$$

Therefore there is no required knowledge about the system A

matrix for the update of the critic or the action. The continuous-time ADP algorithm consists of the iteration between (19) and (14). However the B matrix is required for the update of the control policy (actor), using (14), and this makes the tuning algorithm only partially model free.

It has to be observed that the update of both the actor and the critic is performed at discrete moments in time. However, the control action (14) is a continuous-time control, with gain updated at the sample points. Moreover, the critic update is based on the observations of the continuous-time cost over a finite sample interval. As a result, the algorithm converges to the solution of the continuous-time optimal control problem.

D. Mathematical formulation of the ADP algorithm

In this section we analyze the proposed ADP algorithm to place it into the context of control system theory; this being required prior to acceptance by the control systems community or industry.

Lemma 1. The ADP iteration between (15) and (14) is equivalent to the Quasi-Newton method

$$P_{i+1} = P_i - (Ric'_{P_i})^{-1}\left(Ric(P_i) - e^{A_i T^T} Ric(P_i) e^{A_i T} \right). \quad (20)$$

Proof:

Differentiating (15) with respect to time one obtains

$$\begin{aligned}\dot{V}_{i+1}(x(t)) = &-x^T(t)Qx(t) - u_i^T(t)Ru_i(t) + \\ &+ x^T(t+T)Qx(t+T) + u_i^T(t+T)Ru_i(t+T) \\ &+ \dot{V}_i(x(t+T))\end{aligned} \quad (21)$$

which can be written as

$$\begin{aligned}&(A+BK_i)^T P_{i+1} + P_{i+1}(A+BK_i) + K_i^T RK_i + Q = \\ &e^{(A+BK_i)T^T}(A^T P_i + P_i A - P_i BR^{-1}B^T P_i + Q)e^{(A+BK_i)T}.\end{aligned} \quad (22)$$

Adding and subtracting $A_i^T P_i + P_i A_i$ and making use of (10), (22) becomes

$$A_i^T(P_{i+1} - P_i) + (P_{i+1} - P_i)A_i = -Ric(P_i) + e^{A_i T^T} Ric(P_i) e^{A_i T} \quad (23)$$

and can be written in a Quasi-Newton formulation as

$$P_{i+1} = P_i - (Ric'_{P_i})^{-1}\left(Ric(P_i) - e^{A_i T^T} Ric(P_i) e^{A_i T} \right). \quad ∎$$

Remark 1. If $A_i = A + BK_i$ is stable and $T \to \infty$ one may recover from (19) the standard Newton method, (9), to solve the ARE, for which Kleinman [7] proved convergence conditioned by an initial stabilizing control gain K_0. The last term, appearing in the formulation of the new ADP algorithm, avoids the need for an initial stabilizing gain.

Equations (15) and (14) can be written as

$$P_{i+1} = \int_0^T e^{A_i t^T}(Q + P_i BR^{-1}B^T P_i)e^{A_i t}dt + e^{A_i T^T} P_i e^{A_i T} \quad (24)$$

so that we obtain the next result.

Lemma 2. Iteration (24) is equivalent to

$$P_{i+1} = P_i + \int_0^T e^{A_i t^T} Ric(P_i) e^{A_i t} dt. \quad (25)$$

Proof:

From matrix calculus one may write

$$P_i - e^{A_i T^T} P_i e^{A_i T} = -\int_0^T e^{A_i t^T}(A_i^T P_i + P_i A_i)e^{A_i t}dt \quad (26)$$

From (24) and (26) follows (25). ∎

Note that (25) is just a different way of writing (19).

Remark 2. As $T \to 0$, (25) becomes

$$\begin{aligned}&\dot{P} = A^T P + PA - PBR^{-1}B^T P + Q \\ &P(0) = P_0\end{aligned} \quad (27)$$

which, compared with (5), is a *forward-in-time* computation of the ARE solution, with the terminal boundary condition considered at the starting time, $P_{t_f} = P_0$.

Remark 3. The term $e^{A_i T}$ is the discrete-time version, obtained for the sample time T, of the closed-loop system matrix A_i. Therefore (24) is the expression of a hybrid discrete-time/continuous-time Riccati equation recursion.

Lemma 3. Let the ADP algorithm converge so that $P_i \to P^*$. Then P^* satisfies $Ric(P^*) = 0$, i.e. P^* is the solution the continuous-time ARE.

Proof:

If $\{P_i\}$ converges, then taking the limit in (25),

$$\lim_{P_i \to P^*}(P_{i+1} - P_i) = \lim_{P_i \to P^*}\int_0^T e^{A_i t^T} Ric(P_i) e^{A_i t} dt. \quad (28)$$

This implies

$$\int_0^T e^{A^* t^T} Ric(P^*) e^{A^* t} dt = 0 \quad (29)$$

with $A^* = A - BR^{-1}B^T P^*$, and thus $Ric(P^*) = 0$. ∎

III. ONLINE CT ADP CONTROLLER DESIGN FOR POWER SYSTEMS

In this section the continuous-time V-learning ADP algorithm developed in this paper is used to determine an optimal controller for a power system.

A. Motivation and system model

Power systems are complex nonlinear systems. However during normal operation the system load, which gives the nonlinearity, has only small variations. As such, a linear model can be used to represent the system dynamics around an operating point specified by a constant load value. Although this assumption seems to have simplified the design problem of a load-frequency controller for the system, a problem rises from the fact that in an actual plant the parameter values are not precisely known. For this reason the ADP model free design technique based on V-learning is used in this section for the design of the optimal LQR controller for a given operating point of the system.

The model of the system that is considered here, [17], is

$$\dot{x} = Ax(t) + Bu(t) \tag{30}$$

where

$$x(t) = [\Delta f(t) \quad \Delta P_g(t) \quad \Delta X_g(t) \quad \Delta E(t)]^T$$

$$A = \begin{bmatrix} -1/T_p & K_p/T_p & 0 & 0 \\ 0 & -1/T_T & 1/T_T & 0 \\ -1/RT_G & 0 & -1/T_G & -1/T_G \\ K_E & 0 & 0 & 0 \end{bmatrix}$$

$$B^T = [0 \quad 0 \quad 1/T_G \quad 0]$$

The system states are: $\Delta f(t)$ - incremental frequency deviation (Hz), $\Delta P_g(t)$ - incremental change in generator output (p.u. MW), $\Delta X_g(t)$ - incremental change in governor position (p.u. MW), $\Delta E(t)$ - incremental change in integral control; and the system parameters are: T_G - the governor time constant, T_T - turbine time constant, T_P - plant model time constant, K_P - planet model gain, R - speed regulation due to governor action, K_E - integral control gain.

B. Simulation setup and results

In the simulation only the time constant T_G of the governor, which appears in the B matrix, is considered to be known, while the values for all the other parameters appearing in the system A matrix are not known.

The system parameters, necessary for simulating the system behavior are picked randomly before the simulation is started in some realistic ranges, as specified in [17], such that:

$1/T_p \in [0.033, 0.1]$

$K_p/T_p \in [4, 12]$

$1/T_T \in [2.564, 4.762]$

$1/T_G \in [9.615, 17.857]$

$1/RT_G \in [3.081, 10.639]$

Note however that, even if the parameters' values are chosen to be in the above mentioned ranges, the algorithm does not make use of any of this knowledge, only the exact value of T_G being necessary. Also, although the values of the controller parameters K_E and R are known, as they are specified by the engineer, this information is not used by the CT HDP algorithm to determine the optimal controller.

In the following there will be presented the results of an HDP experiment considering a randomly picked set of values (in the above mentioned ranges) for the systems unknown parameters, i.e. matrix A. In all the simulations the B matrix is $B = [0 \quad 0 \quad 13.7355 \quad 0]$ and it is considered to be known. For the purpose of demonstrating the CT-HDP algorithm the initial state of the system is taken different than zero, $x_0 = [0 \quad 0.1 \quad 0 \quad 0]$, and the matrix $P_0 = 0$.

The online implementation requires the setup of a least-squares problem of the kind presented in Section II-C to solve for the values of the critic parameters, the matrix P_i, at each iteration step i. In the simulations the matrix P_i is determined after collecting 12 points for each least-squares problem. Each such point is calculated after observing the value of the reward over a time interval of $T = 0.1\text{s}$. Therefore a least-squares problem is solved and the critic is updated at each 1.2 s. The simulations were performed over a time interval of 50 s. As such, a number of 41 iterations were performed during each simulation experiment.

For the simulation the unknown values of the system parameters were randomly picked in the specified ranges and the system matrix was

$$A = \begin{bmatrix} -0.0596 & 5.0811 & 0 & 0 \\ 0 & -3.0938 & 3.0938 & 0 \\ -10.0912 & 0 & -13.7355 & -13.7355 \\ 0.6 & 0 & 0 & 0 \end{bmatrix}.$$

The algorithm was run and at each iteration step a solution of (18), explicitly given by (19), was obtained. The convergence of few of the critic parameters $P(1,1)$, $P(1,3)$, $P(2,4)$ and $P(4,4)$ is presented in Fig.1.

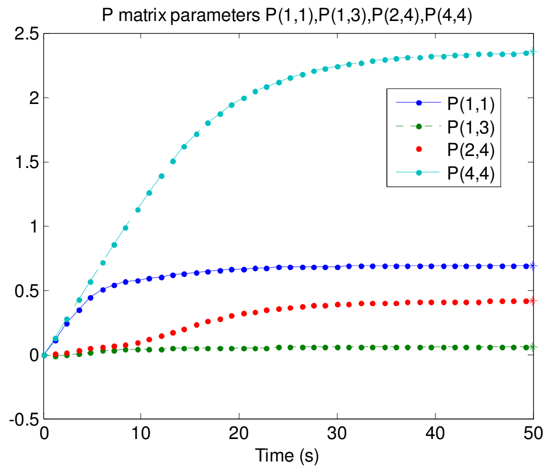

Fig.1 Convergence of P matrix parameters in online CT-HDP

The solution of the ARE (4) for this given matrix A and $Q = I_4, R = 1$ is

$$P = \begin{bmatrix} 0.6920 & 0.5388 & 0.0551 & 0.6398 \\ 0.5388 & 0.7361 & 0.1009 & 0.4173 \\ 0.0551 & 0.1009 & 0.0451 & 0.0302 \\ 0.6398 & 0.4173 & 0.0302 & 2.3550 \end{bmatrix}.$$

After 41 iteration steps the critic is characterized by

$$P_{41} = \begin{bmatrix} 0.6914 & 0.5381 & 0.0551 & 0.6371 \\ 0.5381 & 0.8922 & 0.1008 & 0.4144 \\ 0.0551 & 0.1008 & 0.0451 & 0.0299 \\ 0.6371 & 0.4144 & 0.0299 & 2.3442 \end{bmatrix}.$$

Comparing the values of the two matrices it can be noted that after 41 iteration steps the error between their parameters is of order 10^{-3}, i.e. the algorithm converged to the solution of the ARE.

In Fig. 2 is presented the evolution of the states of the system during the simulation. In Fig. 3 the system states are showed in detail during the first 6 seconds, i.e. the first 5 iteration steps of the simulation. The control signal that was applied to the system during the CT HDP tuning is presented in Fig. 4.

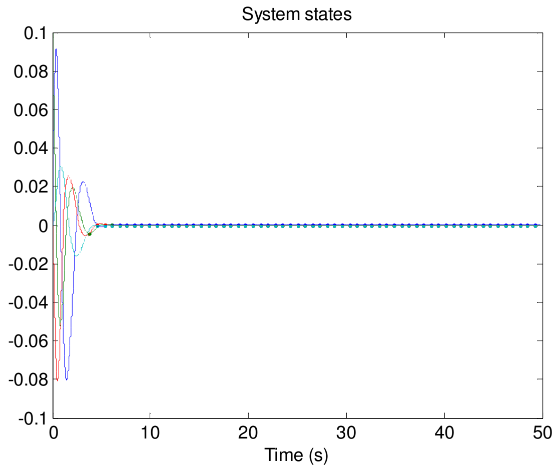

Fig.2 System states during the simulation

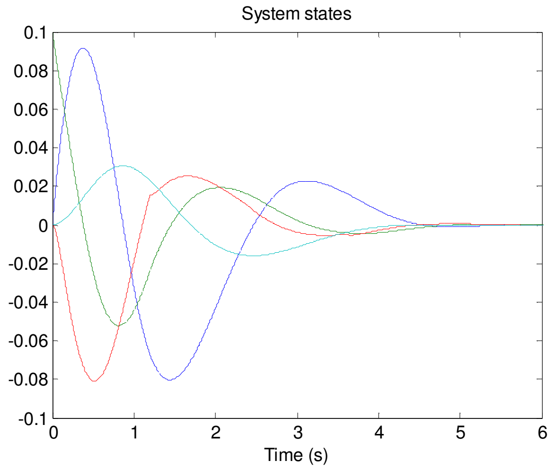

Fig.3 System states during the first 5 iteration steps

C. *Comments on the convergence of CT HDP algorithm*

The relation between the time period T over which the value function is observed at each step and the algorithm convergence is investigated in the following.

Fig. 5 shows the convergence of the critic parameters, in the case of the second simulation setup, when the time period is taken $T = 0.2\,\text{s}$. Over the 50 s duration of the simulation only 20 iterations are performed, the necessary data (12 points) for solving each least-squares problem being collected over an interval of 2.4 s.

By comparing the results plotted in Fig. 1 with the ones presented in Fig. 5 it becomes clear that the amount of time necessary for convergence is not dependent on the sample period that is used for observation. However, the number of iteration steps that are required for convergence is reduced when a large sample period is considered. The reason is that, in case a larger observation sample is used, an increased amount of information regarding the system is carried in the data points collected for the critic update. As such, at each iteration the critic improvement is larger when the time period is increased.

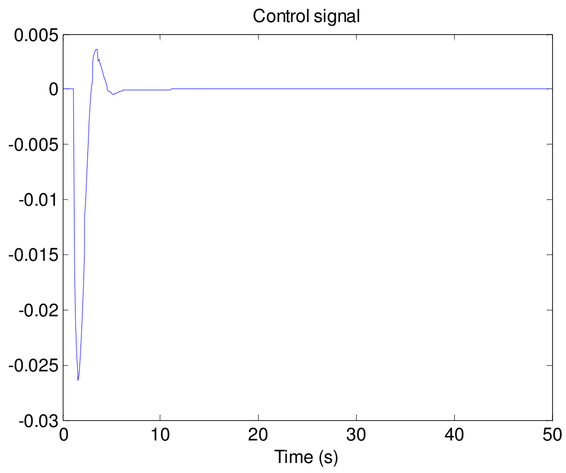

Fig.4 Control signal for simulation of online CT HDP

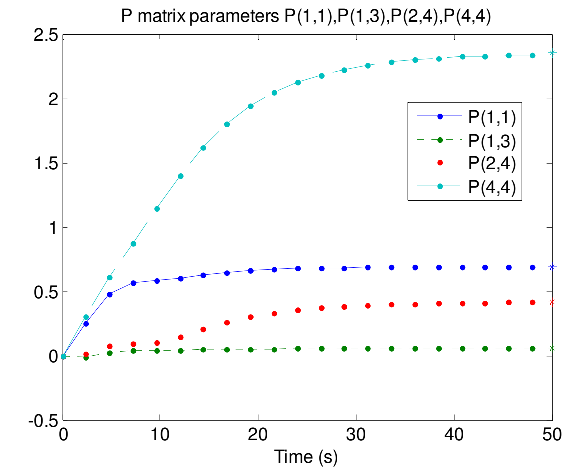

Fig.5 Convergence of P matrix parameters in online CT HDP for *T*=0.2s

IV. Conclusion

In this paper is presented a continuous-time ADP scheme which solves the continuous-time infinite horizon optimal control problem.

The control signal is applied to the system in a continuous time fashion. The actor's continuous time performance is measured over given time intervals and, based on this acquired information data, the critic reevaluates the infinite horizon cost and updates the actor's parameters (i.e. the continuous time system controller) in the sense of improving the over all system performance (i.e. to minimize the infinite horizon continuous time cost). As such, the system performance informational loop, which involves the critic entity, handles discrete information regarding the continuous time performance while the system control loop, which involves the actor, operates entirely in continuous time.

The proposed algorithm, equivalent to a Quasi-Newton method, solves the CARE and obtains the optimal controller in an online, forward in time iteration without using knowledge of the internal dynamics of the plant and without starting with an initial stabilizing policy.

REFERENCES

[1] Baird, L., “Reinforcement Learning in Continuous Time: Advantage Updating,” *Proceedings of the International Conference on Neural Networks,* Orlando, FL, June 1994.

[2] Bradtke S. J., B. E. Ydestie, A. G. Barto, “Adaptive linear quadratic control using policy iteration”, *Proceedings of the American Control Conference*, pp. 3475-3476, Baltmore, Myrland, June, 1994.

[3] Brewer J. W., “Kronecker Products and Matrix Calculus in System Theory”, *IEEE Trans. on Circuit and System*, Vol. CAS-25, No. 9, 1978.

[4] Callier, F. M., C. A. Desoer, *Linear Systems Theory*, Springer-Verlag, New York, 1991.

[5] Doya, K., “Reinforcement Learning in Continuous Time and Space,” *Neural Computation*, vol. 12, pp. 219-245, MIT Press, 2000.

[6] Ferrari, S., R. Stengel, “An Adaptive Critic Global Controller,” *Proceedings of the American Control Conference*, pp. 2665-2670, Anchorage, AK, 2002.

[7] Kleinman D., “On an Iterative Technique for Riccati Equation Computations”, *IEEE Trans. on Automatic Control*, February, 1968.

[8] Landelius, T., *Reinforcement Learning and Distributed Local Model Synthesis*, PhD Dissertation, Linköping University, 1997.

[9] Lewis, F. L., V. Syrmos, *Optimal Control*, John Willey, 2nd Edition, New York,1995.

[10] Liu, X., S. N. Balakrishnan, “Convergence Analysis of Adaptive Critic Based Optimal Control”, *Proceedings of the American Control Conference*, pp. 1929-1933, Chicago, IL, 2000.

[11] Murray J. J., C. J. Cox, G. G. Lendaris, and R. Saeks, “Adaptive Dynamic Programming”, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 32, No. 2, pp 140-153, 2002.

[12] Si J., A. Barto, W. Powel, D. Wunch, *Handbook of Learning and Approximate Dynamic Programming*, John Wiley, New Jersey, 2004.

[13] Sutton, R., “Learning to predict by the method of temporal differences,” *Machine Learning*, 3:9-44, 1988.

[14] Watkins, C., *Learning from Delayed Rewards*, Ph.D. Thesis, Cambridge University, Cambridge, England, 1989.

[15] Werbos, P., *Beyond Regression: New Tools for Prediction and Analysis in the Behavior Sciences*, Ph.D. Thesis, Committee on Appl. Math. Harvard Univ., 1974.

[16] White, D., D. Sofge, Eds., Handbook of Intelligent Control, Neural, Fuzzy, and, Adaptive Approaches, New York: Van Nostrand, 1992.

[17] Wang, Y., R. Zhou, C. Wen, “Robust load-frequency controller design for power systems”, IEE Proc.-C, Vol. 140, No. I , 1993.