Combining Markov Decision Processes with Linear Optimal Controllers

Ekaterina Abramova, Daniel Kuhn, and Aldo Faisal

Imperial College London, Department of Computing,
Exhibition Road, London SW7 2AZ, UK

Abstract. *Linear Quadratic Gaussian (LQG) control has a known analytical solution [1] but non-linear problems do not [2]. The state of the art method used to find approximate solutions to non-linear control problems (iterative LQG) [3] carries a large computational cost associated with iterative calculations [4]. We propose a novel approach for solving non-linear Optimal Control (OC) problems which combines Reinforcement Learning (RL) with OC. The new algorithm, RLOC, uses a small set of localized optimal linear controllers and applies a Monte Carlo algorithm that learns the mapping from the state space to controllers. We illustrate our approach by solving a non-linear OC problem of the 2-joint arm operating in a plane with two point masses. We show that controlling the arm with the RLOC is less costly than using the Linear Quadratic Regulator (LQR). This finding shows that non-linear optimal control problems can be solved using a novel approach of adaptive RL.*

Keywords: Reinforcement Learning, non-linear Optimal Control, locally linear approximations, Linear Quadratic Regulator, robotic arm.

1 Introduction

Optimal Control (OC) theory aims to find a control law which would manipulate a dynamical system while minimizing a cost associated with that system. Non-linear OC deals with systems involving non-linear dynamics and is known to be the most difficult area of the control theory [5]. Difficulties in solution of Hamilton-Jacobi-Bellman partial differential equation for this class of problems [2] resulted in use of iterative methods which yield approximate solutions.

Main methods include: Receding Horizon Control (RHC) [6], Control Lyapunov Functions (CLFs) [7], Differential Dynamic Programming (DDP) [8], [9], Iterative Linear Quadratic Regulator (iLQR) [10] and Iterative Linear Quadratic Gaussian control (iLQG) [3]. The CLF method can lack stability unless it fits closely with the value function and RHC can act suboptimally. The DDP, iLQR and iLQG involve iterative calculations and are computationally costly.

We propose an alternative general method where the adaptive properties of Reinforcement Learning (RL) are combined with the power of OC. This method, RLOC, would be applicable in many different fields as long as the system dynamics and costs can be formally stated or approximated.

Andrew V. Jones (Ed.): Proceedings of the 1st Imperial College Student Workshop (ICCSW '11), pp. 3–9. September 29th–30th 2011, London UK.

RL involves an agent that interacts with the world through actions and receives corresponding rewards [11] thus improving the agent's behaviour. RL problems that satisfy the Markov Property are called Markov Decision Processes (MDPs). An MDP represents a framework used to define control problems.

2 Problem Formulation

2.1 Linear Optimal Control

The Linear Quadratic Regulator (LQR) control has a closed form solution [12]. Its dynamics are linear in $\mathbf{x}$ and costs are assumed to be quadratic. The deterministic linear optimal control problem has the following form

$$\mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k \tag{1}$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ is the control vector, A is the system matrix acting on the state vector and B is the control gain matrix acting on the control vector.

The continuous system dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, where $\dot{\mathbf{x}}$ is the first derivative of the state vector with respect to time, can be represented in discrete form yielding the update rule (Forward Euler)

$$\begin{aligned} \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{\Delta t} &= \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \Delta t \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \end{aligned}$$

where k is the step number.

The total cost, J, is the overall cost, accumulated over finite horizon through incremental (J_k) and final (J_n) cost rates

$$J = \frac{1}{2}{\mathbf{x}_n}^{\mathrm{T}} Q^f \mathbf{x}_n + \sum_{k=0}^{n-1} (\frac{1}{2}{\mathbf{u}_k}^{\mathrm{T}} R \mathbf{u}_k + \frac{1}{2}{\mathbf{x}_k}^{\mathrm{T}} Q \mathbf{x}_k) \tag{2}$$

where n is the number of steps, R is the control cost matrix, Q is the state cost matrix and Q^f is the final state cost matrix.

The control update is linear in $\mathbf{x}$ and is calculated using

$$\mathbf{u}_k = -L_k \mathbf{x}_k \tag{3}$$

where $L_k = ((R + B^{\mathrm{T}} V_{k+1} B)^{-1} B^{\mathrm{T}} V_{k+1} A)$ is the feedback gain matrix and $V_k = Q + A^{\mathrm{T}} V_{k+1} A - A^{\mathrm{T}} V_{k+1} B (R + B^{\mathrm{T}} V_{k+1} B)^{-1} B^{\mathrm{T}} V_{k+1} A$ is the cost to go function. The L matrix does not depend on $\mathbf{x}$ and therefore can be computed offline.

2.2 Non-Linear Optimal Control

The deterministic non-linear optimal control problem we choose to study has the following general form

$$\mathbf{x}_{k+1} = A(\mathbf{x}_k)\mathbf{x}_k + B(\mathbf{x}_k)\mathbf{u}_k \tag{4}$$

It does not have an analytical solution [2] and approximate solutions are computed using iterative methods.

2.3 Proposition

Traditional OC methods used for solving non-linear problems need as many locally linear controllers as there are incremental number of steps. We propose approximating non-linear OC systems by using a reduced number of linear optimal controllers which are joined together for global use with a RL algorithm.

Our problem formulation operates in two spaces: Formulation Space (φ_1) and Process Space (φ_2). The φ_1 describes every aspect of the OC problem and incorporates the Formulation States (FS). The φ_2 describes every aspect of the Markov Decision Process and incorporates the Process States (PS).

The non-linear control problem is converted into a finite MDP problem by:

1 Discretizing the continuous FS (i.e. variable $\mathbf{x}$) into a finite number of equal discrete states that correspond to the PS $\{s_1, s_2, ..., s_n\}$.
2 Using a Linear Quadratic Regulator (LQR) to obtain a small set of localized optimal linear controllers, specifically the feedback gain matrices $L_1, L_2, ..., L_n$.
3 Using a RL algorithm to learn the mapping from the PS to controllers (i.e. optimal way of combining localized linear controllers).

3 Application: Modelling Human Arm

Human motor coordination is well predicted by optimal control [13] and the approximations to non-linear human arm dynamics have been extensively studied [3], [10], [14], [15]. We illustrate RLOC by solving a non-linear optimal control problem of a simplified human arm model (Fig. 1).

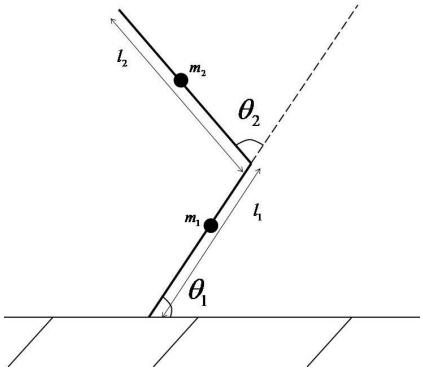

Fig. 1. Simulated arm representation where θ_1 is the angle of the shoulder, θ_2 is the angle of the elbow, both defined on the $[0°, 180°]$ continuous interval, l_1 is the length of the first link, l_2 is the length of the second link, m_1 and m_2 are the positions of the weight on each respective link. Arm is allowed to move in a horizontal plane.

The forward arm dynamics are described by the equation

$$\ddot{\theta} = \mathcal{M}(\theta)^{-1}(\tau - \mathcal{C}(\theta, \dot{\theta}) - \mathcal{B}\dot{\theta}) \tag{5}$$

where $\theta \in \mathbb{R}^2$ is the joint angle vector (shoulder: θ_1, elbow θ_2), $\mathcal{M}(\theta)$ is a PD symmetric inertia matrix, $\mathcal{C}(\theta, \dot{\theta}) \in \mathbb{R}^2$ is a vector centripetal and Coriolis forces, $\mathcal{B} \in \mathbb{R}^{2\times 2}$ is the joint friction matrix and $\tau \in \mathbb{R}^2$ is the joint torque (defined to be the control $\mathbf{u} = \tau$) [14].

The FS is defined as a 4D vector containing joint angles and their velocities

$$\mathbf{x} = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)^{\mathrm{T}} \tag{6}$$

The state dynamics are described as the first derivative of the state vector, $\dot{\mathbf{x}}$

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = (\dot{\theta}_1, \dot{\theta}_2, \ddot{\theta}_1, \ddot{\theta}_2)^{\mathrm{T}} \tag{7}$$

3.1 Simulation

The arm joint angles are discretized into 36 PS states. Six feedback gain matrices, corresponding to RL actions, are obtained by linearizing the arm dynamics using LQR approach. The linearizations are performed around equally spaced points in the φ_1 and vary in elbow angle only. This is due to the fact that non-linearity of the arm depends only on the elbow angle (Fig. 2).

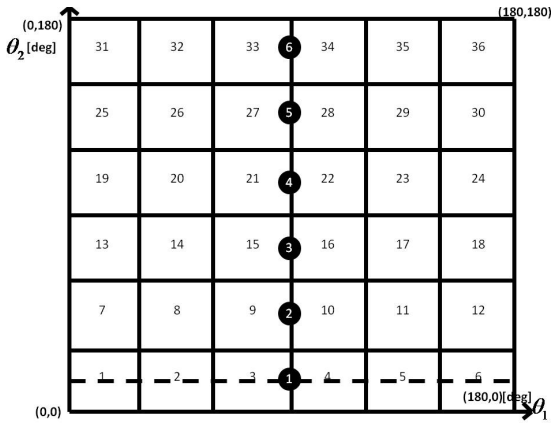

Fig. 2. The $x - axis$ is the shoulder angle, $y - axis$ is the elbow angle. The squares show PS and the black circles point to locations of feedback gain matrices obtained by linearization of arm dynamics. The dashed line shows the φ_1 along which the L matrices are equal, this is true for any such line parallel to the $x - axis$.

We choose an epsilon-greedy on-policy Monte Carlo algorithm [11]. It learns from sampling sequences of states s, actions a and immediate rewards (costs) r. A full sample path from a starting to an absorbing state is called a trace, Γ, we use 20000 traces. Optimal policy is learnt for the trajectory of the arm from the center of any of the 36 states to a target.

Optimal Policy is learnt using the following steps:

1 Initialize a deterministic policy having equal probability of picking any action (linearized controller).
2 For each trace: a) pick a random starting Process State and b) chose current action corresponding to that Process State (determined by the policy).
3 Get a trace: a) start controlling the arm using LQR, b) once enter a new Process State record the {s,a,r} triplet, c) pick a new controller at random, d) repeat until reach the target (i.e. end of obtaining a trace).
4 Examine {s,a,r} triplets to calculate Action-Value Function Q. Improve the Policy by altering probabilities of picking each action in each state. If the Policy yields lower trace cost, store it as the Optimal Policy.
5 Repeat steps 2. to 4. until obtain a specified number of traces (in our case 20000).

3.2 Results and Discussion

The results show that using RLOC algorithm to optimally control the arm from any of the 36 states to an arbitrary target of $(40°, 10°)$ results in either equal to or better performance than the LQR (Fig. 3). The results are presented as 'relative differences' in cost using the following formula $\frac{(J_{LQR} - J_{RLOC})}{|J_{LQR}|}100\%$. This is necessary since some starting states are further away from the target and hence the acquired cost would be higher simply due to the distance travelled. Negative values indicate that RLOC is less costly.

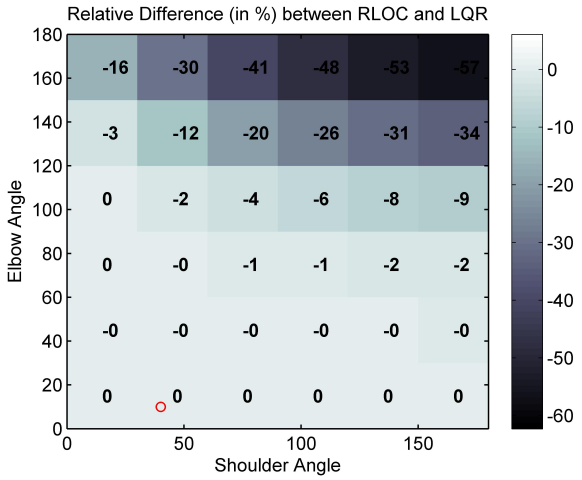

Fig. 3. Relative cost differences in % between the cost of RLOC and LQR (both calculated using J in equation (2). Each square corresponds to a PS, numbered as in (Fig 2) and the target is marked by a circle. The following cost matrices were used: $R = diag[1\ 1]$, $Q = diag[1.5\ 1.5\ 0\ 0]$ and $Q_f = diag[3000\ 3000\ 300\ 300]$.

The total costs accumulated during the MC simulation are shown in (Fig. 4). The algorithm 'learns' throughout the simulation. This can be seen by the decrease in the cost size experienced at the end of each trace for each of the starting states (picked at random at the beginning of a trace). The agent learns a better policy with each sampled trace which results in decreasing cost size as the simulation progresses. Note that the graph has distinctive reduction in the worst cost experienced for each starting state and these transform into lines as the algorithm learns. Each line represents minimal possible cost that could be incurred by controlling the arm from each starting state to the target under RLOC algorithm.

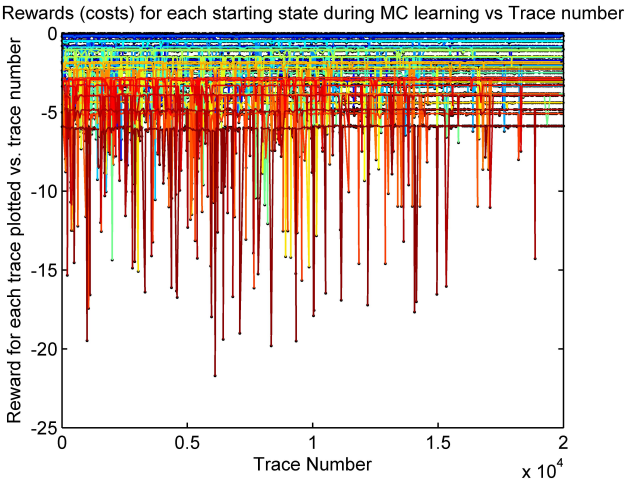

Fig. 4. Plot of the total cost of each trace vs. the trace number (20000 traces obtained). Each starting Process State cost variability is marked with a different line. As the simulation progresses, the variability of the maximum cost encountered by the learner is reduced. Therefore this figure demonstrates that the algorithm learns to use better (less costly) controllers for each Process State.

3.3 Conclusions

We presented the theoretical formulation, supported by a practical example, for the novel approach of solving non-linear optimal control problems by combining RL with OC to produce a new algorithm RLOC. The use of the RL agent is advantageous since the learner is able to explore a large amount of states, experiencing various state-action scenarios. This allows the learner to pick the control policy which provides it with the most overall reward (least cost).

We illustrated the proposed algorithm with a model of the human arm movement, where a combination of LQR control and epsilon greedy on policy Monte

Carlo method was used. The algorithm proved to be able to control the arm from the moment of initialization. This allows a trade off between speed and optimality, which may be desirable in practice for on-line computations. The algorithm was able to reach the desired target in a smooth manner with less accumulated cost than the LQR. This finding is significant because it is the first time it has been shown that RL can be combined with OC to produce better results than using LQR or RL alone. We have therefore taken a step closer to improving our current ability to control complicated non-linear systems.

An important aspect of this research is that it can be applied to many real life problems (such as drug delivery, space exploration, algorithmic trading, air traffic control and robotic control).

References

1. E. Todorov. Optimal control theory. *Bayesian brain: probabilistic approaches to neural coding*, pages 269–298, 2006.
2. A.E. Bryson and Y.C. Ho. *Applied optimal control: optimization, estimation, and control*. Hemisphere Pub, 1975.
3. E. Todorov and W. Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005. Proceedings of the 2005*, pages 300–306. IEEE, 2005.
4. D. Mitrovic, S. Klanke, and S. Vijayakumar. Adaptive optimal feedback control with learned internal dynamics models. *From Motor Learning to Interaction Learning in Robots*, pages 65–84, 2010.
5. J.A. Primbs, V. Nevistić, and J.C. Doyle. Nonlinear optimal control: A control lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1(1):14–24, 1999.
6. WH Kwon, AM Bruckstein, and T. Kailath. Stabilizing state-feedback design via the moving horizon method. *International Journal of Control*, 37(3):631–643, 1983.
7. E.D. Sontag. Lyapunov-like characterization of asymptotic controllability. *SIAM J. CONTR. OPTIMIZ.*, 21(3):462–471, 1983.
8. D.H. Jacobson. New second-order and first-order algorithms for determining optimal control: A differential dynamic programming approach. *Journal of Optimization Theory and Applications*, 2(6):411–440, 1968.
9. D. Jacobson. Differential dynamic programming methods for solving bang-bang control problems. *Automatic Control, IEEE Transactions on*, 13(6):661–675, 1968.
10. W. Li and E. Todorov. Iterative linear-quadratic regulator design for nonlinear biological movement systems. In *Proceedings of the First International Conference on Informatics in Control, Automation, and Robotics*, pages 222–229. Citeseer, 2004.
11. R.S. Sutton and A.G. Barto. *Reinforcement learning*, volume 9. MIT Press, 1998.
12. B.D.O. Anderson and J.B. Moore. *Optimal control: linear quadratic methods*. Prentice-Hall Englewood Cliffs (NJ):, 1989.
13. C.M. Harris and D.M. Wolpert. Signal-dependent noise determines motor planning. *Nature*, 394(6695):780–784, 1998.
14. W. Li and E. Todorov. Iterative linearization methods for approximately optimal control and estimation of non-linear stochastic system. *International Journal of Control*, 80(9):1439–1453, 2007.
15. M.T. Mason. *Mechanics of robotic manipulation*. The MIT Press, 2001.