From Optimal Control to Reinforcement Learning: A Motivating Example from the Perspective of Linear Quadratic Regulators

Fredson Silva de Souza Aguiar

July 8, 2026

We motivate the study of interconnections and relevant results at the interface between Control Theory and Reinforcement Learning, with a special focus on the class of problems known as Linear Quadratic Regulators. We seek to motivate the existence of continuous paths joining techniques and characteristics from the two communities that initially seem to lie in separate universes. By doing so, we highlight how the two fields are not only complementary but, in fact, can benefit greatly from each other.

We start our discussion by walking the reader between two extremes: fully model-based optimal control and fully model-free reinforcement learning. We highlight the advantages and disadvantages of each method, and motivate the existence of intermediate methodologies. Trough a final mixed methodology, we show the power and importance of data-driven techniques, but also how theoretical results are fundamental to justify and guide efficient solutions. This example also justifies how classical optimal control and reinforcement learning are, in fact, part of a single continuum; even further: their frontier is blurry.

Finally, the reproducible codes and trained models used in these notes are made available in the following repository: `https://github.com/DCN-FAU-AvH/oc_rl_lqr`.

1 The Linear Problem and Milestones

In these notes, we will focus on continuous-time Linear Time Invariant (LTI) control systems, with full observability. That is, we are interested in dynamical systems of the form:

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x \in \mathbb{R}^n, \quad u \in \mathbb{R}^k$$

where the *state* signal, $x(t)$, and the *input/control* signal, $u(t)$, are known at any time. For a more complete discussion of linear systems, *input-output* dynamics, and observability, the reader may refer to [Hes18].

Taking this simple model as a starting point, different questions may arise:

- *Is it possible to steer an initial state to any desired state at any time?*
- *If possible, how to do that in the smartest way possible?*
- *Finally, what if we do not know much about the dynamics?*

We will see in our discussion that while the first questions above can be answered easily using classical and modern Optimal Control (OC) theory, the last one requires a complementary approach, which we introduce via Reinforcement Learning (RL). We discuss the drawbacks and advantages of the different techniques, which motivates the importance of the recent efforts developed to bridge the best of both communities; we highlight that the model-based and data-driven approaches to control are not contradictory in nature, but complementary. Finally, we briefly mention extensions of these concepts to nonlinear cases.

We intend to bridge the two worlds through a well-paved path. This is done through discussion of the following results/milestones:

- **Existence:** Under what conditions can we ensure the existence of a stabilizing/steering control? This is done through Kalman's Controllability Rank Condition. This result opens the path to the discussion of optimal linear feedbacks, which is a common problem in classical optimal control and reinforcement learning.
- **Structure:** characterization of *the best* control, after the introduction of quadratic costs, as a linear feedback of the form $u = -Kx$, trough the Riccati Differential Equation, which is a classical optimal control result but will later justify the expression of our cost-to-go and $\mathcal{Q}$-function as a quadratic form.
- **Value Function**: from the optimal control characterization as a linear feedback, we characterize the value function and $\mathcal{Q}$-function as quadratic forms. This characterization is fundamental to characterizing and solving Bellman's equation.
- **Exploration**: we discuss the necessity of *exploration* in determining the $\mathcal{Q}$-function and optimal feedback when no exact model is available, in the form of *persistently exciting controls*. This justifies how it is possible to compute an optimal control through data and exploration instead of an explicit model, as in classical control.

2 Controllability and Linear Feedback

Let us assume that one is interested in steering any given initial state $x(t_0) = x_0$ to a given final state x_f. Without loss of generality, this problem can be rewritten as guiding a given state to the origin, $x_f = 0$.

This is the problem of *controllability*: a system is said to be controllable if any initial state $x(t_0) = x_0$ can be moved to the origin given any positive amount of time, $x(t) = 0$, $t > t_0$. It turns out that, since the system matrix does not depend on time, and the previous definition does not impose a time window, the controllability of an LTI system can be completely characterized by the following rank condition:

Theorem 1 (Kalman rank condition) *A Linear Time Invariant (LTI) system is completely controllable if and only if*

$$\text{rank}\, \mathcal{C} = n, \qquad \mathcal{C} := [B \quad AB \quad A^2B \quad ... \quad A^{n-1}B] \in \mathbb{R}^{n \times kn}.$$

The matrix $\mathcal{C}$ is the so-called controllability matrix.

In this problem, a natural impulse is to impose that the control follows a linear feedback of the form $u(t) = Kx(t)$, where K is called a *state feedback gain matrix*. This gives a closed-loop Ordinary Differential Equation as follows:

$$\dot{x}(t) = Ax(t) + BKx(t) = (A + BK)x(t), \qquad x(t_0) = x_0$$

So, under the controllability condition, by choosing an appropriate gain matrix K, it is possible to assign arbitrary eigenvalues to the system.

Theorem 2 (Eigenvalue Assignment) *Assume the LTI system (A, B) is controllable. Given any set of complex numbers $\{\lambda_1, ..., \lambda_n\}$ in which complex conjugates appear in pairs, there exists a state feedback matrix $K \in \mathbb{R}^{k \times n}$ such that the closed loop system $\dot{x}(t) = (A - BK)x(t)$ has eigenvalues equal to λ_i.*

In particular, this method results in asymptotic feedback stabilization by imposing eigenvalues in the complex half-space with negative real part.

However, an immediate drawback from the previous approach is that an asymptotic result requires an infinite amount of time to steer the initial state to a final state of choice, and by doing so, we do not take into account possible costs arising from time, states, or control assumed on the process. It also does not make clear *how* to choose an appropriate gain matrix K. In fact, most real-world problems incur some cost, and, ideally, a precise method for computing the control is also necessary.

For this reason, in the following section, we introduce the important class of control problems called Linear Quadratic Regulators, and its importance not only for finite-time, but also for infinite-horizon steering/stabilization problems.

3 Linear Quadratic Regulator

Given an LTI system, a Linear Quadratic Regulator (LQR) problem associates each solution to a cost functional of the form:

$$J(u) = \int_0^{t_f} x^\top(t)Qx(t) + u^\top(t)Ru(t)dt + x^\top(t_f)Fx(t_f),$$

where R is positive definite, and Q, F are positive semi-definite. In an LQR problem, the objective is to find an *optimal* control strategy that minimizes the criterion $J(u)$ above.

Through this characterization, a suitable choice of matrices Q,R, and F can be imposed to induce a desirable outcome for the optimal solution: for example, an elevated cost in the final state $x^\top(t_f)Fx(t_f)$ may incur in optimal control u^* that steers the associated trajectory x^* closer to the origin; similar behavior can be achieved by imposing a high running cost on the states trough $x^\top(t)Qx(t)$. In addition, LQR encounters applications for local stabilization of nonlinear systems and infinite-horizon problems, time-varying systems, among others [Ted23].

We will see in the following that, beyond the applicability of this class of problems, the importance of LQR lies in a well-characterized optimal solution in the form of a feedback.

3.1 Optimal Feedback and Example

Interestingly, given an LQR problem, our first proposition of a control following a linear feedback was not completely off. In fact, an optimal strategy can be expressed trough the following

Theorem 3 (Linear Feedback) *The optimal solution to the finite-horizon LQR problem is unique and given by:*

$$u(t) = -K(t)x(t), \qquad K(t) := R^{-1}B^\top P(t),$$

where $P(t)$ is the solution to the Riccati Differential Equation (RDE):

$$-\dot{P}(t) = A^\top P(t) + P(t)A - P(t)BR^{-1}B^\top P(t) + Q, \qquad P(t_f) = F.$$

Thus, while the optimal solution is not a time-invariant feedback, its structure is well defined given a time-dependent matrix $P(t)$. Furthermore, in the particular case of an infinite-horizon LQR problem, $P(t) = P$ is constant, and the feedback gain matrix $K(t) = K$ is also constant.

As an example, consider the following LQR problem with $x_0 = (1, 1)$, $t_0 = 0$, $t_f = 5$, with the linear system given by the time-invariant matrices:

$$A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix},$$

and the cost functional given by the matrices $Q = I_2$, $R = 10I_2$, $F = 100I_2$. We numerically solve the RDE to obtain the solution $P(t)$, and the resulting optimal control is presented in Figure 1. Observe that, as suggested previously, by imposing a high cost on the final deviation from the origin, the optimal solution tends to be brought close to the origin.

Therefore, through the RDE, we were able to obtain an optimal solution to the problem in an efficient manner. On the other hand, while this approach presents a well-defined description of the optimal solution to an LQR, the classical model-based methods tend to assume prior knowledge of the dynamics, which is sometimes not viable. Contexts such as epidemics or economics can involve unknown dynamics that require action. In this context, the use of learning-based techniques in the decision-making process becomes fundamental.

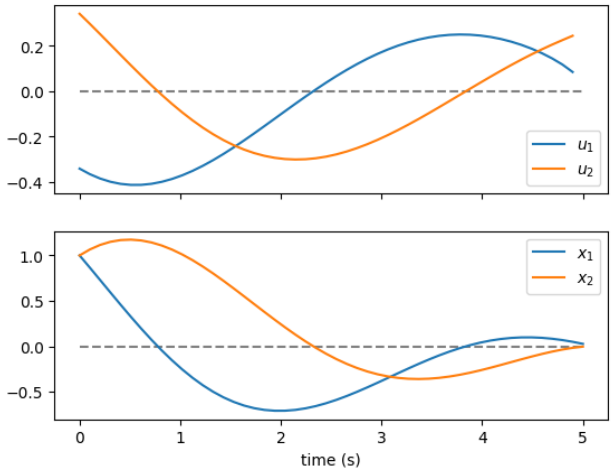

Figure 1: Optimal solution to an LQR problem obtained by solving the Riccati Differential Equation. Total cost realized: $J \approx 8.20$.

4 Reinforcement Learning

A field closely related to OC, due to its portrayal as a decision-making problem, is Reinforcement Learning (RL). It is characterized by the interaction of an agent with its environment, with the task of maximizing a reward function over time; also, the agent does not assume knowledge of the environment and must learn about it through interaction. So another distinctive characteristic of RL is the trade-off between *exploiting* the reward and *exploring* the environment [SB18]. This allows the employment of RL-specific techniques for decision-making even when incomplete knowledge of the environment is available.

More formally, Reinforcement Learning can be modeled as a Markov Decision Process. From a state $s_k \in S$, an agent takes an action $a_k \in A$ that impacts the environment, which returns a new state $s_{k+1} \in S$ and a reward signal, $r_{k+1} \in \mathbb{R}$; here S and A are known respectively as the *state space* and *action space*. The objective of different RL methods is to generate an optimal *policy* $\pi(a|s) : S \to A$, a distribution from state to action that maximizes the (average) cumulated reward over time:

$$J(\pi) = \sum_{k=0}^{N} \gamma^k r_k.$$

where $\gamma \in (0, 1]$ is a *discount-factor*. On this sense, a policy generalizes the notion of a deterministic feedback control. Once more, this characterization highlights the similarities between RL and OC.

4.1 Time-Discrete LQR

A consideration to be made is due to the discrete-time nature of a reinforcement learning problem compared to our initial continuous-time characterization of optimal control. However, the RL environment can be seen as the discretization of a continuous model (e.g., a Newton's method approximation), giving rise to the analogous difference equation model as follows:

$$x_{k+1} = x_k + h(Ax_k + Bu_k), \qquad -r_k = \begin{cases} x_k^\top Q x_k + u_k^\top R u_k, & \text{if} \quad k < N, \\ x_k^\top F x_k, & \text{if} \quad k = N, \end{cases}$$

where $h > 0$ is a small *step-size*; r_k is taken to be negative due to the usual convention of maximizing rewards in the context of reinforcement learning. The previous controllability and feedback results can also be extended to the time-discrete case [Hes18]. From now on, we consider the discretized LQR problem above,

taken with $h = 1/10$, $N = 50$. Through this characterization, the LQR problem can be expressed as an RL environment. Following, established RL techniques can be employed to obtain an optimal strategy.

In the following, we present two established methods to obtain (sub-)optimal solutions to the problem as presented. For now, we focus on fully model-free approaches to the problem, that is, the agents have no knowledge of or access to the dynamics or cost structure; this is the extreme opposite case to the previous RDE-based optimal feedback computation. Both approaches will be later compared to a model-based, parameter-independent solution.

4.2 Parametric Policy Gradient with REINFORCE

A classical reinforcement learning algorithm is called REINFORCE. It consists of a *policy gradient* method, i.e., one learns a parametrized policy $\pi_\theta(a|s)$, commonly in the form of a distribution. In this context, the *gradient* aspect comes from the choice of a parameter update rule based on (approximate) gradient ascent of a criterion $J(\pi_\theta)$:

$$\theta_{t+1} = \theta_t + \alpha \widehat{\nabla_\theta J(\pi_\theta)},$$

where $\alpha > 0$ is a *learning-rate*. Here, a fundamental result is expressed in the form of the

Theorem 4 (Policy Gradient) *Let $\pi_\theta(a|s)$ be a differentiable policy; then the gradient of the performance criterion with relation to the parameter θ is proportional*

$$\nabla_\theta J(\pi_\theta) \propto \mathbb{E}_{\pi_\theta}\left[\sum_{k=0}^{N} G_k \nabla_\theta \ln \pi_\theta(a_k|s_k)\right] \qquad \textit{with the cost-to-go} \qquad G_k = \sum_{n=k+1}^{N} \gamma^{n-k-1} r_n,$$

In the particular case of REINFORCE, the gradients are approximated as:

$$\widehat{\nabla_\theta J(\pi_\theta)} = G_k \nabla_\theta \ln(\pi_\theta(a_k, s_k)),$$

which results from the Policy Gradient theorem.

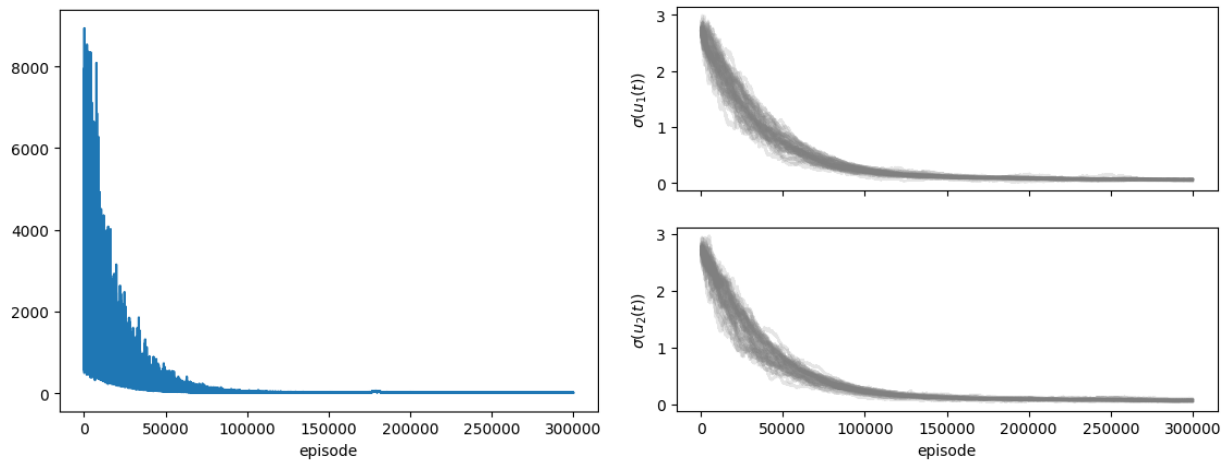

Figure 2: Total episodic loss over episodes (left), and decreased policy variances over episodes (right).

For the continuous action space given by the LQR, $A = R^2$, we propose a parametric policy given by the following distribution:

$$\pi_\theta(a|s) = \frac{1}{\sigma_\theta(s)\sqrt{2\pi}} \exp\left(-\frac{(a - \mu_\theta(s))^2}{2\sigma_\theta^2(s)}\right).$$

We avoid assuming any structure of the solution by imposing only time-dependent actions, that is, the state is taken to be the current time $s = 0, \cdots, N$; this is possible if we assume the initial condition of the problem to be fixed. For this reason, we have $\theta = (\mu_{s,i})_{s=0}^{N-1}, (\sigma_{s,i})_{s=0}^{N-1}$, $i = 0, 1$, the control/action signal index. This also gives the necessary gradients:

$$\nabla_{\mu_{s,i}} \ln \pi_\theta(a_i|s) = \frac{\nabla_{\mu_{s,i}} \pi_\theta(a_i|s)}{\pi_\theta(a_i|s)} = \frac{a_i - \mu_{s,i}}{\sigma_{s,i}^2},$$
$$\nabla_{\sigma_{s,i}} \ln \pi_\theta(a_i|s) = \frac{\nabla_{\sigma_{s,i}} \pi_\theta(a_i|s)}{\pi_\theta(a|s)} = \frac{(a_i - \mu_{s,i})^2}{\sigma_{s,i}^2} - 1,$$

which are then used in the parametric update policy. This approach is implemented with random initial parameters. The training steps are presented in Figure 2, where average losses and policy variances are seen to decrease steadily. The decreased variances are consistent with the expectation that, over time, the agent becomes more confident in the action taken and leaves a policy of exploration towards focus on exploitation.

After training, an example of the computed trajectory is presented in 3. It is possible to notice that the resulting control and, more clearly, the resulting trajectory revolve around the optimal solution.

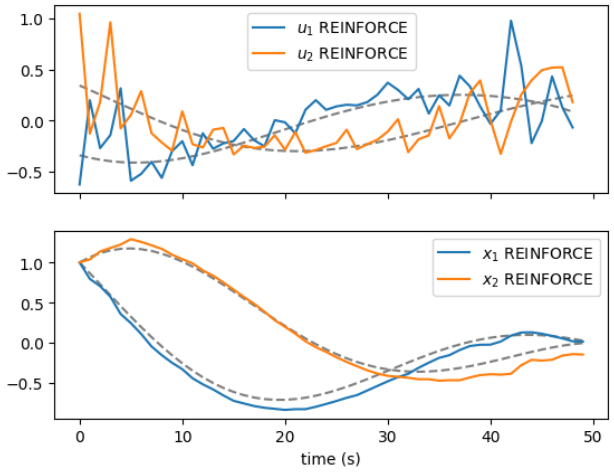

Figure 3: Solution to LQR problem obtained through model-free REINFORCE. In dashed lines, the optimal solution computed previously. Cost realized: $J \approx 15.54$.

We highlight that, although the solution obtained is only suboptimal, it results from a fully model-independent method. For the same reason, the method requires an extensive number of iterations to achieve acceptable performance, which might not be possible in many scenarios, where a limited amount of data is available. In this direction, we compare this approach with a deep-learning-based algorithm, which promises to be more data-efficient, while model-free.

4.3 Actor-Critic Policy Gradient with DDPG

Deep Deterministic Policy Gradient (DDPG) [LHP+19] pertains to a particular class of policy gradient methods called *actor-critic*, which are characterized by the introduction of a *critic*, an approximation to the value function, $\hat{v}_\omega(s)$, or $\mathcal{Q}$-unction, $\hat{Q}_\omega(s, a)$, used to evaluate the quality of actions taken following a given policy, or *actor*.

By using a critic, it is possible to replace the sampled cost-to-go seen previously in REINFORCE with a guess, leading to the following policy update rule:

$$\widehat{\nabla_\theta J(\pi_\theta)} = G_k \nabla_\theta \ln(\pi_\theta(a_k, s_k)) \approx (r_k + \gamma Q_\omega(a_{k+1}, s_{k+1})) \nabla_\theta \ln(\pi_\theta(a_k, s_k))$$

For this reason, the method can update its policy online, or each step at a time, different from the previous one, where a complete episode needed to be observed prior to updating the policy.

Following this framework, In DDPG, a Neural Network is used to approximate the $\mathcal{Q}$-unction, the *critic network*, and a separate Neural Network learns to take actions that minimize this approximation, the *actor network*. Besides that, the authors propose additional improvements, among which two are highlighted:

- Replay Buffer: a finite cache $\mathcal{R}$ containing recent transitions (s_k, a_k, r_k, s_{k+1}) from which past transitions are resampled over time. The use of said buffer increases the data efficiency of the method.
- Soft Updates: at each step, gradient approximation is used to update the actor and critic parameters θ, ω. However, this might be unstable. A way to avoid it is to consider definitive *target* models, with parameters θ', ω' updated softly: $\theta'_{t+1} = (1-\tau)\theta'_t + \tau\theta_{t+1}$, and $\omega'_{t+1} = (1-\tau)\omega'_t + \tau\omega_{t+1}$.

Besides those, the use of other techniques such as batch normalization associated to the running average of the mean and variance and inclusion of random noise for exploration are motivated.

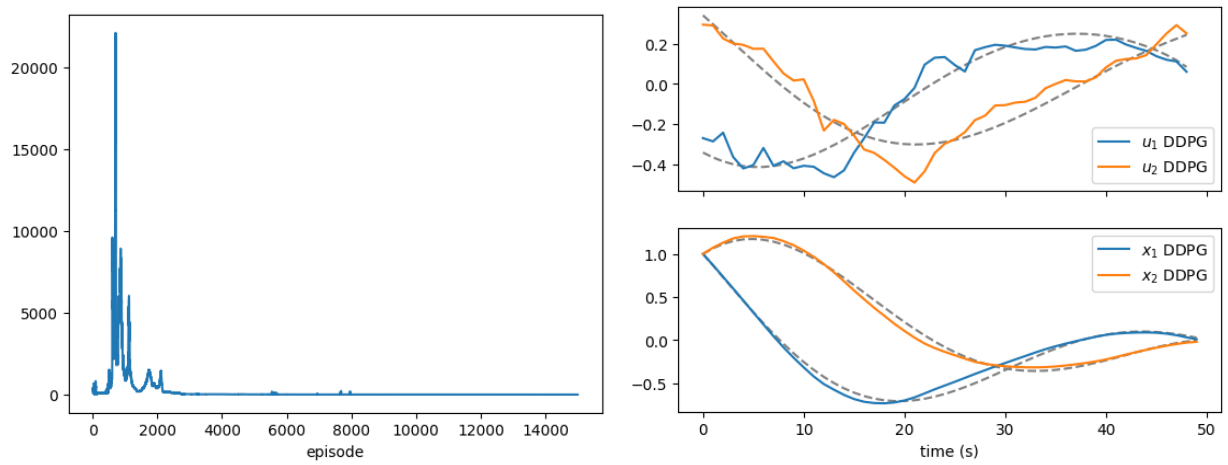

Figure 4: Observed losses over training steps (left) and resulting trajectory and control obtained compared to optimal strategy in dashed lines (right) using DDPG. Cost realized: $J \approx 8.73$.

In Figure 4 we observe the training and resulting solution obtained trough DDPG. We call attention to the smaller number of iteration required to achieve satisfactory results, especially compared to REINFORCE. On the other hand, as discussed by the original authors, the method is prone to instability, which is evident by the sharp spikes in the training loss. Also, while the method is data efficient compared to REINFORCE, it is still computationally costly due to the extensive use of gradients while training the actor and critic neural networks. In fact, while the two fully data-driven approaches deal with the lack of an explicit model, they might still not be viable due to computational or data requirements.

5 Mixed Methodologies

The greatest distinction between the prior methodologies lies in the amount of prior knowledge about the environment. While classical optimal control assumes perfect knowledge of the environment and cost function, reinforcement learning methodologies update this information through interaction. In a sense, however, these techniques are not conflicting but complementary to each other. While classical control provides foundation and structure, reinforcement learning introduces data-driven techniques to address incomplete or complex models. In fact, the boundaries between the methodologies have become unclear, which is seen in recent efforts made to bridge the different communities, such as in [Ber19].

Based on our previous discussions, a method can be proposed to obtain optimal feedback even under uncertain cost and dynamics, in the spirit of, e.g., [Bra92], that is, by only assuming knowledge of the general structure: the matrices A,B,Q,R,F remain unknown for the finite-horizon LQR problem. While an explicit solution can not be presented, it is known that at each time, the optimal feedback is of the form $u_k = K_k x_k$, so the cost-to-go, or value function, can be expressed as:

$$\begin{aligned} v(x_N) &= x_N^\top P_N x_N, \qquad \text{where} \qquad P_N = F \\ v(x_k) &= x_k^\top R x_k + x_k^T K_k^\top Q K_k x_k + x_k^\top (I + A + K_k)^\top F_{k+1}(I + A + K_k) \\ &= x_k^\top (R + K_k^\top Q K_k + (I + A + K_k)^\top F_{k+1}(I + A + K_k)) x_k \\ &=: x_k^\top P_k x_k, \qquad \text{where} \qquad k < N, \end{aligned}$$

and the $\mathcal{Q}$-function has the following quadratic form:

$$\begin{aligned} \mathcal{Q}(x_k, u_k) &= x_k^\top R x_k + u_k^\top Q u_k + x_{k+1}^\top P_{k+1} x_{k+1} \\ &= x_k^\top R x_k + u_k^\top Q u_k + (x_k^\top A^\top + u_k^\top B^\top) P_{k+1}(A x_k + B u_k) \\ &= \begin{bmatrix} x_k^\top & u_k^\top \end{bmatrix} \begin{bmatrix} R + A^\top P_{k+1} A & A^\top P_{k+1} B \\ B^\top P_{k+1} A & Q + B^\top P_{k+1} B \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \\ &=: \begin{bmatrix} x_k^\top & u_k^\top \end{bmatrix} \begin{bmatrix} H_{x,x,k} & H_{x,u,k} \\ H_{u,x,k} & H_{u,u,k} \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \\ &=: \begin{bmatrix} x_k^\top & u_k^\top \end{bmatrix} H_k \begin{bmatrix} x_k \\ u_k \end{bmatrix}, \end{aligned}$$

In fact, this formulation results directly from the following principle, common to the communities of optimal control and reinforcement learning:

Theorem 5 (Principle of Dynamic Programming) *For any initial state (x_k), one has the optimal cost-to-go and $\mathcal{Q}$-function characterized by:*

$$v(x_k) = \max_{u_k} \left[r_k + v(s_{k+1}) \right], \qquad \mathcal{Q}(x_k, u_k) = r_k + \max_{u_{k+1}} \mathcal{Q}(s_{k+1}, u_{k+1}),$$

called the Bellman equation. In the time-continuous case, one has:

$$v(x(t)) = \inf_u \left[v(x(s|u(t), x_t)) + \int_t^s L(x(\tau|u, x_t), u(\tau)) d\tau \right],$$

for $t < s$, where $x(s|u, x_t)$ denotes the trajectory $x(t)$ under the control u and initial condition $x(t) = x_t$. This equation is called the Hamilton-Jacobi-Bellman (HJB) equation.

That is, following the principle of dynamic programming results in an optimal strategy, which is valid in both continuous and discrete cases. This principle is on the core of reinforcement learning methods such as value iteration and $\mathcal{Q}$-learning, but also for necessary and sufficient condition in classical optimal control.

Thus, if the matrices H_k can be identified, an optimal control u_k can be computed by setting the partial derivative $\nabla_w \mathcal{Q}(x_k, w) = 0$, which results in the optimal feedbacks:

$$u_k^* = H_{u,u,k}^{-1} H_{u,x,k} x_k.$$

This is possible even without directly identifying the dynamics matrices (A, B) or cost matrices Q, R, F. In fact, H_k can be estimated under suitable hypotheses over the choice of controls u_k used for adjustment. Observe that, although $\mathcal{Q}$ is not linear with respect to x_k, u_k it is linear with relation to H_k:

$$\mathcal{Q}(x_k, u_k) = z_k^\top H_k z = (z_k \otimes z_k)^\top \mathrm{vec}(H_k),$$

where $z_k = (x_k, u_k)$ and $\otimes$ denotes the Kronecker product. So it can be estimated recursively trough a Linear Regression and Bellman's equation, as follows:

$$Z_N \times \mathrm{vec}(H_N) := \begin{bmatrix} (z_N^{(1)} \otimes z_N^{(1)})^\top \\ \cdots \\ (z_N^{(M)} \otimes z_N^{(M)})^\top \end{bmatrix} \mathrm{vec}(H_N) = \begin{bmatrix} r_N^{(1)} \\ \cdots \\ r_N^{(M)} \end{bmatrix}$$

for the final state, and for prior states, $k < N$, one gets:

$$Z_k \times \mathrm{vec}(H_k) := \begin{bmatrix} (z_k^{(1)} \otimes z_k^{(1)})^\top \\ \cdots \\ (z_k^{(M)} \otimes z_k^{(M)})^\top \end{bmatrix} \mathrm{vec}(H_k) = \begin{bmatrix} r_k^{(1)} \\ \cdots \\ r_k^{(M)} \end{bmatrix} + \begin{bmatrix} (z_{k+1}^{(*,1)} \otimes z_{k+1}^{(*,1)})^\top \\ \cdots \\ (z_{k+1}^{(*,M)} \otimes z_{k+1}^{(*,M)})^\top \end{bmatrix} \mathrm{vec}(H_{k+1})$$

where $z_k^{(m)} = (x_k^{(m)}, u_k^{(m)})$ denotes the k-th step of the m-th trajectory observed, for a number of samples M, and $z_k^{(*,m)} = (x_k, u_k^*)$ with $u_k^{(m)} = H_{u,u,k}^{-1} H_{u,x,k} x_k^{(m)}$ is the minimizing control. By solving the problem backwards in time, one obtains exact estimations in a single step.

5.1 Importance of Exploration

The last requirement necessary for the identification of the $\mathcal{Q}$-function through regression is that the feature matrices Z_k, whose rows consist of the elements $(z_k^{(m)} \otimes z_k^{(m)})^\top$, result in a solvable system. However, one may notice immediately the following:

- if all trajectories observed start at the same initial states and policy, then $\mathrm{rank}(Z_k) = 1$, since all states explored are the same and, thus, rows $(z_k^{(m)} \otimes z_k^{(m)})^\top$ are identical;
- alternatively, even starting at distinct initial states, the particular format of the policy $u_k = K_k x_k$, consisting of linear feedbacks, may still result in a singular features matrix.

We avoid the issue in question by imposing that the solutions are informative, namely *persistently exciting* condition:

$$Z_k^\top Z_k = \sum_{m=1}^{M} (z_k^{(m)} \otimes z_k^{(m)})(z_k^{(m)} \otimes z_k^{(m)})^\top \succ 0,$$

that is, $Z_k^\top Z_k$ is positive definite, which ensures that the system of linear equations has a unique solution trough Moore–Penrose inverse.

Interestingly, in practice, this requirement is achieved by simply introducing noise on the learning phases. So, to introduce informative noise into our data, exploration is done by adding normal deviations from the policy: $u_k^{(m)} = K_k x_k^{(m)} + \epsilon e_k^{(m)}$, where $0 < \epsilon$ is an exploration scale, and $e_k^{(m)} \sim \mathcal{N}(0, 1)$ normal deviations. This discussion clarifies that *exploration* is not only an algorithm peculiarity in reinforcement learning, but a mathematically justified requirement.

Finally, we implement the solution proposed taking $M = 10$, the number of degrees of freedom of H_k, symmetrical. The results are presented in Figure 5, which not only resembles the optimal solution computed previously but, in fact, even surpasses it by a small factor. This probably results from the use of numerical schemes to approximate the solutions to the RDE, compared with direct use of Bellman's equation in the current solution.

5.2 Model Predictive Control

A final consideration can be made on the increasingly unclear distinction between reinforcement learning and recent efforts on data-driven, with special distinction to the field of Model Predictive Control (MPC). For instance, [BKMA21] discusses an approach to obtain linear control feedback of an LTI system without

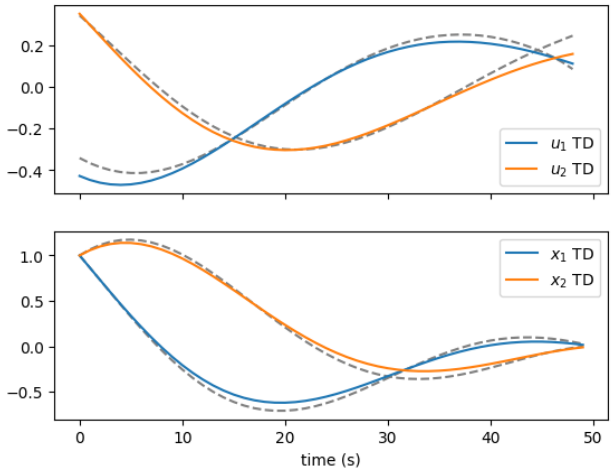

Figure 5: Trajectory obtained through exact $\mathcal{Q}$-learning. Total Cost Achieved: $J \approx 8.06$.

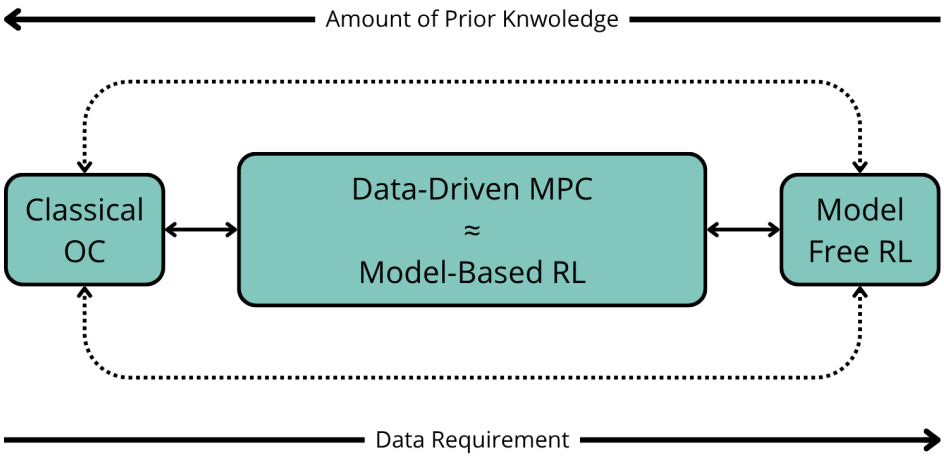

Figure 6: Different paths from classical optimal control to model-free reinforcement learning

system identification under the persistently exciting hypothesis. The method closely resembles that of our discussion.

This close resemblance exists between the two extremes and motivates the existence of continuous paths from classical model-based control to model-free reinforcement learning. The lack of prior information of the model is complemented by data and exploration, as highlighted in Figure 6.

6 Discussion

We walked through different classes of methods from the point of view of Linear Quadratic Regulator stabilization problems, from classical optimal control to reinforcement learning and intermediate approaches. This discussion has the value of highlighting the advantages and drawbacks of the different methods, and their complementarities. That is, while classical model-based OC provides us with well-established and efficient solutions, it is viable under possibly restrictive prior knowledge about the problems. On the other hand, we saw that different fully model-free reinforcement learning techniques may be used to identify

and recommend (sub-)optimal actions even under extreme lack of information; however, this may become data and computationally demanding, and even in that case the identification of optimal strategies is not guaranteed.

On this light, we presented and discussed a method in the intersection of both communities, which showed to be both computationally and sample-efficient, by making use of well established theoretical control results and data-driven techniques. We highlight that, while the approach seems *too close* to classical optimal control, it remains distinct by avoiding direct system and cost identification. Instead, only the $\mathcal{Q}$-function is identified, which also evidentiates the similarity of the method with data-driven off-policy reinforcement learning. In fact, this is the perfect example of how the line distinguishing the two communities is, in fact, blurry, and how techniques from both universes may complement each other.

6.1 Beyond Linear Models

A similar discussion can be extended beyond linear models and LQR problems. Established optimal control theory includes controllability results for nonlinear models through Lie brackets, and the optimality conditions through the Pontryagin Maximum Principle (PMP) and the Hamilton-Jacobi-Bellman (HJB) partial differential equation. These methods are, however, model-based and might result in computationally or analytically complex conditions.

In a similar style, Reinforcement Learning presents modern techniques suited for increasingly complex scenarios, but might be data- and computationally hungry when no underlying solution structure can be assumed.

In that sense, similar to the linear case, the complementary nature of these fields can be seen in works on increasingly complex systems, where OC is used for the analysis and structure of optimal solutions, such as done in [SGS+21] to reduce the dimension of the learning problem, or even on the analysis and application of black-box models, such as done in [RBZ23].

References

[Ber19] Dimitri P. Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, 2019.

[BKMA21] Julian Berberich, Johannes Kohler, Matthias A. Muller, and Frank Allgower. Data-driven model predictive control with stability and robustness guarantees. *IEEE Transactions on Automatic Control*, 66(4):1702–1717, April 2021.

[Bra92] Steven Bradtke. Reinforcement learning applied to linear quadratic regulation. In S. Hanson, J. Cowan, and C. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann, 1992.

[Hes18] João P. Hespanha. *Linear Systems Theory: Second Edition*. Princeton University Press, ned - new edition, 2 edition, 2018.

[LHP+19] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[RBZ23] Domènec Ruiz-Balet and Enrique Zuazua. Neural ode control for classification, approximation, and transport. *SIAM Review*, 65(3):735–773, 2023.

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[SGS+21] Tim Seyde, Igor Gilitschenski, Wilko Schwarting, Bartolomeo Stellato, Martin A. Riedmiller, Markus Wulfmeier, and Daniela Rus. Is bang-bang control all you need? solving continuous control with bernoulli policies. *CoRR*, abs/2111.02552, 2021.

[Ted23] Russ Tedrake. *Underactuated Robotics*. Course Notes for MIT 6.832, 2023.