RLS Algorithms and Convergence Analysis Method for Online DLQR Control Design via Heuristic Dynamic Programming

Watson R. M. Santos, Jonathan A. Queiroz, Jo˜ao Viana da F. Neto, Patr´icia H. M. R´ego, Ewaldo Santana and Gustavo Andrade

Federal University of Maranh˜ao, Federal Institute of Maranh˜ao, State University of Maranh˜ao
Embedded Systems and Intelligent Control Laboratory
S˜ao Luis - Maranh˜ao - Brazil
e-mail: watson.itz@ifma.edu.br jviana@dee.ufma.br

Abstract — In this paper, a method to design online optimal policies that encompasses Hamilton-Jacobi-Bellman (HJB) equation solution approximation and heuristic dynamic programming (HDP) approach is proposed. Recursive least squares (RLS) algorithms are developed to approximate the HJB equation solution that is supported by a sequence of greedy policies. The proposal investigates the convergence properties of a family of RLS algorithms and its numerical complexity in the context of reinforcement learning and optimal control. The algorithms are computationally evaluated in an electric circuit model that represents an MIMO dynamic system. The results presented herein emphasize the convergence behaviour of the RLS, projection and Kaczmarz algorithms that are developed for online applications.

Keywords — Recursive Least Squares; Heuristic Dynamic Programming; RLS Convergence; MIMO Dynamic Systems; Optimal Control; Adaptive Dynamic Programming.

I. INTRODUCTION

In order to overcome the curse of dimensionality problem in applications of Dynamic Programming approach, such as online design of optimal control [1] [2] [3], a lot of efforts has been spent to develop methods to approximate the solution Hamilton-Jacobi-Bellman (HJB) equation [4] [1]. Recently, the development of approximate dynamic programming (ADP) methods and algorithms has been proposed to improve the numerical stability [5] and increase convergence speed of the RLS algorithms family [6] [7]. In this article, a proposal to reduce computational complexity of approximate solution of the HJB equation underlying the discrete linear quadratic regulator (DLQR) problem via derivatives of recursive least square (RLS) method, such as Kacmarz and projection algorithms is presented.

The online DLQR Optimal Control design based on heuristic dynamic programming (HDP) schema is the general context of the main contributions presented in this paper, including recursive least square (RLS) algorithms in addition to convergence analysis method. The RLS algorithms are developed to approximate the Hamilton-Jacobi-Bellman (HJB) equation solution and the convergence is a general procedure to select the parameters of these algorithms. Specifically, an approximation method based on RLS approach to solve the discrete algebraic Riccati equation (DARE), which is a particular form of the HJB equation of the discrete linear quadratic regulator (DLQR), is presented. A convergence analysis method for state and value functions of heuristic dynamic programming algorithms is based on non-singularities of Kronecker transformation of state regressor matrix of RLS estimators. From our point of view, online control design means that the controller gains are automatically adjustable as the dynamic process is in operation mode.

The present proposal is in the context of reinforcement learning (RL), approximate dynamic programming (ADP) and policy iteration to develop online algorithms for the DLQR solution. These algorithms are based on the RLS method to approximate the Hamilton-Jacobi-Bellman equation solution of the DLQR parameterizations. The solutions are given for the discrete algebraic Riccati equation and optimal gain in state value function approximations, respectively.

The reinforcement learning (control) and environment (process) are associated with ADP, as shown in Figure 1 representing the control system with classical feedback in the context of adaptive control that is performed by the critic and actor blocks. Specifically, the RL approach is based on minimizing the Bellman error and heuristic dynamic programming by means of a scheme that combines RLS value function approximation with policy improvement. This approach is directed to online solution of optimal control problems in the sense that the policy improvements are performed at every time step along the realization of state trajectory towards the optimal policy.

Figure 1. Actor Critic Schematic in Control System
Applications in robotics, aerospace and wind generation, require high performance closed-loop controllers. Optimal control has been used to design and implement off-line control laws, the online applications were not exploited until the developments of neural dynamic technics [8] that allow a forward solution of the dynamic programming.

The optimal control theory has been successful for linear systems, but this success is restricted to a few online applications. In a general manner, the methods are offline, because the dynamic system model is needed, which is very difficult to be obtained due to nonlinearities or time constant restrictions of the process. An alternative approach for optimal control solution is based on mathematical concepts and algorithms of dynamic optimization, such as the Bellman method for dynamic programming (DP) and Hamilton-Jacobi-Bellman (HJB) equation [9] [2][10]. The computational resources associated with the realization of those formulations are costly for online implementation [4].

In the context of reinforcement learning, the importance of the least square solution associated with policy iteration (PI) is noticed in [11], where the authors present the state of the art of least square methods for policy iteration as a class of algorithms for approximate reinforcement learning. The component "policy evaluation" of the policy iteration method is performed via least square temporal difference and Bellman residual minimization. The online variants of policy iteration and behavior of representative offline and online methods are focused on the work theses authors. Improvements in the online least square policy iteration method can be seen in [12]. There, the authors propose prior knowledge usage to accelerate online least square policy, what is a paradox, because the RL is seen as a paradigm to work without any prior knowledge of the system.

II. DLQR POLICY ITERATION ALGORITHM

The Discrete linear quadratic regulator (DLQR) is inserted in the context of markovian decision process (MDP) for HDP design purpose. The parametrization of MDP for the DLQR problem and its association with the Bellman equation to develop the approximate policy iteration (PI) algorithms. The main steps of the approximate HDP-PI and ADHDP-PI algorithms for DLQR are commented in terms of actor and critic schemas.

A. DLQR-MDP Framework

The DLQR parametrization in MDP framework is represented by

\[ \mathcal{Y} = \{X, U, f, r\} \]

(1)

where \(X, U, f, r\) are state space, control action space, state transition function and utility function, respectively. These elements are associated with the cost function (also called value function) that is given by

\[ V(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(x_i, u_i) \]

(2)

Expanding the right hand side and manipulating Eq.(2), one obtains a difference equation, called Bellman Equation, that is given by

\[ V(x_k) = r(x_k, u_k) + \gamma V(x_{k+1}) \]

(3)

The parametrization of the state transition \( f(x_k, u_k) \) and the control action \( h(x_k) \) in instant \( k \) are given by

\[ f(x_k, u_k) = Ax_k + Bu_k \]

(4)

and

\[ h(x_k) = -Kx_k. \]

(5)

It is assumed that \((A, B)\) is stabilizable, that is, there is a gain matrix \(K\) that guarantees that the closed loop system

\[ x_{k+1} = (A + BK)x_k \]

(6)

is asymptotically stable. Associated with the impact of the \(K\) gain matrix, there is quadratic utility function that is given by

\[ r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k \]

(7)

where the symmetric matrices \(Q \in \mathbb{R}^{n \times n} \geq 0\) and \(R \in \mathbb{R}^{m \times m} > 0\) represent the state and input weightings, respectively. The objective of the DLQR controller is to select a control policy that minimize a cost function which is given by

\[ V^K(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} (x_i^T Q x_i + u_i^T R u_i) \]

(8)

\[ = \sum_{i=k}^{\infty} \gamma^{i-k} x_i^T (Q + K^T R K) x_i, \quad \forall x_k \in X. \]

(9)

The optimal solution of DLQR, according to [13], admits the following quadratic form

\[ V^K(x_k) = x_k^T P x_k. \]

(10)

The Bellman equation for DLQR is given by

\[ x_k^T P x_k = x_k^T Q x_k + u_k^T R u_k + \gamma (x_{k+1}^T P x_{k+1}) \]

(11)

In terms of the feedback gain of (5) and the dynamics of the closed loop system of Eq.(6), Eq.(11) is expressed by

\[ x_k^T P x_k = x_k^T [Q + K^T R K (A - BK)] x_k \]

(12)

Since Eq.(12) must be satisfied for all states \(x_k\), one has a linear equation in \(P\) that is given by

\[ \gamma (A - BK)^T P (A - BK) - P + Q + K^T R K = 0. \]

(13)

If the gain matrix \(K\) is constant, Equation (13) is known as Lyapunov Equation. Given a stabilizable gain \(K\), the solution of Eq.(13) is \(P = P^T > 0\), such that \(V^K(x_k) = x_k^T P x_k\) is the cost due to policy \(K\), that is,

\[ V^K(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} x_i^T (Q + K^T R K) x_i = x_k^T P x_k. \]

(14)

Replacing the term \(x_{k+1}\) on the right hand side of Eq.(11), the Bellman equation is represented in terms of the current
state $x_k$, which is relevant for real-time applications. The disadvantage of this approach is that the matrices $A$ and $B$ that represent the system dynamics must be known. The Bellman equation, in terms of $x_k$, is given by

$$x_k^T P x_k = x_k^T Q x_k + u_k^T R u_k + \gamma (Ax_k + Bu_k)^T P (Ax_k + Bu_k)$$

(15)

The minimization of Eq.(15) is performed to impose a decision policy (control law) $u_k$ that minimizes the cost function. After derivatives of Eq.(15) with respect to $u_k$, one has that the optimal policy should satisfy

$$R u_k + \gamma B^T P (Ax_k + Bu_k) = 0.$$  

(16)

Equation (16) is solved for $u_k$ and compared with Eq.(5), so it follows that the optimal gain is given by

$$K = \gamma (R + \gamma B^T P B)^{-1} B^T P A.$$ 

(17)

Replacing this into Eq.(13), one obtains the discrete time Hamilton-Jacobi-Bellman (HJB) Equation or the Bellman optimality equation for the DLQR parametrization

$$\gamma (A^T P A) - P + Q - \gamma A^T P B (R/\gamma + B^T P B)^{-1} B^T P A = 0.$$  

(18)

This equation is also known as discrete algebraic Riccati equation (DARE).

### B. HDP-PI Algorithm for DLQR

The main tools for the development of the approximate policy iteration (PI) algorithm for DLQR control design are the approximation method of Lyapunov solution of the Eq.(13) and the procedure of control policy improvement that is established by

$$K_{j+1} = \gamma (R + \gamma B^T \hat{P}_{K_j} B)^{-1} B^T \hat{P}_{K_j} A.$$  

(19)

The Lyapunov approximation $\hat{P}_{K_j}$ method associated to the current policy $K_j$ is based on minimizing the error in Eq.(13) in the least square sense.

The first blocks of the PI for the DLQR are the system parameters and simulation setup. The second block implements the rules for evaluation of approximate PI. The third block implements the improvements of the policy. These steps are shown in Algorithm 1 that implements the approximate PI for the DLQR.

**Algorithm 1**

```
1 1 Setup - Initial conditions
2 2 Weighting and Dynamic System Matrices
3 3 $[Q, R, A, B] \leftarrow []$
4 4 $P$ and $K$ Initial Values
5 5 $[P_{01}, K_{01}] \leftarrow []$
6 6 Iterative Process
7 7 for $j \rightarrow N : -1 : 1$
8 8 do
9 9 Lyapunov Recurrence
10 10 $P_{j+1}(x_{k}) \rightarrow (A_d - B_d K_j)T P_{j}((A_d - B_d K_j) + Q +
11 11 K_{j}^T R K_{j}
12 12 P_{j+1} \leftarrow [
13 13 Feedback - Optimal Gain
14 14 $K_{j+1} \leftarrow (R + \gamma B^T P_{j+1} B_d)^{-1} B^T P_{j+1} A_d
15 15 End - Iterative Process
```

### III. DLQR Value Function Approximation Problem

The HDP algorithms are developed based on the recursive least square (RLS) approach [14] to approximate the state value function. In this section, the RLS estimation and its variants is formulated for approximating the DLQR optimal control policy. The DLQR design is used as a parametrization of MDP mappings. The method of least squares (RLS) has been formulated Gauss in 1795, it is currently applied in the solution of problems of engineering and sciences. In this article a study of equations and changes in RLS method is performed to solve the online form to HJB equation. Details on the deduction of the equations may be obtained from the references.

**A. Problem Formulation**

The main idea behind the state-value RLS-HDP method is the estimation of the cost function $V^{K_j}$ for a given policy $K_j$ that only demands sampling from the instantaneous reward $r_k$ and states, while the models of the environment and the utility function are needed to compute the cost function corresponding to the optimal policy. In this context, the RLS methods play the role of finding parameter $\theta_i$ for estimating $V^{K_j}$. Consequently, the parametric structure of the linear approximation is given by

$$V^{K_j}(x_k) = \varphi^T(x_k) \theta_j,$$  

(20)

and must agree with consistency condition that is given by

$$V^{K_j}(x_k) = r(x_k, h_j(x_k)) + \gamma V^{K_j}(f(x_k, h_j(x_k))).$$  

(21)

**B. RLS Solution**

The formulation of the regression model to the problem of HJB equation approximation using a canonical form of the regression equation is given by

$$\theta_{i+1} = \theta_i + K_{rls}^i (d_i - \varphi^T_i \theta_i)$$  

(22)

$$K_{rls}^{i+1} = P_{rls}^{i+1} \varphi_i (\lambda + \varphi^T_i P_{rls}^{i+1} \varphi_i)^{-1}$$  

(23)

$$P_{rls}^{i+1} = \frac{1}{\lambda} (I - K_{rls}^i \varphi_i^T) P_{rls}^i,$$  

(24)

where $\theta_i$ is the variable representing the unknown parameters, i.e., the solutions for the HJB equation. The variable $P_{rls}$ represents the covariance matrix of the search process of parameters $\theta_i$, the values of $P_{rls}$ on the right side of Eq.(23) means that they were updated by the right side of Eq.(24), the updating of $K_{rls}$ occurs in a similar way. The observation value represents the components of the gradient vector of the cost function $V(x)$ in relation to the state, and $d_i \in \mathbb{R}$.

The quadratic form of the cost function, Eq.(10), is represented in terms of Kronecker product of states and in terms of matrix $P$ vectorization that is the solution of the Hamilton-Jacobi-Bellman equation, Eq.(18). The optimal solution in its vectorized form is given by

$$V^{K_j}(x_k) = x_k^T P_j x_k = \pi_k^T \text{vec}(P_j),$$  

(25)

where $\pi_k \in \mathbb{R}^{(n+1)/2}$ is a vector that is defined according to the Kronecker product given by

$$\pi_k^T = x_k^T \otimes x_k^T = [x_{1k}^T x_k^T \ldots x_{nk}^T x_k^T].$$  

(26)
C. Proposed Solution

The RLS-projection and Kaczmarz methods are investigated as an alternative to compose the central element, which updates the gain of the RLS estimator that is given respectively by

\[ K_{rls}^{i+1} = \mu_{rls} \frac{\phi_i}{\alpha_{rls} + \phi_i^T \phi_i} \tag{27} \]

and

\[ K_{rls}^{i+1} = \mu_{rls} \frac{\phi_i}{\phi_i^T \phi_i} \tag{28} \]

where \( \alpha_{rls} \geq 0 \) is the parameter used to solve the overflow problem in the RLS algorithm which are derived from the occurrence of singularities in the product \( \phi_i^T \phi_i \) of the denominator, and \( 0 < \mu_{rls} < 2 \). Thus, Eq.(22) of the unknown variable and Eq.(23) of the gain update are the equations used to assemble the RLS algorithm core, the updating equation of \( P_{rls} \) is not necessary for these two algorithms. These two algorithms diminish the computing effort when estimating the parameters with respect to the RLS method with forgetting factor \( \lambda \). It can be observed that \( \alpha_{rls} \) realizes the parameter role \( \lambda \) on the standard RLS method equation, but its effect is not similar to the impact provoked by parameter \( \lambda \) which is more comprehensive, producing impacts on the matrix \( P_{rls} \) update.

The convergence results in HDP schemes via RLS approximation has been investigated by ([15]) for linear quadratic regulator (LQR). According to theorem presented in ([15]) and ([16]), the sequence \( \{K_j\}_{j=1}^\infty \) of policies generated by the RLS-HDP-DLQR algorithm converge to optimal policy \( K^* \), that is,

\[ \lim_{j \to \infty} \| K_j - K^* \| = 0, \tag{29} \]

where \( K^* \) is the optimal feedback control matrix that is the mapping of decision policy and \( K_j \) is the policy improvement in \( j \)-th iteration.

IV. HJB APPROACH VIA RLS.

The third part of the procedure concerns with tests and analysis to evaluate the convergence and accuracy of online-RLS-projection algorithm for a fourth-order model of reference [1]. The initial conditions of the experiments are in this section.

A. Initial Conditions

The initial conditions are arranged according to their functionality in the search process of the discrete algebraic equation Riccati (DARE) solution. These conditions are associated with the DLQR dynamic system design, iterative DARE approximation method, as well as the initial conditions are the interface between the dynamic system and DARE approximation method: gain \( K_{HDP} \) generates the new state vector which in turn was estimated by RLS method. The parameter setup are:

1) The initial conditions of the state vector has the following components \( x_1 = 0.22; x_2 = -0.25; x_3 = -0.005; x_4 = 0.005. \)
2) Initial DARE solution is \( P_{HDP} = \text{param}_{HDP} I_{n \times n} \) that generates the initial gain matrix, given by \( K_{HDP} = (R + B_d^T P_{HDP} B_d)^{-1} B_d^T P_{HDP} A_d \).
3) The weighting matrices of DLQR design are given by \( R = I_{2 \times 2} \) and \( Q = I_{4 \times 4} \).

Initial conditions for RLS:

1) The initial conditions of the unknown parameter \( \theta \) are \( \theta_i = 0 \), with \( 1 \leq i \leq 10. \)
2) The covariance matrix of the RLS process is given by \( P_{rls} = \text{param}_{rls} I_{6 \times 6} \).

B. Setup of the Iterative Process

The setup of the iterative process consists in establishing the better parameters for solving a given application, having as reference the designer’s empiricism or the relationships among system variables related to the methods. The parameters that play a relevant role in the method convergence are: system order \( n \) (\( n = 4 \)), sampling interval \( T_{almost} \) (\( T_{almost} = 0.1s \)), forgetting factor (RLS) \( \lambda \) (\( \lambda = 0.96 \)) and the parameters \( \alpha_{rls} \) and \( \mu_{rls} \) projection. The parameter \( T_{almost} \) is used to capture the values (data) and control action for calculating the moving target at each iteration \( k \). The dynamic system matrices represent the behavior of a 4-th electrical circuit with two inputs and two outputs given in [17].

C. Performance Analysis of RLS-HDP Algorithms

The 4-th electrical circuit model is used to evaluate the performance of RLS, projection and Kaczmarz algorithms for optimal controller design. The evolution of the iterative process for HJB-Riccati equation solution by the RLS-HDP algorithm is presented in Figures 2, 5 and 8 for a cycle of 3000 iterations. The curves (a)-(d) of Figure 2 represent the convergence behavior of the elements \( p_{11}, p_{22}, p_{33} \) and \( p_{44} \) of the matrix \( P \) corresponding to the components \( \theta_1, \theta_5, \theta_8 \) and \( \theta_{10} \) of the parameter vector \( \theta \), respectively. Curves (a)-(c) of Figure 5 show the evolution of the elements \( p_{12}, p_{13} \) and \( p_{14} \) of the matrix \( P \) that are associated to the elements \( \theta_2, \theta_3 \) and \( \theta_4 \) of the vector \( \theta \), respectively. The convergence behavior of the parameters \( p_{23}, p_{24} \) and \( p_{34} \) associated to the elements \( \theta_6, \theta_7 \) and \( \theta_9 \) is represented by the curves (a), (b) and (c) of Figure 8, respectively. The results of projection algorithms are presented in a similar way in the Figures 3, 6 and 9, and also the results of Kaczmarz algorithms are presented in the Figures 4, 7 and 10. In terms of steady state values, the graphs of Figures 2, 5 and 8 are compared with the graphs of Figures 3, 6, 9 and 4, 7, 10 respectively, showing that, in general the convergence of the RLS algorithm is reached around 2500 iterations, while that the projection algorithm is achieved 2300, and for the Kaczmarz algorithm is attained 1000 iterations. Thus, it is observed that the Kaczmarz algorithm has a better performance with regard to speed convergence performance when compared to the other two methods.
The HDP-RLS steady state solution of *Riccati/Lyapunov* is given by

Figure 2. Convergence behavior of the parameters of RLS $\theta_1$, $\theta_5$, $\theta_8$, and $\theta_{10}$

Figure 3. Convergence behavior of the parameters of projection $\theta_1$, $\theta_5$, $\theta_8$, and $\theta_{10}$

Figure 4. Convergence behavior of the parameters of Kaczmarz $\theta_1$, $\theta_5$, $\theta_8$, and $\theta_{10}$

Figure 5. Convergence behavior of the parameters of RLS $\theta_2$, $\theta_3$ and $\theta_4$

Figure 6. Convergence behavior of the parameters of projection $\theta_2$, $\theta_3$ and $\theta_4$

Figure 7. Convergence behavior of the parameters Kaczmarz $\theta_2$, $\theta_3$ and $\theta_4$

Figure 8. Convergence behavior of the parameters of RLS $\theta_6$, $\theta_7$ and $\theta_9$

Figure 9. Convergence behavior of the parameters of projection $\theta_6$, $\theta_7$ and $\theta_9
The adaptive dynamic programming had shown to be a feasible alternative for control system design realizations with no parameterizations of the environment and the decision policy. The evaluation tests showed that the RLS methodology presented a satisfactory performance to estimate the optimal policy in actor-critic schema of reinforcement learning.

ACKNOWLEDGMENT

The authors would like to thank to UFMA, IFMA, UEMA and PPGEE.

REFERENCES