# Inverse Reinforcement Learning on Linearly Parametrized Bandits

Abhinav Narayan Harish

December 2021

## 1 Introduction

The K-armed bandit problem involves taking an action from a set of possible actions in order to maximize the long term reward. A class of algorithms called the Upper Confidence Bounds algorithms achieve a strong logarithmic regret guarantee on the K-armed bandit in the finite arm setting. UCB makes an assumption that the rewards for the arms are independently generated and the relationship between arms is inconsequential for reward specification. However, upper confidence bounds become intractable in combinatorial examples like the shortest path algorithm with an exponential number of arm choices.

Linearly parameterized Bandits model a generalized continuous action space and account for relationship between arms. Every arm is treated as a vector in higher dimensional space the per-round reward is computed as a dot product of the action with a reward vector. To select the optimal arm, we follow an adaption of UCB, called as LinUCB which extends the notion of "confidence intervals" to "confidence sets". The reward is modelled as a least squared estimate on the rewards and the optimal reward vector can be bounded with a high probability. By carefully selecting this confidence set we can achieve an algorithm with $O(\sqrt{T})$ regret guarantee.

Instead of generating actions, our goal is to estimate the reward vector $\theta^*$. This problem referred to as the inverse reinforcement learning(RL), finds applications in designing RL systems strongly aligned with human values. We assume that a person $A$ has conducted LinUCB and taken a set of optimal actions at each time step. Person $B$ who has access to only the actions taken by A, has to estimate the reward vector. The eventual goal is to establish a high probability bound of the the optimal reward vector $\theta^*$ that could have resulted in these actions.

This grand objective of establishing a high probability bound on the reward vector is a highly ambitious one. In this project, we take a few initial steps to this eventual goal. Firstly, we study the regret guarantee of the LinUCB algorithm. Secondly, we highlight a technique for obtaining the optimal estimate using linear programming constraints. Lastly, we conduct a few exploratory experiments to gain an understanding of LinUCB to different types of action spaces.

## 2 Linearly Parametrized Bandits

### 2.1 Background

The linear bandit algorithm extends the notion of "optimism in the face of uncertainty" to a continuous action set. At each step, the learner picks an action $a_t$ from a set $A_t \subset \mathcal{R}^d$ and incurs a reward of $c_t = \langle a_t, \theta^\star \rangle + \epsilon_t$. The overall goal of the learner is to pick actions that can maximize the cumulative reward. Given the optimal action $a_t^\star = \text{argmax}_{a \in A_t} \langle a, \theta^\star \rangle$, the learner incurs an excpected pseudo regret of $\overline{\mathcal{R}}_T = E\left[\sum_{t=1}^{T} \langle a_t - a_t^\star, \theta^\star \rangle\right]$.

LinUCB breaks down the learning process into two main steps, i) establishing an estimate $\theta_t$ of $\theta^*$. ii) Utilizing the estimated $\theta_t$ to decide the optimal action of the subsequent time-step. Firstly, the estimation of $\theta_t$ is modelled as a least square problem -

$$\hat{\theta}_t = \underset{\theta \in R^d}{\operatorname{argmin}} \sum_{\tau=1}^{t} \left(\langle a_\tau, \theta \rangle - c_\tau\right)^2 + \lambda \|\theta\|_2^2 \tag{1}$$

Here, $\lambda$ is a regularization parameter to ensure the existence of a solution. The solution of the least square problem is estimated as,

$$\hat{\theta}_t = M_t^{-1} \sum_{\tau=1}^{t} c_\tau a_\tau \tag{2}$$

Here, $M_t = \sum_{\tau=1}^{t} a_\tau a_\tau^\top$ represents the gram matrix of the actions taken upto time-step t. By plugging in $c_\tau = \langle a_\tau, \theta^\star \rangle + \epsilon_\tau$, we get,

$$\widehat{\theta}_t = M_t^{-1} M_t \theta^\star + M_t^{-1} \sum_{\tau=1}^{t} \epsilon_\tau a_\tau = \theta^\star + M_t^{-1} Z_t \tag{3}$$

This estimate captures a relation between the optimal and estimated reward vector. By utilzing our prior knowledge on the gaussian noise distribution, we can bound their difference, $|\theta_t - \theta^*|$ and come up with a $(1-\delta)$ confidence interval.

$$\Pr\left( \left\| M_t^{1/2} \left( \widehat{\theta}_t - \theta^\star \right) \right\|_2^2 \leq d + 2\sqrt{d \ln \frac{1}{\delta}} + 2 \ln \frac{1}{\delta} \right) \geq 1 - \delta \tag{4}$$

### 2.2 Regret Analysis

In this section, we build on the background of LinUCB and demonstrate the $O(\sqrt{T})$ regret gurantee of the LinUCB algorithm.
First for any action $a$, we can bound the difference between the predicted $\langle a, \theta_t \rangle$ and actual reward $\langle a, \theta^* \rangle$ by using Cauchy's Schwatz's inequality,

$$\left| \left\langle a, \theta^\star - \hat{\theta}_t \right\rangle \right| \leq \left\| \theta^\star - \hat{\theta}_t \right\|_{M_t} \|a\|_{M_t^{-1}} \leq \beta_t \|a\|_{M_t^{-1}} \tag{5}$$

Here, we have compactly represented $\beta_t = d + 2\sqrt{d \ln \frac{1}{\delta}} + 2 \ln \frac{1}{\delta}$. Now, substituting $a = a_{t+1}$ in Equation 5 we obtain,

$$\langle a_{t+1}, \theta^\star \rangle \leq \left\langle a_{t+1}, \hat{\theta}_t \right\rangle + \beta_t \left\| a_{t+1}^* \right\|_{M_t^{-1}} \tag{6}$$

Now, for $a = a_{t+1}^*$,

$$\left\langle a_{t+1}^\star, \theta^\star \right\rangle \geq \left\langle a_{t+1}^\star, \hat{\theta}_t \right\rangle - \beta_t \left\| a_{t+1}^\star \right\|_{M_t^{-1}} \geq \left\langle a_{t+1}, \hat{\theta}_t \right\rangle - \beta_t \left\| a_{t+1} \right\|_{M_t^{-1}} \tag{7}$$

Using Equation 6 and 7, we obtain a bound the regret per-time step as,

$$\left\langle a_{t+1}^\star - a_{t+1}, \theta^\star \right\rangle \leq 2\beta_t \left\| a_{t+1} \right\|_{M_t^{-1}} \leq 2\beta_T \left\| a_{t+1} \right\|_{M_t^{-1}} \tag{8}$$

We can now model the regret expression as by adding the regret of the individual time-steps,

$$\begin{aligned}\sqrt{T\sum_{t=1}^{T}\left(\langle a_t - a_t^\star, \theta^\star\rangle\right)^2} &\leq \beta_T\sqrt{4T}\sum_{t=1}^{T}\min\left\{1, \|a_t\|^2_{M_{t-1}^{-1}}\right\} \\ &\leq \beta_T\sqrt{8T\sum_{t=1}^{T}\ln\left(1+\|a_t\|^2_{M_{t-1}^{-1}}\right)} = \beta_T\sqrt{8T\ln\prod_{t=1}^{T}\left(1+\|a_t\|^2_{M_{t-1}^{-1}}\right)}\end{aligned} \tag{9}$$

Now, we observe that the expression $\prod_{t=1}^{T}\left(1+\|a_t\|^2_{M_{t-1}^{-1}}\right)$ can be related to the the determinant of the matrix $M_T$.

$$\begin{aligned}\det\left(M_T\right) &= \det\left(M_{T-1} + a_T a_T^\top\right) = \det\left(M_{T-1}^{\frac{1}{2}}\left(I + M_{T-1}^{-\frac{1}{2}} a_T a_T^\top M_{T-1}^{-\frac{1}{2}}\right)M_{T-1}^{\frac{1}{2}}\right) \\ &= \det\left(M_{T-1}\right)\det\left(I + M_{T-1}^{-\frac{1}{2}} a_T a_T^\top M_{T-1}^{-\frac{1}{2}}\right) = \det\left(M_{T-1}\right)\left(1+\|a_T\|^2_{M_{T-1}^{-1}}\right)\end{aligned} \tag{10}$$

By induction we can establish that,

$$\det\left(M_T\right) = \det\left(M_0\right)\prod_{t=1}^{T}\left(1+\|a_t\|^2_{M_{t-1}^{-1}}\right). \tag{11}$$

We can clearly observe that $\frac{\det(M_T)}{\det(M_0)} = \prod_{t=1}^{T}\left(1+\|a_t\|^2_{M_{t-1}^{-1}}\right)$. Using this we rewrite the overall regret bound as,

$$\mathcal{R}_T \leq \sqrt{8T\ln\frac{\det\left(M_{T-1}\right)}{\det\left(M_0\right)}} \tag{12}$$

Using the AM-GM inequality, we can bound the determinant of the covariate matrix,

$$\begin{aligned}\det\left(M_T\right) &\leq \left(\frac{\mathrm{TR}\left(M_T\right)}{d}\right)^d = \left(\frac{\lambda d + \sum_{t=1}^{T}\mathrm{TR}\left(a_t a_t^\top\right)}{d}\right)^d = \left(\lambda + \frac{\sum_{t=1}^{T}\mathrm{TR}\left(a_t^\top a_t\right)}{d}\right)^d \\ &\leq \left(\lambda + \frac{T}{d}\right)^d\end{aligned}$$

We also know that $\det\left(M_0\right) = \lambda^d$. Putting these two expressions in Equation 12, we obtain,

$$\mathcal{R}_T \leq \sqrt{8T\ln\frac{\det\left(M_{T-1}\right)}{\det\left(M_0\right)}} \tag{13}$$

$$\leq \sqrt{8dT\ln\left(1+\frac{T}{\lambda d}\right)} \tag{14}$$

However, we note that this regret is valid only in the good case (i.e, $\theta^*$ lies in $C_t$). To obtain an overall bound on the regret as a summation of per round regret.

$$\overline{\mathcal{R}}_T = \sum_{t=1}^{T} P(\theta^* \notin C_t)\mathcal{R}_{\theta^*\notin C_t} + \sum_{t=1}^{T} P(\theta^* \in C_t)\mathcal{R}_{\theta^*\in C_t} \tag{15}$$

$$\leq 2T\delta + \beta_T\sqrt{8dT\ln\left(1+\frac{T}{\lambda d}\right)} \tag{16}$$

Now, by substituting the value $\delta = \frac{1}{T}$ in Equation 16, we can obtain our desired regret gurantee,

$$R_T = O(d \ln(T/d)\sqrt{T}) \tag{17}$$

# 3 Inverse Reinforcement Learning on LinUCB

Inverse Reinforcement Learning predicts the rewards of a demonstration given the actions taken by the algorithm. To tackle this problem, we develop a feasibility program based on linear programming that provides a set of constraints. The solver comes up with a valid solution satisfying these constraints.

More concretely, we know the LinUCB objective can be rewritten as,

$$f\left(a, \hat{\theta}_t, \beta_t, M_t\right) = \left\langle a, \hat{\theta}_t \right\rangle + \sqrt{\beta_t}\, a^T M_t a \tag{18}$$

For an optimal $a_t \in \mathcal{A}_t$, we observe that,

$$f\left(a_t, \hat{\theta}_t, \beta_t, M_t\right) - f\left(a, \hat{\theta}_t, \beta_t, M_t\right) > 0 \; \forall \; a_t \in \mathcal{A}_t \tag{19}$$

We know that, $\hat{\theta}_t = M_t^{-1} \sum_{\tau=1}^{t} c_\tau a_\tau$, where Let's define the term, $y_t = \sum_{\tau=1}^{T} c_\tau a_\tau = \sum_{\tau=1}^{t} \left(\langle a_\tau, \theta^\star \rangle + \epsilon_\tau\right) a_\tau$. Then, we know that the value of $y_{t+1}$ can be related to $y_t$ using the following equation,

$$y_{t+1} = y_t + a_t x_t \tag{20}$$

Using this we can generate two more constraints as follows,

$$\begin{aligned} y_{t+1,i} &< \max\left(a_t\right) * \max\left(x_t\right) + y_{t,i} \\ y_{t+1,i} &> \min\left(a_t\right) * \min\left(x_t\right) + y_{t,i} \end{aligned}$$

Now we consider the difference in theta values.

$$\begin{aligned} \hat{\theta}_{t+1} - \hat{\theta}_t &= M_{t+1}^{-1} * \left(y + a_t x_t\right) - M_t^{-1}(y) \\ &= \left(M_t + a_t^T a_t\right)^{-1} * \left(y + a_t x_t\right) - M_t^{-1}(y) \\ = \left(M_t^{-1} - \frac{M_t^{-1} a_t^T a_t M_T^{-1}}{1 + a_t^T M_t^{-1} a_t}\right) &\left(y_t + a_t x_t\right) - M_t^{-1}\left(y_t\right) \end{aligned} \tag{21}$$

It is interesting to note that the difference is aligned along the direction $M_t^{-1} A_t$. Let $P_{M_t^{-1} a_1}$ represent the direction perpendicular to $M_t^{-1} A_t$. We can then formulate the resulting equation as,

$$P_{M_t^{-1} a_1}\left(M_{t+1}^{-1} y_{t+1} - M_t^{-1} y_t\right) = 0 \tag{22}$$

Next, we plug these constraints into the Gurobi solver to obtain a feasible solution. Gurobi uses the cutting plane method to come up with a solution to these linear equations. In Figure 3 we observe that the regret of the low regret of this algorithm with number of arms.

# 4 Experiments

To understand the inverse reinforcement problem better, we need to develop a strong understanding of the forward algorithm for Lin-UCB. In this section, we highlight two additional experiments we conducted and some insights we gathered.

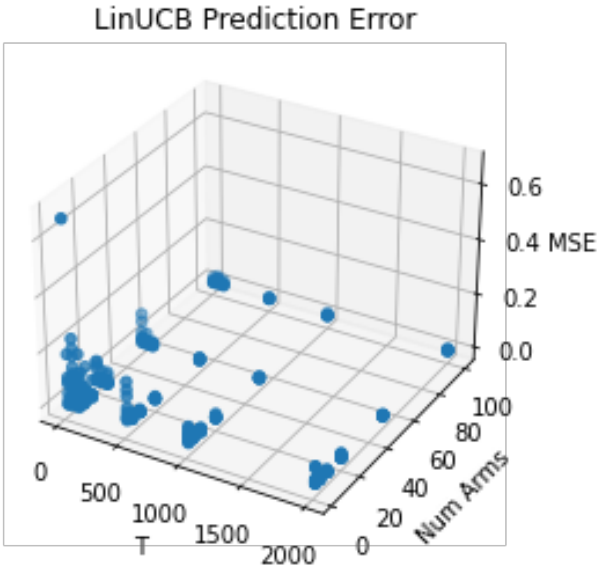


Figure 1: Regret of the Gurobi Solver across time steps for various number of arms.

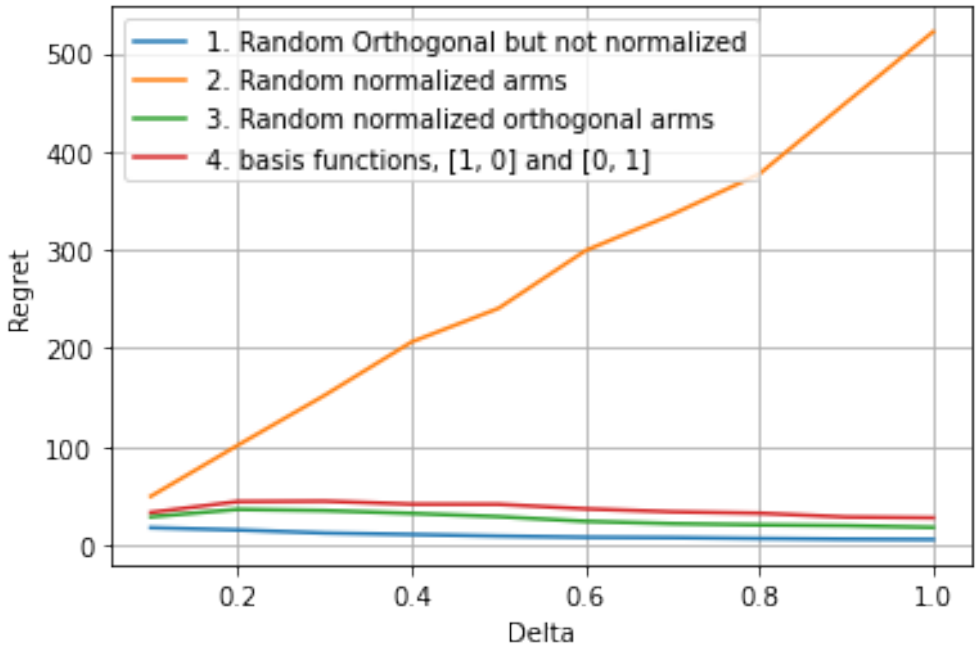


Figure 2: Performance of 4 different arms for varying value of the reward vector $\theta^* = [0, -\delta]$. The numbering of each of these scenarios corresponds to the ones mentioned in the document.

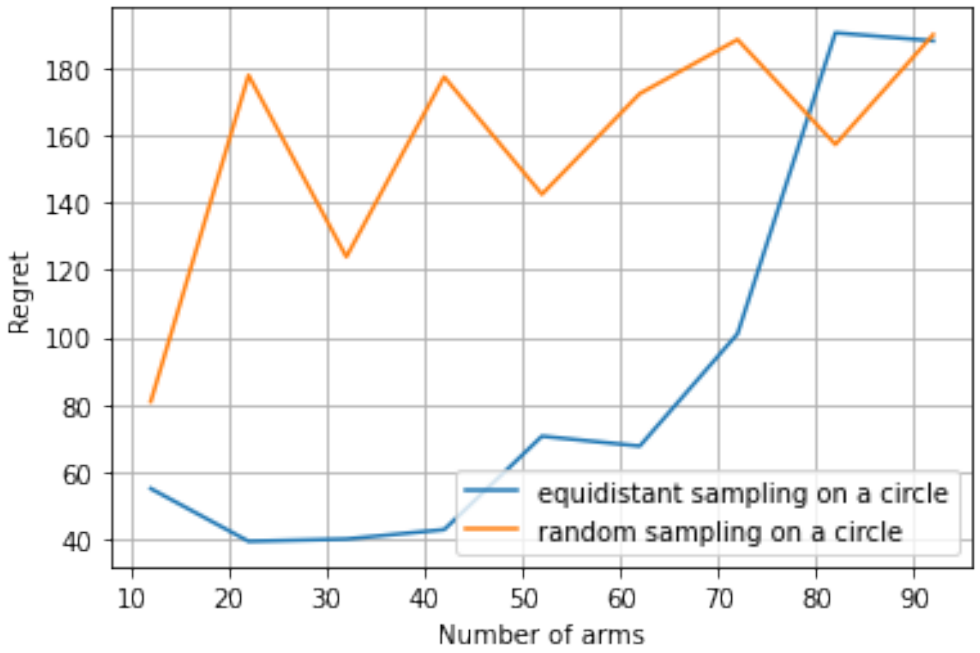


Figure 3: Comparison of the performance of LinUCB on two different arms sets. i) Evenly spaced arms on a circle, ii) Randomly sampled arms on a circle.

### 4.1 Effect of Arm Orthogonality:

In Homework 3, we compared the performance of LinUCB to the upper confidence bound algorithms (UCB) with a finite set of arms for varying values of the suboptimality gap. In this experiment, we will replicate this and consider the effect of different action spaces on the ensuing regret.

More concretely, Given a reward vector $\theta^* = [0, -\delta]$, and an action space containing two arms $A = \{\mathbf{a_1}, \mathbf{a}_2\}$, we will analyze the performance of the LinUCB algorithm under 4 salient cases.

1. **Un-normalized orthogonal:** The two arms $\mathbf{a}_1$ and $\mathbf{a}_2$ are orthogonal $\mathbf{a}_1^T\mathbf{a}_2 = 0$ but need not be normalized.
2. **Random normalized arms:** We have two arms $\mathbf{a}_1$ and $\mathbf{a}_2$ and $||\mathbf{a}_1|| = 1$ and $||\mathbf{a}_2|| = 1$. However, they need not be orthogonal.
3. **Normalized and Orthogonal:** In this scenario, the two arms $\mathbf{a}_1$ and $\mathbf{a}_2$ are normalized and orthogonal, i.e, $\mathbf{a}_1^T\mathbf{a}_2 = 0$ and $||\mathbf{a}_1|| = 1$ and $||\mathbf{a}_2|| = 1$.
4. **Basis Vectors:** We model the two arms as identity vector $\mathbf{a}_1 = \mathbf{e}_1 = [1, 0]$ and $\mathbf{a}_2 = \mathbf{e}_2 = [0, 1]$.

For each round of the algorithm, we sample a new set of random arms to select from. In each round, we run the algorithm for 1000 time steps, for 10 evenly spaced values of of $\delta$ between 0 and 1. Overall we conduct 200 rounds of our algorithm and report the cumulative regret in each case.

While designing our experiment we considered two aspects, i) orthogonality between the arms, ii) importance of normalization. In Figure 2, it's clear that orthogonality plays a key role in ensuring this strong regret guarantee. Among all 4 choices, the randomly sampled non-orthogonal arms are the ones that perform the worst. This experiment highlights the importance of distribution of arms in the action space and it's effect on regret.

### 4.2 Effect of Uniform Sampling

From the previous experiment, we have gained an understanding that orthogonality could play a key role in deciding the regret of each arm. We try to further probe this point by extending this to $|\mathcal{A}_t| > 2$. Note that, sampling $N > 2$ arms with $d = 2$ results in a low-rank space and it's non-trivial to define a notion of maximally separated arms in this case. We consider an intutive choice by considering evenly separated arms on a circle. We sample this by dividing the parametric representation of a circle, i.e $(x, y) = (cos\theta, sin\theta)$. We select $N = 100$ values from $\theta \in (0, 2\pi)$ using this parametric representation.

In Figure 3, our observation is consolidated when we observe that initially the equidistant arms perform better as there is strong separation between arms. However as the number of arms increases, the separation between any two arms reduces. Correspondingly, we observe an increasing regret of the algorithm for larger number of arms.

## 5 Contributions

This project was undertaken with my project partner Etash Guha. The section Inverse Reinforcement Learning on LinUCB describes Etash's ongoing research with Professor Muthukumar. The remainder of the report highlights my experimental and theoretical understanding of this domain.

## 6 Conclusion

In this experiment we conduct a study of the Linear Parametrized Bandits as a first step to understanding Inverse Reinforcement Learning. Firstly, we present the proof of the $O(\sqrt{T})$ guarantee of LinUCB. Secondly, we conduct two experiments to understand better the behaviour of LinUCB to different arm choices. Lastly, we also present an experimental direction which involves incorporating the Gurobi solver along with linear programming constraints to obtain the optimal reward vector.