Contextual Linear Bandits under Interference of Smooth Network Dynamics

Hoang-Son Nguyen
Oregon State University
nguyhoa3@oregonstate.edu

Nam Nguyen
Oregon State University
nguynam4@oregonstate.edu

1 Introduction

1.1 Background

Motivation Contextual linear bandit is a popular framework that extends the traditional multi-armed bandit settings by modeling the reward function as a linear model, with many real-world applications ranging from recommender systems to online advertisements. However, in the age of the Internet where there can be many users to serve simultaneously, it is often required to solve multiple contextual linear bandits at the same time, since every user has a different tastes and preferences, hence yielding different reward functions that the decision-making agent must learn to make good decisions with good regret. Applying linear bandit models in a straightforward manner to solve this multi-task contextual linear bandit would have the accumulated regret scaling up by the number of users, which might be up to the scales of million. The reason is due to the need to learn independent models for each of the users, which in turn requires collecting more data and hence the agent would make more poor decisions before learning enough about the true reward models to make good decisions. However, oftentimes there are underlying structures in the users' rewards that we can exploit, which can be utilized to relax the requirements to learn independent linear models.

Bandits on Network For example, if two users are connected on some online social networks, they likely have similar taste profiles. Therefore, a regularization scheme can be implemented to encourage linear parameters of two users to be close if they are friends on social media. Previous works aimed at leveraging this knowledge about underlying network structure between users to improve learning in bandit algorithms, such as Cesa-Bianchi et al. [2013], Gentile et al. [2014], Kocak et al. [2020], Yang et al. [2020], Agarwal et al. [2024], Anonymous [2024].

Goal Nonetheless, such data about the relationships between user profiles are not always available due to the sheer scale of modern networks, sometimes up to million or even billion. In this project, we deviated from the previous major lines of works to derive a bandit algorithm that can utilize the network structure without seeing the explicit connections between users. Specifically, relying on a smoothness assumption of the payoffs of connected users and the prevalent community structure in real-world networks Barabasi [2016], it is possible to learn useful information about the clustering of users that can serve as a "pseudo-graph" for graph-regularized linear bandit algorithms. We show that our proposed method can outperform running multiple linear bandit algorithms (e.g., LinUCB) simultaneously and neglecting the hidden structure behind the reward signals.

1.2 Literature Review

Graph Regularization for Linear Bandit One of the first works to incorporate similarity encoded in the form of network, available to the learner, is Cesa-Bianchi et al. [2013]. Using the key assumption that the linear parameters of users connected on the given graph are close (with respect to their norm), the authors propose to include a graph-based regularizer that promotes smoothness of learnable parameters into the linear (ridge) regression used in linear upper confidence bound algorithm

Preprint.

(LinUCB) Abbasi-Yadkori et al. [2011]. By including an informative prior into the linear regression, the method reduces the number of data points needed for learning the parameters accurately, in contrast to applying LinUCB without utilizing the smoothness structure of parameters on graph. Several other algorithms were proposed, which also design a graph-based regularization for linear regression used in linear bandit, based on the network topology of users available to learner, such as Kocak et al. [2020] and Yang et al. [2020].

Clustering of Bandits Another line of work relies on the assumption of clustering structure of the parameters: as long as two users are in the same group, their parameters are exactly the same Gentile et al. [2014]. Based on this modeling assumption, the authors propose an iterative procedure to gradually sparsify a fully connected network to a community-structured network, and simultaneously using that for choosing an action.

Enhanced Exploration in Clustering of Bandits A recent study by Anonymous [2024] addresses the challenges of online clustering of bandits under stochastic and smoothed adversarial contexts. The proposed UniCLUB and UniSCLUB algorithms introduce an exploration phase to improve cluster identification and regret minimization, overcoming the limitations of traditional UCB strategies. By relaxing restrictive assumptions like i.i.d. context generation, these methods offer a more practical approach. The smoothed adversarial context setting further enhances robustness, bridging stochastic and adversarial extremes.

2 Model

Contextual Linear Bandits under Network Dynamics Consider a learner interacting with n users over T rounds, connected by an undirected, connected graph $G = (V, E)$, represented by a weighted symmetric adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$ or a Laplacian matrix $\mathbf{L} = Diag(\mathbf{A1}) - \mathbf{A}$. Similar to Gentile et al. [2014], we assume that the similarity of the parameter of each node is encoded in the network, in particular the community structure. At each round $t = 1, 2, ..., T$, the learner receives a node index $i_t \in V$ with a set of context vectors $C_{i_t} = \{\mathbf{x}_{t,1}, \mathbf{x}_{t,2}, ..., \mathbf{x}_{t,c_t}\}$ that each $\mathbf{x}_{t,j} \in \mathbb{R}^n$ is generated i.i.d. from a distribution $\mathcal{D}_t$ such that $\mathbb{E}_{\mathbf{x}_t\sim\mathcal{D}_t}[\mathbf{x}_t\mathbf{x}_t^\top] = \gamma^2\mathbf{I}$ for some $\gamma > 0$. The learner then selects $\bar{\mathbf{x}}_t := \mathbf{x}_{t,k_t} \in C_{i_t}$ as the action, and then observes a payoff $a_{i_t}(\bar{\mathbf{x}}_t)$, which is the i_t entry of the following vector:

$$\mathbf{a}_t(\bar{\mathbf{x}}_t) = \mathcal{H}(\mathbf{L})\bar{\mathbf{x}}_t + \mathbf{e}_t, \tag{1}$$

where $\mathcal{H}(\mathbf{L})$ is a polynomial of $\mathbf{L}$, a symmetric matrix representing the dynamics on network $\mathcal{G}$ Ramakrishna et al. [2020]:

$$\mathcal{H}(\mathbf{L}) = \sum_{k=0}^{\infty} h_k\mathbf{L}^k, \tag{2}$$

and the error vector is $\mathbf{e}_t = [e_{1_t}, ..., e_{n_t}] \in \mathbb{R}^n$ such that $\mathbb{E}[e_{i_t}|\bar{\mathbf{x}}_t] = 0$ and $\mathbb{E}[e_{i_t}^2|\bar{\mathbf{x}}_t] \leq \sigma^2$. Put differently, with $\mathbf{h}_{i_t}$ being the parameter vector of user i_t (in our case, the row i_t of matrix $\mathcal{H}(\mathbf{L})$), the payoff $a_{i_t}(\bar{\mathbf{x}_t})$ can be written as

$$a_{i_t}(\bar{\mathbf{x}}_t) = \mathbf{h}_{i_t}^\top\bar{\mathbf{x}}_t + e_{i_t}. \tag{3}$$

We note that the described model of network dynamics cover many real-world setting, such as opinion dynamics, linear-quadratic network games, etc. Ramakrishna et al. [2020]. For example, it is shown that the DeGroot's and Friedkin-Johnsen's models for opinion dynamics are a special case of the following dynamics

$$\mathbf{y}_{t+1} = (1-\beta)(\mathbf{I} - \alpha\mathbf{L})\mathbf{y}_t + \beta\mathbf{x}_t, \tag{4}$$

where α, β control the spread of opinions, and $\mathbf{y}_t \in \mathbb{R}^n$ is a vector of individuals' opinions at time t, and $\mathbf{x}_t \in \mathbb{R}^n$ is a vector of external opinions perceived by agents in a social network. Assuming that $\mathbf{x}_t := \mathbf{x}$, steady-state of this model indeed has the form of a linear network dynamics:

$$\mathbf{y} = \lim_{t\to\infty}\mathbf{y}_t = (\mathbf{I} + \tilde{\alpha}\mathbf{L})^{-1}\mathbf{x}, \tag{5}$$

where $\tilde{\alpha} > 0$ is defined by α, β. Under this data model, the payoffs of different users are governed by some underlying unknown network dynamics.

In Gentile et al. [2014], the payoff depends on the network via the clustering structure: if i_t and j_t are two users in the same cluster, it is assumed that $\mathbf{h}_{i_t} = \mathbf{h}_{j_t}$. We relax this assumption, and show a

model that has $a_i(\mathbf{x}) \approx a_j(\mathbf{x})$ when i and j are two users in the same community. This phenomenon occurs when the network dynamics is sufficiently smooth, i.e. connected users share similar payoffs.

Smooth Network Dynamics We now consider the smoothness assumption that enables us to reveal the similarity structure of the payoffs. To describe the network dynamics, we will use the framework of graph signal processing Ortega et al. [2018]. A payoff vector $\mathbf{a}(\cdot)$ is called a graph signal supported on G. Let the eigen-decomposition of the Laplacian matrix $\mathbf{L}$ be $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where $\Lambda = \text{Diag}(\lambda_1, ..., \lambda_n)$ is the diagonal of eigenvalues. and $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_n] \in \mathbf{R}^{n \times n}$ is a matrix whose i^{th} column is the eigenvector of $\mathbf{L}$ corresponding to λ_i. The eigenvalues are sorted in descending order, i.e. $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. The eigenvalues are often called *graph frequencies*, as the larger the λ_i, the more oscillatory its associated eigenvector $\mathbf{v}_i$ is.

Now, we can write the network dynamics $\mathcal{H}(\mathbf{L})$ as

$$\mathcal{H}(\mathbf{L}) = \mathbf{U}h(\mathbf{\Lambda})\mathbf{U}^\top \tag{6}$$

where $h(\mathbf{\Lambda}) = \text{Diag}(h(\lambda_1), ..., h(\lambda_n))$ with $h(\lambda) = \sum_{k=0}^{\infty} h_k \lambda^k$, and $\mathbf{U}$ is the eigenvector matrix $\mathbf{V}$ with reordered columns according to ascending order of $\{h(\lambda_1), ..., h(\lambda_n)\}$. A smooth linear network dynamics $\mathcal{H}(\mathbf{L})$ can be modeled by a K-low-pass graph filter Ramakrishna et al. [2020] that amplifies the low frequencies while suppresses the high frequencies, thereby promoting the smooth components that make up a graph signal.

Definition 1 *A graph filter $\mathcal{H}(\mathbf{L})$ is K-low-pass if η_K satisfies $0 \leq \eta_K < 1$, which is defined as*

$$\eta_K := \frac{\max_{i=K+1,...,n} |h(\lambda_i)|}{\min_{j=1,...,K} |h(\lambda_j)|}. \tag{7}$$

If we use the graph quadratic form $\mathbb{E}[\mathbf{y}^\top \mathbf{L}\mathbf{y}]$ for $\mathbf{y} = \mathcal{H}(\mathbf{L})\mathbf{x}$ with the stationary input $\mathbf{x}$ generated i.i.d. and $\mathbb{E}[\mathbf{x}\mathbf{x}^\top] = \mathbf{I}$, we can see that Ramakrishna et al. [2020]

$$\mathbb{E}[\mathbf{y}^\top \mathbf{L}\mathbf{y}] \approx \sum_{i=1}^{K} \lambda_i |h(\lambda_i)|^2 + \text{Tr}(\mathbf{L}). \tag{8}$$

In the cases of $\lambda_i \approx 0$ for $i = 1, ..., K$ (e.g., K-community stochastic block model), we can show that $\mathbb{E}[\mathbf{y}^\top \mathbf{L}\mathbf{y}] \approx 0$, i.e. the signal is smooth on the graph G. Indeed, many real-world network dynamics exhibit this smooth behavior, including the aforementioned opinion dynamics; the readers are referred to Ramakrishna et al. [2020] for more examples.

Revealing the Clusters via Payoff Signals We aim to extract the cluster structure of the payoffs from the payoff signals to serve as a prior that we can incorporate into the online learning procedure. The following observation that enables the identification of clusters is from Wai et al. [2020]. At round t, we can see that in expectation,

$$\mathbf{C}_{\mathbf{a}_t} = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{D}_t}[\mathbf{a}_t \mathbf{a}_t^\top] = \mathcal{H}(\mathbf{L})\mathbb{E}_{\mathbf{x}_t \sim \mathcal{D}_t}[\mathbf{x}_t \mathbf{x}_t^\top]\mathcal{H}(\mathbf{L}) + \sigma^2 \mathbf{I} = \gamma^2 \mathcal{H}(\mathbf{L})^2 = \mathbf{U}(\gamma h(\mathbf{\Lambda}))^2 \mathbf{U}^\top + \sigma^2 \mathbf{I}. \tag{9}$$

Since $h(\mathbf{\Lambda})$ is assumed to be a K-low-pass filter, we can identify the K eigenvectors in $\mathbf{V}_K \text{v}$ of K smallest eigenvalues of $\mathbf{L}$ from the eigenvector matrix $\mathbf{U}$ of $\mathbf{C}_{\mathbf{a}_t}$. Recall that if the nodes of a graph are clustered, one can use spectral clustering von Luxburg [2007] to perform community detection, by grouping the nodes based on the results of K-means clustering of the K eigenvectors associated with K smallest eigenvalues of the corresponding Laplacian matrix. Similarly, we can employ the spectral clustering technique to blindly cluster nodes according to the first K eigenvectors $\mathbf{U}_K$ of $\mathbf{C}_{\mathbf{a}_t}$. The accuracy of this blind clustering method depends on the number of observed signals $\mathbf{a}_1, ..., \mathbf{a}_m$, the low-pass ratio η_K of the data generating process, and K [Wai et al., 2020, Theorem 1].

3 Algorithm

Similar to Explore-then-Commit strategy Lattimore and Szepesvári [2020], we propose to learn to group the users into K clusters $\hat{V}_1, ..., \hat{V}_K$ first, then incorporate this knowledge as a regularizer for

learning the true parameter of each user (i.e., $\mathcal{H}(\mathbf{L})$). The cluster structure is encoded in a matrix $\hat{\mathbf{C}} \in \mathbb{R}^{n \times n}$, defined as

$$\hat{\mathbf{C}}_{i,j} = \begin{cases} 1 \text{ if } i,j \in \hat{V}_k, k = 1, ..., K \\ 0 \text{ otherwise} \end{cases} \tag{10}$$

Compared to GOBLin Cesa-Bianchi et al. [2013] and Yang et al. [2020], we don't require the knowledge of the whole network G beforehand to impose a Laplacian regularization; instead, we rely on the payoff signals to learn the cluster structure that summarizes the graph. The cluster structure $\mathbf{C}$ is used as the adjacency matrix of the input graph requested by GOBLin. The proposed method, Cluster-then-GOBLin, is presented in Algorithm 1.

Algorithm 1 Cluster-then-Bandit

Require: No. of clusters $K \geq 2$, no. of exploration rounds $M \geq 1$

1: Obtain the history of payoffs $\mathbf{a}_1(\bar{\mathbf{x}}_1), ..., \mathbf{a}_M(\bar{\mathbf{x}}_M) \in \mathbb{R}^n$

2: Run BlindCD to obtain $\hat{V}_1, ..., \hat{V}_K$

3: Define $\hat{\mathbf{C}}$ by

$$\hat{\mathbf{C}}_{i,j} := \begin{cases} 1 \text{ if } i \in \hat{V}_k, j \in \hat{V}_k (1 \leq k \leq K) \\ 0 \text{ otherwise} \end{cases} \tag{11}$$

4: Run a bandit algorithm with graph structure such as GOBLin Cesa-Bianchi et al. [2013], SpectralUCB/SpectralTS Kocak et al. [2020], GraphUCB Yang et al. [2020], etc.

Algorithm 2 BlindCD [Wai et al., 2020, Algorithm 1]

Require: Graph signals $\{\mathbf{y}^\ell\}_{\ell=1}^L$; desired number of communities K.

1: Compute the sample covariance $\widehat{\mathbf{C}}_{\mathbf{y}}$ as $\widehat{\mathbf{C}}_{\mathbf{y}} = \frac{1}{L} \sum_{\ell=1}^L \mathbf{y}^\ell (\mathbf{y}^\ell)^\top$.

2: Find the top-K eigenvectors of $\widehat{\mathbf{C}}_y$ (with the eigenvalues sorted in *descending* order). Denote the set of eigenvectors as $\widehat{\mathbf{V}}_K \in \mathbb{R}^{N \times K}$.

3: Apply the K-means method, which seeks to optimize $\min_{c_1,...,c_K \subseteq V} \sum_{k=1}^K \sum_{i \in C_k} \left\| \widehat{\mathbf{v}}_i^{\text{row}} - \frac{1}{|C_k|} \sum_{j \in C_k} \widehat{\mathbf{v}}_j^{\text{row}} \right\|_2^2$, where $\widehat{\mathbf{v}}_i^{\text{row}} := [\widehat{\mathbf{V}}_K]_{i,:} \in \mathbb{R}^K$, to obtain K communities $\widehat{\mathcal{C}}_1, \ldots, \widehat{\mathcal{C}}_K$.

4 Simulation

In the simulation, we implement the Algorithm 1, by first learning the cluster structure of the graph to use as the side observation required by GOBLin Cesa-Bianchi et al. [2013]. There are 30 users on the network. The network underlying the users is randomly generated from a stochastic block model of 3 clusters, with intra-cluster probability of connection being 0.9 and inter-cluster probability of a link being 0.05; see Fig. 1 for a visualization of the generated network. The learner first observes the payoff signals of 10 rounds (e.g., by playing any strategy) as input data generated from the model in Section 2, with the network dynamics $\mathcal{H}(\mathbf{L}) = (\mathbf{I} + \mathbf{L})^{-1}$, the excitation $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, 0.01\mathbf{I})$, and a noise level satisfying $\mathbb{E}[e_{i_t}^2 | \bar{\mathbf{x}}_t] = 0.01$. We then use Algorithm 2 to infer clustering labels to the users and subsequently a "pseudo-graph" with intra-cluster nodes being fully connected and inter-cluster nodes being disconnected. We then feed the constructed graph into GOBLin to play for $T = 3000$ rounds, and run LinUCB independently for each of the use for the same number of rounds, where there are 30 arms at each round for the learner to choose from, which are generated from $\mathcal{N}(\mathbf{0}, 0.01\mathbf{I})$.

The accumulated regret comparison between the independent LinUCB and our approach is presented in Fig. 2. Without assuming an available graph for regularization, we successfully construct a pseudo-graph that serve as an informative inductive bias for the linear bandit algorithm to better identify the linear parameters. Consequently, the accumulated regret of GOBLin with constructed graph is better than running independent LinUCB as the learner moves to later rounds by leveraging better approximated reward parameters.

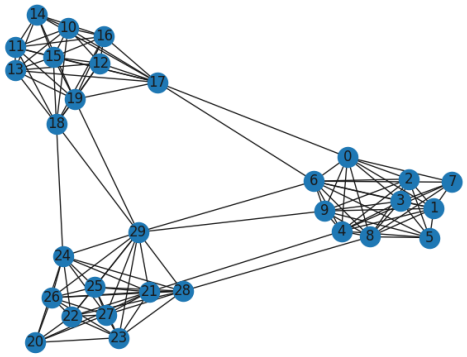

Figure 1: Network Generated from Three-community Stochastic Block Model

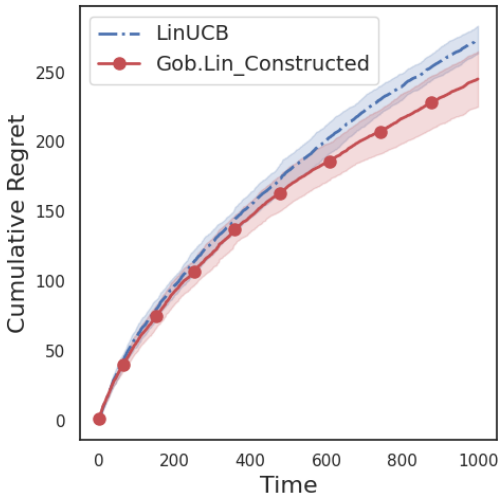

Figure 2: Accumulated Regret of Independent LinUCB vs. Cluster-then-Bandit (10 trials)

5 Conclusion

In this project, we present an approach to learn a prior in the form of a graph constructed from observed payoff signals that is assumed to follow community structure that is prevalent in real-world settings. We empirically show that using a graph-regularized linear bandit algorithm utilized the learned representation can outperform running independent linear bandit problems for each user.

A future extension is to fully analyze a regret bound of the proposed algorithm, relating the accumulated regret with a measure of smoothness of payoff signals on the underlying graph as well as the random graph model behind the observed network (e.g., how well-clustered the latent graph of users is). Another extension is to investigate other forms of representation learning for linear bandit that can potentially give a good prior that can be incorporated into bandit algorithms for improvement in regrets. For example, see Yang et al. [2021] for a study on the effectiveness of representation learning in linear bandit, specifically with matrix factorization models.

Contribution

Hoang-Son Nguyen and Nam Nguyen contributed equally to the literature review and model development. Hoang-Son Nguyen led the algorithm design and simulation efforts.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Abhineet Agarwal, Anish Agarwal, Lorenzo Masoero, and Justin Whitehouse. Multi-armed bandits with network interference. In *Advances in Neural Information Processing Systems*, 2024. URL `https://arxiv.org/abs/2405.18621`.

Anonymous. Demystifying online clustering of bandits: Enhanced exploration under stochastic and smoothed adversarial contexts. In *Submitted to The Thirteenth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=421D67DY3i`. Under Review.

Albert-Laszlo Barabasi. *Network Science*. Cambridge University Press, Cambridge, England, 2016.

Nicolò Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. A gang of bandits. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 757–765. PMLR, 2014.

Tomas Kocak, Remi Munos, Branislav Kveton, Shipra Agrawal, and Michal Valko. Spectral bandits. *Journal of Machine Learning Research*, 21(218):1–44, 2020.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, July 2020. ISBN 9781108486828.

Antonio Ortega, Pascal Frossard, Jelena Kovačević, José M. F. Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, pages 808–828, 2018.

Raksha Ramakrishna, Hoi-To Wai, and Anna Scaglione. A user guide to low-pass graph signal processing and its applications: Tools and applications. *IEEE Signal Processing Magazine*, 37(6): 74–85, 2020.

Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007. ISSN 1573-1375.

Hoi-To Wai, Santiago Segarra, Asuman E. Ozdaglar, Anna Scaglione, and Ali Jadbabaie. Blind community detection from low-rank excitations of a graph filter. *IEEE Transactions on Signal Processing*, 68:436–451, 2020.

Jiaqi Yang, Wei Hu, Jason D. Lee, and Simon Shaolei Du. Impact of representation learning in linear bandits. In *International Conference on Learning Representations*, 2021.

Kaige Yang, Laura Toni, and Xiaowen Dong. Laplacian-regularized graph bandits: Algorithms and theoretical analysis. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3133–3143. PMLR, 2020.