On Sample Complexity of LQR

Pulkit Katdare
Department of Electrical and Computer Engineering
University of Illiinois at Urbana-Champaign
Urbana, IL 61801
`katdare2@illinois.edu`

Abstract

Deep Reinforcement learning has shown a lot of promises since it's seminal paper on Deep Q-networks. However, not a lot of theoretical framework exists in Deep RL regime which provides some sort of a convergence guarantees. One of the alternative that people came up with was Model-based RL, wherein part of the algorithm tries to deal with understanding the model of the system and another part tries to solve the optimization problem. In the main paper that we review here, they have exploited the concepts of a model-based RL methods to solve an LQR problem. Not only does the paper provide convergence guarantees of the estimation of the model parameters, the paper also provides sample complexty guarantees to learn this LQR

1 Introduction

Reinforcement learning is the science of learning decision making so that it minimizes some notion of cumulative cost function. Reinforcement learning problems are generally defined as an MDP, which is worthy of a mention here. An MDP is defined as a 6-tuple as follows,

$$MDP = (S, A, P, c, \gamma, S_0) \tag{1}$$

Where S is the state space of the MDP, A is the action space of the MDP, P is the (stochastic) model of the system. S_0 is the set of initial states of the MDP and for the understanding the γ and c, we need to look the the optimization problem in Reinforcement learning which is defined as follows,

$$\pi^* = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | s_0 \sim S_0, a_t = \pi(s_t)] \tag{2}$$

Now getting back to the MDP, we defined one step cost of the taking action a_t at s_t as $c(s_t, a_t)$ and γ is the impact of future actions may have on the current optimization problem. Note that, the function $\pi(.)$ is called the policy function which spits action to take at state s_t. For someone with the control theory background this is a very general background on how optimal control problems are posed, which are going to be presenting next.

1.1 Linear Quadratic Regulator

After going through an MDP formulation consider a Control system where the state space is $s \in \mathbb{R}^n$ and the action space $a \in \mathbb{R}^p$ and evolution of this control system proceeds as follows,

$$s_{t+1} = f(s_t, a_t, w_t) \tag{3}$$

Here, s_t, a_t os the action taken at time t and w_t is the uncertainty in the system (could be noise) and the next state is s_{t+1}. Moving ahead, the objective of an optimal control problem is to minimize the following cost function as follows,

$$(u_0^*, u_1^*, ...) = arg \min_{(u_0,u_1,...)} \sum_{t=0}^{\infty} c(s_t, a_t) \tag{4}$$

In this particular paper that we are reviewing [1] wherein the authors assume that the model of system is Linear and the cost is quadratic.

$$s_{t+1} = As_t + Ba_t + w_t \tag{5}$$

$$c(s_t, a_t) = s_t^T Q s_t + a_t^T R a_t \tag{6}$$

In the main paper that we review, the algorithm proceeds in two parts. In the first part it tries to estimate the matrix A and B and in the second part using the estimates of A and B it solves the Linear Quadratic Regulator problem. Let us now proceed to the algorithm.

2 Related Work

2.1 Reinforcement learning applied to Linear Quadratic Regulator

Note that using MDPs to solve LQR problem isn't a new thing in the Reinforcement learning literature. One of the first approaches in this area was a 1993 paper [2] where in it is assumed that the system is exactly linear in the LQR problem i.e

$$s_{t+1} = As_t + Bu_t \tag{7}$$

Note that is assumed that $w_t = 0$. Under no noise condition it is a known fact that for LQR cost function is quadratic is states as follows if the policy is linear in state, i.e $\pi(s) = \mathbf{K}s$

$$V(s) = s^T U_K s \tag{8}$$

Let's take a step back and define what value function exactly is. The value function of over state for a particular policy is defined as follows,

$$V_K(s) = \sum_{t=0}^{\infty} \gamma^t (s_t^T Q s_t + a_t^T R a_t) | s_0 = s \tag{9}$$

To solve this LQR problem, the paper uses Q-learning. Let us first understand the definition of the Q-function which is as follows,

$$Q_K(s, a) = (s_0^T Q s_0 + a_0^T R a_0 + \sum_{t=1}^{\infty} s_t^T Q s_t + a_t^T R a_t | a_t = \mathbf{K}s_t, s_0 = s, a_0 = a) \tag{10}$$

The Paper goes around to prove that the Q-function is quadratic in state and action spaces. Using equation 9 we get the following result,

$$\begin{aligned}
Q_K(s, a) &= s^T Q s + a^T R a + \gamma V_K(s) \\
&= s^T Q s + a^T R a + \gamma (As + Ba)^T U_K (As + Ba) \\
&= s^T Q s + a^T R a + \gamma ((s^T A^T U_K + a^T B^T U_K)(As + Ba) \\
&= [s \quad a]^T \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} [s \quad a] + \gamma ([s \quad a]^T \begin{bmatrix} A^T U_K A & A_T U_K B \\ B^T U_K A & B^T U_K B \end{bmatrix} [s \quad a]) \\
&= [s \quad a]^T \begin{bmatrix} Q + \gamma A^T U_K A & \gamma A_T U_K B \\ \gamma B^T U_K A & R + B^T U_K B \end{bmatrix} [s \quad a]^T \\
&= [s \quad a]^T H_K [s \quad a]
\end{aligned} \tag{11}$$

Since the matrix A and B are unknown to us even the matrix H_K and U_K are unknown to us. So, how do we solve this problem. This paper essentially provides two algorithms around this problem. In the first algorithm, the authors essentially estimate the matrix H_K, $\hat{H}_K$ for a certain K. After that, the algorithm updates the policy using policy improvement step. Wherein $K_{new} = -(\hat{H}_K)_{22}^{-}1(\hat{H}_K)_{21}$

In the second algorithm [2] provides another algorithm for estimating the optimal K which is updated on the go. Here, the paper tries to estimate the optimal Q-function Q^* with changing the policy on the go such that it finally converges to Q^*.

2.2 PAC Adaptive Control of Linear Systems

In this paper [3] the authors again try to solve an LQR problem for a linear system but this time the system does have a zero mean noise added to it. The idea is to again come up with an algorithm which is shown to be PAC learnable such that the the system learns an optimal policy. But the idea of the algorithm is simple, first the algorithm estimates the matrix A and B using a linear regressor. Then the algorithm tries to figure out the optimal control of this LQR problem using the estimates of the matrix A and B that have been calculated.
First again define the Linear Control system that we know, which is as follows,

$$x_{t+1} = Ax_t + Bu_t + w_t \tag{12}$$

where, $x_t \in \mathbb{R}^n$ and the control, $u_t \in \mathbb{R}^p$ and the noise w_t is such that,

$$\begin{aligned} \mathbb{E}[w_t] = 0 \quad \forall t \geq 0 \\ Con(w_k, w_j) = \sigma^2 I_n \mathbf{1}_{k=j} \quad \forall k, j \geq 0 \end{aligned} \tag{13}$$

It is well-known fact that the solution of this LQR problem can be described by a Linear controller as follows,

$$\begin{aligned} \pi^*(s) &= -L^* s \\ L^* &= \gamma(\gamma B^T KB + R)^{-1} B^T KA \\ K &= A^T[\gamma K - \gamma^2 KB(\gamma B^T KB + R)^{-1} B^T K]A + Q \end{aligned} \tag{14}$$

Where, the third equation a.k.a can also be solved using Ricatti equation. To determine the regressor, we first define the regressor variables, θ as follows,

$$\theta = [B \quad A]^T \tag{15}$$

We also define a $r = n + p$ dimensional vector as follows,

$$\varphi_k = \begin{bmatrix} u_k \\ y_k \end{bmatrix} \tag{16}$$

If we define, $y_k = x_k^T$ then we get the following updates as follows,

$$y_{k+1} = \varphi_k \theta + w_k^T \tag{17}$$

Since, this is a Regression problem the optimal value of this least-squares regression can be described as follows,

$$\hat{\theta}_m = (\Phi_m^T \Phi_m)^{-1} \Phi_m^T Y_m \tag{18}$$

Where, $_m = \begin{bmatrix} y_0 \\ y_1 \\ \cdot \\ \cdot \\ y_m \end{bmatrix}$, $\Phi_m = \begin{bmatrix} \phi_0 \\ \phi_1 \\ \cdot \\ \cdot \\ \phi_m \end{bmatrix}$ and $W_m = \begin{bmatrix} w_1 \\ w_2 \\ \cdot \\ , \\ w_m \end{bmatrix}$. For the estimation problem we need to update the parameters iteratively with every new data point that comes in. But before doing that lets define the following terms as follows,

$$P_m = Cov(\Phi_m^T, \Phi_m)^{-1} \tag{19}$$

$$Cov(\hat{\theta}(i), \hat{\theta}(i)) = \sigma^2 P_m \tag{20}$$

Thus, the iterative update in the θ with any new data point is as follows,

$$\hat{\theta}_m = \hat{\theta}_m + P_m \varphi_m (y_m - \varphi_m \hat{\theta}_{m-1}) \tag{21}$$

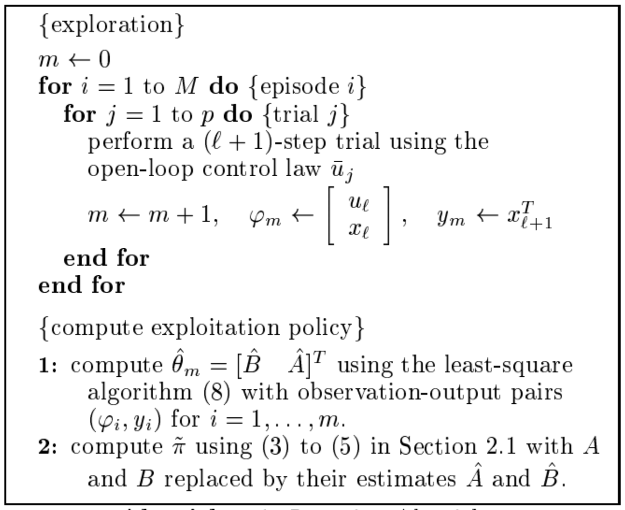
Even the matrix, P_m can be recursively updates as follows,

$$P_m = P_{m-1} - \frac{P_{m-1} \varphi_m \varphi_m^T P_{m-1}}{1 + \varphi_m^T P_{m-1} \varphi_m} \tag{22}$$

Putting it all together the algorithm looks something like this, Where, $\bar{u}_i(j) = \mathbf{1}_{j=i} \quad \forall j = \{1, 2, .., p\}$. Using these approach the authors propose a novel-algorithm wherein they prove PAC learnability of this Linear system as follows,

$$Prob(\frac{v^{\pi^*}(x) - v^{\hat{\pi}(x)}}{1 + \|x\|^2} < \epsilon) \geq 1 - \delta \tag{23}$$

Given that the algorithm is run for M-number of steps which will be determined later. Where, $M = \mathcal{O}(\frac{1}{\delta \epsilon^2})$.



```
{exploration}
$m \leftarrow 0$
for $i = 1$ to $M$ do {episode $i$}
  for $j = 1$ to $p$ do {trial $j$}
    perform a $(\ell + 1)$-step trial using the
    open-loop control law $\bar{u}_j$
    $m \leftarrow m + 1, \quad \varphi_m \leftarrow \begin{bmatrix} u_\ell \\ x_\ell \end{bmatrix}, \quad y_m \leftarrow x_{\ell+1}^T$
  end for
end for

{compute exploitation policy}
1: compute $\hat{\theta}_m = [\hat{B} \quad \hat{A}]^T$ using the least-square
     algorithm (8) with observation-output pairs
     $(\varphi_i, y_i)$ for $i = 1, \ldots, m$.
2: compute $\tilde{\pi}$ using (3) to (5) in Section 2.1 with $A$
     and $B$ replaced by their estimates $\hat{A}$ and $\hat{B}$.
```

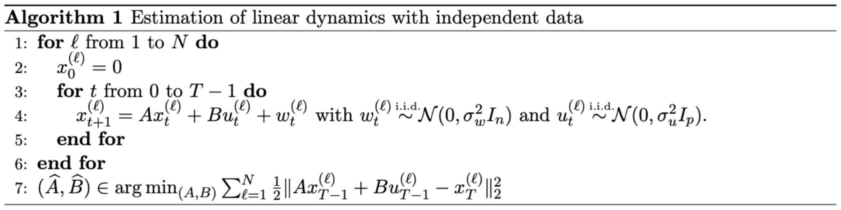
Algorithm 1: Learning Algorithm

3 Sample Complexity of the Linear Quadratic Regulator

The premise of this paper is very simple. Assuming that you have a Linear system which can be defined as follows,

$$x_{t+1} = Ax_t + Bu_t + w_t \tag{24}$$

Where, the notation follows from the previous sections. The idea of this paper is to not only estimate the matrix A and B with a reasonable accuracy but also to estimate the uncertainty in the estimation and pose the optimization problem as a constrained optimization problem which are subject to uncertainties. The paper also claims that the sample complexity to learn this particular kind of controller can be bounded by $\mathcal{O}(\log(1/\delta))$. So, let's proceed ahead with this in mind. In the first step of the algorithm the paper first estimates the matrix A and B. As seen in section 2.2 we saw a least squares estimation to estimate A and B. But due to inability to prove rigorous bounds on A and B, this paper uses a different kind of approach to estimate A and B. Using an identification procedure



```
Algorithm 1 Estimation of linear dynamics with independent data
1: for $\ell$ from 1 to $N$ do
2:   $x_0^{(\ell)} = 0$
3:   for $t$ from 0 to $T-1$ do
4:     $x_{t+1}^{(\ell)} = Ax_t^{(\ell)} + Bu_t^{(\ell)} + w_t^{(\ell)}$ with $w_t^{(\ell)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_w^2 I_n)$ and $u_t^{(\ell)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_u^2 I_p)$.
5:   end for
6: end for
7: $(\widehat{A}, \widehat{B}) \in \arg\min_{(A,B)} \sum_{\ell=1}^{N} \frac{1}{2} \|Ax_{T-1}^{(\ell)} + Bu_{T-1}^{(\ell)} - x_T^{(\ell)}\|_2^2$
```

Figure 1: system Identification algorithm

as described above we can estimate the accuracy of the estimation which can be bounded by $\mathcal{O}(1/\delta)$. We have the following theorem for the proof of this,

Theorem 1. *For $N > 8(n+p) + 16\log(4/\delta)$ and assuming $w_t \sim \mathcal{N}(0, \sigma_u^2 I_p)$ and $u_t \sim \mathcal{N}(0, \sigma_n^2 I_n)$ and $x_0 = 0$. Then with probablity atleast $1 - \delta$. The error in estimation of A and $\hat{A}$, B and $\hat{B}$ can be upper-bounded as follows,*

$$\|\hat{A} - A\| \leq \frac{16\sigma_w}{\sqrt{\lambda_{min}(\sigma_u^2 G_T G_T^* + \sigma_w^2 F_T F_T^*)}} \sqrt{\frac{(n+2p)\log(36/\delta)}{N}} \tag{25}$$

$$\|\hat{B} - B\| \leq \frac{16\sigma_w}{\sigma_u} \sqrt{\frac{(n+2p)\log(36/\delta)}{N}} \tag{26}$$

Where, $G_T = [A^{T-1}B \quad A^{T-2}B \quad ...B]$ and $F_T = [A^{T-1} \quad A^{T-2} \quad ...I_n]$. And λ_{min} is the minimum eigen value which will always be positive.

To prove this theorem, consider the following Lemmas. We are going to accept this without proof for now.

Lemma 1. *For $\delta \in (0,1)$ and $N \geq 2\log(1/\delta)$. Let $f_k, g_k \in \mathbb{R}^m$ with f, g distributed as $\mathcal{N}(0, \Sigma_f)$ and $\mathcal{N}(0, \Sigma_g)$. Then the following statement holds true with probability, $1-\delta$*

$$\|\sum_{k=1}^{N} f_k g_k^T\|_2 \leq 4\|\Sigma_f\|_2^{1/2}\|\Sigma_g\|_2^{1/2}\sqrt{N(m+n)\log(9/\delta)} \tag{27}$$

Lemma 2. *If $X \in \mathbb{R}^{N\times n}$ are i.i.d with $\mathcal{N}(0,1)$. Then with probability at least $1-\delta$*

$$\sqrt{\lambda_{min}(X^TX)} \geq \sqrt{N} - \sqrt{n} - \sqrt{2\log(1/\delta)} \tag{28}$$

Using the first two lemma we can prove the following lemma which will help in the proof of the system identification.

Lemma 3. *Let $z_1, z_2, ...z_N \in \mathbb{R}^n$ distributed as $\mathcal{N}(0,\Sigma)$. Let $W \in \mathbb{R}^{N\times p}$ distributed as $\mathcal{N}(0, \sigma_w^2)$. Let us define $E = (Z^TZ)^{\dagger}Z^TW$. Then if $N \geq 8n + 16\log(2/\delta)$. For a fixed matrix, Q and probablity $1-\delta$ the following holds true.*

$$\|QE\|_2 \leq 16\sigma_w\|Q\Sigma^{-1/2}\|_2\sqrt{\frac{(n+p)\log(18/\delta)}{N}} \tag{29}$$

Proof. Notice that if Z can be written as the following variable,

$$Z = Y\Sigma^{1/2} \tag{30}$$

It is clear from the definition that $Y \sim \mathcal{N}(0,1)$. Using Lemma 2 we can conclude with probablity atleast $1-\delta/2$ that

$$\sqrt{\lambda_{min}(Y^TY)} \geq \sqrt{N} - \sqrt{n} - \sqrt{2\log(2/\delta)} = \sqrt{N} - (\sqrt{n} + \sqrt{2\log(2/\delta)}) \tag{31}$$

Now, it is a well known inequality in math that $(a+b)^2 \leq 2(a^2+b^2)$. Thus, we have $-(a+b) \geq -sqrt2(a^2+b^2)$. substituting this in the above inequality we have the following,

$$\sqrt{\lambda_{min}(Y^TY)} \geq \sqrt{N} - \sqrt{2n + 4log(2/\delta)} = \sqrt{N} - \frac{\sqrt{8n + 16log(2/\delta)}}{2} \geq \sqrt{N} - \sqrt{N}/2 \geq \sqrt{N}/2 \tag{32}$$

Using Lemma 1 combined with the above inequality we get the following upper-bound with probablity atleast $1-\delta/2$

$$\|Y^*W\| \leq 4\sigma_w\sqrt{N(n+p)\log(18/\delta)} \tag{33}$$

Assuming that the above two events are called ϵ_1 and ϵ_2. Then the probability of atleast on these events occuring is atmost, δ (By union bound of probabilities). With probablity of atleast $1-\delta$ that none of these events would occur, we can state the following,

$$\|QE\|_2 = Q(Z^TZ)^{\dagger}Z^TW = Q^{-1/2}(Y^*Y)^{-1}Y^*W \tag{34}$$

Taking 2-norm on both sides and using above two inequalities and Lemma 1. we get the above result. □

Now, lets proceed to the proof of theorem 1.

Proof. (Theorem 1.) Since the estimation algorithm starts from $x_0 = 0$ always with the control inputs being normal, we can calculate this an say the following,

$$\begin{bmatrix} x_T \\ u_T \end{bmatrix} \sim \mathcal{N}(0, \begin{bmatrix} \sigma_u^2 G_T G_T^T + \sigma_w^2 F_T F_T^T & 0 \\ 0 & \sigma_u^2 I_p \end{bmatrix} \tag{35}$$

Assume that we can define, $z_t = \begin{bmatrix} x_t \\ u_t \end{bmatrix}$. Let us define $\Theta = [A \quad B]^T$. Using these things we have,

$$x_t^T = z_t^T \Theta + w_t^T \tag{36}$$

If we now define, $X = \begin{bmatrix} x_1^T \\ x_2^T \\ \cdot \\ \cdot \\ \cdot \\ z_T^T \end{bmatrix}$, $Z = \begin{bmatrix} z_0^T \\ z_1^T \\ \cdot \\ \cdot \\ \cdot \\ z_{T-1}^T \end{bmatrix}$ and $W = \begin{bmatrix} w_1^T \\ w_2^T \\ \cdot \\ \cdot \\ \cdot \\ w_T^T \end{bmatrix}$ Thus, a single rollout can be written in the matrix form as follows,

$$Z = X\Theta + W \tag{37}$$

Call X_N, Z_N, W_N such vertical stacked matrices of all such rollouts one after the another. Then the following updates are also true

$$X_N = Z_N\Theta + W_N \tag{38}$$

Thus, error of the least squares estimator can be written as follows,

$$E = \hat{\Theta} - \Theta = (Z_N^T Z_N)^{-1} Z_N^T W_N \tag{39}$$

Hence, using Lemma 3 and assuming the conditions of Lemma 3 and for $Q_A = [I_n \quad 0]$.

$$\|\hat{A} - A\|_2 \leq \frac{16\sigma_w}{\sqrt{\lambda_{min}(\sigma_u^2 G_T G_T^* + \sigma_w^2 F_T F_T^*)}} \sqrt{\frac{(n+2p)\log(36/\delta)}{N}} \tag{40}$$

Similarly for $Q = [0 \quad I_p]$ we get,

$$\|\hat{B} - B\| \leq \frac{16\sigma_w}{\sigma_u} \sqrt{\frac{(n+2p)\log(36/\delta)}{N}} \tag{41}$$

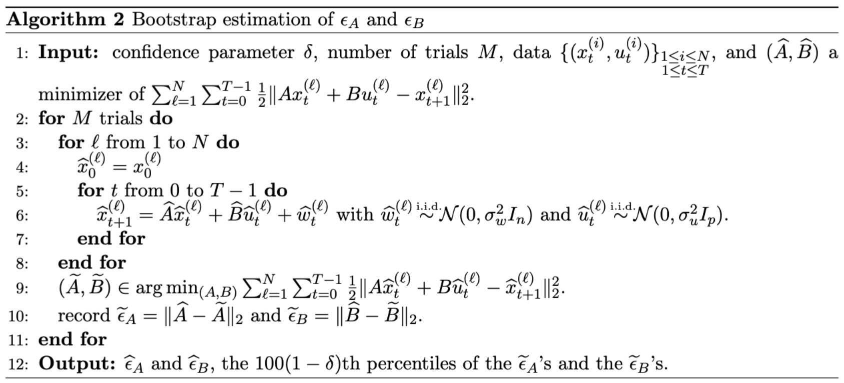
Hence Proved. □

4 Estimating Model Uncertainity

Now, we have estimated the control matrix, A and B it is now time to re-write the optimal control problem subject to these uncertainities. Let us write this down in mathematical form as follows,

$$\begin{aligned} minimize \quad & \sup_{\substack{\|\Delta_A\| \leq \epsilon_A \\ \|\Delta_B\| \leq \epsilon_B}} \mathbb{E}[\sum_{t=0}^{\infty} x_t^T Q x_t + u_{t-1}^T R u_{t-1}] \\ \text{subject to} \quad & x_{t+1} = (\hat{A} + \Delta_A)x_t + (\hat{B} + \Delta_B)u_t + w_t \end{aligned} \tag{42}$$

To solve this optimization problem we first need an estimate of ϵ_A and ϵ_B. Note that we do have an upper-bound on the estimation from theorem 1. but because it is impossible to calculate this upper bound as it involves terms the actual A and B matrices. Thus, to estimate this the paper uses a boot-strapping framework as demonstrated in the algorithm below.



Algorithm 2 Bootstrap estimation of ϵ_A and ϵ_B

```
1: Input: confidence parameter $\delta$, number of trials $M$, data $\{(x_t^{(i)}, u_t^{(i)})\}_{1\leq i\leq N, 1\leq t\leq T}$, and $(\widehat{A}, \widehat{B})$ a
   minimizer of $\sum_{\ell=1}^{N}\sum_{t=0}^{T-1}\frac{1}{2}\|Ax_t^{(\ell)} + Bu_t^{(\ell)} - x_{t+1}^{(\ell)}\|_2^2$.
2: for $M$ trials do
3:     for $\ell$ from 1 to $N$ do
4:         $\widehat{x}_0^{(\ell)} = x_0^{(\ell)}$
5:         for $t$ from 0 to $T-1$ do
6:             $\widehat{x}_{t+1}^{(\ell)} = \widehat{A}\widehat{x}_t^{(\ell)} + \widehat{B}\widehat{u}_t^{(\ell)} + \widehat{w}_t^{(\ell)}$ with $\widehat{w}_t^{(\ell)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_w^2 I_n)$ and $\widehat{u}_t^{(\ell)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_u^2 I_p)$.
7:         end for
8:     end for
9:     $(\widetilde{A}, \widetilde{B}) \in \arg\min_{(A,B)} \sum_{\ell=1}^{N}\sum_{t=0}^{T-1}\frac{1}{2}\|A\widehat{x}_t^{(\ell)} + B\widehat{u}_t^{(\ell)} - \widehat{x}_{t+1}^{(\ell)}\|_2^2$.
10:    record $\widetilde{\epsilon}_A = \|\widehat{A} - \widetilde{A}\|_2$ and $\widetilde{\epsilon}_B = \|\widehat{B} - \widetilde{B}\|_2$.
11: end for
12: Output: $\widehat{\epsilon}_A$ and $\widehat{\epsilon}_B$, the $100(1-\delta)$th percentiles of the $\widetilde{\epsilon}_A$'s and the $\widetilde{\epsilon}_B$'s.
```

Figure 2: Estimating model uncertainty

Using existing framework for estimation using boot-strapping algorithms it can be easily proven that the estimate, $\hat{\epsilon}_A$ and $\hat{\epsilon}_B$ provides the following guarantee

$$\mathbb{P}(\|\hat{A} - A\| \leq \hat{\epsilon}_A) \geq 1 - \delta \tag{43}$$

$$\mathbb{P}(\|\hat{B} - B\| \leq \hat{\epsilon}_B) \geq 1 - \delta \tag{44}$$

Since, we now have estimates of the matrix A and B along with their uncertainity estimation. We can proceed now to solve this optimization problem. Note that the exact solution with zero uncertainties is a know problemc in optimal control theory which is solved by much popular Ricatti equation. In this robust optimization problem they borrowed one small thing from that method, i.e the optimal control is a linear function of state $u^* = Kx^*$. The authors try to solve this problem using a control theory method called system level synthesis. We wiill go through this next and then comes the sample complexity result.

5 System Level Synthesis

The main idea of System Level Synthesis is to cast this control problem in the transfer function domain. The idea behind this is the fact that, this unconstrained optimization turns into a constrained one. As mentioned in equation 35 the equation from x_t can be written in normals such that,

$$\begin{bmatrix} x_k \\ u_k \end{bmatrix} = \sum_{t=1}^{k} \begin{bmatrix} (A+BK)^{k-t} \\ K(A+BK)^{k-t} \end{bmatrix} w_{t-1} = \sum_{t=1}^{k} \begin{bmatrix} \Phi_x(k-t+1) \\ \Phi_u(k-t+1) \end{bmatrix} w_{t-1} \tag{45}$$

It is very clear from the formulation that ϕ_x and ϕ_u should satisfy the following conditions as follows,

$$\phi_x(k+1) = A\phi_x(k) + B\phi_u(k) \quad \phi_x(1) = I \tag{46}$$

Now, since we are converting this in transfer function domain we define,

$$\mathbf{\Phi_x}(z) = \sum_{k=1}^{\infty} \phi_x(k) z^{-k} \tag{47}$$

Thus, for this domain we have the following affine conditions that needs to be satisfied

$$[zI - A \quad B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix} = I \tag{48}$$

Note that in this particular case, $K = \mathbf{\Phi_u}\mathbf{\Phi_x}^{-1}$

Theorem 2. *For $\forall \mathbf{\Phi_x}, \mathbf{\Phi_u}$ the controller, $K = \mathbf{\Phi_u}\mathbf{\Phi_x}^{-1}$ is internally stabilizing if the following affine conditions are satisfied.*

$$[zI - A \quad B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix} = I \tag{49}$$

Theorem 3. *Suppose the transfer matrices $\{\mathbf{\Phi_x}, \mathbf{\Phi_u}\}$ satisfies*

$$[zI - A \quad B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix} = I + \Delta \tag{50}$$

If $\|\Delta\| \leq 1$ then the controller $K = \mathbf{\Phi_u}\mathbf{\Phi_x}^{-1}$ stabilizes the system (A, B)

Since in our problem, $\Delta_A = A - \hat{A}$ and $\Delta_B = B - \hat{B}$. Using this as our baseline we can recast our optimization problem as follows,

$$\min_{\phi_\mathbf{x}, \phi_\mathbf{u}} \sigma_w^2 \| \begin{bmatrix} Q^{1/2} & 0 \\ 0 & R^{1/2} \end{bmatrix} \begin{bmatrix} \phi_\mathbf{x} \\ \phi_\mathbf{u} \end{bmatrix} \|^2 \quad \text{such that equation 49 holds} \tag{51}$$

Given that this holds, the following theorem holds,

Theorem 4. *If a controller* $\mathbf{K}$ *stabilizes* $(\hat{A}, \hat{B})$ *and* $\phi_{\mathbf{x}}, \phi_{\mathbf{u}}$ *is the corresponding system response over estimated system. Suppose that if* K *stabilizes* (A, B) *too then the actual cost achived on this system is as follows,*

$$J(A,B,K) = \left\| \begin{bmatrix} Q^{1/2} & 0 \\ 0 & R^{1/2} \end{bmatrix} \begin{bmatrix} \phi_{\mathbf{x}} \\ \phi_{\mathbf{u}} \end{bmatrix} (I + [\Delta_A \quad \Delta_B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix})^{-1} \right\|_2 \tag{52}$$

Now suppose $\hat{\Delta} = [\Delta_A \quad \Delta_B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix}$ *then a sufficient condition from theorem 3 to stabilize* (A, B) *is that* $\|\hat{\Delta}\|_\infty < 1$

From the above theorem it is very clear that,

$$J(A,B,K) \leq \|(I+\hat{\Delta})^{-1}\|_\infty J(\hat{A},\hat{B},K) \leq \frac{1}{1-\|\hat{\Delta}\|_\infty} J(\hat{A},\hat{B},K) \tag{53}$$

Since, $\hat{A}, \hat{B}$ are known to us, $J(\hat{A}, \hat{B}, K)$ can calculated by us. The only thing remaining is to upper-bound $\|\hat{\Delta}\|_\infty$

Proposition 1. *For any* $\alpha \in (0, 1)$ *and* $\hat{\Delta}$*, the following holds true.*

$$\|\hat{\Delta}\|_\infty \leq \left\| \begin{bmatrix} \frac{\epsilon_A}{\sqrt{\alpha}} \mathbf{\Phi_x} \\ \frac{\epsilon_B}{\sqrt{1-\alpha}} \mathbf{\Phi_u} \end{bmatrix} \right\| = H_\alpha(\phi_{\mathbf{x}}, \mathbf{\Phi_u}) \tag{54}$$

Thus, we have the following bound for $J(A, B, K)$ *as follows,*

$$J(A,B,K) \leq \frac{1}{1 - H_\alpha(\phi_{\mathbf{x}}, \mathbf{\Phi_u})} J(\hat{A},\hat{B},K) \tag{55}$$

Thus, it is very clear that we can convert the optimization problem

$$\begin{gathered} \min_{\gamma \in [0,1)} \frac{1}{1-\gamma} \min_{\mathbf{\Phi_x}, \mathbf{\Phi_u}} \left\| \begin{bmatrix} Q^{1/2} & 0 \\ 0 & R^{1/2} \end{bmatrix} \begin{bmatrix} \phi_{\mathbf{x}} \\ \phi_{\mathbf{u}} \end{bmatrix} \right\|_2 \\ s.t \quad [zI - A \quad B] \begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{\Phi_u} \end{bmatrix} = I \quad H_\alpha(\phi_{\mathbf{x}}, \mathbf{\Phi_u}) \leq \gamma \end{gathered} \tag{56}$$

Using this optimization strategy subject to $\alpha = 0.5$. We have the following relative error.

Theorem 5. *Let* $J_\star$ *be the optimal cost under the actual linear system* (A, B)*. Let* $K_\star$ *be the optimal controller under these conditions. Let* $(\hat{A}, \hat{B})$ *be the error in the estimates of the transition matrices such that* $\|\Delta_A\| \leq \epsilon_A$ *and* $\|\Delta_B\| \leq \epsilon_B$*. If the* K *is synthesized using optimization mentioned above . Then with* $\alpha = 0.5$ *the relative error can be bounded as follows,*

$$\frac{J(A,B,K) - J_\star}{J_\star} \leq 5(\epsilon_A + \epsilon_B \|K_\star\|_2) \|\mathfrak{R}_{A+BK_\star}\|_\infty \tag{57}$$

Using this theorem, we have the following sample complexity bound on the algorithm which is as follows,

Result 1. *Let* $\lambda_G = \lambda_{min}(\sigma_u^2 G_T G_T^T + \sigma_w^2 F_T F_T^T)$ *where* F_T *and* G_T *are as defined earlier. Suppose we estimated* A *and* B *as described earlier* $(\hat{A}, \hat{B})$ *and* $\mathbf{K}$ *is synthesized with* $\alpha = 1/2$*. Then there are universal constants,* C_0 *and* C_1 *such that*

$$\frac{J(A,B,K) - J_\star}{J_\star} \leq C_0 \sigma_w \|\mathfrak{R}_{A+BK_\star}\|_\infty \left(\frac{1}{\sqrt{\lambda_G}} + \frac{\|k_\star\|}{\sigma_u}\right) \sqrt{\frac{(n+p)\log(1/\delta)}{N}} \tag{58}$$

Then with probablity atleast $1 - \delta$, $N \geq C_1 (n+p) \sigma_w^2 \|\mathfrak{R}_{A+BK_\star}\|_\infty^2 (1/\lambda_G + \|K_\star\|_2^2/\sigma_u^2) \log(1/\delta)$

6 Towards understanding model-free Reinforcement learning

There is an interesting criticism of the method that our main paper presents. It is the fact that modern Reinforcement learning became popular because as a algorithmic designed we generally do-not have

any idea whatsoever about model of the system. In that sense, algorithms like those mentioned in this paper defeats the purpose of the RL in general.
But all of this comes with a silver lining. We can use such an algorithm to understand how model-free Reinforcement Learning works. For example, consider one of the popular DeepRL model-free framework called **Deep Deterministic Policy Gradient** (DDPG) algorithm [4]. DDPG showed almost magical results on some of the simple control/robotic environments like Inverted Pendulum. For someone from the controls community it was magical because the algorithm without access to any model whatsover of pendulum it was able to control the pendulum at its unstable equilibrium in around 10^5 sample runs.
One of the possible explanations given at the time is that RL algorithm perform an efficient but random search over policy space to converge to optimal policy. Another possible explanation that is possible maybe just maybe deep RL algorithms are able learn some kind of model of the system. If this hypothesis turns out to be true then it possible for us to get a better and different insight into neural networks that it is known to us. Here are some of the experimental setups that we propose for our experimentation.
In our first set of experiments we run a standard RL algorithm like DDPG on a linear system under zero noise condition. We wish to compare this framework with the solution obtained by solving the Riccati equation. In the second set of experiments we want to compare the solution to one obtained using the paper that we reviewed hoping that we may be able to find a common model for comparison for the two algorithms.

7 Experimental results and Further Work

7.1 No Noise condition

Under no noise condition we tested a DDPG algorithm for 1, 5, 10, 15, 20 dimensional Linear System with the matrix A, B such that the system is controllable. For comparison metric, we considered Riccati equation solution for computing actual optimal control value and for the RL implementation we kept $\gamma = 1.0$. Note that even though there is no noise in the model, there is an exploration noise in the policy training steps.

Dimensions	$\|K_\star - \hat{K}\|_\infty$
1	5.01
5	0.89
10	~ 400
15	~ 6000
20	~ 7000

AS we can see from the table under noise-less condition, the model-free framework is off by the actual policy by around 10^3 order of magnitude for large dimensions. This comes as a surprise because the LQR problem doesn't have a local minima it only has a global minima for the system to converge to. Our best guess for this not working is we believe the approximating capabilities of Neural Networks. From Universal Function approximation theorem Neural networks were shown to approximate any function subject to the fact that the network has sufficient number of weights.
Now for an LQR problem, the critic function will be quadratic in $[x \quad a]$. Since, the Neural net based critic is approximating this actual critic function it could be something like this

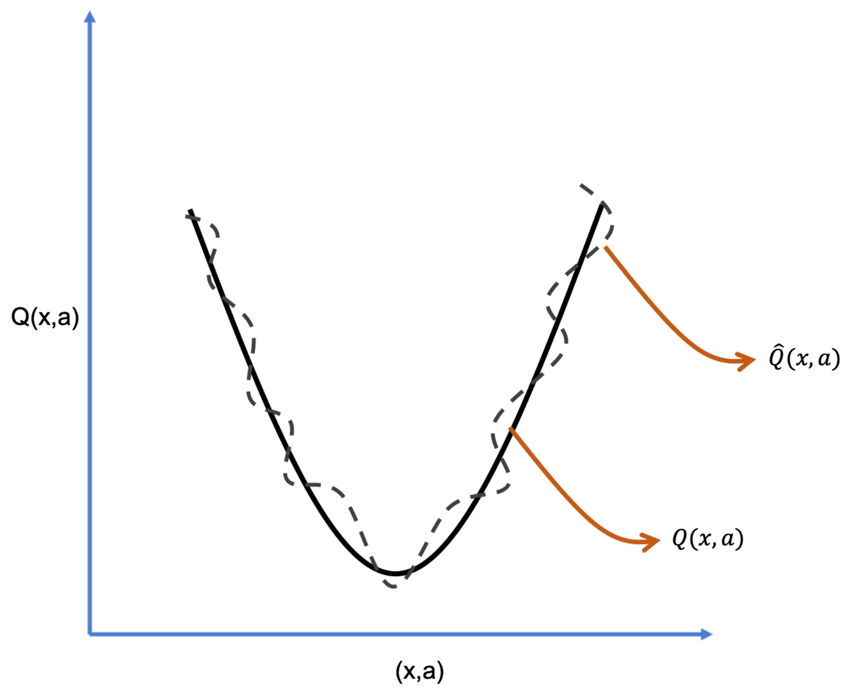


Figure 3: Approximating Critic

Now, if the critic looks something like this then the policy gradient that this algorithm relies on to train the policy [5] is something as follows,

$$\nabla_{\theta_\pi} Q = \nabla_a \hat{Q}(s,a)|_{a=\pi(s)} \nabla_{\theta_\pi} \pi \tag{59}$$

Since, the critic is in itself approximating but since the direction of gradient could be different that the actual actual gradient this will go on to learn a different policy. Thus, we have a different answer about the optimal policy.

A further question this entails would be around how do we go about approximating the critic function such that the form of the approximating critic shifted by a small amount but the structure of the curve looks same. This will ensure that atleast the direction of policy gradient is correct and the convergence will end up near the optimal policy.

References

[1] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. *CoRR*, abs/1805.09388, 2018.

[2] Steven J. Bradtke. Reinforcement learning applied to linear quadratic regulation. In *In Advances in Neural Information Processing Systems 5*, pages 295–302. Morgan Kaufmann, 1993.

[3] Claude-Nicolas Fiechter. PAC adaptive control of linear systems. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory, COLT 1997, Nashville, Tennessee, USA, July 6-9, 1997.*, pages 72–80, 1997.

[4] 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. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[5] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin A. Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 387–395, 2014.