Project: Computing Stabilizing Linear Controllers via Policy Iteration for Noisy Systems

Zaifu Zhan
email: zhan8023@umn.edu
Electrical and Computer Engineering
University of Minnesota Twin Cities

December 19, 2022

Abstract

In recent years, linear quadratic regulator problem has got more and more attentions. Some approaches could compute the linear stabilizing controller from data. However, computing linear stabilizing controller for linear quadratic Gaussian problem remains unsolved. This paper come up with a model-free off-policy reinforcement learning algorithm to find the optimal controller from data where the noisy system and matrices of cost are unknown.

1 Introduction

Reinforcement learning (RL) in active in linear dynamical systems control area, especially in linear quadratic regulators (LQR). The goal of LQR is to find a stailizing linear state-feedback controller to minimiza a specific cost function[D$^+$95]. Some works of LQR have been done in order to explore the sample complexity bounds and regret[AYLS19, DMM$^+$18, CKM19, MTR19], but they assume the initial stabilizing controller is known. At the same time, the ref [DPT19] demostrates some approaches to find the optimal controller from data directly with excitation setting[WRMDM05].

Andrew Lamperski[Lam20] has shown that the LQR problem could be solve via policy iteration. Policy iteration[BYB94] is a typical algorithm of RL, which iterates from random initial policy to optimal policy. Based on his result, we can extend the result by assuming that the data is from the system with process noise..

The paper is organized as follows. The basic problem is introduced in Section 2 and the basic problem is discounted and rescaled in Section 3. The algorithm is given in Section 4 with derivation details. A numerical experiment is presented in Section 5, and finally the conclusion is given in Section 6.

2 Problem Setup

Consider a stochastic linear system with process noise,

$$x_{k+1} = Ax_k + Bu_k + w_k \tag{1}$$

where A and B are unknown matrices and w_k is white Gaussian noise.

The standard linear–quadratic–Gaussian (LQG) control problem is one of the most fundamental optimal control problems, which concerns linear systems driven by additive white Gaussian noise. The problem is to determine an output feedback law that is optimal in the sense of minimizing the expected the infinite horizon quadratic cost:

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\left(x_k^\top Q x_k + u_k^\top R u_k\right)\right] \tag{2}$$

here we will assume that Q and R are unknown positive definite cost matrices.

The standard results of LQG problem is exactly same as the linear-quadratic-regulator problem, and is $u_k = Hx_k$. With this optimal input, $A + BH$ is stable.

Assume we collect a batch of data $\mathcal{D} = \{(x_t, u_t, c_t, y_t) \mid \text{for} t = 1, \ldots, N\}$ from a single trajectory or multiple different paths, where

$$\begin{aligned} y_k &= Ax_k + Bu_k + w_k \\ c_k &= x_k^\top Q x_k + u_k^\top R u_k \end{aligned} \tag{3}$$

The goal is to compute the optimal gain H from the samples we have without any knowledge of system and controller H.

3 Discounting and Rescaling Regulator Problems

To make this problem easier, we could introduce a discount factor γ to our cost. This discount factor gives the system an extra degree of freedom by solving decentralized control problems[FL20]. By steadily increasing the discount factor γ, the discounted cost approaches the cost of standard LQG problem. The discounted problem is as follow:

$$\begin{array}{ll} \min_u & \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^k \left(\tilde{x_k}^\top Q \tilde{x_k} + \tilde{u_k}^\top R \tilde{u_k}\right)\right] \\ \text{subject to} & \tilde{x}_{k+1} = A\tilde{x}_k + B\tilde{u}_k + \tilde{w}_k \end{array} \tag{4}$$

where $\gamma \in [0, 1)$

By rescaling the state and input as $x_k = \gamma^{k/2}\tilde{x}_k$ and $u_k = \gamma^{k/2}\tilde{u}_k$, respectively, the discounted regulator problem is equivalent to the following undiscounted regulator problem:

$$\begin{array}{ll} \min_u & \mathbb{E}\left[\sum_{t=0}^{\infty} \left(x_k^\top Q x_k + u_k^\top R u_k\right)\right] \\ \text{subject to} & x_{k+1} = \sqrt{\gamma}\left(Ax_k + Bu_k + w_k\right) \end{array} \tag{5}$$

where the cost is finite if and only if $\sqrt{\gamma}(A+ BH)$ is stable. For very small γ, the system is easier to stabilize. Indeed, for any H, there is a sufficiently small γ such that $\sqrt{\gamma}(A + BH)$ is stable. Additionally, for $\gamma = 0$, the rescaled system is always stable.

Assume that $\sqrt{\gamma}(A + BH)$ is stable. Then there is a unique positive definite solution, $P_{H,\gamma}$, to the following Lyapunov equation:

$$P_{H,\gamma} = Q + H^\top R H + \gamma(A + BH)^\top P_{H,\gamma}(A + BH) \tag{6}$$

and the cost function is defined as

$$\mathcal{V}_{H,\gamma}(x) = x^\top P_{H,\gamma} x \tag{7}$$

It encodes the cost obtained when starting from $x_0 = x$ and using $u_k = Hx_k$ for all $k \geq 0$. Thus, computing $P_{K,\gamma}$ can be viewed as policy evaluation.

Then, define the action-value function $\mathcal{Q}_{H,\gamma}(x, u)$

$$\begin{aligned} \mathcal{Q}_{H,\gamma}(x, u) &= x^\top Q x + u^\top R u + \gamma \mathcal{V}_{H,\gamma}(Ax + Bu + w) \\ \mathcal{V}_{H,\gamma}(x) &= \mathbb{E}\left[\mathcal{Q}_{H,\gamma}(x, Hx)\right] \end{aligned} \tag{8}$$

If the system $(\sqrt{\gamma}A, \sqrt{\gamma}B)$ is stabilizable, then there is an optimal feedback gain $H^*(\gamma)$ which stabilizes the rescaled system and minimizes the cost. In this case,

$$H^*(\gamma) = -\left(R + \gamma B^\top P_{H^*(\gamma),\gamma} B\right)^{-1} B^\top P_{H^*(\gamma),\gamma} A \tag{9}$$

where $P_{H\cdot(\gamma),\gamma}$ is the associated solution of the Lyapunov equation.

One method for computing $H^\star(\gamma)$ is through policy iteration. Policy iteration alternates between evaluating the current gain by computing $P_{K,\gamma}$, and then updating the gain via the policy improvement step:

$$H' = -\left(R + \gamma B^\top P_{H,\gamma} B\right)^{-1} B^\top P_{H,\gamma} A \tag{10}$$

It can be shown that for stabilizable systems, policy iteration converges geometrically to the optimal feedback gain $H^*(\gamma)$ The value function decreases monotonically as updating controller:

$$P_{H',\gamma} \preceq P_{H,\gamma} \tag{11}$$

4 Algorithm

The algorithm of policy iteration is like this

Algorithm 1 Policy Iteration
Collecting samples
Start with random initial policy
do:
Update action value function
policy improvement
Until find the best policy

Collecting data and randomly generate initial polcy are straightforward. In this section, we are going to focus on how to update action value function and find improved policy. Then we give our final algorithm.

4.1 Action Value function Update

From eq (8),

$$\mathcal{Q}_{H,\gamma}(x,u) = c(x,u) + \gamma \mathbb{E}\left[\mathcal{Q}_{H,\gamma}(y, Hy)|x,u\right] \tag{12}$$

Since our problem is considering noise, so we could assume the form of action value function as folling with a positive definite matrix $M_{H,\gamma}$ and bias $c_{\mathcal{Q}}$ such that

$$\mathcal{Q}_{H,\gamma}(x,u) = \begin{bmatrix} x \\ u \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} x \\ u \end{bmatrix} + c_{\mathcal{Q}} \tag{13}$$

where the bias $c_{\mathcal{Q}}$ is constant.

Then set $K_{H,\gamma} = \sqrt{\gamma}\begin{bmatrix} I \\ H \end{bmatrix}\begin{bmatrix} A & B \end{bmatrix}$ and thus the last term of (12) is

$$\begin{aligned}
&\gamma \mathbb{E}\left[\mathcal{Q}_{H,\gamma}(y, Hy)|x,u\right] \\
=&\gamma \mathbb{E}\left[\begin{bmatrix} y \\ Hy \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} y \\ Hy \end{bmatrix} + c_{\mathcal{Q}}\right] \\
=&\gamma \mathbb{E}\left[y^\top \begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix} y\right] + \gamma c_{\mathcal{Q}} \\
=&\mathbb{E}\left[\left(\sqrt{\gamma}\begin{bmatrix} x \\ u \end{bmatrix}^\top \begin{bmatrix} A & B \end{bmatrix}^\top + \sqrt{\gamma} w^\top\right) \begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix} \left(\sqrt{\gamma}\begin{bmatrix} A & B \end{bmatrix}\begin{bmatrix} x \\ u \end{bmatrix} + \sqrt{\gamma} w\right)\right] + \gamma c_{\mathcal{Q}} \\
=&\begin{bmatrix} x \\ u \end{bmatrix}^\top K_{H,\gamma}^\top M_{H,\gamma} K_{H,\gamma} \begin{bmatrix} x \\ u \end{bmatrix} + \gamma \mathbf{Tr}\left(\begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix}\right) + \gamma c_{\mathcal{Q}}
\end{aligned} \tag{14}$$

Combine (14) and (12), we have

$$\begin{bmatrix} x \\ u \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} x \\ u \end{bmatrix} + c_{\mathcal{Q}} = c(x,u) + \begin{bmatrix} x \\ u \end{bmatrix}^\top K_{H,\gamma}^\top M_{H,\gamma} K_{H,\gamma} \begin{bmatrix} x \\ u \end{bmatrix} + \gamma \mathbf{Tr}\left(\begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix}\right) + \gamma c_{\mathcal{Q}} \tag{15}$$

then for $k = 1, \ldots, N$. (15) can be expressed equivalently as

$$\begin{bmatrix} x_k \\ u_k \end{bmatrix}^\top \left(M_{H,\gamma} - K_{H,\gamma}^\top M_{H,\gamma} K_{H,\gamma}\right) \begin{bmatrix} x_k \\ u_k \end{bmatrix} - \gamma \mathbf{Tr}\left(\begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix}\right) + (1-\gamma) c_{\mathcal{Q}} = c(x_k, u_k) \tag{16}$$

To guarantee that (16) has a unique solution, we use a persistence of excitation condition. Also, with eq (16), we can find M matrix via optimization but it is too less efficient. Sec.4.2 is going to seperate the parameters we want and the data we have to increase the efficience by least square.

4.2 Least square setting

For $z \in \mathbb{R}^q$, let $\phi(z)$ be the corresponding vector of quadratic monomials:

$$\phi(z) = \begin{bmatrix} z_1^2 & z_1 z_2 & \cdots & z_1 z_q & z_2^2 & z_2 z_3 & \cdots & z_q^2 \end{bmatrix}^\top.$$

For a symmetric matrix $S \in \mathbb{R}^{q \times q}$, a corresponding stacking operator is defined by:

$$\omega(S) = \begin{bmatrix} S_{11} & 2S_{12} & \cdots & 2S_{1q} & S_{22} & 2S_{23} & \cdots & S_{qq} \end{bmatrix}^\top.$$

With these definitions, for the first term of eq (16), the quadratic form is given by $z^\top M z = \phi(z)^\top \omega(M)$.

To simplify the second term of eq (16), we can use the property of Trace.

$$\begin{aligned} &\mathbf{Tr}\left(\begin{bmatrix} I \\ H \end{bmatrix}^\top M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix}\right) \\ =&\mathbf{Tr}\left(M_{H,\gamma} \begin{bmatrix} I \\ H \end{bmatrix} \begin{bmatrix} I \\ H \end{bmatrix}^\top\right) \\ =&\mathbf{Tr}\left(M_{H,\gamma} \begin{bmatrix} I & H^\top \\ H & HH^\top \end{bmatrix}\right) \end{aligned} \tag{17}$$

where $\begin{bmatrix} I & H^\top \\ H & HH^\top \end{bmatrix}$ is a symmetric matrix. Let $A = \begin{bmatrix} I & H^\top \\ H & HH^\top \end{bmatrix}$ which has the dimension as matrix $M_{H,\gamma}$. Also, matrices $M_{H,\gamma}$ is also symmetric, so

$$\begin{aligned} &\mathbf{Tr}\left(M_{H,\gamma} A\right) \\ =&\mathbf{Tr}\left(\begin{bmatrix} M_{11} & M_{12} & \dots & M_{1n} \\ M_{21} & M_{22} & \dots & M_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ M_{n1} & M_{n2} & \dots & M_{nn} \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix}\right) \\ =&M_{11}A_{11} + M_{12}A_{21} + \ldots + M_{1n}A_{n1} \\ &+ M_{21}A_{12} + M_{22}A_{22} + \ldots + M_{2n}A_{n2} \\ &+ \vdots + \\ &+ M_{n1}A_{1n} + M_{n2}A_{2n} + \ldots + M_{nn}A_{nn} \\ =&\sum_{i=1}^{n}\sum_{j=1}^{n} M_{ij}A_{ji} \\ =&\sum_{i=1}^{n}\sum_{j=1}^{n} M_{ij}A_{ij} \\ =&\begin{bmatrix} M_{11} & 2M_{12} & \cdots & 2M_{1n} & M_{22} & 2M_{23} & \cdots & M_{nn} \end{bmatrix} \\ &\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} & A_{22} & A_{23} & \cdots & A_{nn} \end{bmatrix}^\top \end{aligned} \tag{18}$$

If we define a new stacking operator ω_2 such that

$$\omega_2(S) = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1q} & S_{22} & S_{23} & \cdots & S_{qq} \end{bmatrix}^\top \tag{19}$$

Therefore,

$$\begin{aligned} &\mathbf{Tr}\left(M_{H,\gamma} A\right) \\ =&\begin{bmatrix} M_{11} & 2M_{12} & \cdots & 2M_{1n} & M_{22} & 2M_{23} & \cdots & M_{nn} \end{bmatrix} \\ &\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} & A_{22} & A_{23} & \cdots & A_{nn} \end{bmatrix}^\top \\ =&\omega^\top(M)\omega_2(A) \\ =&\omega_2^\top(A)\omega(M) \end{aligned} \tag{20}$$

In nutshell, we can transform eq(16) to a new following form which separate the parameters and data via some invertiable operations.

$$\omega^\top(M)\phi(\begin{bmatrix}x_k\\u_k\end{bmatrix} - \gamma \begin{bmatrix}I\\H\end{bmatrix} x_{k+1}) - \gamma\omega^\top(M)\omega_2(\begin{bmatrix}I & H^\top\\H & HH^\top\end{bmatrix}) + (1-\gamma)c_{\mathcal{Q}} = c(x_k,u_k) \tag{21}$$

Then use least square technique to find M and $c_{\mathcal{Q}}$ in each step by following eq (22) which separates the parameters of action value function and the data we have..

$$\begin{bmatrix}\omega(M)\\C_{\mathcal{Q}}\end{bmatrix}^\top \begin{bmatrix}\phi\left(\begin{bmatrix}x_k\\u_k\end{bmatrix} - \gamma\begin{bmatrix}I\\H\end{bmatrix}x_{k+1}\right) - \gamma\omega_2\left(\begin{bmatrix}I & H^\top\\H & HH^\top\end{bmatrix}\right)\\1-\gamma\end{bmatrix} = c(x_k,u_k) \tag{22}$$

4.3 Policy improvement

Once we have the $M_{H,\gamma}$ matrix and bias $c_{\mathcal{Q}}$, we could plug them back to action value function which is quadratic.

$$\begin{aligned}\mathcal{Q}_{H,\gamma}(x,u) &= \begin{bmatrix}x\\u\end{bmatrix}^\top M_{H,\gamma}\begin{bmatrix}x\\u\end{bmatrix} + c_{\mathcal{Q}}\\ &= \begin{bmatrix}x\\u\end{bmatrix}^\top \begin{bmatrix}M^{xx}_{H,\gamma} & M^{xu}_{H,\gamma}\\M^{ux}_{H,\gamma} & M^{uu}_{H,\gamma}\end{bmatrix}\begin{bmatrix}x\\u\end{bmatrix} + c_{\mathcal{Q}}\\ &= x^\top M^{xx}_{H,\gamma}x + 2u^\top M^{xu}_{H,\gamma}x + u^\top M^{uu}_{H,\gamma}u + c_{\mathcal{Q}}\end{aligned} \tag{23}$$

Take derivative over u and equal to zero, we can get the improve policy

$$u = -\left(M^{uu}_{H,\gamma}\right)^{-1} M^{ux}_{H,\gamma} \quad x \tag{24}$$

then the improved policy is

$$H' = -\left(M^{uu}_{H,\gamma}\right)^{-1} M^{ux}_{H,\gamma} \tag{25}$$

4.4 Algorithm

With the results above, we have the final algorithm to solve the LQG problem. Here is a Python implementation of this algorithm at Colab: Code at CoLab

Algorithm 2 Data-Driven Stabilization and Optimization

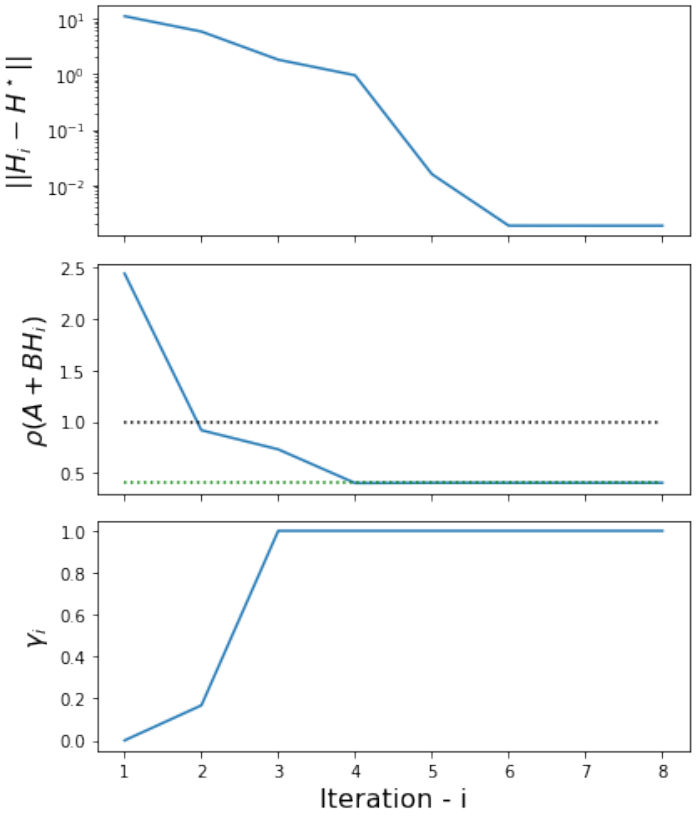
Given samples $\mathcal{D} = \{(x_t,u_t,c_t,y_t)\}_{t=1}^N$
Start with $H_i = 0$ and $\gamma_i = 0$.
for $i \geq 0$ do
  Let γ_{i+1} be the largest value such that:
    $\gamma_{i+1} \leq 1$
    $M_{H_i,\gamma_{i+1}}$ from (22) is positive definite
    $\left\|P_{H_i,\gamma_i} - P_{H_i,\gamma_{i+1}}\right\|_\star \leq 1$
  Let $H_{i+1} = -\left(M^{uu}_{H_i,\gamma_{i+1}}\right)^{-1} M^{ux}_{H_i,\gamma_{i+1}}$
end for

5 Numerical Experiment

Here we describe the performance of the algorithm on a randomly generated unstable system with state dimension 5 and 2 inputs. The open-loop system has a spectral radius of 2.44. After 3 iterations, the discount factor has reached 1. At this point, all controllers are guaranteed to stabilize the system. The discount factor, spectral radius of $A + BH_i$, and the deviation of H_i from the optimal controller H^* are shown in Fig. 1. The green dash line repesent the optimal spectral radius of $A + BH^*$

The controller was computed from 56 samples, (x_t,u_t,c_t,y_t), where inputs are amplified Gaussian white noise with identity covariance because noise may be large and be dominant over our data (we set the amplitude of noise is 100, a very large noise). Due to instability, the states grow large quickly. To avoid numerical problems, the trajectories were restarted at a random initial state when the state reach a large number (bound is 1e8 in our code).

Figure 1: Performance on a random unstable system.



6 Conclusions

This paper proposed a model-free algorithm to find the linear state-feedback stabilizing controller for linear system based on policy iteration. The algorithm perform pretty well and quickly in solving LQG problem.

References

[AYLS19] Yasin Abbasi-Yadkori, Nevena Lazic, and Csaba Szepesvári. Model-free linear quadratic control via reduction to expert prediction. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3108–3117. PMLR, 2019.

[BYB94] Steven J Bradtke, B Erik Ydstie, and Andrew G Barto. Adaptive linear quadratic control using policy iteration. In *Proceedings of 1994 American Control Conference-ACC'94*, volume 3, pages 3475–3479. IEEE, 1994.

[CKM19] Alon Cohen, Tomer Koren, and Yishay Mansour. Learning linear-quadratic regulators efficiently with only $\sqrt{T}$ regret. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1300–1309. PMLR, 09–15 Jun 2019.

[D+95] P Bertsekas Dimitri et al. Dynamic programming and optimal control. *Athena Scientific*, 1995.

[DMM+18] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. *Advances in Neural Information Processing Systems*, 31, 2018.

[DPT19] Claudio De Persis and Pietro Tesi. On persistency of excitation and formulas for data-driven control. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 873–878. IEEE, 2019.

[FL20] Han Feng and Javad Lavaei. Escaping locally optimal decentralized control polices via damping. In *2020 American Control Conference (ACC)*, pages 50–57, 2020.

[Lam20] Andrew Lamperski. Computing stabilizing linear controllers via policy iteration. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1902–1907. IEEE, 2020.

[MTR19] Horia Mania, Stephen Tu, and Benjamin Recht. Certainty equivalent control of lqr is efficient. *arXiv preprint arXiv:1902.07826*, 2019.

[WRMDM05] Jan C Willems, Paolo Rapisarda, Ivan Markovsky, and Bart LM De Moor. A note on persistency of excitation. *Systems & Control Letters*, 54(4):325–329, 2005.