Title

Graph-Informed Sequential Decision Making

ISBN

9798293834754

Author

Wu, Shuang

Publication Date

2025-09-12

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Graph-Informed Sequential Decision Making

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Shuang Wu

2025

ABSTRACT OF THE DISSERTATION

Graph-Informed Sequential Decision Making

by

Shuang Wu
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2025
Professor Arash Ali. Amini, Chair

This dissertation studies *graph-informed sequential decision making*, where graphs enter the bandit problem either as *data*—actions, contexts, rewards—or as *structure* that couples decisions, observations and agents. Algorithms that leverage graph priors to accelerate learning under limited feedback are developed with comprehensive theoretical analysis in this work.

Part I introduces the backgrounds of the models and concepts in both statistical sequential decision making and machine learning on graphs. Chapter 1 elucidates bandit problems and algorithms, while Chapter 2 introduces graph learning models, from graph spectral theory to graph deep learning.

Part II presents the sequential decision making problems where the graph serves as data and our proposed algorithm,`GNN-TS`. Chapter 3 introduces two online problems in which each round presents a graph and only bandit feedback is revealed. First, in *online graph selection*, actions are full graphs (e.g., molecules, program graphs); the learner selects a graph and observes a noisy payoff. This framing highlights the need for graph representations and calibrated exploration at decision time. Second, in *online graph classification*, each input is

a graph and the learner must output a multi-class label with only action-dependent bandit feedback, linking the problem to multinomial logistic bandits over graph encodings. Chapter 4 presents the first project, graph neural Thompson Sampling, which pairs graph neural encoders with Thompson sampling as exploration rules. Theoretically, its performance is characterized via an effective-dimension parameter of a graph neural tangent kernel, yielding sublinear regret of order $\tilde{\mathcal{O}}(\tilde{d}\,T^{1/2})$.

Part III presents the sequential decision making problems where the graph serves as structure and our contribution in novel algorithms and problem unification. Chapter 5 first introduces the problems that decisions are *coupled* by a known graph. A Laplacian-regularized linear unified view that fuses content features with structural smoothness is presented for this problem. The second bandit problem is under the multi-agent setting, with a set of wide applications in interactive systems (recommendation, advertising, personalization). The second project is detailed in Chapter 6. The *Laplacian kernelized bandit* algorithms are proposed by inducing a multi-user kernel and Gaussian process style posterior, with confidence bounds derived from a bias–noise decomposition and regret governed by an effective dimension. The proposals are applied into a generalized design of the gang-of-bandits problem and competitive in both preferred regime and the other regimes.

Part IV introduces the future works and the conclusion on the study about sequential decision making with graph information. Chapter 7 presents the ongoing works and future investigation on this research topic. A novelty algorithm, GCN-Logistic bandit, is proposed as the ongoing project, for online graph classification with bandit feedback. A foundation work on random graph generation model in sequential decision making as well as the innovation for online recommendation with decision making on the item-user graph, are introduced as future works.

The dissertation of Shuang Wu is approved.

Guido F. Montufar

Qing Zhou

Yingnian Wu

Arash Ali. Amini, Committee Chair

University of California, Los Angeles

2025

To my beloved mother.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I want to express my first heartfelt appreciations to my PhD Advisor Prof. Arash A. Amini from UCLA, for his unwavering support, patient guidance, and rigorous mentorship throughout my Ph.D. research on sequential decision making with graph information over the past three years. His generosity with time and ideas has shaped the way I think about research. I will always be thankful for his support.

Secondly, I would also like to thank my committee members—Prof. Guido F. Montufar, Prof. Qing Zhou, and Prof. Yingnian Wu—for their careful critique and insightful, inspiring suggestions toward better status of this dissertation. It is my great pleasure to complete my adventure in Department of Statistics at UCLA.

I would like to express my profound gratitude to my collaborated professors and Post-Doc during my doctoral journey, from Purdue University to UCLA. I want to express my thanks to Prof. Guang Cheng, my previous PhD Advisor at Purdue University, for introducing me to statistical research and for his sustained guidance. I would like to appreciate Prof. Pan Li, for his supports and instructions to build my research abilities and bring me into the graph machine learning community. I also thank Dr. Chi Hua Wang for his guidance and encouragement as I entered the area of sequential decision making.

I am fortunate to have shared this journey with my Ph.D. cohort, collaborators, and colleagues: Dr. Yuantong Li, Dr. Mingxuan Zhang, Dr. Yiran Jiang, Dr. Zhanyu Wang, Dr. Luciano Vinas, Dr. Dehong Xu, and Dr. Shirong Xu. Our discussions and collaborations have been invaluable, and their friendship and support have been constant throughout my Ph.D. I would also like to thank my colleagues and friends: Ms. Xiaoke Zou, Dr. Jingxuan He, Dr. Dong Yuan, Dr. Zeyun Lu, Dr. Junting Ren, Dr. Wenjie Li, Dr. Yitao Li, Dr. Siqi Liang, Dr. Zhou Qin, Dr. Carrie Wu, Dr. Lei Shi, Dr. Yang Xu, Dr. Khan Nouman, Mr. Qining Zhang, Mr. Lin Gan, Mr. Jiale Han, Mr. Shaoxuan Chen, Mr. Yixi Xu, Ms. Chianti Shi, Ms. Qijia He, Ms. Yunong Liu, Ms. Minglu Zhao, Ms. Wenlu Xu, and Ms. Lan Tao.

Thank you for your companionship, encouragement, and help along the way.

Finally, my deepest gratitude goes to my mother for her enduring love. Her unwavering belief in me has been a constant source of strength through every challenge and every triumph. Her support and presence lifted me in difficult moments and made each achievement more meaningful.

VITA

2022–2025 Ph.D. candidate in statistics, Department of Statistics, UCLA.

2023, 2024 Applied Scientist Intern, Amazon Search.

2019–2021 Ph.D. student, Department of Statistics, Purdue University.

2017–2019 M.S. in Biostatistics, Department of Biostatistics, Columbia University

2013–2017 B.S. in Statistics, Department of Mathematics, Sun Yat-Sen University.

PUBLICATIONS

S. Wu, A. A. Amini. Graph Neural Thompson Sampling, *The 1st Reinforcement Learning Conference*, (**RLC 2024**) [paper].

S. Wu, M. Zhang, Y. Li, P. Li. When Federated Learning Meets Graph Neural Network, *The 1st International Workshop on Federated Learning with Graph Data,* **(CIKM 2022 Workshop)**[paper].

S. Wu, C. Wang, Y. Li, G. Cheng. Residual Bootstrap Exploration for Stochastic Linear Bandit, *Uncertainty in Artificial Intelligence*, **(UAI 2022)**[paper].

Part I

Background

CHAPTER 1

Sequential Decision Making

In this chapter, we introduce the sequential decision making problem, which is also usually called bandit problem. The first section in this chapter is the bandit problem formulation and the second section is the algorithms and exploration strategy in bandit.

1.1 Stochastic Bandit Problems

Bandit problems are sequential decision making problems where the only feedback given to the learner is a (noisy) reward of the chosen decision (LS20). Formally, suppose the learner interacts with an environment associated with a action space $\mathcal{A}$. At every time step $t \in [T]$, the learner makes a decision a_t from the available action set $\mathcal{A}_t \subset \mathcal{A}$. Then the learner receives the noisy reward

$$y_t = \mu(a_t) + \epsilon_t$$

where $\mu_a : \mathcal{A} \to \mathbb{R}$ is the true (unknown) reward function for arm a and ϵ_t is zero-mean subgaussian noise with constant σ_ϵ^2. Learner interacts with the environment following some policy and the objective for the learner is maximizing the expected cumulative reward. Bandit literature always use expected regret as evaluation of algorithm with some policy. The expected regret is defined as

$$R_T = \sum_{t=1}^{T} \mu(a_t^*) - \mu(a_t)$$

where $a_t^* = \text{argmax}_{a \in \mathcal{A}_t} \mu(a)$ is the optimal action at time t. In this stochastic interaction process between learner and environment, historical randomness up to round t is denoted as $\mathcal{F}_t$. Furthermore, $\mathbb{P}_t(\cdot) := \mathbb{P}(\cdot|\mathcal{F}_t)$ and $\mathbb{E}_t(\cdot) := \mathbb{E}[\cdot|\mathcal{F}_t]$ are conditional probability given $\mathcal{F}_t$. Note that if σ_ϵ^2 does not depend on actions in $\mathcal{A}$, the problem is called homoscedastic (homogeneity of variance), otherwise it is heteroscedasticity (heterogeneity of variance). We consider homoscedastic setting unless particularly stated.

Multi-Armed Bandit. Suppose the size of action space is fixed: $|\mathcal{A}| = K$, the problem is defined as a K-armed bandit problem, or more broadly, a multi-armed bandit (MAB) problem. The MAB framework posits a situation where there is no additional, arm-independent information obtained from the environment, and actions are simply indexed, leading to a formulation of the action space as $\mathcal{A} = \{1, ..., K\}$. In MAB problem, the key simplification is that the mean function is estimated without model: the learner follows the policy that directly estimates K parameters $\mu(1), ..., \mu(K)$.

Contextual Bandit. Suppose environment provides extra arm-independent contextual information, the bandit problem becomes a contextual setting. Suppose the contexts from environment at time t is denoted as a vector $\boldsymbol{c}_t$. In contextual bandit problem, the true reward function is a function of both action and context: $y_t = \mu(a_t, \boldsymbol{c}_t) + \epsilon_t$ and the expected regret in this setting is defined as $R_T = \sum_{t=1}^T \mu(a_t^*, \boldsymbol{c}_t) - \mu(a_t, \boldsymbol{c}_t)$ where $a_t^* = \text{argmax}_{a \in \mathcal{A}_t} \mu(a, \boldsymbol{c}_t)$. In this contextual setting, learner follows policy that assumes the true reward function is a (parametric or nonparametric) function f on the features $\boldsymbol{x}_t = \psi(a_t, \boldsymbol{c}_t)$ where ψ is some feature map. That means $\mu(a_t, \boldsymbol{c}_t) = f(\boldsymbol{x}_t)$. The feature map ψ can be known or unknown to learner. We also denote the best arm as $\boldsymbol{x}_t^* = \text{argmax}_{a \in \mathcal{A}_t} f(\boldsymbol{x}_t)$.

Linear Bandit. In contextual bandit setting, the reward function μ is assumed to be linearly parametrized as $f_{\boldsymbol{\theta}}(\boldsymbol{x}_t) = \boldsymbol{x}_t^\top \boldsymbol{\theta}$. This problem is commonly called linear bandit problem. The expected regret becomes $R_T = \sum_{t=1}^T \boldsymbol{x}_t^\top \boldsymbol{\theta} - \boldsymbol{x}_t^\top \hat{\boldsymbol{\theta}}_t$ where $\hat{\boldsymbol{\theta}}_t$ is the estimated

parameter up till time t. The estimation for $\hat{\boldsymbol{\theta}}_t$ in the policy for linear bandit is usually from classical linear regression methodologies, including ridge regression, LASSO regression, etc. We will introduce the related methods in the Section 1.2.

Kernelized Bandit. The reward function in contextual bandit setting, can be assumed as a nonparametric function. Formally, suppose the true reward function f is from some reproducing kernel Hilbert space (RKHS) induced by a kernel $K(\cdot, \cdot)$. The expected regret is expressed as $R_T = \sum_{t=1}^{T} f(\boldsymbol{x}_t^*) - f(\boldsymbol{x}_t)$. The policies for kernelized bandit usually use kernel ridge regression, Gaussian Process with kernel, etc.

Remark 1.1.1. *Contexts from environment are assumed to be independent of arm/action in this paper. However, many literature referred context signals to the feature $\psi(a, \boldsymbol{c}_t)$, which is an arm-dependent "context" vectors. With this notations, the learner observe "context" vectors $\{\boldsymbol{x}_{a,t} = \psi(a, \boldsymbol{c}_t)\}_{a \in \mathcal{A}_t}$ at time t. We will use contexts to refer $\boldsymbol{c}_t$ or $\boldsymbol{x}_{a,t}$, depending on the concrete situation.*

1.2 Bandit Algorithms

Bandit algorithm provides the policy for leaner to follow. The core problem in bandit algorithm is balancing exploration and exploitation. A good estimate for mean function μ allow for exploitation and an extra perturbation help the learner keep exploring. The μ function estimation is the model learning in the algorithm and the extra exploration design is the exploration principle. For the model learning, if there is no context, sampled mean of the rewards is the most common estimate and if the problem is contextual bandit, the parametrized model $f_{\boldsymbol{\theta}}$ has many choice including linear model, kernel model, neural model, neural linear model etc. For the exploration principle, the popular and robust types are Upper Confidence Bound (UCB) and Thompson Sampling (TS). In our work, we focus on UCB and TS and following part is the introduction. Denote $\hat{\mu}_t$ as the estimate mean function

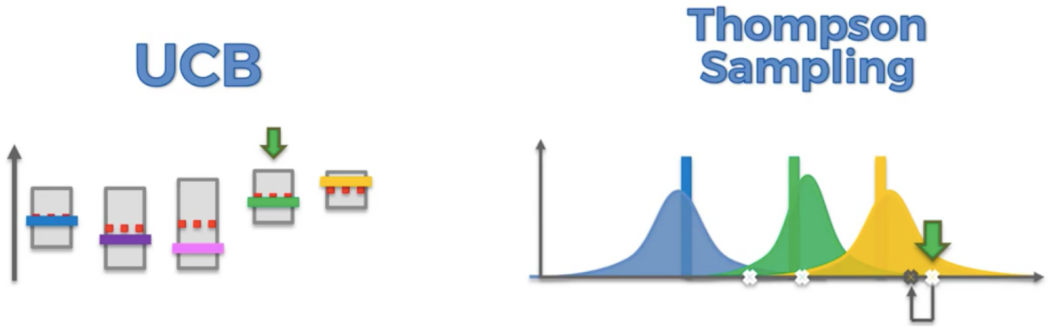

Figure 1.1: Illustration of UCB and TS. UCB provides deterministic optimism and TS provides randomized exploration.

at time t.

Upper Confidence Bound. The Upper Confidence Bound (UCB) algorithm is based on the principle of optimism in the face of uncertainty. The idea is similar to interval estimation in statistical inference: instead of estimating rewards mean by point estimate $\hat{\mu}_t$, a confidence interval for the estimate value is utilized and the upper confidence bound is the guideline for decision making. Formally, in MAB problem, the upper confidence bound with confidence level $1-\delta \in (0,1)$ for arm $i \in [K]$ is

$$\tilde{\mu}_t^{\text{ucb}}(i) = \hat{\mu}_t^{\text{ucb}}(i) + \sigma_\epsilon \sqrt{\frac{2\log(1/\delta)}{T_t(i)}} \tag{1.1}$$

where $\hat{\mu}_t^{\text{ucb}}(i) = T_t(i)^{-1} \sum_{\tau=1}^{t} y_\tau \mathbb{I}\{a_\tau = i\}$ is the sample mean of the rewards from arm i and $T_t(i) := \sum_{\tau=1}^{t} \mathbb{I}\{a_\tau = i\}$ is the number of times to select i up to round t. In practice, if the subgaussian constant is unknown, it can be estimated using sample variance. The policy that UCB provides for learner is $a_t = \text{argmax}_{a \in \mathcal{A}} \tilde{\mu}_t^{\text{ucb}}(a)$. In contextual setting, the upper confidence bound is

$$\tilde{\mu}_t^{\text{ucb}}(a, \boldsymbol{c}_t) = \hat{\mu}_t^{\text{ucb}}(a, \boldsymbol{c}_t) + \beta_t(\delta)\hat{\sigma}_t(a, \boldsymbol{c}_t)$$

where $\hat{\sigma}_t(a, \boldsymbol{c}_t)$ is the estimated standard deviation representing the uncertainty for exploration and $\beta_t(\delta)$ is the constant for confidence set: it satisfies

$$\mathbb{P}(\forall a \in \mathcal{A}_t, |\mu(a, \boldsymbol{c}_t) - \hat{\mu}_t^{\text{ucb}}(a, \boldsymbol{c}_t)| < \beta_t(\delta)\hat{\sigma}_t(a, \boldsymbol{c}_t)) \geq 1 - \delta. \tag{1.2}$$

The $\hat{\sigma}_t(a, \boldsymbol{c}_t)$ in contextual setting is the same as the role of $\sigma_\epsilon/\sqrt{T_t(i)}$ in MAB problem: the standard deviation for estimation. Note that one extension is allowing δ_t which changes with time. The model for reward function determines for concrete equations for $\hat{\mu}_t^{\text{ucb}}(a, \boldsymbol{c}_t)$, $\beta_t(\delta)$ and $\hat{\sigma}_t(a, \boldsymbol{c}_t)$.

Thompson Sampling. Thompson sampling is a Bayesian approach whose idea is sampling from the posterior distribution and playing the optimal action. The extra exploration is the uncertainty from the posterior and the perturbed mean is the guideline for decision making. The TS algorithm is based on the principle of optimism in the face of uncertainty. Precisely, the perturbed mean estimate for arm $a \in \mathcal{A}_t$ is sampled from the posterior

$$\tilde{\mu}_t^{\text{ts}}(a) \sim \mathcal{P}_t(a, \mathcal{F}_{t-1})$$

where $\mathcal{P}_t(a, \mathcal{F}_{t-1})$ is the posterior distribution for $\mu(a)$ given $\mathcal{F}_{t-1}$. This Bayesian stochastic process needs priors on $\{\mu(a)\}_{a\in\mathcal{A}}$. The most popular process is Gaussian process which further assumes the sequence of noises $\{\epsilon_t\}_{t=1}^T$ are Gaussian and priors are also Gaussian with variance ν_0^2. In this Thompson Sampling with Gaussian prior (TS-G), the perturbed mean for arm i becomes

$$\tilde{\mu}_t^{\text{ts}}(i) = \hat{\mu}_t^{\text{ts}}(i) + z\nu_t$$

where $z \sim \mathcal{N}(0,1)$ and the recursive updates for $\hat{\mu}_t^{\text{ts}}$ and ν_t are

$$\begin{aligned}\nu_{t+1}^2 &= \frac{1}{1/\nu_t^2 + 1/\sigma_\epsilon^2} \\ \hat{\mu}_{t+1}^{\text{ts}} &= \frac{\sigma_\epsilon^2}{\sigma_\epsilon^2 + \nu_t^2}\hat{\mu}_t^{\text{ts}} + \frac{\nu_t^2}{\sigma_\epsilon^2 + \nu_t^2} y_t\end{aligned}$$

and σ_ϵ^2 can be further estimated if it is unknown. The policy that TS provides for learner is $a_t = \text{argmax}_{a \in \mathcal{A}} \tilde{\mu}_t^{\text{ts}}(a)$. Note that an extension for Gaussian process in TS is imposing prior on variance of Gaussian, which is called Thompson Sampling with Inverse-Gamma prior (TS-IG) (HT14). We refer Thompson Sampling to TS-G unless otherwise stated. In contextual setting, the perturbed mean is

$$\tilde{\mu}_t^{\text{ts}}(a, \boldsymbol{c}_t) = \hat{\mu}_t^{\text{ts}}(a, \boldsymbol{c}_t) + z\nu\hat{\sigma}_t(a, \boldsymbol{c}_t)$$

where $\hat{\sigma}_t(a, \boldsymbol{c}_t)$ is the estimated posterior standard deviation and μ is a scaling constant from prior and both of them represent the uncertainty for exploration in TS. Similar to UCB, $\hat{\mu}_t^{\text{ts}}(a, \boldsymbol{c}_t)$ and $\hat{\sigma}_t(a, \boldsymbol{c}_t)$ rely on the model assumption for mean function μ.

Linear Bandit Algorithm. In linear bandit setting, a linear model, $f_{\boldsymbol{\theta}}(\boldsymbol{x}_t) = \boldsymbol{x}_t^\top \boldsymbol{\theta}$, is used. Here we present the classical ridge regression protocol. The loss is defined as $\ell(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^t (y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2$ where λ is the regularization hyperparameter. Then the least squared estimate gives

$$\begin{aligned}\hat{\boldsymbol{\theta}}_t &= (\sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i^\top + \lambda \boldsymbol{I})^{-1} \sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \\ \hat{\mu}_t(a, \boldsymbol{c}_t) &= \hat{\boldsymbol{\theta}}_t^\top \psi(a, \boldsymbol{c}_t) \\ \hat{\sigma}_t^2(a, \boldsymbol{c}_t) &= \psi^\top(a, \boldsymbol{c}_t)(\sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i^\top + \lambda \boldsymbol{I})^{-1} \psi(a, \boldsymbol{c}_t)\end{aligned}$$

and $\beta_t(\delta)$ could be the hyperparameter in UCB while ν is the hyparameter in TS. This above ridge regression procedure leads to Linear UCB (`LinUCB` (CLR11; APS11)) and Linear TS (`LinTS` (AG13)) for linear bandit setting.

Remark 1.2.1. *`LinUCB` is proposed by finding a upper bound for $\beta_t(\delta)\hat{\sigma}_t(a, \boldsymbol{c}_t)$ by a confidence ellipsoid for $\hat{\boldsymbol{\theta}}_t$. By introducing complex function approximation for more expressive power, a neural network model with UCB gives `NeuralUCB` (ZLG20) algorithm and a kernel model with UCB gives `KernelUCB` (VKM13) algorithm. Similar works are investigated in the track of TS: `NeuralTS`(ZZL20), `KernelTS` (CG17) are proposed. Consequently, there are many variants of algorithms using UCB or TS exploration principle.*

Remark 1.2.2. *UCB and TS are not the only choices for exploration design in sequential decision making. In fact, the oldest but useful strategy is epsilon-greedy method. The Lower Confidence Bound(LCB) method is used in practice. Also, bootstrapping based pertubation such as residual bootstrapping (WYH20; WWL22) are proposed with good emprical performance. Some index has awareness of the variance, such as information-directed sampling(IDS) is proposed (RV14; HLD21). Finally, we only introduce the fundamental bandit setting here while there are more setting including the adversarial bandit setting.*

CHAPTER 2

Graph Learning

In this section, we introduce some graph learning models and graph-based techniques. The first section in this chapter is the classical graph Laplacian methods and the second section is a brief introduction to the deep learning models on graphs. For simplicity, we refer graphs to the undirected graphs in our works.

Notations. We denote an undirected graph with N nodes as $G = (\boldsymbol{X}, \boldsymbol{A})$ where $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ is the feature matrix with d features and $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is the weighted(or unweighted) adjacency matrix. Rows of $\boldsymbol{X}$ are node features. Corresponding degree matrix is denoted as $\boldsymbol{D}$ and the normalized adjacency matrix is defined as $\bar{\boldsymbol{A}} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ and its eigendecomposition is $\bar{\boldsymbol{A}} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$. The normalized Laplacian matrix is denoted as $\boldsymbol{L} = \boldsymbol{I} - \bar{\boldsymbol{A}}$. We also denote $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$ be the eigenvalues of L with orthonormal eigenvectors $\{u_i\}_{i=1}^N$.

2.1 Graph Laplacian Techniques

Spectral Graph Filtering. In spectral graph theory, feature or outcome of a node is also called signal or attribute. The eigenvalues $(\boldsymbol{I} - \boldsymbol{\Lambda})$ and eigenvectors $(\boldsymbol{U})$ of the normalized Laplacian matrix $\boldsymbol{L}$ represent frequencies and the graph Fourier bases, respectively. Since $\boldsymbol{L}$ and $\bar{\boldsymbol{A}}$ share the same eigenspace, we can use $\boldsymbol{U}$ as the summary of the graph Fourier bases and $\boldsymbol{\Lambda}$ to represent the information about frequencies. Consider one graph signal $\boldsymbol{x} \in \mathbb{R}^N$, the *Graph Fourier transform (GFT)* is defined as $\tilde{\boldsymbol{x}} = \boldsymbol{U}^\top\boldsymbol{x} \in \mathbb{R}^N$, which maps from the

signal domain to the spectral domain. A graph spectral filter $g : [0, 2] \to \mathbb{R}$ is a function on frequencies. Applying this filter on graph signal $\boldsymbol{x}$, the spectral graph convolution or graph convolution is defined as

$$\boldsymbol{h} = \boldsymbol{U} g(\Lambda) \boldsymbol{U}^\top \boldsymbol{x},$$

where filter g is applied on diagonal matrix Λ element-wisely and $\boldsymbol{h} \in \mathbb{R}^N$. In general, the spectral filter g can be any function while it is often set to be polynomial function due to the universal function approximation of polynomials (SNF13). If g is a polynomial function, the graph convolution is also denoted as

$$\boldsymbol{h} = g(\bar{\boldsymbol{A}})\boldsymbol{x} = \boldsymbol{U}(\alpha_0 + \alpha_1 \Lambda + \alpha_2 \Lambda^2 + \cdots + \alpha_k \Lambda^k)\boldsymbol{U}^\top \boldsymbol{x}$$

where k is the order of polynomial and $\{\alpha_\kappa\}_{\kappa=1}^k$ are the polynomial coefficients. In most cases, the attributed graph has multiple signals, which means the node features are assumed to be vectors which summarized in feature matrix $\boldsymbol{X} \in \mathbb{R}^{N \times d}$. The output of convolution channels can be extended from one dimension to multiple dimensions. In this setting with multiple features and multiple output channels, the k-th order polynomial filter is

$$\boldsymbol{H} = \boldsymbol{X}\boldsymbol{W}_0 + \bar{\boldsymbol{A}}^1 \boldsymbol{X}\boldsymbol{W}_1 + \cdots + \bar{\boldsymbol{A}}^k \boldsymbol{X}\boldsymbol{W}_k. \tag{2.1}$$

The spectral graph filtering provides an useful way to fusing structure information $\boldsymbol{A}$ and feature information $\boldsymbol{X}$. Note that this spectral graph filtering/convolution is connected to spectral graph neural networks, which is also introduced in Section 2.2.

Linear Graph Convolution. Suppose the spectral filter g is simply a linear function. The graph convolution in this special case is called linear graph convolution, which is defined as

$$\boldsymbol{h} = \alpha_0 \boldsymbol{x} + \alpha_1 \bar{\boldsymbol{A}} \boldsymbol{x},$$

where α_0 and α_1 are parameters for the linear filter. When α_0 is assumed to be nonzero, this is an inhomogeneous linear graph convolution while we call it homogeneous linear graph convolution when $\alpha_0 = 0$. In the setting with multiple features and multiple output channels, the homogeneous linear graph convolution is defined as

$$\boldsymbol{H} = \bar{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}$$

where $\boldsymbol{W}$ is the parameters for filtering. The linear graph convolution is the concept of a base layer with propagation in graph convolutional networks and message passing networks, presented in Section 2.2.

Laplacian Regularization. For a signal $\boldsymbol{f} \in \mathbb{R}^N$ on the nodes in a graph (i.e., f_v on node v), the *Dirichlet energy* associated with the (normalized) Laplacian $\boldsymbol{L}$ is the quadratic form as

$$\boldsymbol{f}^\top \boldsymbol{L}\boldsymbol{f} = \frac{1}{2}\sum_{u,v} A_{uv}\Big(\frac{f_u}{\sqrt{d_u}} - \frac{f_v}{\sqrt{d_v}}\Big)^2,$$

where d_v represents the degree of node v. This *Dirichlet energy* is known as the (graph) Laplacian regularization, which penalizes variation of f across edges By recalling that $\{u_i\}_{i=1}^N$ forms the eigenbasis, we can expand the signal over graph as $\boldsymbol{f} = \sum_{u=i}^N c_i \boldsymbol{u}_i$. This expansion of $\boldsymbol{f}$ gives

$$\boldsymbol{f}^\top \boldsymbol{L}\boldsymbol{f} = \sum_{u=i}^{N} \lambda_i c_i^2,$$

so Laplacian regularization suppresses high-frequency components (large λ_i), enforcing *graph smoothness* (homophily). We provide two usecases for the Laplacian regularization.

Example 1: Graph signal recovery / denoising. A canonical instance is least-squares denoising of a graph signal from noisy observations $\boldsymbol{y} \in \mathbb{R}^N$:

$$\min_{f\in\mathbb{R}^N} \sum_{i=1}^{N}(y_i - f_i)^2 \;+\; \frac{\lambda}{2}\sum_{(u,v)} A_{uv}\,(f_u - f_v)^2 \;=\; \|\boldsymbol{y} - \boldsymbol{f}\|_2^2 + \lambda\,\boldsymbol{f}^\top \boldsymbol{L}\boldsymbol{f}. \tag{2.2}$$

The solution is $(\boldsymbol{I}+\lambda \boldsymbol{L})^{-1}\boldsymbol{y}$, i.e., a graph low-pass filter that respects topology. This is also the basic template behind semi-supervised learning and label propagation with Laplacian regularization.

Example 2: Laplacian Regularized Linear Regression. Suppose there is an underlying graph across a set of covariates in linear regression. The noisy outcome is assumed as $y = \boldsymbol{\theta}^\top \boldsymbol{x} + \varepsilon$. Enforcing homophily across the covariate graph, the regularized linear regression problem is

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{i=1}^{n}(y_i - \boldsymbol{\theta}^\top \boldsymbol{x})^2 + \frac{\lambda}{2}\sum_{(u,v)} A_{uv}\,(\theta_u - \theta_v)^2$$

implements Laplacian regularization over *parameters*. This regularization also falls into the Tikhonov regularization term family, following the ridge (ℓ_2) regularization and lasso (ℓ_1) regularization. So it involves the sparsity on $\boldsymbol{\theta}$ by enforcing the graph smoothness on it.

Remark 2.1.1. *We use the symmetric normalized version of Laplacian $\boldsymbol{L} = \boldsymbol{I} - \bar{\boldsymbol{A}}$ which is preferable under heavy degree heterogeneity, as they yield scale-invariant smoothness. The unnormalized version $\boldsymbol{D}-\boldsymbol{A}$ and the random walk version $\boldsymbol{I}-\boldsymbol{D}^{-1}\boldsymbol{A}$ is also useful. Different formulation of the regularization will appear while they serve as the same effect essentially.*

2.2 Graph Deep Learning

Spectral Graph Neural Networks. Spectral Graph Neural Networks(GNNs) are built on the *graph Fourier transform (GFT)* and spectral graph filtering, given by the eigen-decomposition of $\boldsymbol{L}$. Recent studies suggest many popular methods use the polynomial spectral filters to achieve graph convolutions (WZ22). For a signal vector on graph $\boldsymbol{x} \in \mathbb{R}^N$, the graph convolution operation on $\boldsymbol{x}$ using a k-th order polynomial spectral graph filter g

is

$$\boldsymbol{y} = \boldsymbol{U}(\alpha_0 + \alpha_1\Lambda + \alpha_2\Lambda^2 + \cdots + \alpha_k\Lambda^k)\boldsymbol{U}^\top\boldsymbol{x}$$

where $\boldsymbol{y}$ is the filtering result and $\{\alpha_i\}_{i=1}^k$ are the polynomial weights. Since the Laplacian matrix $\boldsymbol{L}$ and $\bar{\boldsymbol{A}}$ share the same eigenspace, the graph filtering can be expressed as $\boldsymbol{Y} = \sum_{i=0}^k \theta_i \boldsymbol{L}^i\boldsymbol{x}$ where the polynomial in $\boldsymbol{L}$ is parametrized by $\theta_1, \cdots, \theta_k$. Because $(\boldsymbol{L}^k\boldsymbol{x})_u$ depends only on nodes within k hops of u, polynomial filters are *K-localized* in the vertex domain—yielding message-passing behavior without computing eigenvectors. Thus the graph convolution on the multi-channel signals $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ in (2.1), can be expressed as $\boldsymbol{Y} = \sum_{i=0}^k \boldsymbol{L}^i\boldsymbol{X}\,\boldsymbol{\Theta}_i$ which is called a spectral convolution layer parametrized by $\boldsymbol{\Theta}$. A spectral GNN is denoted as

$$\boldsymbol{Y} = \sum_{i=0}^{k} \theta_i \boldsymbol{L}^i f(\boldsymbol{x}; \boldsymbol{\Theta}_i), \tag{2.3}$$

where $f(\boldsymbol{x}; \boldsymbol{\Theta}_i)$ denotes Multi-Layer Perceptron (MLP).

Example: ChebNet. Spectral convolution and spectral GNN can be extended using different set of polynomial basis, while (2.3) use the monomial basis. For example, one can use Chebyshev polynomials T_k on the scaled operator $\widetilde{\boldsymbol{L}} = 2\boldsymbol{L}/\lambda_{\max}(\boldsymbol{L}) - \boldsymbol{I}$:

$$\boldsymbol{Y} = \sum_{k=0}^{k} \theta_i\, T_i(\widetilde{\boldsymbol{L}}) f(\boldsymbol{x}; \boldsymbol{\Theta}_i), \tag{2.4}$$

which enables filtering via the Chebyshev recurrence (DBV16; HWW22).

Graph Convolutional Networks (GCN). A popular specialization is the first-order (k=1) approximation together. Such filters with nonlinearities yields the *GCN* propagation rule

$$\boldsymbol{H}^{(\ell+1)} \;=\; \sigma\big(\bar{\boldsymbol{A}}\,\boldsymbol{H}^{(\ell)}\,\boldsymbol{W}^{(\ell)}\big), \qquad \boldsymbol{H}^{(0)} = \boldsymbol{X}, \tag{2.5}$$

where $\boldsymbol{W}^{(\ell)}$ are trainable weights for layer ℓ, and σ (e.g., ReLU) is a pointwise nonlinearity. Equation (2.5) can be interpreted as a fixed spectral filter (a particular polynomial in $\boldsymbol{L}$)

followed by learnable channel mixing. Stacking L such layers yields an L-hop receptive field and admits an interpretation as repeated feature diffusion with learnable channel projections.

Message Passing Neural Nets (MPNN). An *MPNN* framework maintains hidden states $\boldsymbol{h}_u^{(\ell)} \in \mathbb{R}^{d_\ell}$ for each node u and iteratively updates them via localized *message* and *update* operators over $\ell = 0, \ldots, L-1$:

$$m_u^{(\ell+1)} = \bigoplus_{v \in \mathcal{N}(u)} \phi_m\big(\boldsymbol{h}_u^{(\ell)}, \boldsymbol{h}_v^{(\ell)}, A_{uv}; \boldsymbol{\Theta}_m^{(\ell)}\big), \qquad \boldsymbol{h}_u^{(\ell+1)} = \phi_u\big(h_u^{(\ell)}, m_u^{(\ell+1)}; \boldsymbol{\Theta}_u^{(\ell)}\big), \tag{2.6}$$

where $\mathcal{N}(u)$) denotes neighbors of u, ϕ_m, ϕ_u are learnable (typically MLP/GRU-style) maps, with parameters $\boldsymbol{\Theta}_m^{(\ell)}$ and $\boldsymbol{\Theta}_u^{(\ell)}$ respectively, and $\bigoplus$ is a permutation-invariant aggregation (sum/mean/max or attention-weighted sum). The initialization is $\boldsymbol{h}_u^{(0)} = \boldsymbol{x}_u$. After L rounds, node embeddings $\{\boldsymbol{h}_u^{(L)}\}$ can be used for node-level tasks; for graph-level tasks one applies a permutation-invariant *readout*

$$\boldsymbol{h}_G = \rho\big(\{\boldsymbol{h}_u^{(L)} : \forall u\}\big), \qquad \rho \in \{\text{sum}, \text{mean}, \text{ max, Set2Set, attention-pooling}\}.$$

Many popular architectures are special cases of (2.6): *GCN* uses a fixed, degree-normalized sum with linear ϕ_u; *GraphSAGE* chooses $\bigoplus \in \{\text{mean}, \text{max}, \text{LSTM}\}$ and concatenates $\boldsymbol{h}_u^{(\ell)}$ with the aggregated message; *GAT* realizes $\bigoplus$ as a learned attention-weighted sum; *GIN* uses sum aggregation plus a powerful MLP update to match the expressivity of the 1-Weisfeiler–Leman (1-WL) test on node color refinement.

Graph Attention Networks (GAT). Graph Attention Networks (GAT) (VCC17) replaces fixed (e.g., degree–normalized) weights with *learned, data–dependent* coefficients on each edge, yielding anisotropic aggregation that adapts per node and per neighbor. Let $\boldsymbol{h}_u^{(\ell)} \in \mathbb{R}^{d_\ell}$ be the node state at layer ℓ and $\boldsymbol{W}^{(\ell)} \in \mathbb{R}^{d_\ell \times d_{\ell+1}}$ a shared linear map. For nodes

u and v , define an *unnormalized attention score* with an additive (Bahdanau-style) kernel:

$$e_{uv}^{(\ell)} \;=\; \mathrm{LeakyReLU}\Big(\, \boldsymbol{a}^{(\ell)\top}\big[\, \boldsymbol{W}^{(\ell)}\boldsymbol{h}_u^{(\ell)} \,\|\, \boldsymbol{W}^{(\ell)}h_v^{(\ell)} \,\big] \;+\; \boldsymbol{b}^{(\ell)} \Big),$$

where $\boldsymbol{a}^{(\ell)}$ and $\boldsymbol{b}^{(\ell)}$ are learnable, and $\|$ denotes concatenation. The *normalized* attention coefficients are

$$\alpha_{uv}^{(\ell)} \;=\; \mathrm{softmax}_{v\in\mathcal{N}(u)\cup\{u\}}\big(e_{uv}^{(\ell)}\big) \;=\; \frac{\exp(e_{uv}^{(\ell)})}{\sum_{w\in\mathcal{N}(u)\cup\{u\}}\exp(e_{uw}^{(\ell)})}, \qquad \sum_v \alpha_{uv}^{(\ell)} = 1,$$

and the layer update is an attention–weighted sum,

$$\boldsymbol{h}_u^{(\ell+1)} \;=\; \sigma\Big(\sum_{v\in\mathcal{N}(u)\cup\{u\}} \alpha_{uv}^{(\ell)}\, \boldsymbol{W}^{(\ell)}\boldsymbol{h}_v^{(\ell)}\Big).$$

This realizes a special case of the MPNN schema with $\bigoplus = \sum$ and a learned, context–dependent kernel $\alpha_{uv}^{(\ell)}$ that can emphasize informative neighbors and de–emphasize noisy ones. Edge features e_{uv} can be incorporated by augmenting the score (e.g., $e_{uv}^{(\ell)} \leftarrow e_{uv}^{(\ell)} + \boldsymbol{w}^\top\psi(e_{uv})$), enabling relation–aware attention. To stabilize training and enrich expressivity, GAT uses J heads in parallel:

$$\boldsymbol{h}_u^{(\ell+1)} = \|_{j=1}^{J}\, \sigma\Big(\sum_v \alpha_{uv}^{(j,\ell)}\, \boldsymbol{W}^{(j,\ell)}\boldsymbol{h}_v^{(\ell)}\Big) \quad \text{(hidden layers)},$$

$$\boldsymbol{h}_u^{(\ell+1)} = \frac{1}{J}\sum_{j=1}^{J}\sum_v \alpha_{uv}^{(j,\ell)}\, \boldsymbol{W}^{(j,\ell)}\boldsymbol{h}_v^{(\ell)} \quad \text{(final layer)}.$$

Compared with fixed spectral/spatial smoothers, attention makes aggregation *anisotropic* and *degree–aware*, often improving performance under heterophily or when only a subset of neighbors is relevant.

Graph Transformers. Graph Transformers(YJK19) transplant the transformer's multi–head self–attention to nodes while injecting *graph structure* as positional/relational bias. Let $\boldsymbol{X} \in \mathbb{R}^{N \times d_0}$ be input node features. For head $j = 1, \ldots, J$, compute queries/keys/values

$$\boldsymbol{Q}_j = \boldsymbol{X}\boldsymbol{W}_Q^{(j)}, \qquad \boldsymbol{K}_j = \boldsymbol{X}\boldsymbol{W}_J^{j)}, \qquad \boldsymbol{V}_j = \boldsymbol{X}\boldsymbol{W}_V^{(j)}$$

with $\boldsymbol{W}_{\{Q,K,V\}}^{(j)} \in \mathbb{R}^{d_0 \times d}$. Pure self–attention is *permutation–equivariant* but needs *structural context*. Denote by $\boldsymbol{B} \in \mathbb{R}^{N \times N}$ a bias matrix —e.g., shortest–path distances, random–walk positional encodings, Laplacian eigenfeatures, centrality, or edge features aggregated into pairwise terms. A single transformer layer then aggregates globally:

$$\mathrm{Attn}_j(\boldsymbol{X}) \;=\; \mathrm{softmax}\Big(\frac{\boldsymbol{Q}_j\boldsymbol{K}_j^\top}{\sqrt{d}} + \boldsymbol{B}\Big)\,\boldsymbol{V}_j, \qquad \boldsymbol{H}^{(1)} \;=\; \Big\|_{j=1}^{J} \mathrm{Attn}_j(\boldsymbol{X})\,\boldsymbol{W}_O^{(j)}, \tag{2.7}$$

followed by residual/normalization and an MLP block. Compared to localized message passing, (2.7) supplies *global* receptive fields in one hop while the bias $\boldsymbol{B}$ injects graph topology and edge semantics; when $\boldsymbol{B}$ masks non–edges it recovers sparse (neighbor–only) attention as a special case. Structural encodings can be concatenated to $\boldsymbol{X}$ (node positional features) and/or folded into $\boldsymbol{B}$ (pairwise/edge priors), yielding anisotropic, structure–aware aggregation (BAY21). With suitable structural encodings, Graph Transformers capture long–range dependencies and alleviate over–squashing typical of deep localized MPNNs, often surpassing 1–WL expressivity. Attention's global routing is effective under heterophily (useful neighbors may be far apart), while $\boldsymbol{B}$ steers the model toward graph–consistent interactions and preserves inductive biases (e.g., distance decay, edge types).

Part II

Graph as Data in Bandit

CHAPTER 3

Graph as Action, Context, Reward in Bandit Problems

In this section, we introduce the bandit problems where graph serves as data. The three components which could be viewed as data in bandit are action, context and reward. Our first section introduces the online graph selection problem where graphs are actions in the online decision making. The second section is the introduction to online graph classification where graphs are served as the environment context in sequential decision making. The last section is a brief introduction to the special bandit problems where graphs are rewards.

3.1 Online Graph Selection

Problem View. In the online graph selection problem, each action is itself a graph. Let $\mathcal{G}$ denote a graph space of candidate graphs. It can be stated as a bandit problem where graphs are the actions in decision making. At round t, the learner selects a graph $G_t \in \mathcal{G}_t \subseteq \mathcal{G}$ and observes a noisy scalar reward $y_t = \mu(G_t) + \epsilon_t$ with $\mu : \mathcal{G} \to \mathbb{R}$ the unknown true graph-to-reward map and ϵ_t is the noise term. The goal is to minimize cumulative regret $R_T = \sum_{t=1}^{T} \big(\mu(G_t^\star) - \mu(G_t)\big)$ where $G_t^\star \in \arg\max_{G \in \mathcal{G}_t} \mu(G)$. The bandit algorithms for online graph selection, at a high level, two ingredients recur: a graph learning model $f_{\boldsymbol{\theta}}(G)$ (via spectral features, MPNNs, GATs, or transformers) and an exploration rule that trades value and uncertainty. Classical bandit ideas, UCB-style and TS-styple, remain applicable.

This setting captures many real systems where the unit of decision is structured: molecules in drug discovery, circuits/netlists in hardware design, program graphs in compilers, and page

or widget-layout graphs in information systems. In all cases, leveraging structural similarity to generalize across $\mathcal{G}$ is essential to reduce trial costs. Unlike standard arms, graphs admit rich notions of similarity (motifs, spectra, substructures), and small edits can have nonlinear reward effects. Exploration is thus not merely "try a different arm," but "probe a new region of a combinatorial space" under bandit feedback.

Graph Neural Network Bandit. In our work to date, we instantiated the above template with an *optimistic* approach, `GNN-UCB` (KKB22), which uses a graph neural encoder to score candidates and adds a confidence bonus for selection. `GNN-UCB` uses a simple GNN encoder architecture: a linear graph convolution plus an MLP. With this simple GNN predictor, the graph neural tangent kernel (GNTK) is applied to linearize the GNN model. This demonstrated the promise of graph-aware representations for rapid discovery in large libraries. We proposed `GNN-TS` in Chapter 4, which retains the same encoder family but adopts *Thompson Sampling* to randomize exploration.

3.2 Online Graph Classification

Problem View. In *online graph classification*, the input at round t is a graph $G_t \in \mathcal{G}$ and the learner must output a class label $a_t \in \mathcal{A} = \{1, \ldots, K\}$ immediately. This can be formulated as a bandit problem where graphs are the environment contexts in bandit. Let $y_t \in \mathcal{A}$ denote the true class. The full reward feedback is (y_t, r_t) where

$$r_t = \mathbf{1}\{a_t = y_t\}, \qquad \mathbb{E}[r_t \mid G_t, a_t] \;=\; \mu_{a_t}(G_t),$$

where $\mu_k(G) = \mathbb{P}(y = k \mid G)$ is the class-probability function. Regret is measured against the Bayes action $a_t^\star \in \arg\max_k \mu_k(G_t)$ as $R_T = \sum_{t=1}^{T} \big(\mu_{a_t^\star}(G_t) - \mu_{a_t}(G_t)\big)$. This is an *online, multi-class* decision making problem in which graphs are the *data (context)*. Real applications for this problem include Molecular function/toxicity typing: assign a discrete

property class to a candidate molecule graph during iterative screening. Program intent or bug triage: Program intent or bug triage: classify program dependence graphs into categories (e.g., optimization/bug types) as code streams in. Social/interaction graphs: categorize dynamic ego-graphs (e.g., bot vs. human cluster types) with limited immediate feedback.

Full Feedback vs. Bandit Feedback. Under *full feedback* (supervised online learning), after predicting a_t the learner observes the true class y_t (equivalently the entire loss vector or all class probabilities), enabling direct updates with the full cross-entropy. In contrast, under *bandit feedback* (JMR19) the agent only observes $r_t \in \{0, 1\}$ for the chosen a_t which indicates that no information about the unchosen classes' losses is revealed. This change in information structure is crucial: it turns supervised online classification into a bias bandit learning, demanding exploration and statistical corrections to debias. Intuitively, full feedback lets us learn "how wrong" each class was; bandit feedback only tells us whether the single choice was correct. In many real applications, full ground-truth labels are scarce or delayed; bandit-style outcomes (click/no-click, success/failure) arrive immediately and are action-dependent, making the bandit formalism appropriate.

Logistic Bandits. A natural probabilistic model for multi-class classification is the multinomial logistic (softmax) link. Thus logistic bandit algorithms (OI19; AT21; LO24; LK24) are good candidate solutions. However, current logistic bandit and general linear bandit algorithms are not design for graph data. In addition, the online classification with the partial feedback is never investigated in bandit. From the graph learning viewpoint, while graph classification under *full* supervision is well-developed, its *online, multi-class, bandit-feedback* counterpart remains comparatively under-explored. Lastly, regarding the research perspective on online multi-class classification, the structure type of data such as graph is never studied. We will propose a GNN based bandit algorithm to solve the online graph classification with bandit feedback in Section 7.1.1.

3.3 Other Topics

Bandits with Graph-Valued Rewards. One interesting but under-explored topic is graph as the reward feedback in the online decision making problem. Consider the classical bandit problem, for example, multi-armed bandit, when the agent make a decision, the environment will return a graph as a feedback, not a scalar reward. In this setting, one need to identify the distance between graphs. With the distance measurement metric, the regret becomes the distance between the selected graph to the optimal graph. Here, the optimal graph is pre-defined. For example, in a online recommendation system for shoppers, a customer's interaction in his shopping journey can be represented as a graph. The agent can define the ideal shopping journey graph as the optimal arm and the agent have to recommend items to the customer. Then the problem is the bandit problem. In addition, consider the interactive settings, annotators or users may provide *comparisons* between graphs rather than absolute labels (e.g., "G is better than G'"). This leads to the bandits rewarding on the space of graphs, where the goal is to identify a near-optimal graph.

Subgraph Selection Bandit. We identify a type of bandit problem that the agent selects the subgraph of the given graph. Note that the graph is not necessarily the structure over arms, which is the reason we mark it as a topic under graph as data in bandit. We list two examples in this type of bandit as follow. The influence maximization bandits (WLW19; ICM22; FTC24) on graphs problem is selecting/activating a set of nodes on graph that maximizing the influence/reward. The bandit problem for traveling on graph is a problem to selecting the routes/shortest-path on the graph (ZJL23). For example, on a transportation or communication graph, each action is a feasible path from a source to a destination.

CHAPTER 4

Graph Neural Thompson Sampling

We consider an online decision-making problem with a reward function defined over graph-structured data. We formally formulate the problem as an instance of graph action bandit. We then propose `GNN-TS`, a Graph Neural Network (GNN) powered Thompson Sampling (TS) algorithm which employs a GNN approximator for estimating the mean reward function and the graph neural tangent features for uncertainty estimation. We prove that, under certain boundness assumptions on the reward function, GNN-TS achieves a state-of-the-art regret bound which is (1) sub-linear of order $\tilde{\mathcal{O}}((\tilde{d}T)^{1/2})$ in the number of interaction rounds, T, and a notion of effective dimension $\tilde{d}$, and (2) independent of the number of graph nodes. Empirical results validate that our proposed `GNN-TS` exhibits competitive performance and scales well on graph action bandit problems.

4.1 Introduction

Thompson Sampling (Tho33) is a widely adopted and effective technique in sequential decision-making problems, known for its ease of implementation and practical success (CL11; KBK15; RVK18; RTS18). The fundamental concept behind Thompson Sampling (TS) is to compute the posterior probability of each action being optimal for the present context, followed by the selection of an action from this distribution. Previous research has extended TS or developed variants of it to incorporate increasingly complex models of the reward function, such as Linear TS (AG13; AL17), Kernelized TS (CG17), and Neural TS (ZZL20). However, these efforts have mainly focused on conventional data types. In contrast, the

application of sequential learning to graph-structured data, such as molecular or biological graph representations, introduces unique challenges that merit further investigation.

Recently, there has been a growing interest in studying bandit optimization over graphs. Several researchers have initiated this line of work by addressing the challenge of encoding graph structures in bandit problems (GWD18; JBJ18; GH20; KXK20). More recently, Graph Neural Network (GNN) bandits have been proposed, which leverage expressive GNNs to approximate reward functions on graphs (KKB22). Despite these advancements, the GNN bandits remain relatively unexplored compared to the extensive research on Neural bandits. Firstly, a formal formulation of this sequential graph selection problem is yet to be proposed. More importantly, there is a significant lack of comprehensive theoretical and empirical investigations regarding the use of TS in sequential graph selection.

Contribution. In this work, we address the online decision-making problem over graph-structured data by contributing a novel algorithm called `GNN-TS`. We begin by formulating the sequential graph selection as graph action bandit. We then propose Graph Neural Thompson Sampling, `GNN-TS`, to incorporate TS exploration with graph neural networks. We establish a regret bound for the proposed algorithm with sub-linear growth of order $\tilde{\mathcal{O}}((\tilde{d}T)^{1/2})$ with respect to the effective dimension $\tilde{d}$ and the number of interaction round T, and independent of the number of graph nodes. Finally, we corroborate the analysis with an empirical evaluation of the algorithm in simulations. Experiments show that `GNN-TS` yields competitive performance and scalability, compared to the state-of-the-art baselines, underscoring its practical value in addition to its strong theoretical guarantees.

Notations. Let $[n] = \{1, 2, ..., n\}$. For a set or event $\mathcal{E}$, we denote its complement as $\bar{\mathcal{E}}$. $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ is the identity matrix. For a matrix $\boldsymbol{A}$, $\boldsymbol{A}_{i*}$ and $\boldsymbol{A}_{*j}$ denote its i-th row and j-th column, respectively. $\lambda_{\max}(\boldsymbol{A})$ and $\lambda_{\min}(\boldsymbol{A})$ represents the maximum and minimum eigenvalues of the matrix $\boldsymbol{A}$. For any vector $\boldsymbol{x}$ and square matrix $\boldsymbol{A}$, $\|\boldsymbol{x}\|_{\boldsymbol{A}} = \sqrt{\boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x}}$. We denote the history of randomness up to (but not including) round t as $\mathcal{F}_t$ and write $\mathbb{P}_t(\cdot) := \mathbb{P}(\cdot \mid \mathcal{F}_t)$ and $\mathbb{E}_t(\cdot) := \mathbb{E}[\cdot \mid \mathcal{F}_t]$ for the conditional probability and expectation given

$\mathcal{F}_t$. We use $\lesssim$ and big-O, to denote "less than", up to a constant factor. We further use $\tilde{\mathcal{O}}(\cdot)$ for big-O up to logarithmic factor.

4.2 Related Works

Graph Bandit. Multiple works have studied graph bandit problems, which can be classified into two categories: graph as structure across arms and graph as data. Most research focuses on the former category, starting from spectral bandit (KVM14; KMK20) to graphical bandit (LZS18; YKW20; GYZ23; TF23). Within this field, bandit problems with graph feedback have garnered significant attention (TDD17; DMM20; CLZ21; KZL22), where learners observe rewards from selected nodes and their neighborhoods. The primary focus of these works have been improving sample efficiency (BDD19; WKK20; IMB22), with some assuming that payoffs are shared according to the graph Laplacian (EFH22; LLZ20; LTW20; TMR22; YTD20). While the existing literature primarily aims to optimize over geometrical signal domains, our work focuses on optimization within graph domains. Specifically, we investigate the online graph selection problem, aligning with the second category of research that considers the entire graph as input data. A related recent work (KKB22) proposed a GNN bandit approach with regret bound based on information gain and an elimination-based algorithm. In contrast, our work explores regret bound based on the effective dimension and builds upon the foundation of Thompson Sampling. This second category of research also encompasses empirical works (UYA20; QBH22; QBH23), particularly those centered around molecule optimization (WSK23b; WSK23a).

Neural Bandit. Our work contributes to the research on neural bandits, where deep neural networks are utilized to estimate the reward function. The work of (ZM19; XWZ20) investigated the Neural Linear bandit, while (ZLG20) developed Neural Upper Confidence Bound (UCB), an extension of Linear UCB. (ZZL20) adapted TS with deep neural networks, proposing Neural TS. (DSL22) makes improvements to neural bandit algorithms to overcome

practical limitations. (NGN21) explores neural bandit in an offline contextual bandit setting and (GKK24) examines batched learning for neural bandit. Our work can be seen as an extension of Neural TS (ZZL20), incorporating significant improvements such as the utilization of graph neural tangent kernel and a distinct definition of effective dimension.

4.3 Problem Formulation and Methodology

4.3.1 Graph Action Bandit Problem

We consider an online decision-making problem in which the learner aims to optimize an unknown reward function by sequentially interacting with a stochastic environment. We identify the actions with graphs from an action space $\mathcal{G}$ and assume that the size of this action space, denoted as $|\mathcal{G}|$, is finite. At time $t \in [T]$, the learner selects a graph G_t from the action space $\mathcal{G}_t \subset \mathcal{G}$. The learner then observes a noisy reward $y_t = \mu(G_t) + \epsilon_t$ where $\mu : \mathcal{G} \to \mathbb{R}$ is the true (unknown) reward function and $\{\epsilon_t\}_{t\in[T]}$ are i.i.d zero-mean sub-gaussian noise with variance proxy σ_ϵ^2. The goal of the learner is to maximize the expected cumulative reward in T rounds, which equivalently entails minimizing the expected (pseudo-)regret denoted as $R_T = \sum_{t=1}^{T} \mathbb{E}[\mu(G_t^*) - \mu(G_t)]$ where $G_t^* = \text{argmax}_{G\in\mathcal{G}_t}\, \mu(G)$ represents the optimal graph at time t.

The graph space $\mathcal{G}$ is a finite set of undirected graphs with at most N nodes. Note that the graphs with less than N nodes can be treated by adding auxiliary isolated nodes with no features. We denote an undirected attributed graph with N nodes as $G = (\mathbf{X}, \mathbf{A})$, where $\mathbf{X} \in \mathbb{R}^{N\times d}$ represents the feature matrix with d features, and $\mathbf{A} \in \{0,1\}^{N\times N}$ is the unweighted adjacency matrix. The rows of $\mathbf{X}$ correspond to node features. The size of the node set of a graph G is denoted as $|\mathcal{V}(G)| \leq N$.

Graph action bandit has several applications such as chemical molecules optimization. Consider the graph structures representing the molecules (Wei88) and rewards are molecular properties. The goal is to sequentially recommend the optimal molecules for experimental

testing.

4.3.2 Graph Neural Network Model

We propose to learn the unknown reward function $\mu(\cdot)$ by fitting a Graph Neural Network (GNN). We consider a relatively simple GNN architecture where the output of a single graph convolution layer is normalized (to unit ℓ_2 norm) and passed through a multilayer perceptron (MLP). A single-layer graph convolution can be compactly stated as $\boldsymbol{AX}$ using the adjacency matrix $\boldsymbol{A}$ of the network. Additionally, we normalize each row of the resulting matrix to have a unit ℓ_2 norm. Letting $u(\boldsymbol{x}) = \boldsymbol{x}/\|\boldsymbol{x}\|_2$ denote the normalization operator, the aggregated feature of node i in a graph G is $\boldsymbol{h}_i^G = u((\boldsymbol{AX})_{i*}) = u(\sum_{j\in\mathcal{N}_i} \boldsymbol{X}_{j*})$ where $\mathcal{N}_j$ is the neighborhood of node j. Our GNN also consists of an L-layer m-width MLP neural network f_{MLP} which is defined recursively as follows

$$\begin{aligned} f^{(1)}(\boldsymbol{h}_i^G) &= \boldsymbol{W}^{(1)}\boldsymbol{h}_i^G, \quad i \in [N], \\ f^{(l)}(\boldsymbol{h}_i^G) &= \frac{1}{\sqrt{m}}\boldsymbol{W}^{(l)}\text{ReLU}(f^{(l-1)}(\boldsymbol{h}_i^G)), \quad 2 \le l \le L, \\ f_{\text{MLP}}(\boldsymbol{h}_i^G;\boldsymbol{\theta}) &= f^{(L)}(\boldsymbol{h}_i^G). \end{aligned} \tag{4.1}$$

Here, $\text{ReLU}(\cdot) = \max(\cdot, 0)$, $\boldsymbol{W}^{(1)} \in \mathbb{R}^{m\times d}$, $\boldsymbol{W}^{(L)} \in \mathbb{R}^{1\times m}$, $\boldsymbol{W}^{(l)} \in \mathbb{R}^{m\times m}$ for any $1 < l < L$ are weight matrices of the MLP and $\boldsymbol{\theta} := (\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(L)}) \in \mathbb{R}^p$ is the collection of parameters of the neural network where $p = dm + (L-2)m^2 + m$. Our GNN model to estimate the reward function is

$$f_{\text{GNN}}(G;\boldsymbol{\theta}) := \frac{1}{N}\sum_{i=1}^{N} f_{\text{MLP}}(\boldsymbol{h}_i^G;\boldsymbol{\theta}). \tag{4.2}$$

The gradient of $\boldsymbol{\theta} \mapsto f_{\text{GNN}}(G;\boldsymbol{\theta})$ denoted as $g(G;\boldsymbol{\theta}) := \nabla_{\boldsymbol{\theta}} f_{\text{GNN}}(G;\boldsymbol{\theta})$ will play a key role in uncertainty quantification, which will be discussed in Section 4.3.3. The GNN model (4.2)

is trained by minimizing the mean-squared loss with ℓ_2 penalty, described concretely in (4.6). A hyperparameter λ is used to tune the strength of ℓ_2 regularization. For the simplicity of exposition, in the theoretical analysis, we solve the optimization via gradient descent with learning rate η, total number of iterations J and initialize parameters $\boldsymbol{\theta}_0$ such that $f_{\text{GNN}}(G; \boldsymbol{\theta}_0) = 0$ for all $G \in \mathcal{G}$, which can be fulfilled based on the work of (ZLG20; KK22).

4.3.3 Graph Neural Thompson Sampling

We adapt Thompson Sampling (TS) for graph exploration, due to its robust performance in balancing exploration against exploitation. Algorithm 1 outlines our proposed GNN Thompson sampling, following the idea of `NeuralTS` in (ZZL20). The key step is the sampling of an estimated reward mean $\widehat{r}_t(G)$ for each graph G in the action space at time t, from a normal distribution as in equation (4.4). The mean of the normal distribution in (4.4) is our current estimate, $f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1})$, of the true mean reward for graph G (i.e., $\mu(G)$). This estimate is obtained by fitting the GNN to all the past data as in (4.6). The variance of the normal distribution $\nu^2\sigma_t^2(G)$ is our current measure of uncertainty about the true reward of graph G. Note that

$$\sigma_t^2(G) = \frac{1}{m}\|g(G; \boldsymbol{\theta}_{t-1})\|^2_{\boldsymbol{U}_{t-1}^{-1}} \quad \text{where} \quad \boldsymbol{U}_{t-1} = \lambda \boldsymbol{I}_p + \frac{1}{m}\sum_{i=1}^{t-1} g(G_i; \boldsymbol{\theta}_{i-1}) g(G_i; \boldsymbol{\theta}_{i-1})^\top. \quad (4.3)$$

The rationale behind $\sigma_t^2(G)$ comes from a linear approximation to $f_{\text{GNN}}(G; \boldsymbol{\theta})$. In particular, the idea is that (4.6) approximately looks like a linear ridge regression problem, with features $\{g(G_i; \boldsymbol{\theta}_i)/\sqrt{m}\}_{i \in [t]}$. The expression (4.3) is then the familiar estimated covariance matrix from linear bandits after we make this identification. This approximation can be made rigorous via the neural tangent kernel idea, as discussed in Section 4.4.

The sampled reward mean $\widehat{r}_t(G)$ is the index for decision-making. The learner selects the graph with the highest index, i.e., $G_t = \text{argmax}_{G \in \mathcal{G}} \widehat{r}_t(G)$. The randomness in $\widehat{r}_t(G)$, due to the positive variance of the sampling distribution, is what allows TS to efficiently explore

Algorithm 1 Graph Neural Thompson Sampling (`GNN-TS`)

Input: T, λ, ν

Initialization: $\boldsymbol{\theta}_0$, $\boldsymbol{U}_0 = \lambda \boldsymbol{I}_p$.

for $t = 1, ..., T$ **do**

Compute $\sigma_t^2(G) := \frac{1}{m} \| g(G; \boldsymbol{\theta}_{t-1}) \|^2_{\boldsymbol{U}_{t-1}^{-1}}$ and sample

$$\widehat{r}_t(G) \sim \mathcal{N}\big(f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1}), \nu^2 \sigma_t^2(G)\big), \quad \text{for all } G \in \mathcal{G}_t. \tag{4.4}$$

Select graph $G_t = \text{argmax}_{G \in \mathcal{G}_t} \widehat{r}_t(G)$, and collect reward $y_t := \mu(G_t) + \epsilon_t$.

Update uncertainty estimate as

$$\boldsymbol{U}_t = \boldsymbol{U}_{t-1} + g(G_t; \boldsymbol{\theta}_{t-1}) g(G_t; \boldsymbol{\theta}_{t-1})^\top / m. \tag{4.5}$$

Update the parameter estimate as

$$\boldsymbol{\theta}_t = \underset{\boldsymbol{\theta}}{\text{argmin}} \frac{1}{2t} \sum_{i=1}^{t} \big(f_{\text{GNN}}(G_i; \boldsymbol{\theta}) - y_i\big)^2 + \frac{m\lambda}{2} \|\boldsymbol{\theta}\|_2^2. \tag{4.6}$$

the action space. We want the uncertainty, as captured by $\sigma_t^2(G)$ not to be too small early on, to allow for effective exploration, but not too large either to miss out on the optimal choice too often. Lemma 4.5.2 in Section 4.5 captures the two sides of this trade-off in our theory.

It is worth noting that our proposed Algorithm 1 is not exact TS. In our approach, (4.4) serves as an *approximation to* a posterior for mean reward function, rather than a true posterior. The difference between our proposed method and an exact Bayesian method will be smaller if the GNN model is better approximated by a linear model.

Lastly, we note that $\widehat{r}_t(G)$ is also referred to as the perturbed mean reward, as it can be expressed as: $\widehat{r}_t(G) = f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1}) + \nu \sigma_t(G) z$ where $z \sim \mathcal{N}(0, 1)$. This perturbed reward includes both the estimated part ($f_{\text{gnn}}(G; \boldsymbol{\theta}_{t-1})$) and the random perturbation part ($\nu \sigma_t(G) \cdot z$). The use of perturbations for exploration has been shown to be a strong strategy in previous works (KT19; KSG19b). Algorithm 1 can be summarized as greedily selecting the graph with the highest *perturbed* mean reward.

4.4 Regret Bound for GNN-TS

Graph Neural Tangent Kernel. Let us briefly review the idea of graph neural tangent kernel (GNTK) (KKB22) which is based on the neural tangent kernel (NTK) of (JGH18). The tangent kernel on graph space $\mathcal{G}$, induced by initialization $\boldsymbol{\theta}_0$, is defined as the inner product of the gradient at initialization, i.e $\tilde{k}(G, G') := g(G; \boldsymbol{\theta}_0)^\top g(G'; \boldsymbol{\theta}_0)$ for any $G, G' \in \mathcal{G}$. The GNTK is the limiting kernel of $\tilde{k}(G, G')/m$. We define the finite-width (empirical) and infinite-width GNTK as

$$\hat{k}(G, G') := \frac{1}{m}\langle g(G; \boldsymbol{\theta}_0), g(G'; \boldsymbol{\theta}_0)\rangle, \quad k(G, G') := \lim_{m\to\infty} \frac{1}{m}\langle g(G; \boldsymbol{\theta}_0), g(G'; \boldsymbol{\theta}_0)\rangle. \tag{4.7}$$

We assume the reward function falls within the RKHS corresponding to the GNTK k defined in (4.7). Define $\boldsymbol{K} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ as the GNTK matrix with entries $k(G, G')$ for all $G, G' \in \mathcal{G}$ and $\boldsymbol{\mu} = (\mu(G))_{G\in\mathcal{G}} \in \mathbb{R}^{|\mathcal{G}|}$ as the reward function vector. The kernel matrix $\boldsymbol{K}$ is positive definite with maximum eigenvalue $\rho_{\max} := \lambda_{max}(\boldsymbol{K})$ and minimum eigenvalue $\rho_{min} := \lambda_{min}(\boldsymbol{K})$. We also define the finite-width GNTK matrix $\hat{\boldsymbol{K}} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ with entries $\hat{k}(G, G')$ for all $G, G' \in \mathcal{G}$ and maximum eigenvalues $\hat{\rho}_{\max} = \lambda_{\max}(\hat{\boldsymbol{K}})$. Note that $\hat{\boldsymbol{K}} \to \boldsymbol{K}$ as $m \to \infty$.

Effective Dimension. We define the effective dimension $\tilde{d}$ of the GNTK matrix $\boldsymbol{K}$ with regularization λ as

$$\tilde{d} := \frac{\log\det(\boldsymbol{I}_{|\mathcal{G}|} + T\boldsymbol{K}/\lambda)}{\log(1 + T\rho_{\max}/\lambda)}. \tag{4.8}$$

This quantity, which appears in our regret bound, measures the actual underlying dimension of the reward function space as seen by the bandit problem (VKM13; BM19). Our definition is adapted from (YW20). The key difference is that our $\tilde{d}$ does not directly depend on $|\mathcal{G}|$, which is replaced by $\rho_{\max}$, compared to the definition in (ZZL20). Our definition is the ratio of the sum over the maximum of the sequence of log-eigenvalues of matrix $\boldsymbol{I}_{|\mathcal{G}|} + T\boldsymbol{K}/\lambda$.

As such, it is a robust measure of matrix rank. In particular, we always have $\tilde{d} \leq |\mathcal{G}|$. Moreover, previous work on GNN bandit (KKB22) utilized the notion of information gain which we replace with the related, but different, notion of effective dimension $\tilde{d}$.

We will make the following assumptions:

Assumption 1 (Bounded RKHS norm for Reward)**.** *The reward function μ has R-bounded RKHS norm with respect to a positive definite kernel k: $\|\mu\|_k = \sqrt{\boldsymbol{\mu}^\top \boldsymbol{K}^{-1}\boldsymbol{\mu}} \leq R$.*

Assumption 2 (Bounded Reward Differences)**.** *Reward differences between any graph in action space are bounded. Formally, $\forall G, G' \in \mathcal{G}$: $|\mu(G) - \mu(G')| \leq B$, for some $B \geq 1$.*

Assumption 3 (Subgaussian Noise)**.** *Noise process $\{\epsilon_t\}_{t\in[T]}$ satisfies $\mathbb{E}_{t-1}[e^{\eta\epsilon_t}] \leq e^{\sigma_\epsilon^2\eta^2/2}, \forall \eta > 0$.*

Assumption 1 aligns with the regularity assumption commonly found in the kernelized and neural bandit literature (SKK09; CG17; KK22). Assumption 2 implies that instantaneous regret is bounded: $|\mu(G_t^*) - \mu(G_t)| \leq B$ for all $t \in [T]$ and Assumption 3 is the conditional subgaussian assumption for stochastic process $\{\epsilon_t\}_{t\in[T]}$.

We are now ready to state our main result. Recall that N is the maximum number of (graph) nodes and L the depth of MLP and m its width.

Theorem 4.4.1. *Suppose Assumption 1,2 and 3 hold. For a fixed horizon $T \in [T]$, let*

$$m \geq \text{poly}\big(T, L, |\mathcal{G}|, \lambda^{-1}, R, \sigma_\epsilon, \rho_{\min}^{-1}, \log(TLN|\mathcal{G}|)\big)$$
$$\nu \gtrsim 1 + \sigma_\epsilon\sqrt{\tilde{d}\log T} + \sqrt{\lambda}R, \quad \lambda \gtrsim (\sigma_\epsilon^2 + R^2)^3 + \rho_{\max}$$

and learning rate $\eta \leq (\tilde{C}mL + m\lambda)^{-1}$, for some constant $\tilde{C}$. Then, the regret of Algorithm 1 is bounded as

$$R_T \leq C\,B\sqrt{\tilde{d}\,T\log(T|\mathcal{G}|)\cdot\log(2 + T\rho_{\max}/\lambda)}$$

for some universal constant $C > 0$.

The order of regret upper bound in Theorem 4.4.1, $\tilde{\mathcal{O}}((\tilde{d}T)^{1/2})$ matches the state-of-the-art regret bounds in the literature of Thompson Sampling (AG13; CG17; KZS20; ZZL20). As in (KKB22), our regret bound is independent of N, indicating that `GNN-TS` is valid for large graphs. Moreover, for low complexity reward functions of effective dimension $\tilde{d} = O(1)$, the regret scales as $\sqrt{\log |\mathcal{G}|}$ in the size of the action space, showing the robust scalability of `GNN-TS`.

4.5 Proof of the Regret Bound

Similar to the previous literature, the key is to to obtain probabilistic control on the 'discrepancy' of the policy in `GNN-TS` Consider the following events

$$\mathcal{E}_t^{\mu} := \left\{ \left| f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1}) - \mu(G) \right| \ \leq \ c_t^{\mu}(G), \ \text{for all } G \in \mathcal{G}_t \right\} \tag{4.9}$$

$$\mathcal{E}_t^{\sigma} := \left\{ \left| \widehat{r}_t(G) - f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1}) \right| \ \leq \ c_t^{\sigma}(G), \ \text{for all } G \in \mathcal{G}_t \right\} \tag{4.10}$$

$$\mathcal{E}_t^{a} := \left\{ \widehat{r}_t(G_t^*) - f_{\mathrm{GNN}}(G_t^*; \boldsymbol{\theta}_{t-1}) \ > \ \nu \sigma_t(G_t^*) \right\} \tag{4.11}$$

where $c_t^{\mu}(G) := \nu\sigma_t(G) + \epsilon(t,m)$ and $c_t^{\sigma}(G) := \nu\sigma_t(G)\sqrt{2\log(t^2|\mathcal{G}_t|)}$ as well as $\epsilon(t,m) = (C_0 \nu L^{9/2})\, m^{-1/6}\sqrt{\log m} \cdot t$ and C_0 is some universal constant. Events $\mathcal{E}_t^{\mu}$ and $\mathcal{E}_t^{\sigma}$ control the discrepancies with constants $c_t^{\mu}(G)$ and $c_t^{\sigma}(G)$ respectively: $c_t^{\mu}(G)$ is bounding the estimation discrepancy while $c_t^{\sigma}(G)$ is bounding the exploration discrepancy. Note that event $\mathcal{E}_t^{a}$ is only for G_t^*, the optimal graph at round t.

4.5.1 Estimation Bound ($\mathcal{E}_t^{\mu}$)

The following lemma ensures that event $\mathcal{E}_t^{\mu}$ happens with high probability.

Lemma 4.5.1. *Fix* $\delta \in (0,1)$. *For* $m \geq \mathrm{poly}(R, \sigma_\epsilon, L, |\mathcal{G}|, \lambda^{-1}, \rho_{\min}^{-1}, \log(TLN|\mathcal{G}|/\delta))$ *and*

(ν, λ, η) satisfying conditions of Theorem 4.4.1, we have $\mathbb{P}(\mathcal{E}_t^{\mu}) \geq 1 - \delta/T$.

In other words, given a large enough width of the GNN (m) and a small enough learning rate (η), there is a high probability upper bound for the estimation error $|f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1}) - \mu(G)|$. This Lemma 4.5.1 also gives an approximate upper confidence bound similar to `GNN-UCB` in (KKB22): $\mu(G) \leq f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1}) + \nu\sigma_t(G) + \epsilon(t, m)$. Since $\epsilon(t, m)$ is negligible for large m, the approximate upper confidence bound, $f_{\text{GNN}}(G; \boldsymbol{\theta}_{t-1}) + \nu\sigma_t(G)$ is used as the index for `GNN-UCB`. Note that this lemma controls the estimation error produced by GNNs, hence applicable to all GNN bandit algorithms using model (4.2). Our $c_t^{\mu}(G) = \nu\sigma_t(G) + \varepsilon(t, m)$ is similar in form to that of (ZZL20) which is different from the earlier analysis of TS in (AG13).

4.5.2 Exploration Bound $(\mathcal{E}_t^{\sigma}, \mathcal{E}_t^{a})$

We also need event $\mathcal{E}_t^{\sigma}$ to quantify the level of exploration achieved by the algorithm. Intuitively, $\mathcal{E}_t^{\sigma}$ ensures our exploration is moderate. On the other hand, indicated by the regret analysis in (KSG19a), instead of controlling the exploration independently, the relation between two sources of explorations needs to be considered because this relation is critical for finding the optimal action. To meet such observation, we define an extra "good" event for anti-concentration on the optimal actions, which is $\mathcal{E}_t^{a}$. Under event $\mathcal{E}_t^{a}$, the policy index $\widehat{r}_t(G_t^*)$ of the optimal graph has the higher future positive exploration, which guides the learner to have higher chance to pick the optimal graph. A formal lemma for exploration discrepancy using TS is given as below:

Lemma 4.5.2. *For `GNN-TS`, for all $t \in [T]$, we have $\mathbb{P}_t(\bar{\mathcal{E}}_t^{\sigma}) \leq t^{-2}$ and $\mathbb{P}(\mathcal{E}_t^{a}) \geq (4e\sqrt{\pi})^{-1}$.*

Lemma 4.5.2 shows that `GNN-TS` has a positive probability of moderate exploration of the optimal arm, which is beneficial to regret reduction.

4.5.3 Proof of Theorem 4.4.1

Let $\Delta_t := \mu(G_t^*) - \mu(G_t)$ be the instantaneous regret. We will need two additional lemmas:

Lemma 4.5.3 (One Step Regret Bound)**.** *Assume the same as Theorem 4.4.1. Suppose* $\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma) > 0$. *Then for any* $t \in [T]$, *almost surely,*

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\mu}] \le \mathbb{I}_{\mathcal{E}_t^\mu} \cdot \left\{ \left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)} + 1 \right) \mathbb{E}_t[\gamma_t(G_t)] - \epsilon(t,m) + B \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma) \right\} \tag{4.12}$$

where $\gamma_t(G) = c_t^\mu(G) + c_t^\sigma(G)$.

Lemma 4.5.4 (Cumulative Uncertainty Bound)**.** *Assume the same as Theorem 4.4.1. Then with probability at least* $1 - \delta/T$,

$$\frac{1}{2} \sum_{t=1}^{T} \min\{1, \sigma_t^2(G_t)\} \le \tilde{d} \log(1 + \lambda^{-1} T \rho_{\max}) + 3 C_\psi |\mathcal{G}|^{3/2} \sqrt{T} \lambda^{-1/2} \epsilon_m \tag{4.13}$$

where $\epsilon_m = o(1)$ *as* $m \to \infty$ *and* C_ψ *is some constant. We always have* $\tilde{d} \le |\mathcal{G}|$.

Main Proof. The expected cumulative regret is

$$R_T = \sum_{t=1}^{T} \mathbb{E}[\Delta_t] = \sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\mathcal{E}_t^\mu}] + \sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\bar{\mathcal{E}}_t^\mu}]. \tag{4.14}$$

By Lemma 4.5.1, letting $\mathbb{P}(\bar{\mathcal{E}}_t^\mu) \le \delta/T$ and $\Delta_t \le B$, we have the upper bound for the second term

$$\sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\bar{\mathcal{E}}_t^\mu}] \le BT(\delta/T) = B\delta. \tag{4.15}$$

Now our focus is controlling the first summation term. By Lemma 4.5.3, almost surely, we have

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\mu}] \le \mathbb{I}_{\mathcal{E}_t^\mu} \cdot \left\{ \left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)} + 1 \right) \mathbb{E}_t[\gamma_t(G_t)] - \epsilon(t,m) + B \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma) \right\} \tag{4.16}$$

where $\gamma_t(G) = c_t^{\mu}(G) + c_t^{\sigma}(G)$. Assuming that $t \geq 5$, we have $t^2 \geq 5e\sqrt{\pi}$. By Lemma 4.5.2, $\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}(\bar{\mathcal{E}}_t^{\sigma}) \geq \frac{1}{4e\sqrt{\pi}} - \frac{1}{t^2} \geq \frac{1}{20e\sqrt{\pi}}$. Then, for $t \geq 5$, dropping $\epsilon(t,m)$ from the bound,

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\mu}}] \leq 194\mathbb{E}_t[\gamma_t(G_t)] + Bt^{-2} \leq \big(194\mathbb{E}_t[\min\{1, \gamma_t(G_t)\}] + t^{-2}\big)B \tag{4.17}$$

using $40e\sqrt{\pi} + 1 \leq 194$, $\Delta_t \leq B$ and $B \geq 1$. Therefore, we have

$$\sum_{t=1}^{T} \mathbb{E}[\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\mu}}]] \leq 194B \sum_{t=5}^{T} \mathbb{E}[\mathbb{E}_t[\min\{1, \gamma_t(G_t)\}]] + 4B + B(\pi^2/6) \tag{4.18}$$

using $\sum_{t=1}^{\infty} t^{-2} = \pi^2/6$. Note that $\gamma_t(G_t) \leq \sigma_t(G_t)\sqrt{8\log(T^2|\mathcal{G}|)} + \epsilon(T,m)$ for all $t \in [T]$. Then by Cauchy-Schwarz inequality,

$$\sum_{t=5}^{T} \min\{1, \gamma_t(G_t)\} \leq \sqrt{8T\log(T^2|\mathcal{G}|)}\Big(\sum_{t=5}^{T} \min\{1, \sigma_t^2(G_t)\}\Big)^{1/2} + T\epsilon(T,m). \tag{4.19}$$

By Lemma 4.5.4 and take m sufficiently large such that $3C_{\psi}|\mathcal{G}|^{3/2}\sqrt{T}\lambda^{-1/2}\epsilon_m \leq \tilde{d}\log(1 + \lambda^{-1}T\rho_{\max})$, we have

$$\sum_{t=1}^{T} \mathbb{E}[\min\{1, \sigma_t^2(G_t)\}] \leq 4\tilde{d}\log(1 + T\rho_{\max}/\lambda) + T(\delta/T). \tag{4.20}$$

Recall that the $\epsilon(T,m) = C_1\, T\, m^{-1/6}\sqrt{\log m}$. Take m large enough we have $T\epsilon(T,m) \leq \sqrt{T}$. Then put the above results back into (4.18), we have:

$$\sum_{t=1}^{T} \mathbb{E}[\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\mu}}]] \leq 194B\big(\sqrt{16T\log(T|\mathcal{G}|)} \cdot \sqrt{4\tilde{d}\log(1 + T\rho_{\max}/\lambda) + \delta} + \sqrt{T}\big) + 4B + B(\pi^2/6) \tag{4.21}$$

by using $\log(T^2|\mathcal{G}|) \leq 2\log(T|\mathcal{G}|)$. Therefore, we have our regret bound:

$$R_T \leq CB\sqrt{\tilde{d}T\log(T|\mathcal{G}|) \cdot \big(1 + \log(1 + T\rho_{\max}/\lambda)\big)} \tag{4.22}$$

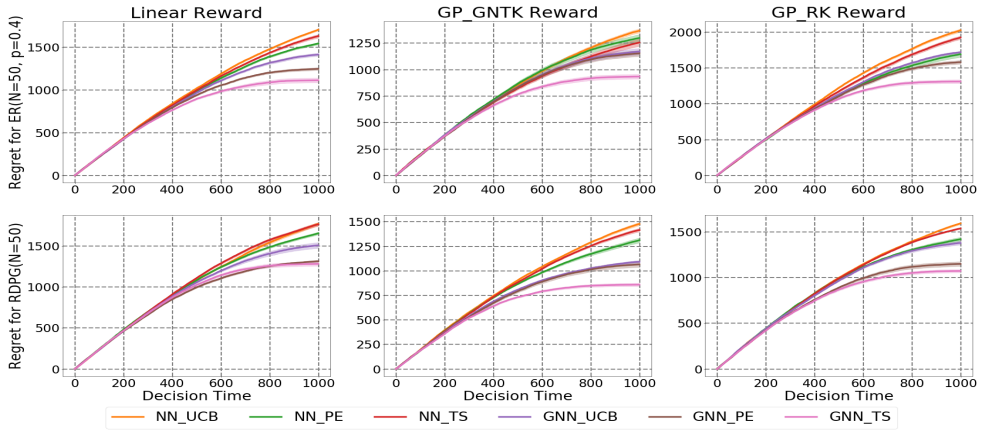

Figure 4.1: Regret over horizon $T = 1000$ for Erdös–Rényi random graphs with $p = 0.4$ and $N = 50$ in the first row and random dot product graphs with $N = 50$. Three columns are three types of reward function generation: linear model, Gaussian process with GNTK, Gaussian process with representation kernel. `GNN-TS` is competitive and robust to different environment settings.

for some universal constant C. We have used $\tilde{d} \geq 1$ and $B \geq 1$, to simplify the bound. Finally, note that $1 + \log(1 + x) \leq 2\log(2 + x)$ for all $x \geq 0$. □

4.6 Experiments

We create synthetic graph data and generate the rewards through three different mechanisms. For the graph structures, we use random graph models including Erdös–Rényi and random dot product graph models. The features are generated i.i.d. from the $\mathcal{N}(0, 1)$. The noisy reward is assumed to have $\sigma_\epsilon = 0.01$. Our experiments investigate `GNN-UCB`, `GNN-PE`, `NN-UCB`, `NN-PE`, and `NN-TS` as baselines from (KKB22). All performance curves in our empirical studies show an average of over 10 repetitions with a standard deviation of the corresponding bandit algorithm with horizon $T = 1000$. We assume the graph domain is fully observable, $\mathcal{G}_t = \mathcal{G}$ for all $t \in [T]$. Below is a brief overview of the simulation elements. For more details, see Appendix 6.6.4.

Random Graph. We use two types of random graphs including Erdös–Rényi (ER) random graphs and random dot product graphs (RDPG). ER graphs are generated with edge probability p and number of nodes N. RDPGs are generated by modeling the expected edge probabilities as the function of the inner product of features. In the first row of Figure 4.1, the graphs in $\mathcal{G}$ are from the ER model with $p = 0.4$ and in the second row from an RDPG, both of size $N = 50$.

Reward Function. To generate the rewards, we use models of three different types: linear model, Gaussian Process (GP) with GNTK, Gaussian process with the representation kernel. For the linear model, we have $\mu(G) = \langle \boldsymbol{\theta}^*, \bar{\boldsymbol{h}}^G \rangle$ with true parameter $\boldsymbol{\theta}^* \sim \mathcal{N}(0, \boldsymbol{I}_d)$ and $\bar{\boldsymbol{h}}^G = \sum_{i=1}^N \boldsymbol{h}_i^G / N$. For the GP with GNTK, we fit a GP regression model with empirical GNTK matrix $\hat{\boldsymbol{K}} \in \mathbb{R}^{|\mathcal{G}| \times |\mathcal{G}|}$ as the covariance matrix of the prior, trained on $\{(G, y_G)\}_{G \in \mathcal{G}}$ where $\{y_G\}_{G \in \mathcal{G}}$ are i.i.d. from $\mathcal{N}(0, 1)$. For the GP with the representation kernel, we trained a GNN for a graph property prediction task and used the mean pooling over all the nodes of the last layer representations as the graph representation, denoted as $\bar{\boldsymbol{h}}_{\text{rep}}^G$. We then define the representation kernel as $k_{\text{rep}}(G, G') := \langle \bar{\boldsymbol{h}}_{\text{rep}}^G, \bar{\boldsymbol{h}}_{\text{rep}}^{G'} \rangle$ and draw $\mu(\cdot)$ from a zero-mean GP with this covariance function (over $\mathcal{G}$).

Algorithms. We investigate two baselines `GNN-UCB` and `GNN-PE` along with our proposed `GNN-TS`. `GNN-PE` is the proposed state-of-the-art algorithm that selects the graph with the highest uncertainty and eliminates the graph candidates by the upper confidence bounds. All the algorithms in our work use the loss function (4.6) which is different from previous work. All gradients used for our experiments are $g(G; \boldsymbol{\theta}_t)$, not $g(G; \boldsymbol{\theta}_0)$, unless otherwise specified. In addition, in order to show the benefit of considering the graph structure, we include `NN-UCB`, `NN-TS`, and `NN-PE` as our baselines. For these NN-based algorithms, we ignore the adjacency matrix of a graph (setting $\boldsymbol{A} = \boldsymbol{I}_N$), and pass through the model in (4.1) and (4.2) with $\boldsymbol{h}_i^G = \boldsymbol{X}_{i*}$. The MLPs in our experiments have $L = 2$ layers and width $m = 512$. We use SGD as the optimizer, with mini-batch size 5, and train for 30 epochs. For the tuning of the hyperparameters (η, λ) and other algorithmic setup, see Appendix 6.6.4. The

matrix inversion in the algorithms is approximated by diagonal inversion across all policy algorithms.

Regret Experiments. In Figure 4.1, we show the performance of all the algorithms for the six possible environments: ER or RDPG model coupled with either of the three reward models. We set the size of the graph domain to $|\mathcal{G}| = 100$ in Figure 4.1 and we experiment across different $|\mathcal{G}|$ in Appendix 6.6.4. Figure 4.1 demonstrates that `GNN-TS` consistently outperforms the baseline algorithms and is robust to all types of random graph models and reward function generations in our experiment. In addition, GNN-based algorithms are clearly better than NN-based algorithms in graph action bandit settings.

4.7 Appendix

4.7.1 Proof for Lemmas in Regret Analysis

4.7.1.1 Notations

In the following parts, we further define the some notations to represent the linear and kernelized models:

$$\begin{aligned}
\boldsymbol{G}_t &= [g(G_1;\boldsymbol{\theta}_0), ..., g(G_t;\boldsymbol{\theta}_{t-1}))] \in \mathbb{R}^{p\times t} \\
\bar{\boldsymbol{G}}_t &= [g(G_1;\boldsymbol{\theta}_0), ..., g(G_t;\boldsymbol{\theta}_0)] \in \mathbb{R}^{p\times t} \\
\boldsymbol{\mu}_t &= [\mu(G_1), ..., \mu(G_t)]^\top \in \mathbb{R}^{t\times 1} \\
\boldsymbol{y}_t &= [y_1, ..., y_t]^\top \in \mathbb{R}^{t\times 1} \\
\boldsymbol{\epsilon}_t &= [\epsilon_1, ..., \epsilon_t]^\top \in \mathbb{R}^{t\times 1}.
\end{aligned} \tag{4.23}$$

Then we define the uncertainty estimate with initial gradient $\boldsymbol{\theta}_0$:

$$\bar{\sigma}_t^2(G) = \frac{1}{m}\|g(G;\boldsymbol{\theta}_0)\|^2_{\bar{\boldsymbol{U}}_{t-1}^{-1}} \quad \text{and} \quad \bar{\boldsymbol{U}}_t = \lambda \boldsymbol{I}_p + \sum_{i=1}^{t} g(G_i;\boldsymbol{\theta}_0)g(G_i;\boldsymbol{\theta}_0)^\top/m. \tag{4.24}$$

4.7.1.2 Proof of Lemma 4.5.1

Let us write

$$\widetilde{\boldsymbol{\theta}}_{t-1} := \bar{\boldsymbol{U}}_{t-1}^{-1}\bar{\boldsymbol{G}}_{t-1}\boldsymbol{y}_{t-1}/m$$

for the ridge regression solution. We will need the following auxiliary lemmas:

Lemma 4.7.1 (Taylor Approximation of a GNN)**.** *Suppose learning rate $\eta \leq (\tilde{C}mL+m\lambda)^{-1}$ for some constant $\tilde{C}$, then for any fixed $t \in [T]$ and $G \in \mathcal{G}$, with probability at least $1-\delta$*

$$|f_{\text{GNN}}(G;\boldsymbol{\theta}_t^{(J)}) - f_{\text{GNN}}(G;\boldsymbol{\theta}_0) - \langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_t^{(J)} - \boldsymbol{\theta}_0\rangle| \leq CL^3\Big(\frac{R^2+\sigma_\epsilon^2}{m\lambda}\Big)^{2/3}\sqrt{m\log(m)} \tag{4.25}$$

where C is some constant independent of m and t.

Lemma 4.7.2. *Suppose $m \geq poly(R, \sigma_\epsilon, L, \lambda^{-1}, |\mathcal{G}|, \rho_{min}^{-1}, \log(LN|\mathcal{G}|/\delta))$ given a fixed $\delta \in (0,1)$ and learning rate $\eta \leq (\tilde{C}mL + m\lambda)^{-1}$ for some constant $\tilde{C}$. For $G \in \mathcal{G}_t$ and $t > 1$, with probability at lease $1-\delta$,*

$$|\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_0 - \widetilde{\boldsymbol{\theta}}_{t-1}\rangle| \leq C\bar{\sigma}_t(G) \tag{4.26}$$

where $C = (C_1(2-\eta m\lambda)^J + C_2)\sqrt{\frac{\sigma_\epsilon^2+R^2}{\lambda}(1+\frac{3\rho_{\max}}{2\lambda})}$ with $C_1 = \mathcal{O}(1)$ and $C_2 = \mathcal{O}(\lambda^{1/3})$.

Lemma 4.7.3. *Fix $\delta \in (0,1)$ and let $m = \Omega(L^{10}T^4|\mathcal{G}|^6\rho_{min}^{-4}\log(LN^2|\mathcal{G}|^2/\delta))$. Then, there exists $\boldsymbol{\theta}^* \in \mathbb{R}^p$ with $\sqrt{m}\|\boldsymbol{\theta}^*\|_2 \leq \sqrt{2}R$ such that with probability at least $1-\delta$,*

$$\begin{aligned} \mu(G) &= \langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}^*\rangle, \quad \text{for all } G \in \mathcal{G} \\ \log\det(\lambda^{-1}\bar{\boldsymbol{U}}_t) &\leq \log\det(\boldsymbol{I}_{|\mathcal{G}|} + \lambda^{-1}t\boldsymbol{K}) + 1. \end{aligned} \tag{4.27}$$

Lemma 4.7.4. *With probability at least $1-\delta$, we have*

$$|\bar{\sigma}_t(G) - \sigma_t(G)| \leq Ct\lambda^{-1/6}L^{9/2}(R^2+\sigma_\epsilon^2)^{1/6}m^{-1/6}\sqrt{\log(m)}. \tag{4.28}$$

We choose an arbitrary small $\delta \in (0,1)$ and set $\delta_i = \delta/(5T)$ for $i = 1, \ldots, 5$. For all $\forall G \in \mathcal{G}_t$, we have

$$|f_{\text{GNN}}(G;\boldsymbol{\theta}_{t-1}) - \mu(G)| \le \underbrace{|f_{\text{GNN}}(G;\boldsymbol{\theta}_{t-1}) - \langle g(G;\boldsymbol{\theta}_0), \widetilde{\boldsymbol{\theta}}_{t-1}\rangle|}_{:=I_1} + \underbrace{|\mu(G) - \langle g(G;\boldsymbol{\theta}_0), \widetilde{\boldsymbol{\theta}}_{t-1}\rangle|}_{:=I_2}. \tag{4.29}$$

We then turn to bounding I_1 and I_2. Throughout the proof, let

$$\gamma_m := m^{-1/6}\sqrt{\log m} \tag{4.30}$$

Bounding I_1: By Lemma 4.7.1 and Lemma 4.7.2, with probability at least $1 - \delta_1 - \delta_2$,

$$\begin{aligned} I_1 &= |f_{\text{GNN}}(G;\boldsymbol{\theta}_{t-1}) - \langle g(G;\boldsymbol{\theta}_0), \widetilde{\boldsymbol{\theta}}_{t-1}\rangle| \\ &\le |f_{\text{GNN}}(G;\boldsymbol{\theta}_{t-1}) - \langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_0\rangle| + |\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}_0 - \widetilde{\boldsymbol{\theta}}_{t-1}\rangle| \\ &\le C_0 L^3 \gamma_m + \tilde{C}_2\, \bar{\sigma}_t(G). \end{aligned} \tag{4.31}$$

where $C_0 := \tilde{C}_1\Big(\frac{R^2+\sigma_\epsilon^2}{\lambda}\Big)^{2/3}$ and $\tilde{C}_2 := (\bar{C}_1(2-\eta m\lambda)^J + \bar{C}_2\lambda^{1/3})\sqrt{\frac{\sigma_\epsilon^2+R^2}{\lambda}(1+\frac{3\rho_{\max}}{2\lambda})}$ for some constant $\bar{C}_1, \bar{C}_2$. For $\lambda \gtrsim (\sigma_\epsilon^2 + R^2)^3 + \rho_{\max}$, we have $C_0, \tilde{C}_2 \lesssim 1$ subject to the constraint in η in Lemma 4.7.2. Thus, we obtain

$$I_1 \lesssim L^3\gamma_m + \bar{\sigma}_t(G).$$

Bounding I_2: By Lemma 4.7.11, with at least probability $1 - \delta_3$, for all $G \in \mathcal{G}$, we have

$$I_2 = |\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}^* - \widetilde{\boldsymbol{\theta}}_{t-1}\rangle|.$$

Recall that $\boldsymbol{y}_{t-1} = \boldsymbol{\mu}_{t-1} + \boldsymbol{\epsilon}_{t-1}$ and by Lemma 4.7.3, we have $\boldsymbol{\mu}_{t-1} = \bar{\boldsymbol{G}}_{t-1}^\top \boldsymbol{\theta}^*$. Then,

$$\widetilde{\boldsymbol{\theta}}_{t-1} = \bar{\boldsymbol{U}}_{t-1}^{-1}\bar{\boldsymbol{G}}_{t-1}\bar{\boldsymbol{G}}_{t-1}^\top\boldsymbol{\theta}^*/m + \bar{\boldsymbol{U}}_{t-1}^{-1}\bar{\boldsymbol{G}}_{t-1}\boldsymbol{\epsilon}_{t-1}/m$$

We have $\bar{\boldsymbol{U}}_t = \lambda \boldsymbol{I}_p + \bar{\boldsymbol{G}}_t \bar{\boldsymbol{G}}_t^\top / m$. Hence, $\bar{\boldsymbol{U}}_t^{-1} \bar{\boldsymbol{G}}_t \bar{\boldsymbol{G}}_t^\top / m = \bar{\boldsymbol{U}}_t^{-1}(\bar{\boldsymbol{U}}_t - \lambda \boldsymbol{I}_p) = \boldsymbol{I}_p - \lambda \bar{\boldsymbol{U}}_t^{-1}$. This gives

$$\widetilde{\boldsymbol{\theta}}_{t-1} = \boldsymbol{\theta}^* - \lambda \bar{\boldsymbol{U}}_{t-1}^{-1} \boldsymbol{\theta}^* + \frac{1}{\sqrt{m}} \bar{\boldsymbol{U}}_{t-1}^{-1} \boldsymbol{S}_{t-1}$$

where we have defined $\boldsymbol{S}_{t-1} := \frac{1}{\sqrt{m}} \bar{\boldsymbol{G}}_{t-1} \boldsymbol{\epsilon}_{t-1}$. Thus, we have

$$I_2 \le \lambda |\langle g(G; \boldsymbol{\theta}_0), \bar{\boldsymbol{U}}_t^{-1} \boldsymbol{\theta}^* \rangle| + \frac{1}{\sqrt{m}} |\langle g(G; \boldsymbol{\theta}_0), \bar{\boldsymbol{U}}_{t-1}^{-1} \boldsymbol{S}_{t-1} \rangle| \tag{4.32}$$

Recall that $\sqrt{m} \bar{\sigma}_t(G) = \|g(G; \boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}}$. Since $\bar{\boldsymbol{U}}_{t-1}^{-1} \preccurlyeq \frac{1}{\lambda} \boldsymbol{I}_p$, for any vector $\boldsymbol{v}$, we have $\|\boldsymbol{v}\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}} \le \frac{1}{\sqrt{\lambda}} \|\boldsymbol{v}\|$. Then, for the first term in (4.32), we have

$$\begin{aligned} \lambda |g(G; \boldsymbol{\theta}_0)^\top \bar{\boldsymbol{U}}_{t-1}^{-1} \boldsymbol{\theta}^*| &\le \lambda \|g(G; \boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}} \cdot \|\boldsymbol{\theta}^*\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}} \\ &\le \sqrt{m} \bar{\sigma}_t(G) \cdot \sqrt{\lambda} \|\boldsymbol{\theta}^*\|_2 \le \bar{\sigma}_t(G) \sqrt{2\lambda} R \end{aligned}$$

where we have used Cauchy-Schwarz inequality for $\langle \cdot, \cdot \rangle_{\boldsymbol{U}_{t-1}^{-1}}$ and Lemma 4.7.3. For the second term in (4.32), we have

$$\frac{1}{\sqrt{m}} |g(G; \boldsymbol{\theta}_0)^\top \bar{\boldsymbol{U}}_{t-1}^{-1} \boldsymbol{S}_{t-1}| \le \frac{1}{\sqrt{m}} \|g(G; \boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}} \|\boldsymbol{S}_{t-1}\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}} = \bar{\sigma}_t(G) \cdot \|\boldsymbol{S}_{t-1}\|_{\bar{\boldsymbol{U}}_{t-1}^{-1}}$$

By Theorem 20.4 of (LS20), with probability at least $1 - \delta_4$, we have

$$\frac{1}{\sigma_\epsilon^2} \|\boldsymbol{S}_t\|_{\bar{\boldsymbol{U}}_t^{-1}}^2 \le 2 \log(1/\delta_4) + \log \deg(\lambda^{-1} \bar{\boldsymbol{U}}_t), \quad \text{for all } t \in [T].$$

By Lemma 4.7.3, with high probability,

$$\log \det(\lambda^{-1} \bar{\boldsymbol{U}}_t) \le \log \det(\boldsymbol{I}_{|\mathcal{G}|} + T\boldsymbol{K}/\lambda) + 1 \le 2\tilde{d} \log(1 + T\rho_{\max}/\lambda). \tag{4.33}$$

Using $\lambda \gtrsim \rho_{\max}$, we have $\log \det(\lambda^{-1} \bar{\boldsymbol{U}}_t) \lesssim \tilde{d} \log(T) + 1 \lesssim \tilde{d} \log(T)$. We also have $\log(1/\delta_4) = \log(5T) \lesssim \log(T) \lesssim \tilde{d} \log(T)$.

Putting the pieces together, we have

$$\frac{1}{\sqrt{m}}|g(G;\boldsymbol{\theta}_0)^\top \bar{\boldsymbol{U}}_{t-1}^{-1}\boldsymbol{S}_{t-1}| \lesssim \sigma_\epsilon\sqrt{\tilde{d}\log T}\cdot\bar{\sigma}_t(G).$$

Combining with the first term, we obtain

$$I_2 \lesssim \big(\sigma_\epsilon\sqrt{\tilde{d}\log T}+\sqrt{\lambda}R\big)\bar{\sigma}_t(G).$$

Combining with the bound on I_1, we have

$$\begin{aligned}|f_{\mathrm{GNN}}(G;\boldsymbol{\theta}_{t-1})-\mu(G)| &\lesssim L^3\gamma_m + \big(1+\sigma_\epsilon\sqrt{\tilde{d}\log T}+\sqrt{\lambda}R\big)\,\bar{\sigma}_t(G)\\ &=: L^3\gamma_m + \alpha\,\bar{\sigma}_t(G)\end{aligned}$$

where we have set $\alpha := 1+\sigma_\epsilon\sqrt{\tilde{d}\log T}+\sqrt{\lambda}R$ for simplificty.

By Lemma 4.7.4, with probability at least $1-\delta_5$,

$$\bar{\sigma}_t(G)-\sigma_t(G) \le CtL^{9/2}\Big(\frac{R^2+\sigma_\epsilon^2}{\lambda}\Big)^{1/6}\gamma_m \lesssim t\cdot L^{9/2}\gamma_m$$

using the assumption $\lambda \gtrsim R^2+\sigma_\epsilon^2$. We obtain

$$\begin{aligned}|f_{\mathrm{GNN}}(G;\boldsymbol{\theta}_{t-1})-\mu(G)| &\lesssim L^3\gamma_m + t\cdot\alpha L^{9/2}\gamma_m + \alpha\sigma_t(G)\\ &\le 2t\cdot\alpha L^{9/2}\gamma_m + \alpha\sigma_t(G)\end{aligned}$$

since $t\ge 1$ and $\alpha\ge 1$. Taking $\nu\ge\alpha$ finishes the proof.

4.7.1.3 Proof of Lemma 4.5.2

Proof of Lemma 4.5.2. Conditioned on $\mathcal{F}_t$, we have

$$\widehat{r}_t(G) \,|\, \mathcal{F}_t \sim \mathcal{N}(f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1}), \nu^2 \sigma_t^2(G)). \tag{4.34}$$

Using standard Gaussian tail bound, followed by a union bound gives

$$\mathbb{P}_t(|\widehat{r}_t(G) - f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1})| \geq \nu \sigma_t(G) \cdot u) \leq |\mathcal{G}_t| e^{-u^2/2} \tag{4.35}$$

which shows the first assertion by letting $u = \sqrt{2 \log(t^2 |\mathcal{G}_t|)}$.

For the second assertion, it is enough to note that $\mathbb{P}(Z \geq 1) \geq (4e\sqrt{\pi})^{-1}$ for $Z \sim \mathcal{N}(0,1)$. □

4.7.1.4 Proof of Lemma 4.5.3

Proof of Lemma 4.5.3. Our proof is inspired from the proof in (WWL22). Recall that $c_t^{\mu}(G) = \nu \sigma_t(G) + \epsilon(t,m)$ and $c_t^{\sigma}(G) := \nu \sigma_t(G) \sqrt{2 \log(t^2 |\mathcal{G}_t|)}$ and

$$\begin{aligned} \mathcal{E}_t^{\mu} &= \{\forall G \in \mathcal{G}_t, |f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1}) - \mu(G)| \leq c_t^{\mu}(G)\} \\ \mathcal{E}_t^{\sigma} &= \{\forall G \in \mathcal{G}_t, |\widehat{r}_t(G) - f_{\mathrm{GNN}}(G; \boldsymbol{\theta}_{t-1})| \leq c_t^{\sigma}(G)\} \end{aligned} \tag{4.36}$$

Let $\gamma_t(G) = c_t^{\mu}(G) + c_t^{\sigma}(G)$ and $c_t(G) = \gamma_t(G) + \epsilon(t,m)$. Then, on $\mathcal{E}_t^{\mu} \cap \mathcal{E}_t^{\sigma}$, by triangle inequality,

$$|\widehat{r}_t(G) - \mu(G)| \leq \gamma_t(G). \tag{4.37}$$

We also recall that $\mathcal{E}_t^a := \{\widehat{r}_t(G_t^*) - f_{\mathrm{GNN}}(G_t^*; \boldsymbol{\theta}_{t-1}) > \nu\sigma_t(G_t^*)\}$. Then, on $\mathcal{E}_t^\mu \cap \mathcal{E}_t^a$, we have

$$\begin{aligned}\widehat{r}_t(G_t^*) &> f_{\mathrm{GNN}}(G_t^*; \boldsymbol{\theta}_{t-1}) + \nu\sigma_t(G_t^*) \\ &\geq \mu(G_t^*) - c_t^\mu(G_t^*) + \nu\sigma_t(G_t^*) \\ &= \mu(G_t^*) - \epsilon(t, m) \end{aligned} \tag{4.38}$$

Recall that $\Delta_t := \mu(G_t^*) - \mu(G_t)$ for convenience. Consider the set of unsaturated actions

$$\mathcal{U}_t = \big\{G \in \mathcal{G}_t : \ \mu(G_t^*) < \mu(G) + c_t(G)\big\}$$

and let $\bar{G}_t$ be the least uncertain unsaturated action at time t:

$$\bar{G}_t := \operatorname*{argmin}_{G \in \mathcal{U}_t} c_t(G). \tag{4.39}$$

By $\bar{G}_t \in \mathcal{U}_t$, we have $\Delta_t \leq c_t(\bar{G}_t) + \mu(\bar{G}_t) - \mu(G_t)$. Applying (4.37), twice, on $\mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma$, we have

$$\begin{aligned}\Delta_t &\leq c_t(\bar{G}_t) + \gamma_t(\bar{G}_t) + \gamma_t(G_t) + \widehat{r}_t(\bar{G}_t) - \widehat{r}_t(G_t) \\ &\leq \ c_t(\bar{G}) + \gamma_t(\bar{G}_t) + \gamma_t(G_t)\end{aligned}$$

for all $G \in \mathcal{G}_t$ where the second inequality follows since G_t maximizes $\widehat{r}_t(\cdot)$ over $\mathcal{G}_t$, by design.

Recall that $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \,|\, \mathcal{F}_t]$, where $\mathcal{F}_t$ is the history up to (but not including) time t. Given $\mathcal{F}_t$, the event $\mathcal{E}_t^\mu$ is deterministic while $\mathcal{E}_t^\sigma$ is only random due to the independent randomness in the sampling step (4.4). Next, we have

$$\begin{aligned}\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\mu}] &= \mathbb{I}_{\mathcal{E}_t^\mu} \cdot \mathbb{E}_t[\Delta_t] \\ &= \mathbb{I}_{\mathcal{E}_t^\mu} \cdot \big(\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\sigma}] + \mathbb{E}_t[\Delta_t \mathbb{I}_{\bar{\mathcal{E}}_t^\sigma}]\big) \\ &\leq \mathbb{I}_{\mathcal{E}_t^\mu} \cdot \big(\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\sigma}] + B\,\mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)\big)\end{aligned} \tag{4.40}$$

using the boundedness Assumption 2. Here, we are using the fact that $\mathcal{E}_t^\mu$ is measurable w.r.t. $\mathcal{F}_t$, hence it is deterministic conditioned on $\mathcal{F}_t$. Due to factor $\mathbb{I}_{\mathcal{E}_t^\mu}$ in the above, the bound is trivial when $\mathcal{E}_t^\mu$ fails, so for the rest of the proof we assume that $\mathcal{E}_t^\mu$ holds (conditioned on $\mathcal{F}_t$).

We have

$$\begin{aligned}\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\sigma}] &\leq c_t(\bar{G}_t) + \gamma_t(\bar{G}_t) + \mathbb{E}_t[\gamma_t(G_t)\mathbb{I}_{\mathcal{E}_t^\sigma}] \\ &\leq 2c_t(\bar{G}_t) - \epsilon(t,m) + \mathbb{E}_t[\gamma_t(G_t)]\end{aligned}$$

where we have used the definition of $c_t(\cdot)$ and dropped the indicator $\mathbb{I}_{\mathcal{E}_t^\sigma}$ to get a further upper bound. It remains to bound $c_t(\bar{G}_t)$ in terms of $\gamma_t(G_t)$.

Since $\bar{G}_t$ is the least uncertain unsaturated action, we have

$$c_t(\bar{G}_t)\mathbb{I}\{G_t \in \mathcal{U}_t\} \leq c_t(G_t).$$

Multiplying both sides by $\mathbb{I}_{\mathcal{E}_\sigma^t}$, taking $\mathbb{E}_t[\cdot]$, and rearranging

$$c_t(\bar{G}_t) \leq \frac{\mathbb{E}_t[c_t(G_t)\mathbb{I}_{\mathcal{E}_\sigma^t}]}{\mathbb{P}_t(\{G_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^\sigma)} \leq \frac{\mathbb{E}_t[\gamma_t(G_t)]}{\mathbb{P}_t(\{G_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^\sigma)}.$$

It remains to bound the denominator.

Recall that G_t maximizes $\widehat{r}_t(\cdot)$ over the entire $\mathcal{G}_t$. Thus, if

$$\widehat{r}_t(G_t^*) > \max_{G \in \bar{\mathcal{U}}_t} \widehat{r}_t(G) \tag{4.41}$$

then G_t cannot belong to $\bar{\mathcal{U}}_t$, hence $G_t \in \mathcal{U}_t$. On $\mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma$, for any $G \in \bar{\mathcal{U}}_t$, we have

$$\begin{aligned}\widehat{r}_t(G) \leq \mu(G) + \gamma_t(G) &\leq \mu(G_t^*) - c_t(G) + \gamma_t(G) \\ &\leq \mu(G_t^*) - \epsilon(t,m)\end{aligned}$$

where the second inequality is by the definition of $\bar{\mathcal{U}}_t$. Then for (4.41) to hold on $\mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma$, it is enough to have $\widehat{r}_t(G_t^*) > \mu(G_t^*) - \epsilon(t,m)$. But this holds on $\mathcal{E}_t^\mu \cap \mathcal{E}_t^a$ by (4.38). That is,

$$\begin{aligned}\mathcal{E}_t^a \cap \mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma &\subset \{\widehat{r}_t(G_t^*) > \mu(G_t^*) - \epsilon(t,m)\} \cap \mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma \\ &\subset \{\widehat{r}_t(G_t^*) > \max_{G \in \bar{\mathcal{U}}_t} \widehat{r}_t(G)\} \cap \mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma \\ &\subset \{G_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^\mu \cap \mathcal{E}_t^\sigma.\end{aligned}$$

Assuming as before that $\mathcal{E}_t^\mu$ holds, we have

$$\mathbb{P}_t(\mathcal{E}_t^a \cap \mathcal{E}_t^\sigma) \leq \mathbb{P}_t(\{G_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^\sigma). \tag{4.42}$$

We have $\mathbb{P}_t(\mathcal{E}_t^a \cap \mathcal{E}_t^\sigma) \geq \mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)$. Putting the pieces together

$$c_t(\bar{G}_t) \leq \frac{\mathbb{E}_t[\gamma_t(G_t)]}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)}$$

and we obtain

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^\sigma}] \leq \Big(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^\sigma)} + 1\Big)\mathbb{E}_t[\gamma_t(G_t)] - \epsilon(t,m)$$

Combining with (4.40) the result follows. □

4.7.1.5 Proof of Lemma 4.5.4

Proof of Lemma 4.5.4. For simplicity, we define

$$g_t := \frac{1}{\sqrt{m}} g(G_t; \boldsymbol{\theta}_{t-1}), \quad \bar{g}_t := \frac{1}{\sqrt{m}} g(G_t; \boldsymbol{\theta}_0). \tag{4.43}$$

Then, recall that

$$\sigma_t^2(G_t) = \|g_t\|^2_{\boldsymbol{U}_{t-1}^{-1}}, \quad \boldsymbol{U}_{t-1} = \lambda \boldsymbol{I}_p + \sum_{i=1}^{t-1} g_t g_t^\top.$$

Note that $\boldsymbol{U}_t = \boldsymbol{U}_{t-1} + g_t g_t^\top$.

Then we introduce following Lemmas:

Lemma 4.7.5 (Elliptical Potential)**.** *Assume that $\boldsymbol{U}_t = \boldsymbol{U}_{t-1} + g_t g_t^\top$ for all $t \in [T]$. Then,*

$$\sum_{t=1}^{T} \min\{1, \|g_t\|^2_{\boldsymbol{U}_{t-1}^{-1}}\} \leq 2 \log\left(\frac{\det \boldsymbol{U}_T}{\det \boldsymbol{U}_0}\right). \tag{4.44}$$

Lemma 4.7.6. *Let $\boldsymbol{A} = [\boldsymbol{a}_1 \ \boldsymbol{a}_2 \ \cdots \ \boldsymbol{a}_n]$ and $\bar{\boldsymbol{A}} = [\bar{\boldsymbol{a}}_1 \ \bar{\boldsymbol{a}}_2 \ \cdots \ \bar{\boldsymbol{a}}_n]$ be $p \times n$ matrices, with columns $\{\boldsymbol{a}_i\}$ and $\{\bar{\boldsymbol{a}}_i\}$, respectively. Assume that for $\epsilon \leq C$, we have*

$$\|\boldsymbol{a}_i - \bar{\boldsymbol{a}}_i\| \leq \epsilon, \quad \|\boldsymbol{a}_i\| \leq C$$

for all i. Then,

$$\log\det(\boldsymbol{I}_p + \boldsymbol{A}\boldsymbol{A}^\top) \leq \log\det(\boldsymbol{I}_p + \bar{\boldsymbol{A}}\bar{\boldsymbol{A}}^\top) + p\log(1 + 3Cn\epsilon) \tag{4.45}$$

$$\log\det(\boldsymbol{I}_p + \boldsymbol{A}\boldsymbol{A}^\top) \leq \log\det(\boldsymbol{I}_n + \bar{\boldsymbol{A}}^\top\bar{\boldsymbol{A}}) + 3Cn^{3/2}\epsilon. \tag{4.46}$$

By Lemma 4.7.5, we have

$$\frac{1}{2}\sum_{t=1}^{T} \min\{1, \sigma_t^2(G_t)\} \leq \log\left(\frac{\det \boldsymbol{U}_T}{\det \boldsymbol{U}_0}\right) = \log\det(\lambda^{-1}\boldsymbol{U}_T) =: \log\det(\boldsymbol{V}_T) \tag{4.47}$$

using $\det(\boldsymbol{U}_0) = \det(\lambda \boldsymbol{I}_p) = \lambda^p$, and defining $\boldsymbol{V}_t := \lambda^{-1}\boldsymbol{U}_t$.

Let $\mathcal{G} = \{G^j : j \in [|\mathcal{G}|]\}$ be the collection of all the graphs and $n_j(t)$ be the number of graphs which are equal to $G^j \in \mathcal{G}$ in our selection of graphs up to and including time t, i.e

$n_j(t) := \sum_{t=1}^{t} \mathbb{I}_{G_i = G^j}$. Let

$$\boldsymbol{\psi}_j := \frac{1}{\sqrt{m}} g(G^j; \boldsymbol{\theta}_{t-1}), \quad \bar{\boldsymbol{\psi}}_j := \frac{1}{\sqrt{m}} g(G^j; \boldsymbol{\theta}_0) \tag{4.48}$$

and let $\boldsymbol{\Psi}$ and $\bar{\boldsymbol{\Psi}}$ be the corresponding $p \times |\mathcal{G}|$ matrices with the above columns. Then, we have

$$\sum_{i=1}^{T} g_i g_i^\top = \sum_{j=1}^{|\mathcal{G}|} n_j(T) \boldsymbol{\psi}_j \boldsymbol{\psi}_j^\top = \boldsymbol{\Psi} \boldsymbol{D} \boldsymbol{\Psi}^\top \preceq T \cdot \boldsymbol{\Psi} \boldsymbol{\Psi}^\top \tag{4.49}$$

where $\boldsymbol{D} \in \mathbb{R}^{|\mathcal{G}| \times |\mathcal{G}|}$ is the diagonal matrix with diagonal elements $\{n_j(T)\}_{j=1}^{|\mathcal{G}|}$ and the last inequality due to $n_j(T) \leq T$ for all $j \in [|\mathcal{G}|]$.

Note that $\boldsymbol{V}_T = \boldsymbol{I}_p + \lambda^{-1} \sum_{i=1}^{T} g_i g_i^\top$, hence

$$\log\det(\boldsymbol{V}_T) \leq \log\det(\boldsymbol{I}_p + \lambda^{-1} T \cdot \boldsymbol{\Psi} \boldsymbol{\Psi}^\top). \tag{4.50}$$

By Lemma 4.7.19, fix a $\delta_1 \in (0, 1)$, we have the following bound for $\|\boldsymbol{\psi}_j\|_2$ and $\|\boldsymbol{\psi}_j - \bar{\boldsymbol{\psi}}_j\|_2$, with probability at least $1 - \delta_1$,

$$\begin{aligned} \|\boldsymbol{\psi}_j\|_2 &\leq \frac{1}{N} \sum_{i \in \mathcal{V}(G^j)} \|g_{\mathrm{MLP}}(\boldsymbol{h}_i^{G^j}; \boldsymbol{\theta}_{t-1})/\sqrt{m}\|_2 \leq C_\psi \\ \|\boldsymbol{\psi}_j - \bar{\boldsymbol{\psi}}_j\|_2 &\leq \frac{1}{N} \sum_{i \in \mathcal{V}(G^j)} \|g_{\mathrm{MLP}}(\boldsymbol{h}_i^{G^j}; \boldsymbol{\theta}_{t-1})/\sqrt{m} - g_{\mathrm{MLP}}(\boldsymbol{h}_i^{G^j}; \boldsymbol{\theta}_0)/\sqrt{m}\|_2 \leq \epsilon_m \end{aligned} \tag{4.51}$$

where $\epsilon_m = o(1)$ as $m \to \infty$ and C_ψ is $C_7\sqrt{L}$ in Lemma 4.7.19.

Then, applying Lemma 4.7.6 with $n = |\mathcal{G}|$, $\boldsymbol{A} = \sqrt{\lambda^{-1}T}\boldsymbol{\Psi}$, $\bar{\boldsymbol{A}} = \sqrt{\lambda^{-1}T}\bar{\boldsymbol{\Psi}}$ and ϵ replaced with $\sqrt{\lambda^{-1}T}\epsilon_m$, we obtain

$$\log\det(\boldsymbol{V}_T) \leq \log\det(\boldsymbol{I}_{|\mathcal{G}|} + \lambda^{-1} T \cdot \bar{\boldsymbol{\Psi}}^\top \bar{\boldsymbol{\Psi}}) + 3C_\psi |\mathcal{G}|^{3/2} \sqrt{T} \lambda^{-1/2} \epsilon_m \tag{4.52}$$

Recall $\hat{\boldsymbol{K}} = \bar{\boldsymbol{\Psi}}^\top \bar{\boldsymbol{\Psi}}$ and $\hat{\rho}_{\max} = \lambda_{\max}(\hat{\boldsymbol{K}})$ and note that $\hat{\boldsymbol{K}}$ is the finite-width GNTK matrix.

By Lemma 4.7.12, with high probability, $\hat{\rho}_{\max} \le \rho_{\max} + \epsilon_{\rho,m}$ and note that $\epsilon_{\rho,m} = \Omega(m^{-1/4})$. Dropping $\epsilon_{\rho,m}$ by large enough m, we have

$$\log\det(\boldsymbol{I}_{|\mathcal{G}|} + \lambda^{-1}T \cdot \bar{\boldsymbol{\Psi}}^\top \bar{\boldsymbol{\Psi}}) \le |\mathcal{G}| \log(1 + T\rho_{\max}/\lambda).$$

Putting the pieces together with the definition of effective dimension $\tilde{d}$ in (4.8) finishes the proof. □

4.7.1.6 Proof of Lemma 4.7.5

Proof of Lemma 4.7.5. Since $\min\{1, x\} \le 2\log(1+x)$ for $x \ge 0$, we have

$$\begin{aligned}\sum_{t=1}^{T} \min\{1, \|g_t\|^2_{\boldsymbol{U}_{t-1}^{-1}}\} &\le 2\sum_{t} \log(1 + \|g_t\|^2_{\boldsymbol{U}_{t-1}^{-1}}) \\ &= 2\sum_{t=1}^{T} \log\Big(\frac{\det \boldsymbol{U}_t}{\det \boldsymbol{U}_{t-1}}\Big) = 2\log\Big(\frac{\det \boldsymbol{U}_T}{\det \boldsymbol{U}_0}\Big)\end{aligned}$$

where the first equality follows from $\det(\boldsymbol{A} + \boldsymbol{v}\boldsymbol{v}^\top) = \det(\boldsymbol{A})(1 + \boldsymbol{v}^\top \boldsymbol{A}^{-1}\boldsymbol{v})$, obtained by an application of Sylvester's determinant identity: $\det(\boldsymbol{I} + \boldsymbol{A}\boldsymbol{B}) = \det(\boldsymbol{I} + \boldsymbol{B}\boldsymbol{A})$. □

4.7.1.7 Proof of Lemma 4.7.6

Proof of Lemma 4.7.6. Note that

$$\begin{aligned}\|\boldsymbol{a}_i\boldsymbol{a}_i^\top - \bar{\boldsymbol{a}}_i\bar{\boldsymbol{a}}_i^\top\|_{op} &= \|\boldsymbol{a}_i(\boldsymbol{a}_i - \bar{\boldsymbol{a}}_i)^\top - (\bar{\boldsymbol{a}}_i - \boldsymbol{a}_i)\bar{\boldsymbol{a}}_i^\top\|_{op} \\ &\le (\|\boldsymbol{a}_i\| + \|\bar{\boldsymbol{a}}_i\|)\|\boldsymbol{a}_i - \boldsymbol{a}_i\| \le (2C + \epsilon)\epsilon \le 3C\epsilon\end{aligned}$$

Let $\boldsymbol{V} = \boldsymbol{I}_p + \boldsymbol{A}\boldsymbol{A}^\top$ and $\bar{\boldsymbol{V}} = \boldsymbol{I}_p + \bar{\boldsymbol{A}}\bar{\boldsymbol{A}}^\top$. We have

$$\|\boldsymbol{V} - \bar{\boldsymbol{V}}\|_{op} \le \sum_{i=1}^{n} \|\boldsymbol{a}_i\boldsymbol{a}_i^\top - \bar{\boldsymbol{a}}_i\bar{\boldsymbol{a}}_i^\top\|_{op} \le n \cdot 3C\epsilon$$

Write $\lambda_i(\boldsymbol{V})$ for the ith eigenvalue of matrix $\boldsymbol{V}$. By Weyl's inequality $|\lambda_i(\boldsymbol{V}) - \lambda_i(\bar{\boldsymbol{V}})| \leq 3Cn\epsilon$. Then,

$$\begin{aligned}\log\det(\boldsymbol{V}) = \sum_{i=1}^{p} \log\lambda_i(\boldsymbol{V}) &\leq \sum_i \log\big(\lambda_i(\bar{\boldsymbol{V}}) + 3Cn\epsilon\big) \\ &= \sum_i \log(\lambda_i(\bar{\boldsymbol{V}})) + \sum_i \log\Big(1 + \frac{3Cn\epsilon}{\lambda_i(\bar{\boldsymbol{V}})}\Big) \\ &\leq \log\det(\bar{\boldsymbol{V}}) + p\log(1 + 3Cn\epsilon)\end{aligned}$$

using $\lambda_i(\bar{\boldsymbol{V}}) \geq 1$. This proves one of the bounds.

For the second bound, let $\boldsymbol{W} = \boldsymbol{I}_n + \boldsymbol{A}^\top\boldsymbol{A}$ and $\bar{\boldsymbol{W}} = \boldsymbol{I}_n + \bar{\boldsymbol{A}}^\top\bar{\boldsymbol{A}}$. Then, then by concavity of the $\boldsymbol{X} \mapsto \log\det(\boldsymbol{X})$ and the fact that its derivative is $\boldsymbol{X}^{-1}$ over symmetric matrices, we have

$$\log\det(\boldsymbol{X} + \boldsymbol{\Delta}) - \log\deg(\boldsymbol{X}) \leq \mathrm{tr}(\boldsymbol{X}^{-1}\boldsymbol{\Delta}) \leq \|\boldsymbol{X}^{-1}\|_F\|\boldsymbol{\Delta}\|_F.$$

Let $\boldsymbol{\Delta} = \boldsymbol{W} - \bar{\boldsymbol{W}}$. We have $|\boldsymbol{\Delta}_{ij}| = |\langle\boldsymbol{a}_i, \boldsymbol{a}_j\rangle - \langle\bar{\boldsymbol{a}}_i, \bar{\boldsymbol{a}}_j\rangle| \leq 3C\epsilon$, hence $\|\boldsymbol{\Delta}\|_F \leq 3Cn\epsilon$ Then,

$$\begin{aligned}\log\det(\boldsymbol{V}) - \log\det(\bar{\boldsymbol{W}}) &\overset{(a)}{=} \log\det(\boldsymbol{W}) - \log\det(\bar{\boldsymbol{W}}) \\ &\leq \mathrm{tr}(\bar{\boldsymbol{W}}^{-1}\boldsymbol{\Delta}) \\ &\leq \sqrt{n}\|\bar{\boldsymbol{W}}^{-1}\|_{op}\|\boldsymbol{\Delta}\|_F \overset{(b)}{\leq} \sqrt{n}\cdot 3Cn\epsilon.\end{aligned}$$

where (a) is by Sylvester's identity and (b) uses the fact that $\bar{\boldsymbol{W}} \succeq \boldsymbol{I}_n$, hence $\bar{\boldsymbol{W}}^{-1} \preceq \boldsymbol{I}_n$ giving $\|\bar{\boldsymbol{W}}^{-1}\|_{op} \leq 1$. □

4.7.2 Technical Lemmas

In this Section, we provides the Proof for Lemmas in Appendix 4.7.1 and other Technical Lemmas supporting the proofs. Most technical Lemmas are related to NTK and optimization in depp learning theory, mainly modified from the GNN helper Lemmas in (KKB22) and

technical Lemmas in (ZLG20; VBJ21).

4.7.2.1 Notations for MLP

Recall our GNN with one layer of linear graph convolution and an MLP:

$$\begin{aligned} f^{(1)}(\boldsymbol{h}_i^G) &= \boldsymbol{W}^{(1)}\boldsymbol{h}_i^G, \quad i \in [N], \\ f^{(l)}(\boldsymbol{h}_i^G) &= \frac{1}{\sqrt{m}}\boldsymbol{W}^{(l)}\mathrm{ReLU}(f^{(l-1)}(\boldsymbol{h}_i^G)), \quad 2 \le l \le L, \\ f_{\mathrm{MLP}}(\boldsymbol{h}_i^G;\boldsymbol{\theta}) &= f^{(L)}(\boldsymbol{h}_i^G) \\ f_{\mathrm{GNN}}(G;\boldsymbol{\theta}) &= \frac{1}{N}\sum_{i=1}^{N} f_{\mathrm{MLP}}(\boldsymbol{h}_i^G;\boldsymbol{\theta}). \end{aligned} \tag{4.53}$$

We denote the gradients for GNN and associated MLP as

$$\begin{aligned} g(G;\boldsymbol{\theta}) &:= \nabla_{\boldsymbol{\theta}} f_{\mathrm{GNN}}(G;\boldsymbol{\theta}) \\ g_{\mathrm{MLP}}(\cdot;\boldsymbol{\theta}) &:= \nabla_{\boldsymbol{\theta}} f_{\mathrm{MLP}}(\cdot;\boldsymbol{\theta}) \end{aligned} \tag{4.54}$$

and the connection between gradients for the MLP and the gradient for the whole GNN is

$$g(G;\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} g_{\mathrm{MLP}}(\boldsymbol{h}_i^G;\boldsymbol{\theta}) \tag{4.55}$$

Similarly, we define a tangent kernel for the an MLP as

$$\tilde{k}_{MLP}(\boldsymbol{x},\boldsymbol{x}') := g_{MLP}(G;\boldsymbol{\theta}_0)^\top g_{MLP}(G';\boldsymbol{\theta}_0) \tag{4.56}$$

for any MLP inputs $\boldsymbol{x}$, $\boldsymbol{x}'$ and the associated neural tangent kernel $k_{MLP}(\boldsymbol{x},\boldsymbol{x}')$ is defined as limiting kernel of $\tilde{k}_{MLP}(\boldsymbol{x},\boldsymbol{x}')/m$:

$$k_{MLP}(\boldsymbol{x},\boldsymbol{x}') := \lim_{m\to\infty} \tilde{k}_{MLP}(\boldsymbol{x},\boldsymbol{x}')/m. \tag{4.57}$$

By the connection between f_{GNN} and f_{MLP}, we have

$$k(G, G') = \frac{1}{N^2} \sum_{i \in \mathcal{V}(G)} \sum_{j \in \mathcal{V}(G')} k_{\text{MLP}}(\boldsymbol{h}_i^G, \boldsymbol{h}_j^{G'}). \tag{4.58}$$

4.7.2.2 Proof for Lemmas in Appendix 4.7.1

Proof of Lemma 4.7.1. By Lemma 4.7.19, with probability at least $1 - \delta \in (0, 1)$

$$\begin{aligned}
&|f_{\text{GNN}}(G; \boldsymbol{\theta}_t^{(J)}) - f_{\text{GNN}}(G; \boldsymbol{\theta}_0) - \langle g(G; \boldsymbol{\theta}_0), \boldsymbol{\theta}_t^{(J)} - \boldsymbol{\theta}_0 \rangle| \\
&\quad \leq \frac{1}{N} \sum_{j \in \mathcal{V}(G)} |f_{\text{MLP}}(\boldsymbol{h}_j^G; \boldsymbol{\theta}_t^{(J)}) - f_{\text{MLP}}(\boldsymbol{h}_j^G; \boldsymbol{\theta}_0) - \langle g_{\text{MLP}}(\boldsymbol{h}_j^G; \boldsymbol{\theta}_0), \boldsymbol{\theta}_t^{(J)} - \boldsymbol{\theta}_0 \rangle| \\
&\quad \leq C_1 \tau^{4/3} L^3 \sqrt{m \log(m)} \\
&\quad \leq C_1 (\tilde{C} \sqrt{(R^2 + \sigma_\epsilon^2)/m\lambda})^{4/3} L^3 \sqrt{m \log(m)}
\end{aligned} \tag{4.59}$$

where the last inequality is from the choice of $\tau = \tilde{C} \sqrt{(R^2 + \sigma_\epsilon^2)/m\lambda}$ such that $\|\boldsymbol{\theta}_t^{(J)} - \boldsymbol{\theta}_0\|_2 \leq \tau$. Since $\tau \propto 1/\sqrt{m}$, it can be verified that technical condition (4.130) in Lemma 4.7.19 is satisfied when m is large. Therefore, set $C_2 = C_1 \tilde{C}^{4/3}$,

$$|f_{\text{GNN}}(G; \boldsymbol{\theta}_t^{(J)}) - f_{\text{GNN}}(G; \boldsymbol{\theta}_0) - \langle g(G; \boldsymbol{\theta}_0), \boldsymbol{\theta}_t^{(J)} - \boldsymbol{\theta}_0 \rangle| \leq C_2 L^3 (\frac{R^2 + \sigma_\epsilon^2}{m\lambda})^{2/3} \sqrt{m \log(m)}. \tag{4.60}$$

□

Proof of Lemma 4.7.2. In this proof, set $\delta_1 = \delta_2 = \delta/2$ where $\delta \in (0, 1)$ is an arbitrary small real value. We introduce $\{\tilde{\boldsymbol{\theta}}_t^{(j)}\}_{j=1}^J$ be the gradient descent update sequence of the following proximal optimization (KKB22):

$$\min_{\boldsymbol{\theta}} \frac{1}{2t} \sum_{i=1}^{t} (\langle g(G_i; \boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle - y_i)^2 + \frac{m\lambda}{2} \|\boldsymbol{\theta}\|_2^2 \tag{4.61}$$

and $\{\boldsymbol{\theta}_t^{(j)}\}_{j=1}^J$ be the gradient descent update sequence of parameters of our primary opti-

mization (4.6). In GNN training step in algorithms, we let $\boldsymbol{\theta}_t := \boldsymbol{\theta}_t^{(J)}$. Recall that $\bar{\boldsymbol{U}}_t = \lambda \boldsymbol{I} + \bar{\boldsymbol{G}}_t \bar{\boldsymbol{G}}_t^\top / m$. By Lemma 4.7.11, with probability at least $1-\delta_1 \in (0,1)$, $\bar{\boldsymbol{U}}_t \preccurlyeq (\lambda + \frac{3}{2}\rho_{\max})\boldsymbol{I}$. Therefore,

$$
\begin{aligned}
&|\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_t - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\rangle| \\
\leq &\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\|_{\bar{\boldsymbol{U}}_t} \\
\leq &\sqrt{\lambda + 3\rho_{\max}/2}\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\|_2 \\
\leq &\sqrt{\lambda + 3\rho_{\max}/2}\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}} (\|\tilde{\boldsymbol{\theta}}_t^{(J)} - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\|_2 + \|\tilde{\boldsymbol{\theta}}_t^{(J)} - \boldsymbol{\theta}_t\|_2
\end{aligned} \tag{4.62}
$$

By Lemma 4.7.9 and Lemma 4.7.7, with probability at least $1-\delta_2 \in (0,1)$, for some constants C_1 and C_2, we have

$$
\begin{aligned}
&|\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_t - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\rangle| \\
\leq &\sqrt{\lambda + 3\rho_{\max}/2}\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}} \left(C_1(2-\eta m\lambda)^J \sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} + \|\tilde{\boldsymbol{\theta}}_t^{(J)} - \boldsymbol{\theta}_t\|_2 \right) \\
\leq &\sqrt{\lambda + 3\rho_{\max}/2}\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}} \times \left(C_1(2-\eta m\lambda)^J \sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} + C_2\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} \right) \\
= &\sqrt{m(1 + \frac{3\rho_{\max}}{2\lambda})}(C_1(2-\eta m\lambda)^J + C_2)\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}}\bar{\sigma}_{t+1}(G)
\end{aligned} \tag{4.63}
$$

where the first inequality is from Lemma 4.7.9 and the second inequality is from Lemma 4.7.7 and the last equality is obtained from the definition of $\bar{\sigma}_{t+1}^2(G)$, which is

$$
\bar{\sigma}_{t+1}^2(G) = \lambda g^\top(G;\boldsymbol{\theta}_0)\bar{\boldsymbol{U}}_t^{-1} g(G;\boldsymbol{\theta}_0)/m = \frac{\lambda}{m}\|g(G;\boldsymbol{\theta}_0)\|_{\bar{\boldsymbol{U}}_t^{-1}}^2.
$$

Now we let $\tilde{C} = \sqrt{m(1 + \frac{3\rho_{\max}}{2\lambda})}(C_1(2-\eta m\lambda)^J + C_2)\sqrt{\frac{\sigma_\epsilon^2+R^2}{m\lambda}}$. Note that this constant $\tilde{C} = \mathcal{O}(1)$ with respect to m since $\eta = \mathcal{O}(m^{-1})$. Then we have the desired result:

$$
|\langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}_t - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t/m\rangle| \leq \tilde{C}\bar{\sigma}_{t+1}(G) \tag{4.64}
$$

where $\tilde{C} = (C_1(2-\eta m\lambda)^J + C_2)\sqrt{\frac{\sigma_\epsilon^2+R^2}{\lambda}(1+\frac{3\rho_{\max}}{2\lambda})}$ with $C_1 = \mathcal{O}(1)$ and $C_2 = \mathcal{O}(\lambda^{1/3})$. □

Proof of Lemma 4.7.3. See Appendix 4.7.2.4. □

Proof of Lemma 4.7.4. Define function ψ_λ for vectors $\{\boldsymbol{v}, \boldsymbol{a}_1, ..., \boldsymbol{a}_{t-1}\}$ as followed:

$$\psi_\lambda(\boldsymbol{v}, \boldsymbol{a}_1, ..., \boldsymbol{a}_{t-1}) := \sqrt{\boldsymbol{v}^\top(\lambda \boldsymbol{I} + \sum_{i=1}^{t-1} \boldsymbol{a}_i \boldsymbol{a}_i^\top)^{-1}\boldsymbol{v}}, \tag{4.65}$$

and denote the gradients for ψ_λ as

$$\begin{aligned} \nabla_0 \psi_\lambda &:= \nabla_{\boldsymbol{v}} \psi_\lambda(\boldsymbol{v}, \boldsymbol{a}_1, ..., \boldsymbol{a}_{t-1}) \\ \nabla_i \psi_\lambda &:= \nabla_{\boldsymbol{a}_i} \psi_\lambda(\boldsymbol{v}, \boldsymbol{a}_1, ..., \boldsymbol{a}_{t-1}), \forall i \in [t-1]. \end{aligned} \tag{4.66}$$

By setting $\boldsymbol{A} = (\lambda \boldsymbol{I} + \sum_{i=1}^{t-1} \boldsymbol{a}_i \boldsymbol{a}_i^\top)^{-1} \preccurlyeq \frac{1}{\lambda}\boldsymbol{I}$ with eigendecomposition $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^\top$. The gradients are bounded as followed

$$\begin{aligned} \|\nabla_0 \psi_\lambda\|_2 &= \frac{\|\boldsymbol{A}\boldsymbol{v}\|_2}{\sqrt{\boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{v}}} = \sqrt{\frac{\boldsymbol{v}^\top \boldsymbol{A}^2 \boldsymbol{v}}{\boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{v}}} \le \sqrt{\lambda_{max}(\boldsymbol{A})} \le 1/\sqrt{\lambda} \\ \|\nabla_i \psi_\lambda\|_2 &= \frac{\|\boldsymbol{A}\boldsymbol{v}\boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{a}_i\|_2}{\sqrt{\boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{v}}} \le \|\boldsymbol{a}_i\|_2 \frac{\boldsymbol{v}^\top \boldsymbol{A}^2 \boldsymbol{v}}{\sqrt{\boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{v}}} \le \|\boldsymbol{a}_i\|_2 \|\boldsymbol{v}\|_2/\lambda \end{aligned} \tag{4.67}$$

We can express $\bar{\sigma}_t(G)$ and $\sigma_t(G)$ by ψ_λ:

$$\begin{aligned} \bar{\sigma}_t(G) &= \psi_\lambda(\frac{g(G;\boldsymbol{\theta}_{t-1})}{\sqrt{m}}, \frac{g(G_1;\boldsymbol{\theta}_1)}{\sqrt{m}}, ..., \frac{g(G_{t-1};\boldsymbol{\theta}_{t-1})}{\sqrt{m}}) \\ \sigma_t(G) &= \psi_\lambda(\frac{g(G;\boldsymbol{\theta}_0)}{\sqrt{m}}, \frac{g(G_1;\boldsymbol{\theta}_0)}{\sqrt{m}}, ..., \frac{g(G_{t-1};\boldsymbol{\theta}_0)}{\sqrt{m}}). \end{aligned} \tag{4.68}$$

From Lemma 4.7.19, there exists positive constants such that the gradients and gradient differences are bounded with high probability, which indicates for some constant C_1 with

probability greater than $1-\delta$,

$$\|g(G;\boldsymbol{\theta})\|_2 = \|\frac{1}{N}\sum_{j\in\mathcal{V}(G)} g_{\mathrm{MLP}}(\boldsymbol{h}_j^G;\boldsymbol{\theta})\|_2 \leq C_1\sqrt{mL} \tag{4.69}$$

Note that ψ_λ is Lipschitz continuous, then with high probability, we have

$$\begin{aligned}
&|\bar{\sigma}_t(G) - \sigma_t(G)| \\
&= |\psi_\lambda(\frac{g(G;\boldsymbol{\theta}_{t-1})}{\sqrt{m}}, \frac{g(G_1;\boldsymbol{\theta}_1)}{\sqrt{m}}, ..., \frac{g(G_{t-1};\boldsymbol{\theta}_{t-1})}{\sqrt{m}}) - \psi_\lambda(\frac{g(G;\boldsymbol{\theta}_0)}{\sqrt{m}}, \frac{g(G_1;\boldsymbol{\theta}_0)}{\sqrt{m}}, ..., \frac{g(G_{t-1};\boldsymbol{\theta}_0)}{\sqrt{m}})| \\
&\leq \sup\{\|\nabla_0\psi_\lambda\|_2\}\|\frac{g(G;\boldsymbol{\theta}_{t-1})}{\sqrt{m}} - \frac{g(G;\boldsymbol{\theta}_0)}{\sqrt{m}}\|_2 + \sum_{i=1}^{t-1}\sup\{\|\nabla_i\psi_\lambda\|_2\}\|\frac{g(G_i;\boldsymbol{\theta}_i)}{\sqrt{m}} - \frac{g(G_i;\boldsymbol{\theta}_0)}{\sqrt{m}}\|_2 \\
&\leq \frac{1}{\sqrt{\lambda}}\|\frac{g(G;\boldsymbol{\theta}_{t-1})}{\sqrt{m}} - \frac{g(G;\boldsymbol{\theta}_0)}{\sqrt{m}}\|_2 + \frac{C_1^2L}{\lambda}\sum_{i=1}^{t-1}\|\frac{g(G_i;\boldsymbol{\theta}_i)}{\sqrt{m}} - \frac{g(G_i;\boldsymbol{\theta}_0)}{\sqrt{m}}\|_2(\text{ by (4.67) and (4.69)}) \\
&\leq C_2\sqrt{\log(m)}\tau^{1/3}L^3\|g(G;\boldsymbol{\theta}_0)\|_2/\sqrt{m}(\frac{1}{\sqrt{\lambda}} + \frac{C_1^2Lt}{\lambda}) \quad (\text{by Lemma 4.7.19}) \\
&\leq C_1C_2\sqrt{\log(m)}\tau^{1/3}L^{7/2}(\frac{1}{\sqrt{\lambda}} + \frac{C_1^2Lt}{\lambda}) \quad (\text{by (4.69)})
\end{aligned} \tag{4.70}$$

Therefore, if $\lambda \leq C_1^4L^2t^2$ and let $\tau = \tilde{C}\sqrt{\frac{R^2+\sigma_\epsilon^2}{m\lambda}}$, $C_3 = 2\tilde{C}C_2C_1^3$,

$$|\bar{\sigma}_t(G) - \sigma_t(G)| \leq C_3t\lambda^{-7/6}L^{9/2}(R^2+\sigma_\epsilon^2)^{1/6}m^{-1/6}\sqrt{\log(m)} \tag{4.71}$$

□

4.7.2.3 Lemmas for GNN training

Lemma 4.7.7 (Parameter Bound for Primary Optimization)**.** *Let* $\{\boldsymbol{\theta}_t^{(j)}\}_{j=1}^J$ *be the gradient descent update sequence of parameters of the optimization* (4.6) *which is,*

$$\min_{\boldsymbol{\theta}} \frac{1}{2t}\sum_{i=1}^{t}(f_{\mathrm{GNN}}(G_i;\boldsymbol{\theta}) - y_i)^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \tag{4.72}$$

then if $m \geq poly(R, \sigma_\epsilon, L, \lambda^{-1}, \log(\frac{N}{\delta}))$ *and learning rate* $\eta \leq (\tilde{C}mL + m\lambda)^{-1}$ *for some constant* $\tilde{C}$. *Then for a constant* $C = \mathcal{O}(\lambda^{1/3})$ *which is independent of* m *and* t, *with probability at least* $1 - \delta$

$$\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 \leq C\sqrt{\frac{R^2 + \sigma_\epsilon^2}{m\lambda}} \tag{4.73}$$

where $\{\tilde{\boldsymbol{\theta}}_t^{(j)}\}_{j=1}^J$ *be the gradient descent update sequence of parameters of the proximal optimization with loss function* $\frac{1}{2t}\sum_{i=1}^t(\langle g(G_i;\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0\rangle - y_i)^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2$. *Both optimization have the same initialization at* $\tilde{\boldsymbol{\theta}}_t^{(0)} = \boldsymbol{\theta}_t^{(0)} = \boldsymbol{\theta}_0$ *and same learning rate* η.

Proof. In this proof, set $\delta_1 = \delta_2 = \delta/2$ where $\delta \in (0, 1)$ is an arbitrary small real value. Define $\boldsymbol{G}_t^{(j)} := [g(G_1;\boldsymbol{\theta}_t^{(j)}), ..., g(G_t;\boldsymbol{\theta}_t^{(j)}))] \in \mathbb{R}^{p\times t}$ as the j-th updates in our primary optimzation with loss (4.6) at round t. Also define $\boldsymbol{f}_{gnn,t}^{(j)} := [f_{\text{GNN}}(G_1;\boldsymbol{\theta}_t^{(j)}), ..., f_{\text{GNN}}(G_t;\boldsymbol{\theta}_t^{(j)})]^\top \in \mathbb{R}^{t\times 1}$. The gradient descent updates for sequences $\{\boldsymbol{\theta}_t^{(j)}\}_{j=1}^J$ and $\{\tilde{\boldsymbol{\theta}}_t^{(j)}\}_{j=1}^J$ are

$$\begin{aligned}
\boldsymbol{\theta}_t^{(j+1)} &= \boldsymbol{\theta}_t^{(j)} - \eta\left(\frac{1}{t}[\boldsymbol{G}_t^{(j)}]^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t) + m\lambda\boldsymbol{\theta}_t^{(j)}\right) \\
\tilde{\boldsymbol{\theta}}_t^{(j+1)} &= \tilde{\boldsymbol{\theta}}_t^{(j)} - \eta\left(\frac{1}{t}\bar{\boldsymbol{G}}_t^\top(\bar{\boldsymbol{G}}_t(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0) - \boldsymbol{y}_t) + m\lambda\tilde{\boldsymbol{\theta}}_t^{(j)}\right)
\end{aligned} \tag{4.74}$$

Therefore,

$$
\begin{aligned}
&\|\boldsymbol{\theta}_t^{(j+1)} - \tilde{\boldsymbol{\theta}}_t^{(j+1)}\|_2 \\
=&\|(1-\eta m\lambda)(\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}) - \frac{\eta}{t}[\boldsymbol{G}_t^{(j)}]^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t) + \frac{\eta}{t}\bar{\boldsymbol{G}}_t^\top(\bar{\boldsymbol{G}}_t(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0) - \boldsymbol{y}_t)\|_2 \\
=&\|(1-\eta m\lambda)(\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}) - \frac{\eta}{t}(\boldsymbol{G}_t^{(j)} - \bar{\boldsymbol{G}}_t)^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t) - \frac{\eta}{t}\bar{\boldsymbol{G}}_t^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \bar{\boldsymbol{G}}_t(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0))\|_2 \\
=&\|(1-\eta m\lambda)(\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}) - \frac{\eta}{t}(\boldsymbol{G}_t^{(j)} - \bar{\boldsymbol{G}}_t)^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t) \\
&\qquad - \frac{\eta}{t}\bar{\boldsymbol{G}}_t^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \bar{\boldsymbol{G}}_t(\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0) + \bar{\boldsymbol{G}}_t(\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}))\|_2 \\
=&\|(\boldsymbol{I} - \eta(m\lambda\boldsymbol{I} + \bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/t))(\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}) - \frac{\eta}{t}(\boldsymbol{G}_t^{(j)} - \bar{\boldsymbol{G}}_t)^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t) \\
&\qquad - \frac{\eta}{t}\bar{\boldsymbol{G}}_t^\top(\boldsymbol{f}_{gnn,t}^{(j)} - \bar{\boldsymbol{G}}_t(\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0))\|_2 \\
\le&\underbrace{\|(\boldsymbol{I} - \eta(m\lambda\boldsymbol{I} + \bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/t))\|_2\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2}_{I_1} + \underbrace{\frac{\eta}{t}\|\bar{\boldsymbol{G}}_t\|_2\|\boldsymbol{f}_{gnn,t}^{(j)} - \bar{\boldsymbol{G}}_t(\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)\|_2}_{I_2} \\
&\qquad + \underbrace{\frac{\eta}{t}\|\boldsymbol{G}_t^{(j)} - \bar{\boldsymbol{G}}_t\|_2\|\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t\|_2}_{I_3}
\end{aligned}
\tag{4.75}
$$

For I_1, due to $\bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/t \succcurlyeq \boldsymbol{0}$, we have

$$
I_1 = \|(\boldsymbol{I} - \eta(m\lambda\boldsymbol{I} + \bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/t))\|_2\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 \le (1-\eta m\lambda)\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 \tag{4.76}
$$

For I_2, by Lemma 4.7.10, set $\tau = \tilde{C}\sqrt{(R^2+\sigma_\epsilon^2)/m\lambda}$. Since $\tau \propto 1/\sqrt{m}$, it can be verified that technical condition (4.130) in Lemma 4.7.19 is satisfied when m is large. Then with probability at least $1-\delta_1 \in (0,1)$,

$$
I_2 = \frac{\eta}{t}\|\bar{\boldsymbol{G}}_t\|_2\|\boldsymbol{f}_{gnn,t}^{(j)} - \bar{\boldsymbol{G}}_t(\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)\|_2 \le \eta C_1(\tilde{C}\frac{R^2+\sigma_\epsilon^2}{m\lambda})^{2/3}L^{7/2}m\sqrt{\log(m)} \tag{4.77}
$$

For I_3, by Lemma 4.7.8 and Lemma 4.7.10, and Lemma 4.7.19, with probability at least

$1-\delta_2 \in (0,1)$,

$$I_3 = \frac{\eta}{t}\|\boldsymbol{G}_t^{(j)} - \bar{\boldsymbol{G}}_t\|_2 \|\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t\|_2 \leq \eta C_2 (\tilde{C}\frac{R^2+\sigma_\epsilon^2}{m\lambda})^{1/6} L^{7/2}\sqrt{m\log(m)}\sqrt{R^2+\sigma_\epsilon^2} \tag{4.78}$$

Put the upper bound for I_1, I_2., I_3 together and set $C_3 = (\lambda^{1/3}C_1 + C_2)\tilde{C} = \mathcal{O}(\lambda^{1/3})$, then we get,

$$\|\boldsymbol{\theta}_t^{(j+1)} - \tilde{\boldsymbol{\theta}}_t^{(j+1)}\|_2 \leq (1-\eta m\lambda)\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 + C_3\eta(R^2+\sigma_\epsilon^2)^{2/3}L^{7/2}m^{1/3}\lambda^{-1/6}\sqrt{\log(m)} \tag{4.79}$$

Therefore, there exists $m = poly(R, \sigma_\epsilon, \lambda, L)$ satisfies that $(R^2+\sigma_\epsilon^2)^{1/6}L^{7/2}\lambda^{1/3}\sqrt{\log(m)} \leq m^{1/6}$, which indicates

$$\|\boldsymbol{\theta}_t^{(j)} - \tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 \leq C_3(R^2+\sigma_\epsilon^2)^{2/3}L^{7/2}m^{-2/3}\lambda^{-1/6}\sqrt{\log(m)} \ \leq C_3\sqrt{\frac{R^2+\sigma_\epsilon^2}{m\lambda}} \tag{4.80}$$

□

Lemma 4.7.8 (Prediction Error Bound in Gradient Descent)**.** *Let $\{\boldsymbol{\theta}_t^{(j)}\}_{j=1}^J$ be the gradient descent update sequence of parameters of the optimization (4.6). Define $\boldsymbol{f}_{gnn,t}^{(j)} := [f_{\text{GNN}}(G_1;\boldsymbol{\theta}_t^{(j)}), ..., f_{\text{GNN}}(G_t;\boldsymbol{\theta}_t^{(j)})]^\top \in \mathbb{R}^{t\times 1}$. Assume τ is set such that $\|\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0\|_2 \leq \tau$ for all t and $\forall j \leq J$. Suppose $m \geq poly(L, \lambda^{-1}, \log(N/\delta))$ where $\delta \in (0,1)$ and learning rate $\eta \leq (\tilde{C}mL + m\lambda)^{-1}$ for some constant $\tilde{C}$, then with probability at least $1-\delta$,*

$$\|\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t\|_2 \leq C\sqrt{t(R^2+\sigma_\epsilon^2)} \tag{4.81}$$

where C is some constant which does not depend on m and t.

Proof. Define $\boldsymbol{f}_t(\boldsymbol{\theta})$ and $\boldsymbol{G}_t(\boldsymbol{\theta})$ as follow

$$\begin{aligned} \boldsymbol{f}_t(\boldsymbol{\theta}) &= [f_{\text{GNN}}(G_1;\boldsymbol{\theta}), ..., f_{\text{GNN}}(G_t;\boldsymbol{\theta})]^\top \in \mathbb{R}^{t\times 1} \\ \boldsymbol{G}_t(\boldsymbol{\theta}) &= [g(G_1;\boldsymbol{\theta}), ..., g(G_t;\boldsymbol{\theta})] \in \mathbb{R}^{p\times t} \end{aligned} \tag{4.82}$$

Also define $\mathcal{L}_t(\boldsymbol{\theta}) := \frac{1}{2t}\sum_{i=1}^{t}(f_{\text{GNN}}(G_i;\boldsymbol{\theta}) - y_i)^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2$ as the loss function in primary optimization. Note that $\mathcal{L}_t(\boldsymbol{\theta}) := \frac{1}{2t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2$ First notice that loss function $\mathcal{L}_t(\boldsymbol{\theta})$ is convex due to the strongly convexity of $\|\cdot\|_2^2/2$. We are going to use the following two-sided bound from strongly convexity in this proof:

$$\|\boldsymbol{y}\|_2^2/2 - \|\boldsymbol{x}\|_2^2/2 = \boldsymbol{x}^\top(\boldsymbol{y}-\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{x}\|_2^2 \tag{4.83}$$

By 1-strongly convexity of $\|\cdot\|_2^2/2$, we have

$$\begin{aligned}
&\mathcal{L}_t(\boldsymbol{\theta}') - \mathcal{L}_t(\boldsymbol{\theta}) \\
=&\frac{1}{2t}\left(\|\boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{y}_t\|_2^2 - \|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2^2\right) + \frac{m\lambda}{2}\left(\|\boldsymbol{\theta}'\|_2^2 - \|\boldsymbol{\theta}\|_2^2\right) \\
\leq&\frac{1}{t}\left((\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top(\boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{f}_t(\boldsymbol{\theta})) + \frac{1}{2}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{f}_t(\boldsymbol{\theta}')\|_2^2\right) + m\lambda\left(\boldsymbol{\theta}^\top(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \frac{1}{2}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_2^2\right).
\end{aligned} \tag{4.84}$$

Define $\boldsymbol{e}_t := \boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta})$. By Lemma 4.7.10, with probability at least $1-\delta_1 \in (0,1)$

$$\begin{aligned}
&\mathcal{L}_t(\boldsymbol{\theta}') - \mathcal{L}_t(\boldsymbol{\theta}) \\
\leq&\frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top(\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \boldsymbol{e}_t) + \frac{1}{2t}\|\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \boldsymbol{e}_t\|_2^2 \\
&\quad + m\lambda\left(\boldsymbol{\theta}^\top(\boldsymbol{\theta}-\boldsymbol{\theta}) + \frac{1}{2}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_2^2\right) \\
=&\frac{1}{t}[\boldsymbol{G}_t(\boldsymbol{\theta})(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t) + m\lambda\boldsymbol{\theta}]^\top(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top\boldsymbol{e}_t + \frac{1}{2t}\|\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \boldsymbol{e}_t\|_2^2 \\
&\quad + \frac{m\lambda}{2}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_2^2 \\
=&\nabla\mathcal{L}_t(\boldsymbol{\theta})^\top(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top\boldsymbol{e}_t + \frac{1}{2t}\|\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \boldsymbol{e}_t\|_2^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_2^2 \\
\leq&\nabla\mathcal{L}_t(\boldsymbol{\theta})^\top(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 + \frac{1}{t}\|\boldsymbol{G}_t(\boldsymbol{\theta})\|_2^2\|\boldsymbol{\theta}'-\boldsymbol{\theta}\|_2^2 + \frac{1}{t}\|\boldsymbol{e}_t\|_2^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|_2^2 \\
\leq&\nabla\mathcal{L}_t(\boldsymbol{\theta})^\top(\boldsymbol{\theta}'-\boldsymbol{\theta}) + \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 + \frac{1}{t}\|\boldsymbol{e}_t\|_2^2 + (C_1^2 mL + m\lambda/2)\|\boldsymbol{\theta}'-\boldsymbol{\theta}\|_2^2
\end{aligned} \tag{4.85}$$

where the last inequality is from Lemma 4.7.10. Similarly by 1-strongly convexity of $\|\cdot\|_2^2/2$, we also investigate the lower bound:

$$\begin{aligned}\mathcal{L}_t(\boldsymbol{\theta}') - \mathcal{L}_t(\boldsymbol{\theta}) \geq & \frac{1}{t}\left((\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top(\boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{f}_t(\boldsymbol{\theta})) + \frac{1}{2}\|\boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{f}_t(\boldsymbol{\theta}_)\|_2^2\right) \\ & + m\lambda\left(\boldsymbol{\theta}^\top(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{1}{2}\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|_2^2\right)\end{aligned} \tag{4.86}$$

Using $\boldsymbol{e}_t := \boldsymbol{f}_t(\boldsymbol{\theta}') - \boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}' - \boldsymbol{\theta})$, we obtain

$$\begin{aligned}\mathcal{L}_t(\boldsymbol{\theta}') - \mathcal{L}_t(\boldsymbol{\theta}) \geq & \frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top(\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \boldsymbol{e}_t) + m\lambda\boldsymbol{\theta}^\top(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{m\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 \\ = & \frac{1}{t}[\boldsymbol{G}_t(\boldsymbol{\theta})(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t) + m\lambda\boldsymbol{\theta}]^\top(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top\boldsymbol{e}_t + \frac{m\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2\end{aligned} \tag{4.87}$$

Then using $\nabla\mathcal{L}_t(\boldsymbol{\theta}) = \boldsymbol{G}_t(\boldsymbol{\theta})(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t) + m\lambda\boldsymbol{\theta}$, we have

$$\begin{aligned}\mathcal{L}_t(\boldsymbol{\theta}') - \mathcal{L}_t(\boldsymbol{\theta}) \geq & \nabla\mathcal{L}_t(\boldsymbol{\theta})^\top(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{1}{t}(\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t)^\top\boldsymbol{e}_t + \frac{m\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 \\ \geq & \nabla\mathcal{L}_t(\boldsymbol{\theta})^\top(\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{m\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 - \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 \\ \geq & -\frac{\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_2^2}{2m\lambda} - \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 \quad \text{(by Lemma 4.7.13)}\end{aligned} \tag{4.88}$$

Now recall the update step $\boldsymbol{\theta}_t^{(j+1)} = \boldsymbol{\theta}_t^{(j)} - \eta\nabla\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)})$ and combine the above upper and lower bounds,

$$\begin{aligned}& \mathcal{L}_t(\boldsymbol{\theta} - \eta\nabla\mathcal{L}_t(\boldsymbol{\theta})) - \mathcal{L}_t(\boldsymbol{\theta}) \\ \leq & -\eta\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_2^2 + \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 + \frac{1}{t}\|\boldsymbol{e}_t\|_2^2 + \eta^2(C_1^2mL + m\lambda/2)\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_2^2 \\ = & -\eta\left(1 - \frac{\eta}{2}(2C_1^2mL + m\lambda)\right)\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_2^2 + \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 + \frac{1}{t}\|\boldsymbol{e}_t\|_2^2 \\ \leq & -\frac{\eta}{2}\|\nabla\mathcal{L}_t(\boldsymbol{\theta})\|_2^2 + \frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2 + \frac{1}{t}\|\boldsymbol{e}_t\|_2^2 \quad \text{(by choice of } \eta)\end{aligned} \tag{4.89}$$

and we can further apply (4.88),

$$
\begin{aligned}
&\mathcal{L}_t(\boldsymbol{\theta}-\eta\nabla\mathcal{L}_t(\boldsymbol{\theta}))-\mathcal{L}_t(\boldsymbol{\theta})\\
\leq&\eta m\lambda\bigg(\mathcal{L}_t(\boldsymbol{\theta}')-\mathcal{L}_t(\boldsymbol{\theta})+\frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2\bigg)+\frac{1}{t}\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2\|\boldsymbol{e}_t\|_2+\frac{1}{t}\|\boldsymbol{e}_t\|_2^2\\
\leq&\eta m\lambda\bigg(\mathcal{L}_t(\boldsymbol{\theta}')-\mathcal{L}_t(\boldsymbol{\theta})+\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2^2/8t+2\|\boldsymbol{e}_t\|_2^2/t\bigg)+\\
&\qquad\frac{1}{t}(\frac{\eta m\lambda}{8}\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2^2+\frac{2\|\boldsymbol{e}_t\|_2^2}{\eta m\lambda})+\frac{1}{t}\|\boldsymbol{e}_t\|_2^2 \qquad (4.90)\\
=&\eta m\lambda(\mathcal{L}_t(\boldsymbol{\theta}')-\mathcal{L}_t(\boldsymbol{\theta}))+\frac{\eta m\lambda}{4t}\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2^2+(\frac{2\eta m\lambda}{t}+\frac{2}{\eta m\lambda t}+\frac{1}{t})\|\boldsymbol{e}_t\|_2^2\\
\leq&\eta m\lambda(\mathcal{L}_t(\boldsymbol{\theta}')-\mathcal{L}_t(\boldsymbol{\theta}))+\eta m\lambda\mathcal{L}_t(\boldsymbol{\theta})/2+(\frac{2\eta m\lambda}{t}+\frac{2}{\eta m\lambda t}+\frac{1}{t})\|\boldsymbol{e}_t\|_2^2\\
=&\eta m\lambda(\mathcal{L}_t(\boldsymbol{\theta}')-\mathcal{L}_t(\boldsymbol{\theta})/2)+(\frac{2\eta m\lambda}{t}+\frac{2}{\eta m\lambda t}+\frac{1}{t})\|\boldsymbol{e}_t\|_2^2
\end{aligned}
$$

where the last inequality is because $\|\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{y}_t\|_2^2\leq 2t\mathcal{L}(\boldsymbol{\theta})$. For $\|\boldsymbol{e}_t\|_2^2$, by Lemma 4.7.19, with probability at least $1-\delta_2\in(0,1)$ for some constant C_2, we have

$$
\begin{aligned}
\|\boldsymbol{e}_t\|_2&=\|\boldsymbol{f}_t(\boldsymbol{\theta}')-\boldsymbol{f}_t(\boldsymbol{\theta})-\boldsymbol{G}_t^\top(\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta})\|_2\\
&\leq\sqrt{t}\max_{i\in[t]}|f_{\mathrm{GNN}}(G_i;\boldsymbol{\theta}')-f_{\mathrm{GNN}}(G_i;\boldsymbol{\theta})+g^\top(G_i;\boldsymbol{\theta})(\boldsymbol{\theta}'-\boldsymbol{\theta})|\\
&\leq\frac{\sqrt{t}}{N}\max_{i\in[t]}\sum_{j\in\mathcal{V}(G_i)}|f_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}')-f_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta})+g_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta})^\top(\boldsymbol{\theta}'-\boldsymbol{\theta})| \qquad (4.91)\\
&\leq C_2\tau^{4/3}L^3\sqrt{tm\log(m)}
\end{aligned}
$$

where $\mathcal{V}(G)$ as vertice set of a graph G. Moreove, by Lemma 4.7.16, we have the high probability upper bound for $\frac{1}{t}\|\boldsymbol{y}_t\|_2^2$: with probability at least $1-\delta_3\in(0,1)$ and some constant C_3 depends on δ_3,

$$
\frac{1}{t}\|\boldsymbol{y}_t\|_2^2\leq\frac{1}{t}(tR^2+\|\boldsymbol{\epsilon}_t\|_2^2+2\sqrt{t}R\|\boldsymbol{\epsilon}_t\|_2)\leq C_3(\sigma_\epsilon^2+R^2) \qquad (4.92)
$$

Then let $\boldsymbol{\theta}'=\boldsymbol{\theta}_0$ and plug in $\boldsymbol{\theta}_t^{(j+1)}$ and $\boldsymbol{\theta}_t^{(j)}$ in (4.90), by Lemma 4.7.9, with probability at

least $1-\delta_4$,

$$
\begin{aligned}
&\mathcal{L}_t(\boldsymbol{\theta}_t^{(j+1)}) - \mathcal{L}_t(\boldsymbol{\theta}_0) \\
&\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + \frac{\eta m\lambda}{2}\mathcal{L}_t(\boldsymbol{\theta}_0) + (\frac{2\eta m\lambda}{t} + \frac{2}{\eta m\lambda t} + \frac{1}{t})\|\boldsymbol{e}_t\|_2^2 \\
&\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + \frac{\eta m\lambda}{2}(\frac{1}{t}\|\boldsymbol{y}_t\|_2^2 + m\lambda\|\boldsymbol{\theta}_0\|_2^2) \\
&\quad + (2\eta m\lambda + 2/\eta m\lambda + 1)C_2^2\tau^{8/3}L^6 m\log(m) \quad \text{(by (4.92))} \\
&\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + \frac{\eta m\lambda}{2}(C_3(\sigma_\epsilon^2 + R^2) + m\lambda\|\boldsymbol{\theta}_0\|_2^2) \\
&\quad + \frac{5}{\eta m\lambda}C_2^2\tau^{8/3}L^6 m\log(m) \quad \text{(by (4.91) and } \eta m\lambda \le 1) \\
&\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + C_4\eta m\lambda(\sigma_\epsilon^2 + R^2) + \frac{5}{\eta m\lambda}C_2^2\tau^{8/3}L^6 m\log(m) \\
&\quad \text{(by Lemma 4.7.9)}
\end{aligned}
\tag{4.93}
$$

Now we further set $\tau = \tilde{C}\sqrt{\frac{\sigma_\epsilon^2+R^2}{m\lambda}}$ and the upper bound for $\mathcal{L}_t(\boldsymbol{\theta}_t^{(j+1)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)$ is

$$
\begin{aligned}
\mathcal{L}_t(\boldsymbol{\theta}_t^{(j+1)}) - \mathcal{L}_t(\boldsymbol{\theta}_0) &\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + C_4\eta m\lambda(\sigma_\epsilon^2 + R^2) \\
&\quad + \frac{5}{\eta m\lambda}\tilde{C}^2C_2^2(\sigma_\epsilon^2 + R^2)\tau^{2/3}\lambda^{-1}L^6\log(m) \quad \text{(by } \tau = \tilde{C}\sqrt{\frac{\sigma_\epsilon^2+R^2}{m\lambda}}) \\
&\le (1-\eta m\lambda/2)(\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) - \mathcal{L}_t(\boldsymbol{\theta}_0)) + C_4\eta m\lambda(\sigma_\epsilon^2 + R^2) + C_5\eta m\lambda(\sigma_\epsilon^2 + R^2) \\
&\quad \text{(by choice of } \tau \text{ in Lemma 4.7.19)}
\end{aligned}
\tag{4.94}
$$

where C_4 is a constant depends on δ_3 and δ_4 and C_5 depends on δ_2, δ_3 and δ_4. Then by recursion,

$$
\mathcal{L}_t(\boldsymbol{\theta}_t^{(j+1)}) - \mathcal{L}_t(\boldsymbol{\theta}_0) \le \frac{C_6\eta m\lambda(\sigma_\epsilon^2 + R^2)}{\eta m\lambda/2} = \tilde{C}_6(\sigma_\epsilon^2 + R^2) \tag{4.95}
$$

where $C_6 = C_4 + C_5$ and $\tilde{C}_6 = 2C_6$. Recall that $\|\boldsymbol{f}_t(\boldsymbol{\theta}) - \boldsymbol{y}_t\|_2^2 = 2t\mathcal{L}_t(\boldsymbol{\theta}) - \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \le 2t\mathcal{L}_t(\boldsymbol{\theta})$,

with some constant C_7 derived from C_6 and C_4, then we have

$$\begin{aligned}\|\boldsymbol{f}_{gnn,t}^{(j)} - \boldsymbol{y}_t\|_2^2 \leq 2t\mathcal{L}_t(\boldsymbol{\theta}_t^{(j)}) &\leq 2t\tilde{C}_6(\sigma_\epsilon^2 + R^2) + 2t\mathcal{L}_t(\boldsymbol{\theta}_0) \\ &= 2t\tilde{C}_6(\sigma_\epsilon^2 + R^2) + 2t(\frac{1}{t}\|\boldsymbol{y}_t\|_2^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}_0\|_2^2) \\ &\leq C_7 t(\sigma_\epsilon^2 + R^2) \quad \text{(by Lemma 4.7.9)}\end{aligned} \tag{4.96}$$

which implies our result by setting $\delta_1 = \delta_2 = \delta_3 = \delta_4 = \delta/4$ where $\delta \in (0, 1)$ is arbitrary small.

□

Lemma 4.7.9 (Parameter Bound for Proximal Optimization)**.** *Let* $\{\tilde{\boldsymbol{\theta}}_t^{(j)}\}_{j=1}^J$ *be the gradient descent update sequence of parameters of the following optimization,*

$$\min_{\boldsymbol{\theta}} \frac{1}{2t}\sum_{i=1}^{t}(\langle g(G_i;\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0\rangle - y_i)^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \tag{4.97}$$

Then if $m \geq poly(L, \lambda^{-1}, \log(N/\delta))$ *and learning rate* $\eta \leq (\tilde{C}mL + m\lambda)^{-1}$ *for some constant* $\tilde{C}$*. Then for some constant* C *and for any* $\forall t \in [T]$ *and* $\forall j \in [J]$*, with probability at least* $1 - \delta \in (0, 1)$*,*

$$\begin{aligned}\|\tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 &\leq C\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} \\ \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0\|_2 &\leq C\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} \\ \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t\boldsymbol{y}_t/m\|_2 &\leq C(2 - \eta m\lambda)^j\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}}\end{aligned} \tag{4.98}$$

for some constant C *which is independent of* m *and* t*.*

Proof. Denote $\mathcal{L}_t(\boldsymbol{\theta}) := \frac{1}{2t}\sum_{i=1}^{t}(\langle g(G_i;\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0\rangle - y_i)^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}\|_2^2$ as the loss function in our proximal optimization. By Lemma 4.7.10, with probability at least $1 - \delta_1 \in (0, 1)$ the

Hessian of $\mathcal{L}_t(\boldsymbol{\theta})$ satisfies:

$$\mathbf{0} \prec \nabla^2 \mathcal{L}_t = \bar{\boldsymbol{G}}_t \bar{\boldsymbol{G}}_t^\top / t + m\lambda \boldsymbol{I} \preccurlyeq (\|\bar{\boldsymbol{G}}_t\|_F^2 / t + m\lambda)\boldsymbol{I} \preccurlyeq (C_1^2 mL + m\lambda)\boldsymbol{I} \tag{4.99}$$

which reveals that $\mathcal{L}_t$ is strongly convex and $(C_1^2 mL + m\lambda)$-smooth. Thus if $\eta \leq (C_1^2 mL + m\lambda)^{-1}$, $\mathcal{L}_t$ is a monotonically decreasing function:

$$\frac{1}{2t}\|\bar{\boldsymbol{G}}_t^\top(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0) - \boldsymbol{y}_t\|_2^2 + \frac{m\lambda}{2}\|\tilde{\boldsymbol{\theta}}_t^{(j)}\|_2^2 \leq \frac{1}{2t}\|\boldsymbol{y}_t\|_2^2 + \frac{m\lambda}{2}\|\boldsymbol{\theta}_0\|_2^2 \tag{4.100}$$

which indicates

$$\begin{aligned} \|\tilde{\boldsymbol{\theta}}_t^{(j)}\|_2^2 \leq & \frac{1}{tm\lambda}\|\boldsymbol{y}_t\|_2^2 + \|\boldsymbol{\theta}_0\|_2^2 \\ \leq & \frac{1}{tm\lambda}(\|\boldsymbol{\mu}_t\|_2^2 + \|\boldsymbol{\epsilon}_t\|_2^2 + 2\|\boldsymbol{\mu}_t\|_2\|\boldsymbol{\epsilon}_t\|_2) + \|\boldsymbol{\theta}_0\|_2^2 \end{aligned} \tag{4.101}$$

Note that the proximal optimization is optimization for ridge regression which has the closed form solution:

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t \boldsymbol{y}_t / m \tag{4.102}$$

and $\tilde{\boldsymbol{\theta}}_t^{(j)}$ converges to $\boldsymbol{\theta}^*$ with the following rate:

$$\begin{aligned} & \|\tilde{\boldsymbol{\theta}}_t^{(j+1)} - \boldsymbol{\theta}^*\|_2^2 \\ = & \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \eta\nabla\mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)}) - \boldsymbol{\theta}^*\|_2^2 \\ = & \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 + \eta^2\|\nabla\mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)})\|_2^2 - 2\eta(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*)^\top\nabla\mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)}) \\ \leq & \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 + \eta^2(C_1^2 mL + m\lambda)^2\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 - 2\eta(\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*)^\top\nabla\mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)}) \quad \text{(by smoothness)} \\ \leq & \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 + \eta^2(C_1^2 mL + m\lambda)^2\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 + 2\eta(\mathcal{L}(\boldsymbol{\theta}^*) - \mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)})) \quad \text{(by convexity)} \\ \leq & 2\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 + 2\eta(\mathcal{L}(\boldsymbol{\theta}^*) - \mathcal{L}(\tilde{\boldsymbol{\theta}}_t^{(j)})) \quad (\text{by } \eta \leq (C_1^2 mL + m\lambda)^{-1}) \\ \leq & 2\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 - \eta m\lambda\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 \quad (\text{by } m\lambda\text{-strongly convexity}) \\ = & (2 - \eta m\lambda)\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}^*\|_2^2 \end{aligned} \tag{4.103}$$

Therefore,

$$\begin{aligned}\|\tilde{\boldsymbol{\theta}}_t^{(j+1)} - \boldsymbol{\theta}^*\|_2^2 &\leq (2-\eta m\lambda)^j \|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|_2^2 \\ &\leq (2-\eta m\lambda)^j \frac{2}{m\lambda}(\mathcal{L}(\boldsymbol{\theta}_0)) - \mathcal{L}(\boldsymbol{\theta}^*)) \quad \text{(by } m\lambda\text{-strongly convexity)} \\ &\leq (2-\eta m\lambda)^j \frac{2}{m\lambda}\mathcal{L}(\boldsymbol{\theta}_0) \\ &= (2-\eta m\lambda)^j \left(\frac{1}{tm\lambda}\|\boldsymbol{y}_t\|_2^2 + \|\boldsymbol{\theta}_0\|_2^2\right)\end{aligned} \tag{4.104}$$

Then combine with Lemma 4.7.16 and $\|\boldsymbol{\mu}_t\|_2 \leq \sqrt{t}\|\mu\|_{\mathcal{H}} \leq \sqrt{t}R$, we have that with probability at least $1-\delta_2 \in (0,1)$,

$$\frac{1}{tm\lambda}\|\boldsymbol{y}_t\|_2^2 \leq \frac{1}{tm\lambda}(tR^2 + \|\boldsymbol{\epsilon}_t\|_2^2 + 2\sqrt{t}R\|\boldsymbol{\epsilon}_t\|_2) \leq \tilde{C}_1(\sigma_\epsilon^2 + R^2)/m\lambda \tag{4.105}$$

where $\tilde{C}_1$ is some constant depends on δ_2. Therefore, for any $\delta \in (0,1)$, set $\delta_1 = \delta_2 = \delta/2$, with probability at least $1-\delta_2$,

$$\begin{aligned}\|\tilde{\boldsymbol{\theta}}_t^{(j)}\|_2 &\leq \tilde{C}_2\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} \\ \|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0\|_2 &\leq \tilde{C}_2\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}}\end{aligned} \tag{4.106}$$

and

$$\|\tilde{\boldsymbol{\theta}}_t^{(j)} - \boldsymbol{\theta}_0 - \bar{\boldsymbol{U}}_t^{-1}\bar{\boldsymbol{G}}_t\boldsymbol{y}_t/m\|_2 \leq (2-\eta m\lambda)^j\tilde{C}_2\sqrt{\frac{\sigma_\epsilon^2 + R^2}{m\lambda}} \tag{4.107}$$

where $\tilde{C}_2$ is some constant depends on δ_2 and $\|\boldsymbol{\theta}_0\|_2$.

□

Lemma 4.7.10 (Gradient Descent Norm Bound)**.** *Define* $\boldsymbol{G}_t^{(j)} := [g(G_1;\boldsymbol{\theta}_t^{(j)}), ..., g(G_t;\boldsymbol{\theta}_t^{(j)}))]$ *from* $\mathbb{R}^{p\times t}$ *for the gradients in the* j*-th updates in GNN training (optimization of (4.6)) at round* t*. Also define* $\boldsymbol{f}_{gnn,t}^{(j)} := [f_{\text{GNN}}(G_1;\boldsymbol{\theta}_t^{(j)}), ..., f_{\text{GNN}}(G_t;\boldsymbol{\theta}_t^{(j)})]^\top \in \mathbb{R}^{t\times 1}$*. Assume* τ *is set such that* $\|\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0\|_2 \leq \tau$ *for all* t *and* $\forall j \leq J$*. Suppose* $m \geq poly(L, \lambda^{-1}, \log(N/\delta))$ *where*

$\delta \in (0,1)$, *then with probability at least* $1-\delta$,

$$\begin{aligned}
\|\bar{\boldsymbol{G}}_t\|_F &\le C_1\sqrt{tmL} \\
\|\boldsymbol{G}_t^{(j)}\|_F &\le C_1\sqrt{tmL} \\
\|\bar{\boldsymbol{G}}_t - \boldsymbol{G}_t^{(j)}\|_F &\le C_2\tau^{1/3}L^{7/2}\sqrt{tm\log(m)} \\
\|\boldsymbol{f}_{gnn,t}^{(j)} - (\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)^\top \bar{\boldsymbol{G}}_t\|_2 &\le C_3\tau^{4/3}L^3\sqrt{tm\log(m)}
\end{aligned} \tag{4.108}$$

for some constant C_1, C_2, C_3 *which does not depend on* m *and* t.

Proof. From Lemma 4.7.19, we can bounding the $\|g(G;\boldsymbol{\theta}_0)\|_2$ with probability at least $1-\delta \in (0,1)$, which provides the high probability upper bound for the Frobenius norm of $\bar{\boldsymbol{G}}_t$:

$$\|\bar{\boldsymbol{G}}_t\|_F \le \sqrt{t}\max_{i\in[t]}\|g(G_i;\boldsymbol{\theta}_0)\|_2 \le \frac{\sqrt{t}}{N}\max_{i\in[t]}\sum_{j\in\mathcal{V}(G_i)}\|g_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_0)\|_2 \le C_1\sqrt{tmL} \tag{4.109}$$

and the high probability upper bound for the Frobenius norm of $\boldsymbol{G}_t^{(j)}$:

$$\|\boldsymbol{G}_t^{(j)}\|_F \le \sqrt{t}\max_{i\in[t]}\|g(G_i;\boldsymbol{\theta}_t^{(j)})\|_2 \le \frac{\sqrt{t}}{N}\max_{i\in[t]}\sum_{j\in\mathcal{V}(G_i)}\|g_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_t^{(j)})\|_2 \le C_1\sqrt{tmL} \tag{4.110}$$

For the gradients difference, by Lemma 4.7.19, with probability at least $1-\delta$,

$$\begin{aligned}
\|\bar{\boldsymbol{G}}_t - \boldsymbol{G}_t^{(j)}\|_F &\le \sqrt{t}\max_{i\in[t]}\|g(G_i;\boldsymbol{\theta}_0) - g(G_i;\boldsymbol{\theta}_t^{(j)})\|_2 \\
&\le \frac{\sqrt{t}}{N}\max_{i\in[t]}\sum_{j\in\mathcal{V}(G_i)}\|g_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_0) - g_{\mathrm{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_t^{(j)})\|_2 \\
&\le C_2\tau^{1/3}L^{7/2}\sqrt{tm\log(m)}
\end{aligned} \tag{4.111}$$

The last norm for difference between the GNN prediction and linearized prediction is bounded

due to Lemma 4.7.19, with probability at least $1-\delta$,

$$\begin{aligned}\|\boldsymbol{f}_{gnn,t}^{(j)} - (\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)^\top \boldsymbol{G}_t^{(j)}\|_2 &\leq \sqrt{t}\max_{i\in[t]} |f_{\text{GNN}}(G_i;\boldsymbol{\theta}_t^{(j)}) - (\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)^\top g(G_i;\boldsymbol{\theta}_0)| \\ &\leq \frac{\sqrt{t}}{N}\max_{i\in[t]} \sum_{j\in\mathcal{V}(G_i)} |f_{\text{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_t^{(j)}) - (\boldsymbol{\theta}_t^{(j)} - \boldsymbol{\theta}_0)^\top g_{\text{MLP}}(\boldsymbol{h}_j;\boldsymbol{\theta}_0)| \\ &\leq C_3\tau^{4/3}L^3\sqrt{tm\log(m)}\end{aligned} \tag{4.112}$$

□

4.7.2.4 Lemmas for GNTK

Lemma 4.7.11 (Approximation from GNTK)**.** *Set $\delta \in (0,1)$ and*

$$m = \Omega(L^{10}T^4|\mathcal{G}|^6\rho_{min}^{-4}\log(LN^2|\mathcal{G}|^2/\delta)).$$

Then with probability at least $1-\delta$,

(i) (Approximate Linearized Nerual Network) $\exists\boldsymbol{\theta}^*$ such that, for $\forall G \in \mathcal{G}$

$$\begin{aligned}\mu(G) &= \langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}^*\rangle \\ \sqrt{m}\|\boldsymbol{\theta}^*\|_2 &\leq \sqrt{2}R\end{aligned} \tag{4.113}$$

(ii) (Spectral Bound for Uncertainty Matrix $\bar{\boldsymbol{U}}_t$ by GNTK)

$$\begin{aligned}\lambda_{max}(\bar{\boldsymbol{U}}_t) &\leq \lambda + \frac{3}{2}\rho_{\max} \\ \log\det(\lambda^{-1}\bar{\boldsymbol{U}}_t) &\leq \log\det(\boldsymbol{I}_{|\mathcal{G}|} + \lambda^{-1}t\boldsymbol{K}) + 1\end{aligned} \tag{4.114}$$

Proof. In this proof, set $\delta_1 = \delta_2 = \delta/2$ where $\delta \in (0,1)$ is an arbitrary real value. Recall the definition of the true reward function $\mu : \mathcal{G} \to \mathbb{R}$ and the GNTK matrix $\boldsymbol{K} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$. We further define the vector of function values $\boldsymbol{\mu} \in \mathbb{R}^{|\mathcal{G}|\times 1}$ as well as the gradient matrix

$\bar{\boldsymbol{G}} \in \mathbb{R}^{p\times|\mathcal{G}|}$ on initialization $\boldsymbol{\theta}_0$.

$$\begin{aligned}[\boldsymbol{K}]_{ij} &= k(G^i, G^j) \quad \forall G^i, G^j \in \mathcal{G} \\ [\boldsymbol{\mu}]_i &= \mu(G^i) \quad \forall G^i \in \mathcal{G} \\ \bar{\boldsymbol{G}}_{*i} &= g(G^i; \boldsymbol{\theta}_0)\end{aligned} \tag{4.115}$$

Proof for (i): By the connection between GNTK and NTK,

$$\begin{aligned}&\|\boldsymbol{K} - \bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\|_F \\ =&\sqrt{\sum_{i=1}^{|\mathcal{G}|}\sum_{j=1}^{|\mathcal{G}|}(k(G^i, G^j) - g^\top(G^i;\boldsymbol{\theta}_0)g(G^j;\boldsymbol{\theta}_0)/m)^2} \\ =&\sqrt{\sum_{i=1}^{|\mathcal{G}|}\sum_{j=1}^{|\mathcal{G}|}\left(\frac{1}{N^2}\sum_{u\in\mathcal{V}(G^i)}\sum_{v\in\mathcal{V}_{G^j}}(k_{\text{MLP}}(\boldsymbol{h}_u^{G^i}, \boldsymbol{h}_v^{G^j}) - g_{\text{MLP}}^\top(\boldsymbol{h}_u^{G^i};\boldsymbol{\theta}_0)g_{\text{MLP}}(\boldsymbol{h}_v^{G^j};\boldsymbol{\theta}_0)/m)\right)^2} \\ \leq&\sqrt{\sum_{i=1}^{|\mathcal{G}|}\sum_{j=1}^{|\mathcal{G}|}\sum_{u\in\mathcal{V}(G^i)}\sum_{v\in\mathcal{V}(G^j)}(k_{\text{MLP}}(\boldsymbol{h}_u^{G^i}, \boldsymbol{h}_v^{G^j}) - g_{\text{MLP}}^\top(\boldsymbol{h}_u^{G^i};\boldsymbol{\theta}_0)g_{\text{MLP}}(\boldsymbol{h}_v^{G^j};\boldsymbol{\theta}_0)/m)^2}\end{aligned} \tag{4.116}$$

where $\mathcal{V}_G$ denotes the vertice set of a graph G. By Lemma 4.7.18, when

$$m = \Omega(L^{10}N^4|\mathcal{G}|^4\rho_{min}^{-4}\log(LN^2|\mathcal{G}|^2/\delta_1),$$

then with probability at least $1 - \delta_1/(N^2|\mathcal{G}|^2)$,

$$|k_{\text{MLP}}(\boldsymbol{h}_u^{G^i}, \boldsymbol{h}_v^{G^j}) - g_{\text{MLP}}^\top(\boldsymbol{h}_u^{G^i};\boldsymbol{\theta}_0)g_{\text{MLP}}(\boldsymbol{h}_v^{G^j};\boldsymbol{\theta}_0)/m| \leq \frac{\rho_{min}}{2N|\mathcal{G}|}$$

. Then apply union bound over all pairs $(\boldsymbol{h}_u^{G^i}, \boldsymbol{h}_v^{G^j})$, the following holds with probability at least $1 - \delta_1$,

$$\|\boldsymbol{K} - \bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\|_F \leq \rho_{min}/2 \tag{4.117}$$

which shows that

$$\begin{aligned}\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m &\succcurlyeq \boldsymbol{K} - \|\boldsymbol{K} - \bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\|_2\boldsymbol{I}_{|\mathcal{G}|} \\ &\succcurlyeq \boldsymbol{K} - \|\boldsymbol{K} - \bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\|_F\boldsymbol{I}_{|\mathcal{G}|} \\ &\succcurlyeq \boldsymbol{K} - \frac{\rho_{min}}{2}\boldsymbol{I}_{|\mathcal{G}|} \\ &\succcurlyeq \boldsymbol{K}/2 \succ \boldsymbol{0}\end{aligned} \tag{4.118}$$

Suppose $\bar{\boldsymbol{G}} = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{Q}^\top$ is the decomposition of $\bar{\boldsymbol{G}}$ where $\boldsymbol{P} \in \mathbb{R}^{p\times|\mathcal{G}|}$, $\boldsymbol{Q} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ are unitary and $\boldsymbol{\Lambda} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$. By (4.118), we know $\boldsymbol{\Lambda} \succ \boldsymbol{0}$ with probability at least $1-\delta_1$. Now denote $\boldsymbol{\theta}^* = \boldsymbol{P}\boldsymbol{\Lambda}^{-1}\boldsymbol{Q}^\top\boldsymbol{\mu}$ and it satisfies

$$\begin{aligned}&\bar{\boldsymbol{G}}^\top\boldsymbol{\theta}^* = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{P}^\top\boldsymbol{P}\boldsymbol{\Lambda}^{-1}\boldsymbol{Q}^\top\boldsymbol{\mu} = \boldsymbol{\mu} \\ \Rightarrow &\mu(G) = \langle g(G;\boldsymbol{\theta}_0), \boldsymbol{\theta}^*\rangle \quad \forall G \in \mathcal{G}\end{aligned} \tag{4.119}$$

Moreover, the norm of $\boldsymbol{\theta}^*$ is also bounded:

$$\|\boldsymbol{\theta}^*\|_2^2 = \boldsymbol{\mu}^\top\boldsymbol{Q}\boldsymbol{\Lambda}^{-2}\boldsymbol{Q}^\top\boldsymbol{\mu} = \boldsymbol{\mu}^\top(\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}})^{-1}\boldsymbol{\mu} \leq \frac{2}{m}\boldsymbol{\mu}^\top\boldsymbol{K}^{-1}\boldsymbol{\mu} \leq \frac{2R^2}{m} \tag{4.120}$$

which completes our proof for (i).

Proof for (ii): From the definition of $\bar{\boldsymbol{G}}_t$, we have

$$
\begin{aligned}
&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+\lambda^{-1}\bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/m)\\
=&\log\det\left(\boldsymbol{I}_{|\mathcal{G}|}+\sum_{i=1}^{t}g(G_i;\boldsymbol{\theta}_0)g^\top(G_i;\boldsymbol{\theta}_0)/(m\lambda)\right)\\
\le&\log\det\left(\boldsymbol{I}_{|\mathcal{G}|}+t\sum_{G\in\cup_{i=1}^{t}\mathcal{G}_i}g(G;\boldsymbol{\theta}_0)g^\top(G;\boldsymbol{\theta}_0)/(m\lambda)\right)\\
\le&\log\det\left(\boldsymbol{I}_{|\mathcal{G}|}+t\sum_{G\in\mathcal{G}}g(G;\boldsymbol{\theta}_0)g^\top(G;\boldsymbol{\theta}_0)/(m\lambda)\right)\quad(\text{by }\mathcal{G}_t\in\mathcal{G}\text{ for }\forall t\in[T])\\
=&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/(m\lambda))\\
=&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda+t(\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K})/\lambda)\\
\le&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+\langle(\boldsymbol{I}+t\boldsymbol{K}/\lambda)^{-1},t(\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K})/\lambda\rangle_F\quad(\text{by concavity of }\log\det(\cdot))\\
\le&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+\|(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)^{-1}\|_F\|t(\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K})/\lambda\|_F\\
\le&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+t\sqrt{|\mathcal{G}|}\|(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)^{-1}\|_2\|\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K}\|_F/\lambda\\
=&\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+\sqrt{|\mathcal{G}|}(\lambda/t+\rho_{min})^{-1}\|\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K}\|_F
\end{aligned}
\tag{4.121}
$$

By Lemma 4.7.18, when $m=\Omega(L^{10}N^4|\mathcal{G}|^6\rho_{min}^{-4}\log(LN^2|\mathcal{G}|^2/\delta_2)$, then with probability at least $1-\delta_2/(N^2|\mathcal{G}|^2)$, $|k_{\text{MLP}}(\boldsymbol{h}_u^{G^i},\boldsymbol{h}_v^{G^j})-g_{\text{MLP}}^\top(\boldsymbol{h}_u^{G^i};\boldsymbol{\theta}_0)g_{\text{MLP}}(\boldsymbol{h}_v^{G^j};\boldsymbol{\theta}_0)/m|\le\frac{\rho_{min}}{N|\mathcal{G}|^{3/2}}$. Then apply union bound over all pairs $(\boldsymbol{h}_u^{G^i},\boldsymbol{h}_v^{G^j})$, with probability at least $1-\delta_2$, $\|\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K}\|_F\le\frac{\rho_{min}}{\sqrt{|\mathcal{G}|}}$, which indicates that

$$
\begin{aligned}
\log\det(\boldsymbol{I}_{|\mathcal{G}|}+\lambda^{-1}\bar{\boldsymbol{G}}_t^\top\bar{\boldsymbol{G}}_t/m)&\le\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+\sqrt{|\mathcal{G}|}(\lambda/t+\rho_{min})^{-1}\|\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m-\boldsymbol{K}\|_F\\
&\le\log\det(\boldsymbol{I}_{|\mathcal{G}|}+t\boldsymbol{K}/\lambda)+1
\end{aligned}
\tag{4.122}
$$

Finally, with probability at least $1-\delta_1$,

$$
\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\preccurlyeq\boldsymbol{K}+\|\boldsymbol{K}-\bar{\boldsymbol{G}}^\top\bar{\boldsymbol{G}}/m\|_2\boldsymbol{I}_{|\mathcal{G}|}\preccurlyeq\boldsymbol{K}+\frac{\rho_{\max}}{2}\boldsymbol{I}_{|\mathcal{G}|}\preccurlyeq\frac{3}{2}\rho_{\max}\boldsymbol{I}_{|\mathcal{G}|}
\tag{4.123}
$$

which indicates that $\lambda_{max}(\bar{\boldsymbol{U}}_t) \leq \lambda + \frac{3}{2}\rho_{\max}$. □

Lemma 4.7.12. *Fix $\delta \in (0,1)$. Then, for $m = \Omega(L^{10}|\mathcal{G}|^4\epsilon^{-4}\log(L/\delta))$, with probability at least $1-\delta$,*

$$|\rho_{\max} - \hat{\rho}_{\max}| \leq \epsilon.$$

Proof. Let m be as in Lemma 4.7.18. Recall that $\|\boldsymbol{h}_u^G\| = 1$ for all $u \in \mathcal{V}(G)$ and $G \in \mathcal{G}$, by construction. Let $N_i := |\mathcal{V}(G^i)|$. Then, we have, with probability at least $1-\delta$,

$$\begin{aligned}
&|k(G^i, G^j) - \hat{k}(G^i, G^j)| \\
&\quad \leq \frac{1}{N_i N_j} \sum_{\substack{u \in \mathcal{V}(G^i) \\ v \in \mathcal{V}(G^j)}} \left| k_{\mathrm{MLP}}(\boldsymbol{h}_u^{G^i}, \boldsymbol{h}_v^{G^j}) - g_{\mathrm{MLP}}(\boldsymbol{h}_u^{G^i}; \boldsymbol{\theta}_0)^\top g_{\mathrm{MLP}}(\boldsymbol{h}_v^{G^j}; \boldsymbol{\theta}_0)/m \right| \leq \epsilon
\end{aligned}$$

by Lemma 4.7.18. Then

$$\|\boldsymbol{K} - \hat{\boldsymbol{K}}\|_{op} \leq \|\boldsymbol{K} - \hat{\boldsymbol{K}}\|_F \leq |\mathcal{G}|\epsilon.$$

Then, from Weyl's inequality, $|\rho_{\max} - \hat{\rho}_{\max}| \leq |\mathcal{G}|\epsilon$. Replacing ϵ with $\epsilon/|\mathcal{G}|$ the result follows. □

4.7.3 Supporting Lemmas

Lemma 4.7.13. *Suppose $\boldsymbol{a}$, $\boldsymbol{b}$ are vectors and $\boldsymbol{A}$ is a matrix. c is assumed to be positive scalar. Then we have the following results: (i) $|\boldsymbol{a}^\top \boldsymbol{A}\boldsymbol{b}| \leq \sqrt{\boldsymbol{a}^\top \boldsymbol{A}\boldsymbol{a}}\sqrt{\boldsymbol{b}^\top \boldsymbol{A}\boldsymbol{b}}$. (ii) $\boldsymbol{a}^\top \boldsymbol{b} + c\|\boldsymbol{a}\|_2^2 \geq -\|\boldsymbol{b}\|_2^2/4c$.*

Lemma 4.7.14. *Suppose $X \sim \mathcal{N}(\mu, \sigma^2)$ and $\beta > 0$, then*

$$\mathbb{P}(|X - \mu| \leq \beta\sigma) \geq 1 - e^{-\beta^2/2} \tag{4.124}$$

Lemma 4.7.15. *Suppose $X \sim \mathcal{N}(\mu, \sigma^2)$ and $\beta > 0$, then*

$$\mathbb{P}(X - \mu > \beta\sigma) \geq \frac{e^{-\beta^2}}{4\beta\sqrt{\pi}} \tag{4.125}$$

Lemma 4.7.16. *Suppose $\boldsymbol{\epsilon} \in \mathbb{R}^t$ is a subgaussian random vector with subgaussian constant σ^2, then*

$$\mathbb{E}[\|\boldsymbol{\epsilon}\|_2] \leq 4\sigma\sqrt{t} \tag{4.126}$$

and with probability at least $1 - \delta$ for $\delta \in (0, 1)$,

$$\|\boldsymbol{\epsilon}\|_2 \leq C\sigma\sqrt{t}. \tag{4.127}$$

where C is some constant depending on δ.

Lemma 4.7.17. *(Theorem 1 (CG17)) Let $\{\boldsymbol{x}_t\}_{t=1}^{\infty}$ be an $\mathbb{R}^d$-valued discrete time stochastic process that is predictable with respect to the filtration $\{\mathcal{F}_t\}_{t=1}^{\infty}$. Let $\{\epsilon_t\}_{t=1}^{\infty}$ be a real-valued stochastic process and for any $\forall t$, ϵ_t is $\mathcal{F}_t$-measurable and subgaussian with constant R conditionally on $\mathcal{F}_{t-1}$. Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a symmetric positive-definite kernel. Then for any $\eta > 0$, $\delta \in (0, 1)$, with probability at least $1 - \delta$,*

$$\|\boldsymbol{\epsilon}_t\|^2_{((\boldsymbol{K}_t + \eta\boldsymbol{I}_t)^{-1} + \boldsymbol{I}_t)^{-1}} \leq R^2 \log\det((1 + \eta)\boldsymbol{I}_t + \boldsymbol{K}_t) + 2R^2 \log(1/\delta) \tag{4.128}$$

where $\boldsymbol{\epsilon}_t := (\epsilon_1, ..., \epsilon_t)^\top \in \mathbb{R}^t$ and $\boldsymbol{K}_t \in \mathbb{R}^{t \times t}$ is a matrix with $[\boldsymbol{K}_t]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $1 \leq i, j \leq t$.

Lemma 4.7.18 (Theorem 3.1 (ADH19))**.** *Fix $\epsilon > 0$ and $\delta \in (0, 1)$. Suppose an MLP $f_{\text{MLP}}(\cdot; \boldsymbol{\theta})$ with ReLU activation has L layers and width $m = \Omega(L^{10}\epsilon^{-4}\log(L/\delta))$. Then for any input $\boldsymbol{x}$, $\boldsymbol{x}'$ such that $\|\boldsymbol{x}\|_2 \leq 1$, $\|\boldsymbol{x}'\|_2 \leq 1$, with probability at least $1 - \delta$,*

$$|k_{\text{MLP}}(\boldsymbol{x}, \boldsymbol{x}') - g_{\text{MLP}}(\boldsymbol{x}; \boldsymbol{\theta}_0)^\top g_{\text{MLP}}(\boldsymbol{x}'; \boldsymbol{\theta}_0)/m| \leq \epsilon \tag{4.129}$$

where k_{MLP} is the neural tangent kernel associated with f_{MLP} and $g_{\text{MLP}}(\,\cdot\,; \boldsymbol{\theta}_0) = \nabla f_{\text{MLP}}(\,\cdot\,; \boldsymbol{\theta}_0)$

.

Lemma 4.7.19 (Lemma B.4/Lemma B.5/Lemma B.6 (ZLG20) / Lemma C.4 (ZZL20))**.** *Suppose $\boldsymbol{\theta}$ is parameters for an MLP $f_{\text{MLP}}(\cdot;\boldsymbol{\theta})$ with L layers and width m and this neural network $f_{\text{MLP}}(\cdot;\boldsymbol{\theta})$ is trained via gradient descent with initialization $\boldsymbol{\theta}_0$, learning rate η and ℓ_2 regularization constant λ in a mean squared loss. The input feature set is denoted as $\mathcal{X} = \{\boldsymbol{x}_i\}_{i\in[T]}$. Then there are positive constants $\{C_i\}_{i=1}^7$ such that for $\forall\delta \in (0,1)$, if τ satisfies*

$$\begin{aligned}\tau &\geq C_1 m^{-3/2}L^{-3/2}max((\log(TL^2/\delta))^{3/2}, (\log(m))^{-3/2})\\ \tau &\leq \min(C_2L^{-6}(\log(m))^{-3/2}, C_3L^{-9/2}(\log(m))^{-3}, C_4m^3\lambda^{9/2}\eta^3L^{-9}(\log(m))^{-3/2})\end{aligned} \tag{4.130}$$

then with probability at least $1-\delta$, for $\|\boldsymbol{\theta}-\boldsymbol{\theta}_0\|_2 \leq \tau$ and $\|\boldsymbol{\theta}'-\boldsymbol{\theta}_0\|_2 \leq \tau$, for $\forall \boldsymbol{x} \in \mathcal{X}$, we have

$$\|g_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta}) - g_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta}_0)\|_2 \leq C_5\sqrt{\log(m)}\tau^{1/3}L^3\|g_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta}_0)\|_2 \tag{4.131}$$

and

$$|f_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta}) - f_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta}') - \langle g_{\text{MLP}}(\boldsymbol{x},\boldsymbol{\theta}'), \boldsymbol{\theta}-\boldsymbol{\theta}'\rangle| \leq C_6\tau^{4/3}L^3\sqrt{m\log(m)} \tag{4.132}$$

and

$$\|g_{\text{MLP}}(\boldsymbol{x};\boldsymbol{\theta})\|_2 \leq C_7\sqrt{mL}. \tag{4.133}$$

4.7.4 Supplement to Experiments

4.7.4.1 Data Generation

We use synthetic data environments for our experiments. The datasets are generated from two different random graph models and three different reward function generating models. The random graph models are Erdös–Rényi random graph model and random dot product graph model. We use a linear model, Gaussian process with GNTK model, Gaussian process

with representation kernel to generate our reward function. In all data environments, the feature dimension is set as $d = 10$. For any synthetic graph, all entries of the associated feature matrix $\{\boldsymbol{X}_{ji}\}_{j\in[N],i\in[d]}$ are i.i.d from a standard Gaussian distribution. The noisy reward is assumed to have standard deviation $\sigma_\epsilon = 0.01$. All performance curves in our empirical studies show an average of over 10 repetitions with a standard deviation of the corresponding bandit problem with horizon $T = 1000$. Our experiment assumes the graph domain is fully observable, $\mathcal{G}_t = \mathcal{G}$ for all $t \in [T]$. We experiment four graph size $|\mathcal{G}| \in \{10, 50, 100, 200\}$ in the random dot product graphs with $N = 100$ and representation kernel.

Erdös–Rényi Random Graphs. Erdös–Rényi random graphs are generated by edge probability p and number of nodes N. Set the graph has N nodes and for any node pair $(i, j) \in [N]^2$, there is an edge linking i and j with probability p. We investigate $p \in \{0.2, 0.4, 0.6, 0.8\}$ and $N \in \{10, 50, 100, 500\}$ in our experiment. Including 3 types of reward function generating and 4 sizes of graph space $\mathcal{G}$, there are 192 combinations of datasets of Erdös–Rényi random graph environments.

Random Dot Product Graphs. Random dot product graphs are generated by modeling the expected edge probabilities as the function of the inner product of features. In our experiment, we set the latent embeddings observed as features, i.e. X_{i*} is the latent embedding of node i. Formally, the edge probability for node i and j is generated by $p_{ij} = \text{sigmoid}(\boldsymbol{X}_{i*}^\top \boldsymbol{X}_{j*})$. We also investigate $N \in \{10, 50, 100, 500\}$. Including 3 types of reward function generating and 4 sizes of graph space $\mathcal{G}$, there are 48 combinations of datasets of random dot product graph environments.

Linear Reward Function Generation. We generate a true parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$ whose elements are i.i.d standard Gaussian. Then the true reward mean is

$$\mu(G) = \langle \boldsymbol{\theta}^*, \bar{\boldsymbol{h}}^G \rangle \tag{4.134}$$

where $\bar{\boldsymbol{h}}^G = \sum_{i=1}^N \boldsymbol{h}_i^G/N$.

Gaussian Process with GNTK for Reward Function Generation. We also use Gaussian process and Graph Neural Tangent Kernel(GNTK) as introduced from experiment in (KKB22). We approximately construct the GNTK matrix $\boldsymbol{K}$ by the empirical GNTK matrix $\hat{\boldsymbol{K}} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ whose entries are $\hat{\boldsymbol{K}}_{ij} = \frac{1}{m}\langle g(G^i;\boldsymbol{\theta}_0), g(G^j;\boldsymbol{\theta}_0)\rangle$ for any $G^i, G^j \in \mathcal{G}$. We use this empirical GNTK matrix $\hat{\boldsymbol{K}}$ as the covariance matrix of prior, i.e, $\mathcal{N}(0, \boldsymbol{K}^{gntk})$ and use $\{(G, y_G)\}_{G\in\mathcal{G}}$ where $\{y_G\}_{G\in\mathcal{G}}$ are i.i.d from $\mathcal{N}(0,1)$ as our training data. To train this Gaussian process model, we use negative log-likelihood loss with Adam optimizer with learning rate 0.01 and 30 epochs. The true reward means are sampled from the posterior in this Gaussian process.

Gaussian Process with Representation Kernel for Reward Function Generation. For the Gaussian process with representation kernel, we trained a GNN for a graph property prediction task and used the mean pooling over all nodes of the last layer representations as the graph representation. In our experiment, we utilize the average degree prediction as our task. That is, suppose outcome is $d_G = \frac{1}{N}\sum_{j=1}^N \deg(j)$ and train GNN in (4.2) to predict this outcome. Then denote the last layer representation as $\bar{\boldsymbol{h}}_{\text{rep}}^G = \frac{1}{N}\sum_{j=1}^N f^{(L-1)}(\boldsymbol{h}_j^G)$. Then we define the representation kernel as the inner product of the graph representations

$$k_{\text{rep}}(G, G') := \langle \bar{\boldsymbol{h}}_{\text{rep}}^G, \bar{\boldsymbol{h}}_{\text{rep}}^{G'}\rangle. \tag{4.135}$$

The associated kernel matrix is denoted as $\boldsymbol{K}^{rep} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{G}|}$ with entries $\{k^{rep}(G, G')\}_{G,G'\in mG}$. In this Gaussian process, we sample the true reward means by $\{\mu(G)\}_{G\in\mathcal{G}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}^{rep})$. To train this Gaussian process model, we use MSE loss with Adam optimizer with learning rate 0.01 mini-batch size 2 and 30 epochs.

4.7.4.2 Algorithms Set Up

We provide the practical details and set up on our proposed algorithms and baseline algorithms.

Algorithms. We investigate 3 GNN-based bandit algorithms (`GNN-TS`, `GNN-UCB` and `GNN-PE`) and 3 corresponding NN-based bandit algorithms (`NN-TS`, `NN-UCB` and `NN-PE`). All algorithms in our work use the loss function (4.6) which is different from previous work. All gradients used for in our experiments are $g(G;\boldsymbol{\theta}_t)$ not $g(G;\boldsymbol{\theta}_0)$ unless special stated. In addition, in order to show the benefit of considering the graph structure, we include `NN-UCB`, `NN-TS`, `NN-PE` as our baselines. For this NN-based algorithm, we ignore the adjacency matrix for a graph (assume $\boldsymbol{A} = \boldsymbol{I}$), and pass through the model in (4.1) and (4.2) by $\boldsymbol{h}_i^G = \boldsymbol{X}_{i*}$. For `GNN-TS`, we tuned the exploration scale with grid search on $\nu \in \{0.01, 0.1, 1.0, 10.0\}$ and `NN-TS` follows the same value. For `GNN-UCB`, we tuned the hyperparameter with grid search on $\beta \in \{0.01, 0.1, 1.0, 10.0\}$ and `NN-UCB` follows the same value. For `GNN-PE`, we tuned the hyperparameter with grid search on $\beta \in \{0.01, 0.1, 1.0, 10.0\}$ and `NN-PE` follows the same value. All the hyperparameter tuning is performed in Erdös–Rényi random graphs with $p = 0.4$, $N = 50$, $|\mathcal{G}| = 100$ and Gaussian process with GNTK for all the Erdös–Rényi random graphs settings and random dot product graphs with 50 nodes and $|\mathcal{G}| = 100$ and Gaussian process with GNTK for all the random dot product graphs settings.

Neural Networks. The MLPs in our experiments have 2 layers ($L = 2$) and width $m = 512$. We use SGD optimizer with mini-batch size 5 and 30 epochs. Learning rates (η) we tuned from and the regularization hyperparameters λ we tuned from $\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. Initialization for the trainable GNN parameter $\boldsymbol{\theta}$ satisfies the condition $f_{\text{GNN}}(G;\boldsymbol{\theta}_0) = 0$ for all $G \in \mathcal{G}$, which is handle by the treatment in (KK22). Suppose the initialization is $\boldsymbol{\theta}_0$. The matrix inversion in the algorithms is approximated by diagonal inversion across all policy algorithms.

4.7.4.3 Experiments on Scalability ($|\mathcal{G}|$)

We set the size of the graph domain to $|\mathcal{G}| = 100$ in Figure 4.1 and we experiment across different sizes $|\mathcal{G}| \in \{10, 50, 100, 200\}$ to check the scalability of the algorithms. Figure 4.2 shows that given a fixed horizon length, larger $|\mathcal{G}|$ leads to a harder bandit problem. It also shows that `GNN-TS` can achieve top performance across all algorithms in all scales of the graph space. This empirical observation shows that `GNN-TS` is robust to the scalability of the action space, supporting our theoretical justification in Section 4.4.

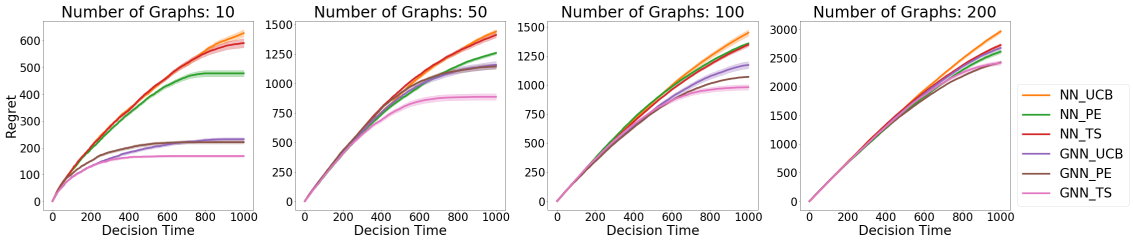

Figure 4.2: Competitive performance of `GNN-TS` is consistent across different sizes of graph space.

4.7.4.4 Effect of m and Initial Gradients

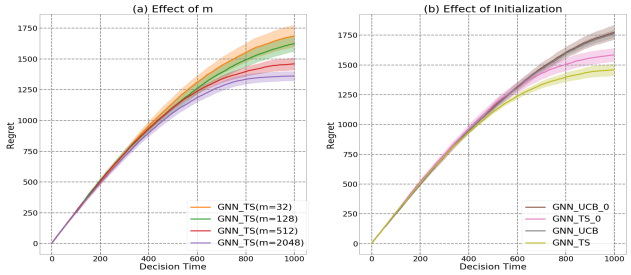

Figure 4.3: Increasing m can improve the performance of `GNN-TS` and no improvement of using $g(G_t; \boldsymbol{\theta}_0)$.

Our regret analysis depends on the assumption that the width of the neural network m must be large enough. We conduct an experiment to observe the effect from the width which is chosen from $\{32, 128, 512, 2048\}$. As some previous works on Neural bandit use

the gradients at initialization ($g(G_t; \boldsymbol{\theta}_0)$) for uncertainty calculation (ZLG20; KKB22) while some works use $g(G_t; \boldsymbol{\theta}_{t-1})$ which aligns with ours (ZZL20). Formally, instead of the update of uncertainty estimate in (4.5), using initial gradient means performing the following

$$\bar{\sigma}_t^2(G) = \frac{1}{m}\|g(G; \boldsymbol{\theta}_0)\|^2_{\bar{\boldsymbol{U}}_t^{-1}}, \qquad \bar{\boldsymbol{U}}_t = \bar{\boldsymbol{U}}_{t-1} + g(G_t; \boldsymbol{\theta}_0) g(G_t; \boldsymbol{\theta}_0)^\top / m. \tag{4.136}$$

Part (a) of Figure 4.3 reflects that the wider MLP has better performance which matches our expectation. Moreover, part (b) of Figure 4.3 reflects that there are no benefits from setting gradients used in algorithms to be the initial gradients for all $t \in [T]$. One small final observation is that the effects of m and initialization are not strong.

4.7.4.5 Additional Figures and Tables

Results for Erdös–Rényi Random Graphs. For better visualization of the 192 synthetic data environments using Erdös–Rényi random graphs, we summarised the result in Table 4.1. The metrics are relative regret and top rate, which are defined based on regret as follow. The relative regret of one algorithm in one data environment is defined as

$$\text{Relative Regret:} \tilde{R}^{\text{alg, env}} = \frac{R_T^{\text{alg, env}}}{\max_{\text{alg}} R_T^{\text{alg, env}}} \tag{4.137}$$

where $R_T^{\text{alg, env}}$ is the cumulative regret of algorithm alg, and data environment env.

We define the top rate for the policy in algorithm as the number of times such that the policy achieve the least two cumulative regret R_T. The denomnator is the number of total trails, which is the 1920, the 10 repetition and 192 combinations of ER environments. The top rate of one algorithm is defined as

$$\text{Top Rate:} \alpha_{\text{alg}} = \frac{\#\text{ times alg achieves "Top 2"}}{\#\text{ trails}}. \tag{4.138}$$

	NN-UCB	NN-PE	NN-TS	GNN-UCB	GNN-PE	GNN-TS
Top Rate	0.0%	1.6%	0.0%	9.4%	90.6%	**98.4 %**
Relative Regret	0.994(0.02)	0.891(0.06)	0.943(0.05)	0.762(0.15)	0.690(0.14)	**0.595(0.16)**

Table 4.1: Results on Erdös–Rényi random graphs. 192 data environments with 10 repetitions.

Results for Random Dot Product Graphs We provide the experiment results for regret on all random dot product graph settings. In thee plots, different rows represents different sizes of the graph space ($|\mathcal{G}|$) and columns represents the choices of the number of nodes in the graph (N).

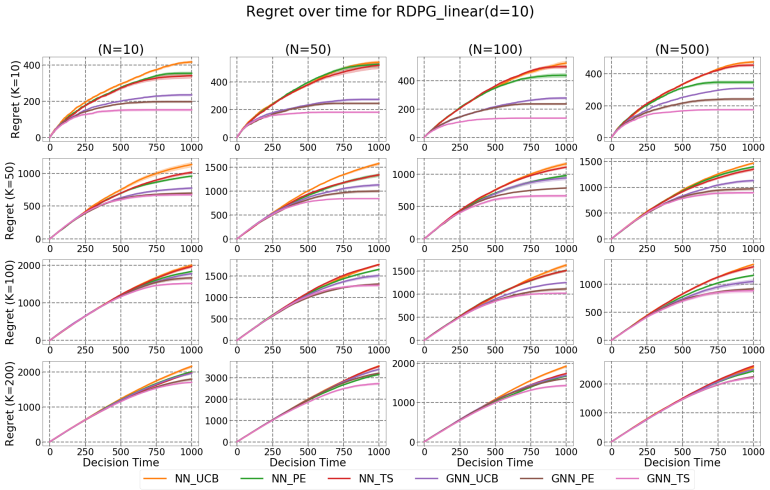

Figure 4.4: Random Dot Product Graphs with linear reward.

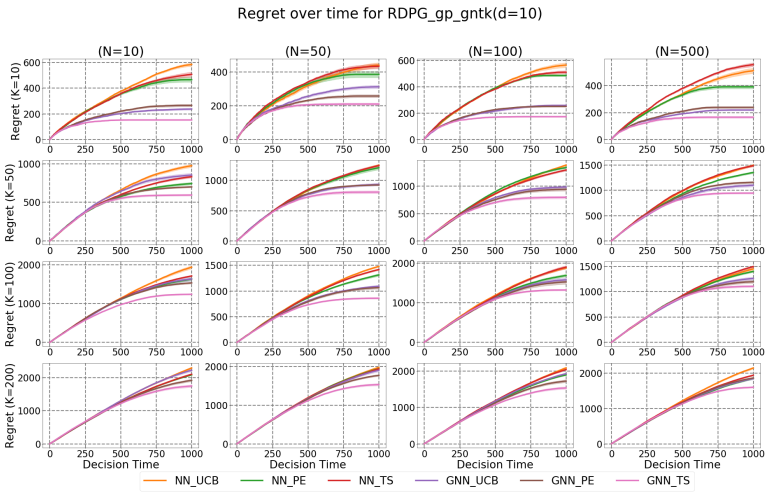

Figure 4.5: Random Dot Product Graphs with GP and GNTK for reward.

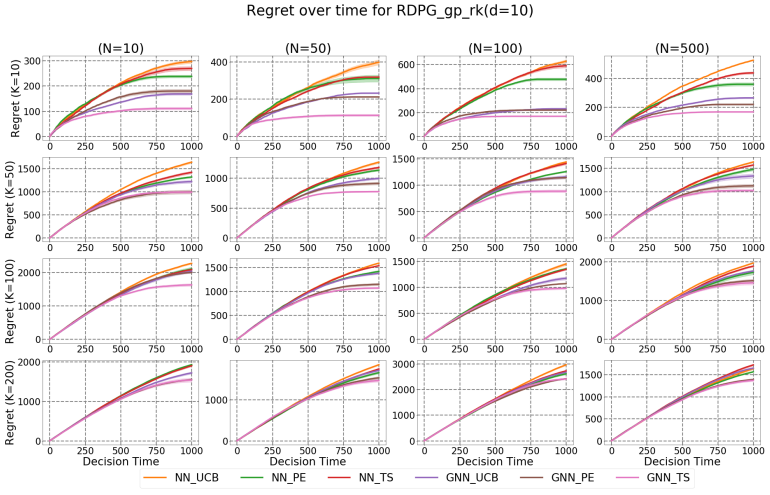

Figure 4.6: Random Dot Product Graphs with GP and representation kernel for reward.

Part III

Graph as Structure in Bandit

CHAPTER 5

Graph over Arms, Agents and more in Bandit Problems

5.1 Structured Arm Bandit

Problem View. Consider a stochastic bandit with a finite action set (arms) $\mathcal{A} = \{1, \dots, K\}$. We are given a weighted, undirected graph $G = (\mathcal{V}, \mathcal{E})$ *over the arms*, where $\mathcal{V} = \mathcal{A}$ and $w_{ij} \geq 0$ quantifies similarity between arms i and j. Let $\boldsymbol{L}$ be the (unnormalized or normalized) graph Laplacian of G. The mean-reward vector is $\boldsymbol{\mu} = (\mu_1, \dots, \mu_K)^\top \in \mathbb{R}^K$. A central inductive bias is *graph smoothness*: nearby arms should have similar means, which is formally measured by (BNS06)

$$\boldsymbol{\mu}^\top \boldsymbol{L}\, \boldsymbol{\mu} \;=\; \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} w_{ij}\, (\mu_i - \mu_j)^2. \tag{5.1}$$

The learner selects $a_t \in \mathcal{V}_t \in \mathcal{V}$ and observes $y_t = \mu_{a_t} + \epsilon_t$ where ϵ_t is the zero mean noise. The graph provides *structure* that allows the learner to share information across arms, accelerating identification of good arms when K is large or feedback is sparse.

Spectral bandit. Spectral methods exploit the Laplacian eigen-decomposition $\boldsymbol{L} = \boldsymbol{Q}\Lambda\boldsymbol{Q}^\top$ with eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_K$ and eigenvectors $\boldsymbol{U} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_K]$. Spectral bandit utilize the eigenbasic to represent the reward $\boldsymbol{\mu}$ (KVM14; KMK20) as

$$\mu_k = \boldsymbol{\theta}^\top \boldsymbol{x}_k.$$

Note that the $\{\boldsymbol{x}_k\}_{k=1}^K$ are the spectral feature which matches that $\phi(k, c) = \boldsymbol{x}_k$ in Chapter 1. Then problem is turned into a *linear* bandit with a Lapalacian regularization

$$\min_{\boldsymbol{\theta}} \frac{1}{2} \sum_{i=1}^{t} (y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2 + \frac{\lambda}{2} \|\boldsymbol{\theta}\|^2 + \frac{1}{2} \|\boldsymbol{\Lambda}\|^2 \tag{5.2}$$

where λ is the regularization parameter. The solution to (5.2) is the regularized least squared estimator. Following the same protocol of `LinUCB` or `LinTS`, the algorithms for spectral bandit, `SpectralUCB` or `SpectralTS` are proposed.

Laplacian Linear Bandit. Our contribution to the graph over arms problem is unifying some existing bandit methods as well as incorporating the attributes from the graph. Suppose $\boldsymbol{X} \in \mathbb{R}^{N \times d}$ is not the spectral feature matrix but the attributes feature matrix, where each node has d feature signals. The linear assumption is $\boldsymbol{\mu} = \boldsymbol{X}\boldsymbol{\theta}$. The *Laplacian linear* view couples these base features through the arm-graph by penalizing variation of the *predicted means* across edges. The graph smoothness in (5.1) becomes $\boldsymbol{\mu}^\top \boldsymbol{L}\, \boldsymbol{\mu} = \boldsymbol{\theta}^\top \big(\boldsymbol{X}^\top \boldsymbol{L} \boldsymbol{X}\big) \boldsymbol{\theta}$. which shrinks models whose *predicted* rewards vary sharply across adjacent arms. We also define $\boldsymbol{\Omega} = \boldsymbol{X}^\top \boldsymbol{L} \boldsymbol{X} + \lambda \boldsymbol{I}$ as the Laplacian-plus-ridge regularized matrix. Intuitively, $\boldsymbol{X}^\top \boldsymbol{L} \boldsymbol{X}$ encodes *where* in feature space the graph deems arms similar; directions that would induce non-smooth predictions are penalized. The resulting linear bandit (UCB/TS) uses the standard value-plus-uncertainty index, but the uncertainty metric is induced by $\boldsymbol{V}_t = \boldsymbol{V}_0 + \sum_{s<t} x_s x_s^\top$, so exploration prioritizes regions that are both data-poor and poorly constrained by the graph.

Our problem is an unified framework with with (1) Multi-Armed Bandit (MAB); (2) Contextual Bandit; (3) Spectral Bandit. For MAB, there is no attribute for the actions, which leads to $\boldsymbol{X} = \boldsymbol{I} \in \mathbb{R}^{K \times K}$ and $d = K$. In this setting, if actions are not isolated ($\boldsymbol{W} \neq \boldsymbol{D}$), we called the problem as Laplacian MAB. Otherwise, if the actions are isolated (a trivial graph with $\boldsymbol{W} = \boldsymbol{D}$), the problem is reduced to classical MAB. For contextual

bandit, the features are viewed as the context signals for the actions (d does not necessary equal to K). If the actions are isolated, the problem is reduced to classical linear bandit problem under linear model assumption. if actions are not isolated, we called the problem Laplacian linear bandit, which is our focus in this paper. For spectral bandit, there is no attribute for the actions but the spectral bases are considered as the features for the actions, which means $\boldsymbol{X} = \boldsymbol{Q}$ where $\boldsymbol{Q}$ is the orthogonal matrix with eigenvectors of $\boldsymbol{L}$.

5.2 Multi-Agent Bandit on Graphs

In many interactive systems (recommendation, advertising, personalization), decisions are made repeatedly for *many* users/agents. Let the user set be $\mathcal{U}$ and let $G = (\mathcal{U}, \mathcal{E}, w)$ be a weighted, undirected similarity graph over users, with Laplacian $\boldsymbol{L}$. At round t, the platform serves user $u_t \in \mathcal{U}$, observes a available action feature set $\mathcal{D}_t \subseteq \mathbb{R}^d$, selects an action $\boldsymbol{x}_t \in \mathcal{D}_t$ (or equivalently an item/context pair), and receives a noisy reward $y_t = f_{u_t}(\boldsymbol{x}_t) + \epsilon_t$ where ϵ_t is the zero mean noise with conditional subgaussian constant σ^2 and each user u has an (unknown) reward function f_u. A core inductive bias is *homophily across the user graph*: adjacent users tend to have similar response functions, formalized by a Laplacian smoothness prior $\sum_{(u,v)\in\mathcal{E}} w_{uv}\|f_u - f_v\|^2$. Leveraging this structure lets us "borrow strength" across users, accelerating cold-start and reducing regret when per-user data are sparse.

Gang-of-Bandits (GoB) Generally, *Gang-of-Bandits* (GoB) problem is the multi-agent bandit problem described above. So the GoB introduces a transductive multi-agent contextual bandit that couples all users through the graph and reduces the problem to a *single* bandit in a lifted space. The original work on GoB assume the linear bandit problem for the users (CGZ13): $f_u(\boldsymbol{x}) = \boldsymbol{\theta}_u^\top \boldsymbol{x}$. The graph smoothness is transferred as the quadratic form of user parameters $\{\boldsymbol{\theta}_u\}_{u\in\mathcal{U}}$: $\sum_{(u,v)\in\mathcal{E}} w_{uv}\|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_2^2$. Denote $\boldsymbol{\Theta} := [\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_n]^\top$ and $\boldsymbol{\theta} := [\boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_n^\top]$ as the stacking matrix and the stacking vector for parameters. We will

introduce the algorithms in the previous works for GoB with linear assumption.

GoB.Lin. The original work for *Gang-of-Bandits* problem proposed an algorithm called `GoB.Lin` which use the transformed feature to perform the linear bandit protocol. The least squared recipe in `GoB.Lin` is

$$\hat{\boldsymbol{w}}_t = \boldsymbol{M}^{-1} \sum_{i=1}^{t} y_i \phi_{u_i}(\boldsymbol{x}_i), \qquad \boldsymbol{M}_t = \sum_{i=1}^{t} \phi_{u_i}(\boldsymbol{x}_i)\phi_{u_i}(\boldsymbol{x}_i)^\top + \boldsymbol{I}.$$

with the UCB-type decision rule is

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x} \in \mathcal{D}_t} \hat{\boldsymbol{w}}_{t-1}^\top \phi_{u_t}(\boldsymbol{x}) + \beta_t \|\phi_{u_t}(\boldsymbol{x})\|_{\boldsymbol{M}_t^{-1}}$$

where β_t is the confidence scale parameter.

`G-EG` and `G-TS`. `GoB.Lin` is believed as an expensively computational algorithm and several works are scaling up the GoB algorithms. `G-EG`(graph-based epoch greedy) and `G-TS`(graph-based thompson sampling) are proposed to improve scalability by using Gaussian Markov Random Field (GMRF) (VSL17). Concretely, the authors consider $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{0}, \lambda^{-1}(\boldsymbol{L} \otimes \boldsymbol{I})^{-1})$ as the prior Gaussian model for $\boldsymbol{\theta}$. Then they transfer the minimization problem to maximizing the posterior $\boldsymbol{\theta}|\mathcal{F}_t \sim \mathcal{N}(\hat{\boldsymbol{\theta}}_t, \boldsymbol{\Sigma}_t^{-1})$ where $\mathcal{F}_t$ is the historical filtration up to time t and

$$\hat{\boldsymbol{\theta}}_t = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_t^{-1}\boldsymbol{b}_t, \quad \boldsymbol{\Sigma}_t = \frac{1}{\sigma^2}\sum_{\tau=1}^{t} \phi_{u_\tau}(\boldsymbol{x}_\tau)\phi_{u_\tau}(\boldsymbol{x}_\tau)^\top + \lambda(\boldsymbol{L} \otimes \boldsymbol{I}), \quad \boldsymbol{b}_t = \sum_{\tau=1}^{t} y_\tau \phi_{u_\tau}(\boldsymbol{x}_\tau).$$

The key improvement of `G-EG` is the efficient estimation of the posterior mean by solving the linear system $\boldsymbol{\Sigma}_t\boldsymbol{\theta} = \boldsymbol{b}_t$ using conjugate gradient descent. `G-TS` also has an efficient way to solve a linear system while the system has random perturbation terms which required sampling using Cholesky decomposition of $\boldsymbol{L}$.

GraphUCB. *GraphUCB* refines the GoB/Laplacian idea to improve scalability while preserving its statistical coupling (YTD20). Instead of working in the full lifted space, GraphUCB operates in the original feature space with per-user (or per-cluster) statistics, and incorporates the graph via *Laplacian-regularized least squares*. Concretely, Laplacian-regularized linear regression is

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\sum_{i=1}^{t}(y_i - \boldsymbol{\theta}^\top \boldsymbol{x}_i)^2 + \lambda \operatorname{tr}(\boldsymbol{\Theta}^\top \boldsymbol{\mathcal{L}}\boldsymbol{\Theta}) \tag{5.3}$$

where $\boldsymbol{\mathcal{L}}$ is the random walk Laplacian. This algorithm performs updates by blocks/agents: each agent has own empirical precision matrix and empirical Gram matrix:

$$\boldsymbol{A}_{i,t} = \sum_{\tau \in \mathcal{T}_{i,t}} \boldsymbol{x}_\tau \boldsymbol{x}_\tau^\top, \quad \boldsymbol{\Lambda}_{i,t} = \boldsymbol{A}_{i,t} + 2\lambda \boldsymbol{\mathcal{L}}_{ii}\boldsymbol{I} + \lambda^2 \sum_{j=1}^{n} \boldsymbol{\mathcal{L}}_{ij}^2 \boldsymbol{A}_{j,t}^{-1}$$

where $\mathcal{T}_{i,t}$ is the set of time at which agent i is served up to time t. To improve efficiency, the author proposed the following approximation to the solution in (5.3)

$$\hat{\boldsymbol{\theta}}_{i,t} \approx \boldsymbol{A}_{i,t}^{-1}\boldsymbol{X}_{i,t}^\top \boldsymbol{Y}_{i,t} - \lambda \boldsymbol{A}_{i,t}^{-1} \sum_{j=1}^{n} \boldsymbol{\mathcal{L}}_{ij} \boldsymbol{A}_{j,t}^{-1} \boldsymbol{X}_{j,t}^\top \boldsymbol{Y}_{j,t}$$

where $\boldsymbol{X}_{i,t} \in \mathbb{R}^{|\mathcal{T}_{i,t}| \times d}$ is the collection of features of arms that are selected for agent i up to time t with $\{\boldsymbol{x}_{\tau,j_\tau}\}_{\tau \in \mathcal{T}_{i,t}}$ as columns and $\boldsymbol{Y}_{i,t} \in \mathbb{R}^{|\mathcal{T}_{i,t}|}$ is the collection of rewards associated with agent i. up to time t. The UCB-style update is then designed as

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x}} \boldsymbol{x}^\top \hat{\boldsymbol{\theta}}_{u_t,t-1} + \beta_{u_t,t-1} \|\boldsymbol{x}\|_{\boldsymbol{\Lambda}_{u_t,t-1}^{-1}}.$$

5.3 Other Topics

Bandits with graph as structure is a large research area with lots of related problems. We will introduce two popular types in this section. For the other topics, we list some worth

studying topics as follow. Bandit with feedback graphs (RSP25) considers a setting that selecting a node on a graph will return a random linear combination of all nodes. Bandit with network interference (AAM24) considers to assign action on all nodes on the given network and observe rewards from all nodes where the true reward means are modeled by the neighborhood effect. Graph-triggering bandit (GMG24b; GMG24a) is a bandit problem with nonstationarity on the reward means. In this setting, given a graph over arms, pulling an arm will trigger an evolution of the expected reward of the chosen arm and of all its neighbor arms. The agent only observe the reward from the chosen arm and don't observe the evolution of the true reward means.

Bandits with Graphical Feedback. Bandit with graphical feedback is also a bandit on graph over arms (CLZ21). Comparing to what we discussed in Section 5.1, the key difference is that in bandit with graphical feedback, the learner can observe the noisy rewards from the neighborhood of the selected node, which is also referred to the *side observations* (GZ23). This also falls in between pure bandit feedback (scalar reward for the selected arm) and full information (reward for all arms). The graph is *structure over actions*: it does not change the reward-generating process for each arm, but it governs what we learn each round. Typical interactions include monitoring or recommendation scenarios where choosing one item also reveals (through logging or shared subcomponents) outcomes about a small neighborhood of related items.

Bandit on User-Item Graphs. An interesting extension to graph over arms or agents is to explore decision making on a User-Item graph, where the nodes on the graph include both users and the items and the decision is to recommend items to users on the graph: taking one actions means recommending one item to one user. Note that users are also as agents in a decentralized setting while users are not the agent in centralized setting. At each round, agent select an action (recommending an item to a user) on the graph and receive a reward feedback. Two examples are listed as follow. One is matching/assignment bandits on

bipartite graphs (YWH23). With a bipartite graph (e.g., drivers–riders, ads–slots), a round consists of observing contexts on nodes, choosing a feasible matching subject to capacity or fairness constraints, and receiving rewards on the *matched edges only* (semi-bandit) or an aggregate metric (bandit). Another example is linking user and item in a online manner. The interaction exposes only the outcome of chosen pairs, so exploration requires trying diverse matchings that inform unknown compatibilities while respecting instantaneous constraints. The graph encodes which pairings are even admissible and couples observations across time when the same nodes reappear.

CHAPTER 6

Laplacian Kernelized Bandit

6.1 Introduction

Graphs are pervasive in sequential decision making: they encode similarity or interaction among entities (users, items, sensors), and thus determine how information should be shared. We study a multi-user contextual bandit in which a known user graph promotes *homophily* of reward functions across users. At round t, the learner observes a user u_t and a candidate set $\mathcal{D}_t$ of arms (contexts), selects $\boldsymbol{x}_t \in \mathcal{D}_t$, and receives reward feedback y_t. Naively learning a separate model per user leads to regret that scales linearly with the number of users, whereas exploiting graph structure can yield dramatic improvements in both sample efficiency and computation. The research objective for this problem is to design multi-user algorithms that can leverage communication to improve overall performance (SBH13; LSL16; Dub20).

This multi-user bandit problem is originally studied as gang of bandit (CGZ13), which model user parameters or reward functions as *smooth signals* on the graph and penalize roughness via the graph Laplacian. `GoB.Lin` from this earliest work leverage the graph Laplacian to transform the model and the contexts of arms, which leads to a linear bandit solution. To scale up the computational cost in running algorithm with inversion, improved algorithms using Gaussian Markov random field (VSL17) and Taylor approximation (YTD20) are proposed. However, most of existing works on bandit problem with multi-user graph consider the linear function space and follows the linear bandit protocol. Our work will expand the problem into a *generalized gang-of-bandits* which considers the reward functions from

reproducing kernel Hilbert space (RKHS), inspired from the remarkable research track on kernelized bandit (CG17; DCK21; BEL21; LWW22; ZJ22).

A general idea to combine the kernlized bandit on a multi-user network is to cooperate the users with kernelized bandit (Dub20). Our work falls in this track but provide a more concrete and practical solution to the problem. We instantiate the agent-similarity kernel *explicitly* as a regularized Laplacian, $K_z(u, u') = [\boldsymbol{L}_\rho^{-1/2}]_{u,u'}$, and couple it with an arm kernel K_x to obtain a *multi-user kernel* $K((\boldsymbol{x}, u), (\boldsymbol{x}', u')) = K_z(u, u')\, K_x(\boldsymbol{x}, \boldsymbol{x}')$. This yields Gaussian-process posteriors over user–arm pairs and enables UCB and Thompson-sampling policies with calibrated uncertainty that *jointly* leverage nonlinear arm structure and Laplacian homophily.

Contributions. We first formalize a Laplacian–kernelized GP model for the multi-user bandit and then derive `LK-GP-UCB` and `LK-GP-TS` with scalable hybrid updates. We also provide confidence and regret guarantees in terms of an *effective dimension* that captures spectral decay of the combined kernel and graph. Empirically, our methods are competitive in linear regimes and substantially outperform linear and no-graph baselines when rewards are nonlinear yet graph-smooth.

6.2 Problem Formulation

Gang of Bandits. We consider the Gang of bandits(GOB) problem (CGZ13) with m arms and n users. Denote the arm set as $\mathcal{D} \subseteq \mathbb{R}^d$ and each arm is represented by a feature vector $\boldsymbol{x} \in \mathcal{D}$. In the GOB, a graph across users are given and the true reward functions of the users are assumes to satisfy the homophily on the given graph. We denote the known undirected graph as $G = (\mathcal{U}, \mathcal{E})$ where $\mathcal{U} = \{1, \ldots, n\}$ represents a set of n users(nodes in the graph) and $\mathcal{E}$ represents the links over the pairs of users which share the similar preference. Let $\boldsymbol{W}$ be the $n \times n$ matrix of edge weights (w_{ij}) and $\boldsymbol{D}$ be the diagonal degree matrix with

entries $d_i := \sum_j w_{ij}$. The graph Laplacian is defined as $\boldsymbol{L} := \boldsymbol{D} - \boldsymbol{W}$.

At each round t, the learner receives an agent $u_t \in \mathcal{U}$ (for example, randomly/uniformly receive one) with a set of context vectors $\mathcal{D}_t \subseteq \mathcal{D}$. The learner chooses an arm with vector $\boldsymbol{x}_t \in \mathcal{D}_t$ following some decision policy π. The learner then observe $y_t = f(\boldsymbol{x}_t, u_t) + \epsilon_t$ where ϵ_t is a conditionally zero-mean and bounded variance noise term for round t and $\{f(\cdot, i)\}_{i=1}^n$ are the reward functions for the users. ϵ_t is commonly assumed as conditionally sub-Gaussian with variance proxy σ^2. For the illustrative purpose, we also denote $f_i(\cdot) := f(\cdot, i)$ as the reward function for user i and use $f_{1:n} := \{f_i\}_{i=1}^n$ as the collection of the user-level reward functions.

The goal of the learner is to maximize the cumulative reward or alternatively minimize the cumulative regret with respect to the optimal strategy, which always selects the best arm for user. The instantaneous regret incurred at time t is $\Delta_t = f(\boldsymbol{x}_t^*, u_t) - f(\boldsymbol{x}_t, u_t)$ where $\boldsymbol{x}_t^* = \operatorname{argmin}_{\boldsymbol{x} \in \mathcal{D}_t} f(\boldsymbol{x}, u_t)$ and and the cumulative regret in a time horizon T (not necessarily known a priori) is defined to be $\mathcal{R}_T = \sum_{t=1}^T \mathbb{E}[\Delta_t]$. A sub-linear growth of $\mathcal{R}_T$ in T signifies that $\mathcal{R}_T / T \to 0$ as $T \to \infty$, or vanishing per-round regret.

Graph Regularity. Our core regularity assumption starts from presuming that the true reward functions $f_{1:n}$ are from the same reproducing kernel Hilbert space (RKHS), denoted by $\mathcal{H}_x$ with a semi-definite kernel function $K_x(\cdot, \cdot)$. The associated feature map is denoted as φ such that $K_x(\boldsymbol{x}, \boldsymbol{x}') = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}') \rangle$. We then consider the *user similarity assumption*, which posits that the users connected by the edges in the graph have similar reward functions. This is quantified by the graph smoothness

$$\mathrm{PEN}_{graph}(f_{1:n}) := \sum_{(i,j) \in \mathcal{E}} \|f_i - f_j\|_{\mathcal{H}_x}^2 = \frac{1}{2} \sum_{i,j} w_{ij} \|f_i - f_j\|_{\mathcal{H}_x}^2. \tag{6.1}$$

and we effectively assume that the above graph smoothness as well as the ridge smoothness

are small:

$$\text{PEN}(f_{1:n};\rho) := \text{PEN}_{graph}(f_{1:n}) + \rho\text{PEN}_{ridge}(f_{1:n}) \tag{6.2}$$

where the ridge penalty term is defined as $\text{PEN}_{ridge}(f_{1:n}) := \sum_{i=1}^{n} \|f_i\|_{\mathcal{H}_x}^2$ and ρ is the hyperparameter for balancing the ridge penalty and graph smoothness.

6.3 Methodology

6.3.1 Kernel Laplacian Regularized Regression

Our policy design is to learn f by using Kernel Laplacian Regularized Regression(KLRR) as the offline optimization problem. Formally, at each round t, the learner solves the KLRR problem presented as below

$$\min_{f_1,\cdots,f_n\in\mathcal{H}_x} \sum_{i=1}^{t}(f_{u_i}(\boldsymbol{x}_i) - y_i)^2 + \lambda\text{PEN}(f_{1:n};\rho) \tag{6.3}$$

where λ is the hyperpameter for Laplacian regularization. We denote $\boldsymbol{L}_\rho := \boldsymbol{L} + \rho\boldsymbol{I}_n$ as the Laplacian matrix with ridge correction.

Multi-user Kernel. Provided that user-level reward functions $f_{1:n}$ are from a RKHS $\mathcal{H}_x$ and users are connected by a graph with Lalplacian $\boldsymbol{L}$, we define *multi-user kernel* and *multi-user RKHS*. Our reward function f is assumed to be in a Laplacian-induced *multi-user RKHS* $\mathcal{H}$. We define the inner product on $\mathcal{H}$ as:

$$\langle f, g\rangle_{\mathcal{H}} := \sum_{u,u'=1}^{n} [\boldsymbol{L}_\rho]_{u,u'}\langle f_u, g_{u'}\rangle_{\mathcal{H}_x}$$

where $f, g \in \mathcal{H}$ and $g_u = g(\cdot, u) \in \mathcal{H}_x$. Clearly, $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}})$ is a Hilbert space, and the associated norm is $\|f\|_{\mathcal{H}} = \sum_{u,u'=1}[\boldsymbol{L}_\rho]_{u,u'}\langle f_u, f_{u'}\rangle_{\mathcal{H}_x}$

The corresponding *multi-user kernel*, is raised by a new feature map ϕ which embeds information for both user and context. Formally, for a user-context pair $(\boldsymbol{x}, u) \in [n] \times \mathbb{R}^d$, the feature map ϕ is defined as

$$\phi(\boldsymbol{x}, u) := (\boldsymbol{L}_\rho^{-1/2} \otimes \boldsymbol{I}_m)(\boldsymbol{e}_u \otimes \varphi(\boldsymbol{x})) = (\boldsymbol{L}_\rho^{-1/2} \otimes \boldsymbol{e}_u) \otimes \varphi(\boldsymbol{x}) \tag{6.4}$$

and it gives rise to our *multi-user kernel* $K((\boldsymbol{x}, u), (\boldsymbol{x}', u'))$ via the inner product:

$$\begin{aligned} K((\boldsymbol{x}, u), (\boldsymbol{x}', u')) &= \langle \phi(\boldsymbol{x}, u), \phi(\boldsymbol{x}', u') \rangle \\ &= ((\boldsymbol{L}_\rho^{-1/2} \otimes \boldsymbol{e}_u) \otimes \varphi(\boldsymbol{x}))^\top ((\boldsymbol{L}_\rho^{-1/2} \otimes \boldsymbol{e}_{u'}) \otimes \varphi(\boldsymbol{x}')) \\ &= [\boldsymbol{L}_\rho^{-1/2}]_{u,u'} K_x(\boldsymbol{x}, \boldsymbol{x}') \end{aligned} \tag{6.5}$$

which couples users u and u' through the graph structure encoded in the inverse regularized Laplacian.

By properties of the functions in RKHS, our reward functions admit a linear parametrization with form $f(\boldsymbol{x}, u) = \boldsymbol{\theta}^\top \phi(\boldsymbol{x}, u)$ where $\boldsymbol{\theta}$ is the unknown true coefficients. Note that the key challenge here is the infinite dimensionality of $\boldsymbol{\theta}$, which is not the same case of the ordinary linear bandit set up. Lastly, we assume the bound on the RKHS norm of the unknown reward function in Assumption 6, which is exactly bounding the ℓ^2 norm for the infinite dimensional parameter $\boldsymbol{\theta}$.

6.3.2 Gaussian Process Bandit

Motivated by the connection between the equivalence of online kernel ridge regression and Gaussian process (CG17), we propose algorithms based on the Gaussian process equivalent to the kernel Laplacian regularized regression 6.3. Similar to the kernelized bandit literatures, our Bayesian modeling is assumed for deriving our estimators, it is not necessarily the true

model. We consider a Gaussian process over $\mathcal{D}$, which has an initial prior

$$[f_1(\cdot), \ldots, f_n(\cdot)] \sim \mathcal{GP}(0, (\sigma^2/\lambda)K(\cdot, \cdot)).$$

We can also denote the prior distribution in this Gaussian process by $[f_{u_1}(\boldsymbol{x}_1), \cdots, f_{u_t}(\boldsymbol{x}_t)]^\top \sim \mathcal{N}(\mathbf{0}, K_t)$ where $K_t \in \mathbb{R}^{t\times t}$ has entries $[K_t]_{ij} = K((\boldsymbol{x}_i, u_i), (\boldsymbol{x}_j, u_j))$. At round t, given user u_t and selected arm $\boldsymbol{x}_t$, the noise in the observation model is $\epsilon_t = y_t - f(\boldsymbol{x}_t, u_t)$, which is Gaussian with variance σ^2 in the design of Gaussian process. Therefore, conditioned on the history $\mathcal{F}_t$, the posterior distribution over $f_1(\cdot), \ldots, f_n(\cdot)$ is $f_u(\cdot)|\mathcal{F}_t \sim \mathcal{N}(\mu_{u,t}(\cdot), \sigma^2_{u,t}(\cdot))$ where

$$\begin{aligned}
\mu_{u,t}(\boldsymbol{x}) &= \boldsymbol{k}_t(\boldsymbol{x}, u)^\top (\boldsymbol{K}_t + \lambda \boldsymbol{I}_t)^{-1} \boldsymbol{y}_t \\
\sigma^2_{u,t}(\boldsymbol{x}) &= K((\boldsymbol{x}, u), (\boldsymbol{x}, u)) - \boldsymbol{k}_t(\boldsymbol{x}, u)^\top (\boldsymbol{K}_t + \lambda \boldsymbol{I}_t)^{-1} \boldsymbol{k}_t(\boldsymbol{x}, u).
\end{aligned} \tag{6.6}$$

Here $\boldsymbol{k}_t(\boldsymbol{x}, u) := [K((\boldsymbol{x}_1, u_1), (\boldsymbol{x}, u)), \ldots, K((\boldsymbol{x}_t, u_t), (\boldsymbol{x}, u))]^\top \in \mathbb{R}^t$ is the kernel vector between past selected user-action pairs $\{(\boldsymbol{x}_s, u_s)\}_{s=1}^t$ and new pair $(\boldsymbol{x}, u)$, and $\boldsymbol{y}_t = [y_1, \ldots, y_t]^\top \in \mathbb{R}^t$ is the observed reward. Also, to simplify the notations, we let $\boldsymbol{\Sigma}_t := \boldsymbol{K}_t + \lambda \boldsymbol{I}_t$.

Decision Strategy. We design UCB Algorithm which follows the principle of optimism in face of uncertainty

$$\boldsymbol{x}_t = \underset{\boldsymbol{x}\in\mathcal{D}_t}{\operatorname{argmax}} \Big(\mu_{u_t,t-1}(\boldsymbol{x}) + \beta_t \sigma_{u_t,t-1}(\boldsymbol{x})\Big). \tag{6.7}$$

where β_t is the hyperparameter which ensures the appropriate scale of exploration in the bandit algorithms. We provide a upper confidence bound in Theorem 6.4.1 which shows a theoretical explicit form (c_t) for β_t, while it is usually tuned by learner in practice.

We also propose the TS Algorithm by adding Gaussian perturbation

$$\boldsymbol{x}_t = \underset{\boldsymbol{x}\in\mathcal{D}_t}{\operatorname{argmax}} \Big(\mu_{u_t,t-1}(\boldsymbol{x}) + \nu_t z_t(\boldsymbol{x}) \sigma_{u_t,t-1}(\boldsymbol{x})\Big). \tag{6.8}$$

where ν_t is the scale hyperparameter for exploration and $z_t(\boldsymbol{x}) \sim \mathcal{N}(0, 1)$ is the Gaussian

Algorithm 2 LK-GP-UCB

Input: T, λ, $\{\beta_t\}_{t=1}^T$
Initialization: $\mu_{u,0}(\boldsymbol{x})$, $\sigma_{u,0}(\boldsymbol{x})$
for $t = 1, ..., T$ **do**
  Observe user u_t and arm set $\mathcal{D}_t$.
  Select arm $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x}\in\mathcal{D}_t} \mu_{u_t,t-1}(\boldsymbol{x}) + \beta_t \sigma_{u_t,t-1}(\boldsymbol{x})$.
  Receive feedback $y_t = f(\boldsymbol{x}_t, u_t) + \epsilon_t$.
  Update $\mu_{u_t,t}(\boldsymbol{x})$ and $\sigma^2_{u_t,t}(\boldsymbol{x})$.

perturbation. Aligned with common Thompson Sampling literature, our decision strategy in (6.8) can be separated to two steps: sampling $\widetilde{\mu}_t(\boldsymbol{x})$ from $\mathcal{N}(\mu_{u_t,t-1}(\boldsymbol{x}), \nu_t^2 \sigma^2_{u_t,t-1}(\boldsymbol{x}))$ for all $\boldsymbol{x} \in \mathcal{D}_t$ and choosing an arm by $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x}\in\mathcal{D}_t} \widetilde{\mu}_t(\boldsymbol{x})$. Similar to UCB algorithm, we also use the theoretical explicit choice (c_t) for ν_t, while it is a tuning hyperparameter in real application.

Practical Implementation. The updates in (6.6) involves matrix inversion, making a high computational cost. We use practical recursive implementation for posterior mean and variance estimation (CG17). Concretely, we use

$$\begin{aligned}
\mu_{u,t}(\boldsymbol{x}) &= \mu_{u,t-1}(\boldsymbol{x}) + \frac{q_{t-1}((\boldsymbol{x},u),(\boldsymbol{x}_t,u_t))}{\lambda + \sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}(y_t - \mu_{u_t,t-1}(\boldsymbol{x}_t)) \\
q_t((\boldsymbol{x},u),(\boldsymbol{x}',u')) &= q_{t-1}((\boldsymbol{x},u),(\boldsymbol{x}',u')) - \frac{q_{t-1}((\boldsymbol{x},u),(\boldsymbol{x}_t,u_t)) q_{t-1}((\boldsymbol{x}_t,u_t),(\boldsymbol{x}',u'))}{\lambda + \sigma^2_{u_t,t-1}(\boldsymbol{x}_t)} \\
\sigma^2_{u,t}(\boldsymbol{x}) &= \sigma^2_{u,t-1}(\boldsymbol{x}) - \frac{q^2_{t-1}((\boldsymbol{x},u),(\boldsymbol{x}_t,u_t))}{\lambda + \sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}.
\end{aligned} \tag{6.9}$$

where $q_t((\boldsymbol{x},u),(\boldsymbol{x}',u'))$ is the estimated posterior covariance at round t. We explain how to obtain the updates in Appendix 6.6.1.2.

For the original update (6.6) at round t, the inversion takes $\mathcal{O}(t^3 m)$ time. The practical updates is efficient for each pair $(\boldsymbol{x}, u)$ while it requires the updates for all pairs, leading to $\mathcal{O}(mn)$ time. Therefore, high-level idea is to perform original updates (6.6) when $t \leq n^{1/3}$ and perform practical updates (6.9) when when $t \leq n^{1/3}$. Details in our implementation are provided in Appendix 6.6.4.4.

Algorithm 3 LK-GP-TS

Input: T, λ, $\{\nu_t\}_{t=1}^T$
Initialization: $\mu_{u,0}(\boldsymbol{x})$, $\sigma_{u,0}(\boldsymbol{x})$
for $t = 1, ..., T$ **do**
  Observe user u_t and arm set $\mathcal{D}_t$.
  Sample $\widetilde{\mu}_t(\boldsymbol{x})$ from $\mathcal{N}(\mu_{u_t,t-1}(\boldsymbol{x}), \nu_t^2\sigma_{u_t,t-1}^2(\boldsymbol{x}))$ for all $\boldsymbol{x} \in \mathcal{D}_t$
  Select arm $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x}\in\mathcal{D}_t} \widetilde{\mu}_t(\boldsymbol{x})$.
  Receive feedback $y_t = f(\boldsymbol{x}_t, u_t) + \epsilon_t$.
  Update $\mu_{u_t,t}(\boldsymbol{x})$ and $\sigma_{u_t,t}^2(\boldsymbol{x})$.

6.4 Regret Analysis

We begin by listing the Assumptions, which are mostly mentioned in the previous Sections.

Assumption 4 (Subgaussian Noise)**.** $\{\epsilon_t\}_{t=1}^T$ *is a* $\mathcal{F}_t$*-measurable stochastic process and is conditionally sub-Gaussian with constant* σ^2*.*

Assumption 5 (Rewards in Bounded RKHS)**.** *The true reward functions* $f_{1:n}$ *are from the same RKHS* $\mathcal{H}_x$ *induced by a semi-definite kernel function* $K_x(\cdot,\cdot)$ *with the following diagonal bound* $\sup_{\boldsymbol{x}\in\mathcal{D}} |K_x(\boldsymbol{x},\boldsymbol{x})| \leq \alpha$*.*

Assumption 6 (User Similarity)**.** *The user-level reward functions are similar across the given graph, quantified by the boundness of the graph regularity:* $PEN(f_{1:n};\rho) = \|f\|_{\mathcal{H}}^2 \leq B_\rho^2$ *where* $PEN(f_{1:n};\rho)$ *is defined in*(6.2)*.*

Assumption 4 is common assumption in bandit literature. Assumption 5 and 6 indirectly align with the regularity assumptions in kernelized bandit and graph smoothness literatures (BNS06; KMK20). Note that above Assumptions 4, 5 and 6 lead to the following results.

1. The off-diagonal kernel values are bounded. Formally, for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{D}$

$$|K_x(\boldsymbol{x},\boldsymbol{x}')| = \langle K_x(\cdot,\boldsymbol{x}), K_x(\cdot,\boldsymbol{x}')\rangle_{\mathcal{H}_x} \leq \sqrt{K_x(\boldsymbol{x},\boldsymbol{x})}\sqrt{K_x(\boldsymbol{x}',\boldsymbol{x}')} \leq \alpha.$$

2. The *multi-user kernel* is also bounded:

$$|K((\boldsymbol{x}, u)(\boldsymbol{x}', u'))| \leq \alpha \cdot \|\boldsymbol{L}_\rho^{-1/2}\|_{\max}.$$

3. Reward Means are bounded:

$$\begin{aligned} |f(\boldsymbol{x}, u)| = |\langle f, K(\cdot, (\boldsymbol{x}, u))\rangle_{\mathcal{H}}| &\leq \|f\|_{\mathcal{H}} \|K(\cdot, (\boldsymbol{x}, u))\|_{\mathcal{H}} \\ &= \|f\|_{\mathcal{H}} \sqrt{K((\boldsymbol{x}, u), (\boldsymbol{x}, u))} \\ &\leq B_\rho \cdot \alpha \cdot \|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \end{aligned}$$

4. Optimality gap is bounded:

$$|\Delta_t| = |f(\boldsymbol{x}_t^*, u_t) - f(\boldsymbol{x}_t, u_t)| \leq 2 \sup_{\boldsymbol{x}, u} |f(\boldsymbol{x}, u)| \leq 2\alpha B_\rho \|\boldsymbol{L}_\rho^{-1/2}\|_{\max}$$

We denote the bound for the optimality gap as $B_\Delta := 2\alpha B_\rho \|\boldsymbol{L}_\rho^{-1/2}\|_{\max}$ for simplicity.

Now we can state the confidence bound inferred from the concentration results:

Theorem 6.4.1 (Confidence Bound)**.** *Suppose Assumptions4, 5 and 6 hold. Let* $\{(\boldsymbol{x}_t, u_t)\}_{t=1}^{\infty}$ *be the* $\mathcal{F}_{t-1}$*-measurable discrete time stochastic process. Using the posterior estimators* $\mu_{u,t}(\boldsymbol{x})$ *and* $\sigma_{u,t}(\boldsymbol{x})$ *n Equation (6.6) yields to a high probability upper bound: with probability at least* $1 - \delta$*,* $\forall t \geq 1$*,* $\delta \in (0, 1)$*,* $\boldsymbol{x} \in \mathcal{D}$ *and* $u \in [n]$*:*

$$|\mu_{u,t}(\boldsymbol{x}) - f(\boldsymbol{x}, u)| \leq \left(B_\rho + \sqrt{\frac{\sigma^2}{\lambda} \cdot \log\det(\boldsymbol{I}_t + \lambda^{-1}\boldsymbol{K}_t) + \frac{2\sigma^2}{\lambda} \log\frac{1}{\delta}} \right) \cdot \sigma_{u,t}(\boldsymbol{x}) \tag{6.10}$$

where $\boldsymbol{K}_t \in \mathbb{R}^{t \times t}$ *is the Gram matrix up to round* t *with entries* $[\boldsymbol{K}_t]_{ij} = K((\boldsymbol{x}_i, u_i), (\boldsymbol{x}_j, u_j))$*.*

Theorem 6.4.1 states the confidence bound following the similar form in the kernelized bandit literatures(CG17; VKM13; Dub20). Our confidence bound has two primary key

differences. First one is the relaxation on the hyperparameter λ, which was assumed to be greater than 1. Second difference is retaining the Gram matrix in the bound, while classical way on Gaussian process bandit is to bound the logarithm on the product of eigenvalues using the information gain from the heuristic Gaussian assumption. Instead of the role of information gain, we consider the following effective dimension

$$\tilde{d} := \frac{\log\det(\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T)}{\log(1 + \lambda^{-1}\alpha T\|\boldsymbol{L}_\rho^{-1/2}\|_{\max})} \tag{6.11}$$

The quantity measures the actual underlying dimension of the reward function space as seen by the bandit problem(WA24; BM19; YW20). Our definition of the effective dimension is interpreted as the sum-to-maximum ratio of the log-eigenvalues of matrix $\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T$. As such, it is a robust measure of matrix rank. Using the confidence bound in Theorem 6.4.1, we provide the regret upper bound for `KL-GP-UCB` and `KL-GP-TS` as follow.

Theorem 6.4.2 (Regret Bound of KL-GP-UCB)**.** *Suppose Assumptions4, 5 and 6 hold. By setting $\beta_t = c_t$, Algorithm 2 for the gang of bandits problem has the high probability regret upper bound with order of upper bound $\mathcal{O}(\tilde{d}\log(T)\sqrt{T})$.*

Theorem 6.4.3 (Regret Bound of LK-GP-TS)**.** *Suppose Assumptions4, 5 and 6 hold. By setting $\nu_t = c_t$, Algorithm 3 for the gang of bandits problem has the high probability regret upper bound with order of upper bound $\mathcal{O}(\tilde{d}\log(T)^{3/2}\sqrt{T})$.*

6.5 Experiments

We evaluate Laplacian Kernelized bandit algorithms, `LK-GP-UCB` and `LK-GP-TS` on several synthetic data environments that capture user–user homophily on a known graph while varying reward structure (linear vs. nonlinear) and problem difficulty. Baseline algorithms include `GraphUCB`(YTD20), `GoB.Lin`(CGZ13), `GP-UCB`(CG17), `Pooled LinUCB` and `Per-User LinUCB`. Full implementation and reproducibility details are in Appendix 6.6.4.

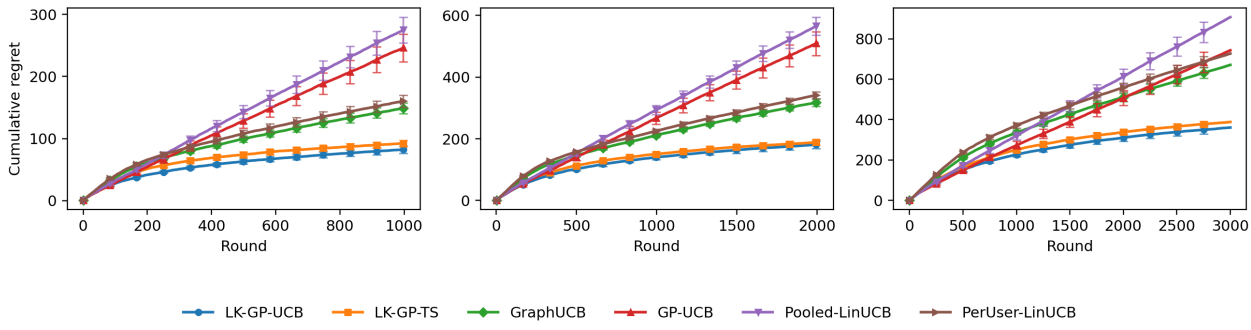

Figure 6.1: Cumulative Regret uneder *Linear-GOB* regime.

Environments. We draw a context pool $\mathcal{D}$ by sampling from $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ first and then normalize the context vectors. At round t we present $\mathcal{D}_t$ by sampling m_t distinct items from $\mathcal{D}$ without replacement. We generate the user graphs by Erdős–Rényi (ER) random graph model or Radial basis function(RBF) random graph model. After giving the generated graph, we consider one linear regime and two kernelized(nonlinear) regimes for synthetic data simulation. First synthetic data environment is called *Linear–GOB*. We consider simulating the true graph graph-smooth user parameters $\boldsymbol{\Theta} = (\boldsymbol{I} + \eta \boldsymbol{L})^{-1}\boldsymbol{\Theta}_0$, which enforce graph homophily on the random initial parameters $\boldsymbol{\Theta}_0 \in \mathbb{R}^{n \times d}$ (YTD20). The homophily strength is controlled by η in *Linear–GOB* regime. We also generate the true reward functions by simulating *multi-user kernel*, which is called the *Laplacian–Kernel* regime. We first use *Squared Exponential* as our base kernel K_x over arms $\mathcal{U}$ and construct the *multi-user kernel* using (6.5). Next, we design two choices to generate f, including a GP draw and a representer draw. We leave all the details for synthetic data simulation in Appendix 6.6.4.1.

Task Design. Our experiment has following design of the bandit tasks for a general comparison. In these tasks, the noise of reward is set as $\sigma = 0.1$ and the number of users is $n = 20$. The simplest level task is a 10-arm bandit problem ($m = 10$) with 50% viewability ($m_t = 5$) at each round for all users, under $T = 1000$ interaction rounds. Medium level task is a 20-arm bandit problem ($m = 20$) with 25% viewability ($m_t = 5$) at each round for all users, under $T = 3000$ interaction rounds. The hard task is a 50-arm bandit problem

($m = 50$) with 10% viewability ($m_t = 5$) at each round for all users, under $T = 5000$ interaction rounds.

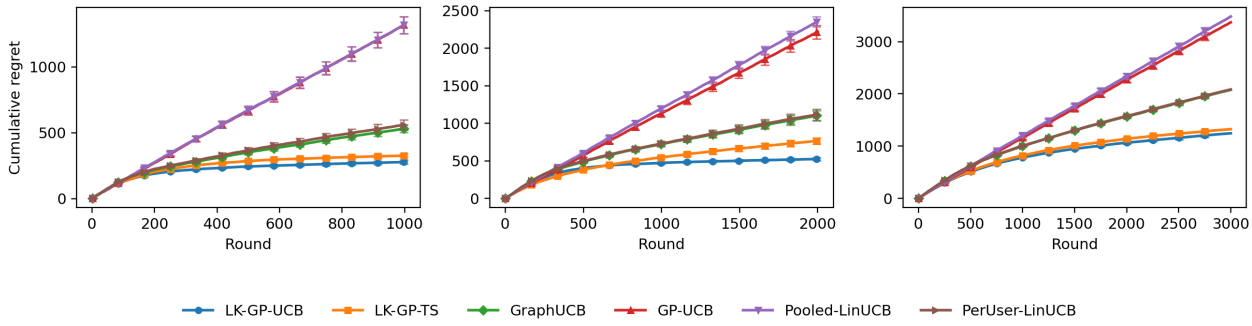

Figure 6.2: Cumulative Regret uneder *Laplacian–Kernel* regime using GP draw.

Algorithms Configurations. Our proposals `LK-GP-UCB` and `LK-GP-TS` are given in Algorithm 2 and Algorithm 3 and we implement the hybrid updates using practical recursive update in (6.9) and exact update in (6.6) with Cholesky decomposition. Details are in Appendix 6.6.4.4. Hyperparameters ν and β are tuned. The UCB hyperparameter β is also tuned. The classical baselines for GOB problem, `GoB.Lin`, `GraphUCB`, and all the remaining baselines, `Pooled LinUCB`, `Per-User LinUCB` and `GP-UCB`, are all `UCB`-based algorithms. We also tune their hyperparameter for the confidence bound. See Appendix 6.6.4.5 for details of hyperparameters tuning. All methods run in a centralized, no-delay setting.

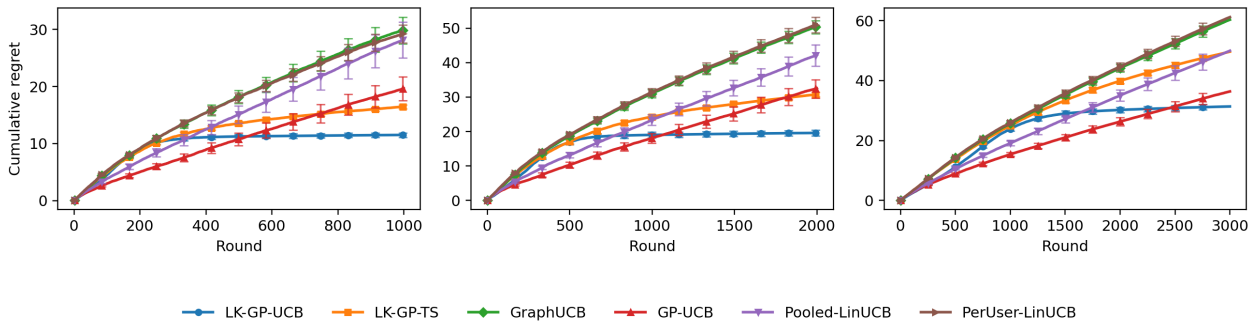

Figure 6.3: Cumulative Regret uneder *Laplacian–Kernel* regime using representer draw.

Algorithm	$n = 20$	$n = 50$	$n = 100$	$n = 200$
LK-GP-UCB	650.82 ± 37.65	903.07 ± 22.23	1054.93 ± 19.24	1147.39 ± 20.25
LK-GP-TS	634.46 ± 21.23	945.89 ± 17.17	1176.30 ± 15.78	1260.06 ± 13.57
GOB.Lin	1092.86 ± 71.70	1203.32 ± 18.57	1370.51 ± 16.78	1432.48 ± 18.72
GraphUCB	1105.20 ± 68.54	1192.30 ± 22.12	1360.02 ± 15.32	1453.21 ± 17.81
GP-UCB	2222.20 ± 90.26	1964.65 ± 61.40	1641.43 ± 37.43	1444.83 ± 36.33
Pooled-LinUCB	2360.95 ± 70.55	1909.81 ± 49.49	1723.27 ± 40.23	1438.74 ± 26.44
PerUser-LinUCB	1117.87 ± 72.04	1221.99 ± 22.03	1432.89 ± 18.81	1527.04 ± 17.61

Table 6.1: Ablation over number of users n (final cumulative regret; mean±SE).

Ablations. Our ablation study is based on the medium level task with *Laplacian–Kernel* regime using GP draw. In this study, we vary number of users $n \in \{20, 50, 100, 200\}$.

Main Findings. Our proposals `LK-GP-UCB` and `LK-GP-TS` have robust performance in all the 9 data environments. In the *Linear-GOB* regime, which is the preferred setting for linear bandit algorithms, our proposals can beat the baselines with clear gaps. In the *Laplacian-Kernel* regime, our proposals are consistently the best choices. For the GP draw setting, our proposals are always the top algorithms in our experiment. For setting using representer draw, even though `LK-GP-UCB` and `LK-GP-TS` are not the lowest cumulative regret when the task become hard, shown in Figure 6.3, only our proposals can achieve the sublinear regret. We believe our proposed algorithms can clearly outperform others in a long-term manner due to the achievement of the clear sublinear regret.

6.6 Appendix

6.6.1 Additional Discussion

6.6.1.1 Comparison to Cooperative Multi-Agent kernelized Bandits.

Comparison on the problem. Our work consider the Gang-of-Bandits problem, which is exactly the cooperative multi-agent kernelized bandit (Dub20). There are 3 key differences

between the problems we focus. We explicitly consider the regularized Laplacian $\boldsymbol{L}_\rho^{-1/2}$ as the agent similarity kernel. Formally, $K_z(u,v) = [\boldsymbol{L}_\rho^{-1/2}]_{u,v}$. Additionally, we assume the user-level reward functions $f_{1:n}$ are from the same RKHS while the previous paper assumes f_u has its own RKHS $\mathcal{H}_{x,u}$ for all $u \in \mathcal{U}$. Lastly, we consider a centralized agent/learner to observe with on-time feedback while they consider a decentralized version of the problem. There are also other differences in detail, such as the assumption on graph smoothness.

Comparison on the method. We propose to use Laplacian Kernelized Gaussian Process (LK-GP) method in bandit algorithm design, leading to `LK-GP-UCB` and `LK-GP-TS`. The `CoopKernelUCB` for cooperative multi-agent kernelized bandit problem is a UCB-based algorithm. In addition, the similarity kernel across the agents are learned in `CoopKernelUCB` and we apply the regularized Laplacian. Regarding the regret analysis, we use the effective dimension while the analysis for `CoopKernelUCB` follows the idea of information gain, which is classical but distribution-dependent for kernel-based method.

As a summary, we consider a more concrete and practical setting comparing to the problem in the paper for `CoopKernelUCB`. Also, we propose TS-based algorithm with kernel method to solve the problem. The advantage of using `CoopKernelUCB` is to solve more general and hard setting while our `LK-GP` bandit algorithms provide the explicit form choice of the similarity kernel and show superiority on our preferred regime.

6.6.1.2 Recursive Update of Posterior Mean and Variance

This sections refers to the derivation of incremental update of the posterior mean and posterior variance (CG17), via the properties of Schur complement. Recall that we need to handle the inversion of $\boldsymbol{\Sigma}_t = \boldsymbol{I} + \lambda K_t \in \mathbb{R}^{t\times t}$ which grows with the number of rounds. To compute the inversion of $\boldsymbol{\Sigma}_t$ efficiently, we use the recursive formula from $\boldsymbol{\Sigma}_{t-1}$ by block matrix inverse

formula

$$\boldsymbol{\Sigma}_t^{-1} = \begin{bmatrix} \boldsymbol{M}_{11,t} & \boldsymbol{M}_{12,t} \\ \boldsymbol{M}_{12,t}^\top & d_t^{-1} \end{bmatrix} \tag{6.12}$$

where

$$\begin{aligned} \boldsymbol{M}_{11,t} &= \boldsymbol{\Sigma}_{t-1}^{-1} + d_t^{-1}\boldsymbol{G}_t \\ \boldsymbol{M}_{12,t} &= -d_t^{-1}\boldsymbol{\Sigma}_{t-1}^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t) \end{aligned} \tag{6.13}$$

and

$$\begin{aligned} d_t &= K((\boldsymbol{x}_t, u_t), (\boldsymbol{x}_t, u_t)) - \boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t)^\top \boldsymbol{\Sigma}_{t-1}^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t) + \lambda = \sigma_{u_t,t-1}^2(\boldsymbol{x}_t) + \lambda \\ \boldsymbol{G}_t &= \boldsymbol{\Sigma}_{t-1}^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t)\boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \end{aligned}$$

Here d_t is the Schur complement.

Thus we have the posterior mean using (6.12)

$$\begin{aligned} & \mu_{u,t}(\boldsymbol{x}) \\ =& \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1}\boldsymbol{y}_t \\ =& \begin{bmatrix} \boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top & K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) \end{bmatrix} \begin{bmatrix} \boldsymbol{M}_{11,t} & \boldsymbol{M}_{12,t} \\ \boldsymbol{M}_{12,t}^\top & d_t^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_{t-1} \\ y_t \end{bmatrix} \\ =& \boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{M}_{11,t}\boldsymbol{y}_{t-1} + K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t))\boldsymbol{M}_{12,t}^\top \boldsymbol{y}_{t-1} + \boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{M}_{12,t} y_t \\ & \quad + K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t))d_t^{-1} y_t \\ =& \underbrace{\boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_{t-1}^{-1}\boldsymbol{y}_{t-1}}_{\mu_{u,t-1}(\boldsymbol{x})} + d_t^{-1}(\beta_1 \boldsymbol{y}_{t-1} - \beta_2 \boldsymbol{y}_{t-1} - \beta_3 y_t + \beta_4 y_t) \end{aligned}$$

where

$$\beta_1 = \boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{G}_t \Rightarrow \beta_1 \boldsymbol{y}_{t-1} = \Big(\boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t)\Big) \mu_{u_t, t-1}(\boldsymbol{x}_t)$$

$$\beta_2 = K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) \boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \Rightarrow \beta_2 \boldsymbol{y}_{t-1} = K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) \mu_{u_t, t-1}(\boldsymbol{x}_t)$$

$$\beta_3 = \boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \boldsymbol{k}_{t-1}$$

$$\beta_4 = K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)).$$

Thus we have the recursive update of posterior mean

$$\begin{aligned}
&\mu_{u,t}(\boldsymbol{x}) \\
&= \mu_{u,t-1}(\boldsymbol{x}) + d_t^{-1} \Big(\boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \boldsymbol{k}_{t-1}(\boldsymbol{x}_t, u_t) (\mu_{u_t, t-1}(\boldsymbol{x}_t) - y_t) \\
&\qquad + K((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) (y_t - \mu_{u_t, t-1}(\boldsymbol{x}_t)) \Big) \\
&= \mu_{u,t-1}(\boldsymbol{x}) + d_t^{-1} q_{t-1}((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) (y_t - \mu_{u_t, t-1}(\boldsymbol{x}_t))
\end{aligned}$$

where $q_{t-1}((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t))$ is defined from

$$q_t((\boldsymbol{x}, u), (\boldsymbol{x}', u')) = K((\boldsymbol{x}, u), (\boldsymbol{x}', u')) - \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{k}_t(\boldsymbol{x}', u')$$

which can be transferred into a recursive form using (6.12)

$$\begin{aligned}
&q_t((\boldsymbol{x}, u), (\boldsymbol{x}', u')) \\
=&K((\boldsymbol{x}, u), (\boldsymbol{x}', u')) - \Big(\boldsymbol{k}_{t-1}(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_{t-1}^{-1} \boldsymbol{k}_{t-1}(\boldsymbol{x}', u') \\
&\qquad + d_t^{-1} (\beta_1 \boldsymbol{k}_{t-1}(\boldsymbol{x}', u') - \beta_2 \boldsymbol{k}_{t-1}(\boldsymbol{x}', u') - \beta_3 K((\boldsymbol{x}_t, u_T), (\boldsymbol{x}', u')) + \beta_4 K((\boldsymbol{x}_t, u_T), (\boldsymbol{x}', u'))) \Big) \\
=&q_{t-1}((\boldsymbol{x}, u), (\boldsymbol{x}', u')) - d_t^{-1} q_{t-1}((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)) q_{t-1}((\boldsymbol{x}_t, u_t), (\boldsymbol{x}', u')).
\end{aligned}$$

Now using the incremental update of the posterior covariance, we can easily obtain the

recursive update for the posterior variance

$$\sigma^2_{u,t}(\boldsymbol{x}) = \sigma^2_{u,t-1}(\boldsymbol{x}) - d_t^{-1} q^2_{t-1}((\boldsymbol{x}, u), (\boldsymbol{x}_t, u_t)).$$

Now replace d_t by $\sigma^2_{u_t,t-1}(\boldsymbol{x}_t) + \lambda$ and we achieve the recursive updates in (6.9).

6.6.2 Proofs in Analysis

We first define following additional notations

$$\begin{aligned}
\boldsymbol{\Phi}_t &:= [\phi(\boldsymbol{x}_1, u_1), \cdots, \phi(\boldsymbol{x}_t, u_t)]^\top \\
\boldsymbol{J}_t &:= \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t \\
\boldsymbol{\Gamma}_t &:= \boldsymbol{J}_t + \lambda \boldsymbol{I}_\infty
\end{aligned}$$

Here we have $\boldsymbol{\Phi}_t \in \mathbb{R}^{t \times \infty}$ and $\boldsymbol{J}_t$, $\boldsymbol{\Gamma}_t$ are from $\mathbb{R}^{\infty \times \infty}$.

Then we define some useful events for concentration:

$$\begin{aligned}
\mathcal{E}_t^{\text{ts}} &= \{|z_t(\boldsymbol{x})| \le \sqrt{2\log(t^2|\mathcal{D}_t|)}, \text{ for all } \boldsymbol{x} \in \mathcal{D}_t\} \\
\mathcal{E}_t^{a} &= \{\mu_{u_t,t-1}(\boldsymbol{x}_t^*) + c_t z_t(\boldsymbol{x}_t^*)\sigma_{u_t,t-1}(\boldsymbol{x}_t^*) > f(\boldsymbol{x}_t^*, u_t)\}
\end{aligned}$$

where $z_t(\boldsymbol{x}) \sim \mathcal{N}(0,1)$ stands for the resampling randomness in Thompson Sampling. We also define the confidence set at round t:

$$\mathcal{C}_t := \{|\mu_{u_t,t-1}(\boldsymbol{x}_t) - f(\boldsymbol{x}_t, u_t)| \le c_t \cdot \sigma_{u_t,t-1}(\boldsymbol{x}_t)\} \tag{6.14}$$

where

$$c_t := \left(B_\rho + \sqrt{\frac{\sigma^2}{\lambda} \cdot \log\det(\boldsymbol{I}_{t-1} + \lambda^{-1}\boldsymbol{K}_{t-1}) + \frac{2\sigma^2}{\lambda}\log\frac{1}{\delta}} \right).$$

In addition, recall the following effective dimension

$$\tilde{d} := \frac{\log\det(\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T)}{\log(1 + \lambda^{-1}\alpha T\|\boldsymbol{L}_\rho^{-1/2}\|_{\max})}$$

Lastly, we provide the following Lemmas, which are commonly required in regret analysis.

Lemma 6.6.1 (Concentrations for TS)**.** *For all $t \in [T]$, we have $\mathbb{P}_t(\bar{\mathcal{E}}_t^{ts}) \leq t^{-2}$ and $\mathbb{P}_t(\mathcal{E}_t^a|\mathcal{C}_t) \geq (4e\sqrt{\pi})^{-1}$.*

Lemma 6.6.2 (One Step Regret Bound for TS)**.** *Suppose $\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{ts}) > 0$. Then for any t, almost surely,*

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}] \leq \mathbb{I}_{\mathcal{C}_t} \cdot \left\{ \left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{ts})} + 1 \right) \cdot \mathbb{E}_t[\gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)] + B_\Delta \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^{ts}) \right\}$$

where $\gamma_t := c_t + c_t\sqrt{2\log(t^2|\mathcal{D}_t|)}$ and $B_\Delta := 2\alpha B_\rho \|\boldsymbol{L}_\rho^{-1/2}\|_{\max}$

Lemma 6.6.3 (Cumulative Uncertainty Bound)**.**

$$\sum_{t=1}^{T} \sigma_{u_t,t-1}(\boldsymbol{x}_t) \leq \sqrt{2T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \log\det(\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T)}$$

Lemma 6.6.4 (Dual Identities)**.** *With the defined notations, we have two key identities:*

$$\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Phi}_t = \boldsymbol{\Phi}_t\boldsymbol{\Gamma}_t^{-1}, \text{ and } \sigma_{u,t}^2(\boldsymbol{x}) = \lambda\|\phi(\boldsymbol{x},u)\|_{\boldsymbol{\Gamma}_t^{-1}}^2.$$

6.6.2.1 Proof of Confidence Set

Proof of Theorem 6.4.1. We first decompose

$$\begin{aligned}\mu_{u,t}(\boldsymbol{x}) - f(\boldsymbol{x}, u) &= \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1}(\boldsymbol{\Phi}_t \boldsymbol{\theta} + \boldsymbol{\epsilon}_t) - \boldsymbol{\theta}^\top \phi(\boldsymbol{x}, u) \\ &= (\boldsymbol{\Phi}_t^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{k}_t(\boldsymbol{x}, u))^\top \boldsymbol{\theta} + \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\epsilon}_t - \boldsymbol{\theta}^\top \phi(\boldsymbol{x}, u) \\ &= \underbrace{\langle \boldsymbol{\theta}, \delta_t(\boldsymbol{x}, u) \rangle}_{\mathrm{bias}_t(\boldsymbol{x}, u)} + \underbrace{\boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\epsilon}_t}_{\mathrm{noise}_t(\boldsymbol{x}, u)}\end{aligned}$$

where $\delta_t(\boldsymbol{x}, u) = \boldsymbol{\Phi}_t^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{k}_t(\boldsymbol{x}, u) - \phi(\boldsymbol{x}, u) \in \ell^2$. Our target is to bound the $\mathrm{bias}_t(\boldsymbol{x}, u)$ and $\mathrm{noise}_t(\boldsymbol{x}, u)$. We state the following Lemmas:

Lemma 6.6.5 (Bias Identity)**.** *The squared bias is the degraded variance for noise:*

$$\|\delta_t(\boldsymbol{x}, u)\|_{\ell^2}^2 = \sigma_{u,t}^2(\boldsymbol{x}) - \lambda \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-2} \boldsymbol{k}_t(\boldsymbol{x}, u) \tag{6.15}$$

In particular, we have $\|\delta_t(\boldsymbol{x}, u)\|_{\ell^2} \leq \sigma_{u,t}(\boldsymbol{x})$ *and* $\lambda \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-2} \boldsymbol{k}_t(\boldsymbol{x}, u) < \sigma_{u,t}^2(\boldsymbol{x})$.

Lemma 6.6.6 (Noise Bound)**.**

$$\|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|_{\Gamma_t^{-1}} \leq \sqrt{\sigma^2 \log \det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t) + 2\sigma^2 \log \frac{1}{\delta}}$$

From Lemma 6.6.5, we could bound the bias by

$$\mathrm{bias}_t(\boldsymbol{x}, u) \leq \|\boldsymbol{\theta}\|_{\ell^2} \|\delta_t(\boldsymbol{x}, u)\|_{\ell^2} \leq B_\rho \sigma_{u,t}(\boldsymbol{x}). \tag{6.16}$$

Using the identities in above Lemma 6.6.4, we note that

$$\begin{aligned}
\text{noise}_t(\boldsymbol{x}, u) &= \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\epsilon}_t \\
&= \phi(\boldsymbol{x}, u)^\top \boldsymbol{\Gamma}_t^{-1} \boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t \\
&= \langle \phi(\boldsymbol{x}, u), \boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t \rangle_{\boldsymbol{\Gamma}_t^{-1}} \\
&\leq \|\phi(\boldsymbol{x}, u)\|_{\boldsymbol{\Gamma}_t^{-1}} \cdot \|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|_{\boldsymbol{\Gamma}_t^{-1}} \\
&= \frac{\sigma_{u,t}(\boldsymbol{x})}{\sqrt{\lambda}} \cdot \|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|_{\boldsymbol{\Gamma}_t^{-1}}
\end{aligned}$$

where the inequality is from the Cauchy-Schwarz inequality for the inner product $\langle \cdot, \cdot \rangle_{\boldsymbol{\Gamma}_t^{-1}}$.

Our Lemma 6.6.6 gives the high probability upper bound for the norm $\|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|_{\boldsymbol{\Gamma}_t^{-1}}$, leading to

$$\text{noise}_t(\boldsymbol{x}, u) \leq \frac{\sigma_{u,t}(\boldsymbol{x})}{\sqrt{\lambda}} \cdot \sqrt{\sigma^2 \log \det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t) + 2\sigma^2 \log \frac{1}{\delta}} \tag{6.17}$$

Now combine (6.16) and (6.17) together, we have

$$\begin{aligned}
|\mu_{u,t}(\boldsymbol{x}) - f(\boldsymbol{x}, u)| &\leq |\text{bias}_t(\boldsymbol{x}, u)| + |\text{noise}_t(\boldsymbol{x}, u)| \\
&\leq \sigma_{u,t}(\boldsymbol{x}) \left(B_\rho + \sqrt{\frac{\sigma^2}{\lambda} \cdot \log \det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t) + \frac{2\sigma^2}{\lambda} \log \frac{1}{\delta}} \right).
\end{aligned}$$

□

6.6.2.2 Proof of Regret Bound of `LK-GP-UCB`

Proof of Theorem 6.4.2. Recall the instantaneous regret at time t is $\Delta_t = f(\boldsymbol{x}_t^*, u_t) - f(\boldsymbol{x}_t, u_t)$ and the cumulative regret in a time horizon T is $\mathcal{R}_T = \sum_{t=1}^T \Delta_t$. We note event $\mathcal{C}_t := \{|\mu_{u_t,t-1}(\boldsymbol{x}_t) - f(\boldsymbol{x}_t, u_t)| \leq c_t \cdot \sigma_{u_t,t-1}(\boldsymbol{x}_t)\}$ happens with high probability $(1 - \delta)$, according

to Theorem 6.4.1,

$$c_t := \left(B_\rho + \sqrt{\frac{\sigma^2}{\lambda} \cdot \log\det(\boldsymbol{I}_{t-1} + \lambda^{-1}\boldsymbol{K}_{t-1}) + \frac{2\sigma^2}{\lambda}\log\frac{1}{\delta}} \right) \tag{6.18}$$

By Theorem 6.4.1, for all $t \geq 2$ with probability at least $1-\delta$,

$$\begin{aligned}
\Delta_t = f(\boldsymbol{x}_t^*, u_t) - f(\boldsymbol{x}_t, u_t) &\leq \mu_{u_t,t-1}(\boldsymbol{x}_t^*) + c_t\sigma_{u_t,t-1}(\boldsymbol{x}_t^*) - f(\boldsymbol{x}_t, u_t) \\
&\leq \mu_{u_t,t-1}(\boldsymbol{x}_t) + c_t\sigma_{u_t,t-1}(\boldsymbol{x}_t) - f(\boldsymbol{x}_t, u_t) \\
&\leq 2c_t\sigma_{u_t,t-1}(\boldsymbol{x}_t).
\end{aligned}$$

Thus we have high probability bound for the cumulative regret

$$\mathcal{R}_T \leq 2\mathbb{E}\Big[c_T \sum_{t=2}^{T} \sigma_{u_t,t-1}(\boldsymbol{x}_t)\Big] + B_\Delta.$$

Then we apply Lemma 6.6.3 and the definition of effective dimension in (6.11)

$$\begin{aligned}
\sum_{t=1}^{T} \sigma_{u_t,t-1}(\boldsymbol{x}_t) &\leq \sqrt{2T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \log\det(\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T)} \\
&= \sqrt{2T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \tilde{d}\log(1 + T\lambda^{-1}\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max})}.
\end{aligned}$$

Therefore, we have the final high probability upper bound for regret:

$$\mathcal{R}_T \leq 2\mathbb{E}[c_T]\sqrt{2T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \tilde{d}\log(1 + T\lambda^{-1}\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max})} + B_\Delta.$$

Next step is analyzing the order of the upper bound. By using the effective dimension $\tilde{d}$

again and dropping constants, we have

$$c_T \leq B_\rho + \sqrt{\frac{\sigma^2}{\lambda} \cdot \tilde{d} \log(1 + T\lambda^{-1}\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max}) + \frac{2\sigma^2}{\lambda} \log \frac{1}{\delta}} = \mathcal{O}(\sqrt{\tilde{d} \log(T)})$$
$$\Rightarrow \mathcal{R}_T = \mathcal{O}(\tilde{d} \log(T) \sqrt{T}).$$

□

6.6.2.3 Proof of Regret Bound of `LK-GP-TS`

Proof of Theorem 6.4.3. We start from the decomposition of the cumulative regret

$$\mathcal{R}_T = \sum_{t=1}^{T} \mathbb{E}[\Delta_t] = \sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\mathcal{C}_t}] + \sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\bar{\mathcal{C}}_t}].$$

By Theorem 6.4.1 and the upper bound for the optimality gap, we know the second term is bounded:

$$\sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\bar{\mathcal{C}}_t}] \leq \delta B_\Delta$$

by letting $\mathbb{P}(\mathcal{C}_t) \leq \delta/T$ for all t in Theorem 6.4.1.

For the regret on the event $\mathcal{C}_t$, by Lemma 6.6.2, almost surely, we have

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}] \leq \mathbb{I}_{\mathcal{C}_t} \cdot \left\{ \left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})} + 1 \right) \cdot \mathbb{E}_t[\gamma_t \sigma_{u_t, t-1}(\boldsymbol{x}_t)] + B_\Delta \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}}) \right\}$$

where $\gamma_t := c_t + c_t \sqrt{2 \log(t^2 |\mathcal{D}_t|)}$. Note that $\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}}) \geq \frac{1}{4e\sqrt{\pi}} - \frac{1}{t^2} \geq \frac{1}{20e\sqrt{\pi}}$ by Lemma 6.6.1 and the fact that $t^2 \geq 5e\sqrt{\pi}$ for all $t \geq 5$. Thus we have

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}] \leq \mathbb{I}_{\mathcal{C}_t} \cdot \left\{ 194 \mathbb{E}_t[\gamma_t \sigma_{u_t, t-1}(\boldsymbol{x}_t)] + B_\Delta t^{-2} \right\}$$

by using $40e\sqrt{\pi}+1 \leq 194$. Taking summation on both side for our target cumulative regret, we get

$$\begin{aligned}
\sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\mathcal{C}_t}] &= \mathbb{E}[\sum_{t=1}^{T} \mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}]] \\
&\leq \mathbb{E}[\sum_{t=5}^{T} \left(194\mathbb{E}_t[\gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)] + B_\Delta t^{-2}\right) + 4B_\Delta] \\
&\leq \mathbb{E}[194 \sum_{t=5}^{T} \mathbb{E}_t[\gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)] + (4 + \frac{\pi^2}{6})B_\Delta] \\
&\leq \mathbb{E}[194\gamma_T \mathbb{E}_t[\sum_{t=1}^{T} \sigma_{u_t,t-1}(\boldsymbol{x}_t)] + (4 + \frac{\pi^2}{6})B_\Delta]
\end{aligned}$$

where the second equality is using $\sum_{t=1}^{\infty} t^{-2} = \pi^2/6$ and the last step is from the monotonicity of the γ_t and the nonnegative of $\sigma_{u,t}(\boldsymbol{x})$. Our next focus is bounding the summation of uncertainty. As the same approach in the proof of Theorem 6.4.2, we apply Lemma 6.6.3 and the definition of effective dimension in (6.11)

$$\begin{aligned}
\sum_{t=1}^{T} \sigma_{u_t,t-1}(\boldsymbol{x}_t) &\leq \sqrt{2T\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \log\det(\boldsymbol{I}_T + \lambda^{-1}\boldsymbol{K}_T)} \\
&= \sqrt{2T\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \tilde{d}\log(1 + T\lambda^{-1}\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max})}.
\end{aligned}$$

Thus we have

$$\sum_{t=1}^{T} \mathbb{E}[\Delta_t \mathbb{I}_{\mathcal{C}_t}] \leq 194\mathbb{E}[\gamma_T]\sqrt{2T\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \tilde{d}\log(1 + T\lambda^{-1}\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max})} + (4 + \frac{\pi^2}{6})B_\Delta$$

leading to the high probability $(1-\delta)$ regret upper bound:

$$\mathcal{R}_T \leq 194\mathbb{E}[\gamma_T]\sqrt{2T\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max} \cdot \tilde{d}\log(1 + T\lambda^{-1}\alpha \|\boldsymbol{L}_\rho^{-1/2}\|_{\max})} + (4 + \frac{\pi^2}{6})B_\Delta + \delta B_\Delta.$$

For the order of the upper bound, we first analyze $\mathbb{E}[\gamma_T]$, by using the definition of effective

dimension $\tilde{d}$ again and dropping constants

$$\gamma_T \leq \left(1+\sqrt{2\log(T^2 m)}\right)\cdot\left(B_\rho+\sqrt{\frac{\sigma^2}{\lambda}\cdot\tilde{d}\log(1+T\lambda^{-1}\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max})+\frac{2\sigma^2}{\lambda}\log\frac{1}{\delta}}\right) \tag{6.19}$$

$$=\mathcal{O}(\log(T)\sqrt{\tilde{d}}). \tag{6.20}$$

Therefore,

$$\mathcal{R}_T=\mathcal{O}(\tilde{d}\log(T)^{3/2}\sqrt{T}).$$

□

6.6.3 Proof of Lemmas

6.6.3.1 Proof of Lemma 6.6.1

Proof. Using the standard Gaussian tail bound and the classical union bound, we have

$$\mathbb{P}_t(|z_t(\boldsymbol{x})|>u)\leq|\mathcal{D}_t|e^{-u^2/2}.$$

By letting $u=\sqrt{2\log(t^2|\mathcal{D}_t|)}$, we obtain $\mathbb{P}_t(\bar{\mathcal{E}}_t^{ts})\leq t^{-2}$.

For the result of event $\mathcal{E}_t^a$, we have

$$\begin{aligned}
&\mathbb{P}_t\left(\mu_{u_t,t-1}(\boldsymbol{x}_t^*)+c_t z_t(\boldsymbol{x}_t^*)\sigma_{u_t,t-1}(\boldsymbol{x}_t^*)>f(\boldsymbol{x}_t^*,u_t)|\mathcal{C}_t\right)\\
=&\mathbb{P}_t\left(z_t(\boldsymbol{x}_t^*)>\frac{f(\boldsymbol{x}_t^*,u_t)-\mu_{u_t,t-1}(\boldsymbol{x}_t^*)}{c_t\sigma_{u_t,t-1}(\boldsymbol{x}_t^*)}|\mathcal{C}_t\right)\\
\geq&\mathbb{P}_t(z_t(\boldsymbol{x}_t^*)>1)\\
\geq&(4e\sqrt{\pi})^{-1}
\end{aligned}$$

where the first inequality is from the fact that $\mathcal{C}_t$ holds and the last step is directly obtain by the fact that $\mathbb{P}(Z \geq 1) \geq (4e\sqrt{\pi})^{-1}$ for $Z \sim \mathcal{N}(0,1)$.

□

6.6.3.2 Proof of Lemma 6.6.2

Proof. This proof is following the classical analysis for Thompson Sampling algorithms ().

We first recall $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_t]$. Given the randomness from the history $\mathcal{F}_t$, event $\mathcal{C}_t$ becomes deterministic and the randomness is only from the resampling step. So we have

$$\begin{aligned}
\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}] &= \mathbb{I}_{\mathcal{C}_t} \cdot \mathbb{E}_t[\Delta_t] \\
&= \mathbb{I}_{\mathcal{C}_t} \cdot \left(\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\text{ts}}}] + \mathbb{E}_t[\Delta_t \mathbb{I}_{\bar{\mathcal{E}}_t^{\text{ts}}}]\right) \\
&\leq \mathbb{I}_{\mathcal{C}_t} \cdot \left(\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\text{ts}}}] + B_\Delta \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})\right)
\end{aligned}$$

where the last step is from the boundness of the optimality gap $\Delta_t \leq B_\Delta$. Our following focus is bounding $\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\text{ts}}}]$, indicating $\mathcal{C}_t$ holds in the remaining part of proof.

We then define the concept of "least uncertain undersampled" action, which is called unsaturated actions, defined as

$$\mathcal{U}_t := \{\boldsymbol{x} \in \mathcal{D}_t : f(\boldsymbol{x}_t^*, u_t) < f(\boldsymbol{x}, u_t) + \gamma_t \sigma_{u_t, t-1}(\boldsymbol{x})\}$$

where

$$\gamma_t := c_t + c_t \sqrt{2 \log(t^2 |\mathcal{D}_t|)}$$

and let $\bar{\boldsymbol{x}}_t$ be the least uncertain unsaturated action at time t:

$$\bar{\boldsymbol{x}}_t = \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{U}_t} \gamma_t \sigma_{u_t, t-1}(\boldsymbol{x}).$$

Recall the notation for the resampled index is $\tilde{\mu}_t(\boldsymbol{x}) = \mu_{u_t,t-1}(\boldsymbol{x}) + c_t z_t(\boldsymbol{x})\sigma_{u_t,t-1}(\boldsymbol{x})$. On the good situation $\mathcal{C}_t \cap \mathcal{E}_t^{\text{ts}}$, we have

$$|\tilde{\mu}_t(\boldsymbol{x}) - f(\boldsymbol{x}, u_t)| \leq |\tilde{\mu}_t(\boldsymbol{x}) - \mu_{u_t,t-1}(\boldsymbol{x})| + |\mu_{u_t,t-1}(\boldsymbol{x}) - f(\boldsymbol{x}, u_t)| \leq \gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}).$$

Thus we can provide an initial upper bound for regret

$$\begin{aligned}
\Delta_t &= f(\boldsymbol{x}_t^*, u_t) - f(\boldsymbol{x}_t, u_t) \\
&= f(\boldsymbol{x}_t^*, u_t) - f(\bar{\boldsymbol{x}}_t, u_t) + f(\bar{\boldsymbol{x}}_t, u_t) - f(\boldsymbol{x}_t, u_t) \\
&\leq \gamma_t \sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t) + f(\bar{\boldsymbol{x}}_t, u_t) - f(\boldsymbol{x}_t, u_t) + \tilde{\mu}_t(\boldsymbol{x}_t) - \tilde{\mu}_t(\boldsymbol{x}_t) \quad (\text{by } \bar{\boldsymbol{x}}_t \in \mathcal{U}_t) \\
&\leq 2\gamma_t \sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t) + \gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t) + \tilde{\mu}_t(\bar{\boldsymbol{x}}_t) - \tilde{\mu}_t(\boldsymbol{x}_t) \quad (\text{ since } \mathcal{C}_t \cap \mathcal{E}_t^{\text{ts}}) \\
&\leq 2\gamma_t \sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t) + \gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t) \quad (\text{ by } \tilde{\mu}_t(\bar{\boldsymbol{x}}_t) < \tilde{\mu}_t(\boldsymbol{x}_t)).
\end{aligned} \tag{6.21}$$

Note that

$$\gamma_t \sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t)\mathbb{I}\{\boldsymbol{x}_t \in \mathcal{U}_t\} \leq \gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)$$

and by taking $\mathbb{E}_t[\cdot]$ after multiplying both sides by $\mathbb{I}_{\mathcal{E}_t^{\text{ts}}}$, we have

$$\sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t)\mathbb{P}_t(\{\boldsymbol{x}_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^{\text{ts}}) \leq \mathbb{E}_t[\sigma_{u_t,t-1}(\boldsymbol{x}_t)\mathbb{I}_{\mathcal{E}_t^{\text{ts}}}].$$

Thus it remains to bound the probability $\mathbb{P}_t(\{\boldsymbol{x}_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^{\text{ts}})$ from below.

We notice the following two facts. First, if $\tilde{\mu}_t(\boldsymbol{x}_t^*) > \tilde{\mu}_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in \bar{\mathcal{U}}_t$, then $\boldsymbol{x}_t$ must belong to $\mathcal{U}_t$, which means $\{\tilde{\mu}_t(\boldsymbol{x}_t^*) > \max_{\boldsymbol{x} \in \bar{\mathcal{U}}_t} \tilde{\mu}_t(\boldsymbol{x})\} \subseteq \{\boldsymbol{x}_t \in \mathcal{U}_t\}$. Second, for any $\boldsymbol{x} \in \bar{\mathcal{U}}_t$, on the good situation $\mathcal{C}_t \cap \mathcal{E}_t^{\text{ts}} \cap \mathcal{E}_t^a$, we have

$$\tilde{\mu}_t(\boldsymbol{x}) \leq f(\boldsymbol{x}, u_t) + \gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}) \leq f(\boldsymbol{x}_t^*, u_t) < \tilde{\mu}_t(\boldsymbol{x}^*)$$

which leads to $\mathcal{E}_t^a \subseteq \{\tilde{\mu}_t(\boldsymbol{x}_t^*) > \max_{\boldsymbol{x} \in \bar{\mathcal{U}}_t} \tilde{\mu}_t(\boldsymbol{x})\}$

Therefore, on event $\mathcal{C}_t$, we have

$$\begin{aligned}\mathbb{P}_t(\{\boldsymbol{x}_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^{\text{ts}}) &\geq \mathbb{P}_t(\{\tilde{\mu}_t(\boldsymbol{x}_t^*) > \max_{\boldsymbol{x} \in \bar{\mathcal{U}}_t} \tilde{\mu}_t(\boldsymbol{x})\} \cap \mathcal{E}_t^{\text{ts}}) \\ &\geq \mathbb{P}_t(\mathcal{E}_t^a \cap \mathcal{E}_t^{\text{ts}}) \\ &\geq \mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})\end{aligned}$$

Now we have a upper bound for $\sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t)$:

$$\sigma_{u_t,t-1}(\bar{\boldsymbol{x}}_t) \leq \frac{\mathbb{E}_t[\sigma_{u_t,t-1}(\boldsymbol{x}_t)\mathbb{I}_{\mathcal{E}_t^{\text{ts}}}]}{\mathbb{P}_t(\{\boldsymbol{x}_t \in \mathcal{U}_t\} \cap \mathcal{E}_t^{\text{ts}})} \leq \frac{\mathbb{E}_t[\sigma_{u_t,t-1}(\boldsymbol{x}_t)]}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})}$$

which gives the upper bound for instantaneous regret by plugging above result in (6.21):

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{E}_t^{\text{ts}}}] \leq \left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})} + 1\right) \cdot \mathbb{E}_t[\gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)].$$

Therefore,

$$\mathbb{E}_t[\Delta_t \mathbb{I}_{\mathcal{C}_t}] \leq \mathbb{I}_{\mathcal{C}_t} \cdot \left\{\left(\frac{2}{\mathbb{P}_t(\mathcal{E}_t^a) - \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})} + 1\right) \cdot \mathbb{E}_t[\gamma_t \sigma_{u_t,t-1}(\boldsymbol{x}_t)] + B_\Delta \cdot \mathbb{P}_t(\bar{\mathcal{E}}_t^{\text{ts}})\right\}$$

□

6.6.3.3 Proof of Lemma 6.6.3

Proof. We first apply Cauchy-Schwartz inequality and obtain

$$\begin{aligned}\sum_{t=1}^{T}\sigma_{u_t,t-1}(\boldsymbol{x}_t) &\le \sqrt{T\sum_{t=1}^{T}\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}\\ &= \sqrt{\lambda T\sum_{t=1}^{T}\frac{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}{\lambda}}\\ &\le \sqrt{T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}\sum_{t=1}^{T}\min\{1,\frac{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}{\lambda}\}}\end{aligned} \tag{6.22}$$

where the last inequality is because

$$\frac{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}{\lambda} \le \min\{\frac{\alpha}{\lambda}\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}, \frac{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}{\lambda}\} \le \frac{\alpha}{\lambda}\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}\min\{1,\frac{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}{\lambda}\}$$

using $\sigma^2_{u_t,t-1}(\boldsymbol{x}_t) \le |K((\boldsymbol{x}_t,u_t)(\boldsymbol{x}_t,u_t))| \le \alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}$ and $\lambda \le \alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}$.

Then we apply the fact that $\min\{1,x\} \le 2\log(1+x)$ for $x \ge 0$ to obtain

$$\sum_{t=1}^{T}\min\{1,\frac{1}{\lambda}\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)\} \le 2\sum_{t=1}^{T}\log(1+\frac{1}{\lambda}\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)).$$

Now we can use the property of the Shur complement for K_t:

$$\begin{aligned}\det(\boldsymbol{I}_t+\frac{1}{\lambda}\boldsymbol{K}_t) =& \det(\boldsymbol{I}_{t-1}+\frac{1}{\lambda}\boldsymbol{K}_{t-1})\\ &\times\Big[1+\frac{1}{\lambda}\big(\underbrace{K((\boldsymbol{x}_t,u_t),(\boldsymbol{x}_t,u_t))-\boldsymbol{k}_{t-1}(\boldsymbol{x}_t,u_t)^\top(\boldsymbol{K}_{t-1}+\lambda\boldsymbol{I})^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}_t,u_t)}_{\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)}\big)\Big]\end{aligned}$$

which leads to

$$\sum_{t=1}^{T}\log(1+\frac{1}{\lambda}\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)) = \sum_{t=1}^{T}\log\frac{\det(\boldsymbol{I}_t+\frac{1}{\lambda}\boldsymbol{K}_t)}{\det(\boldsymbol{I}_{t-1}+\frac{1}{\lambda}\boldsymbol{K}_{t-1})} = \log\det(\boldsymbol{I}_T+\lambda^{-1}\boldsymbol{K}_T).$$

As a consequence, we have

$$\frac{1}{2}\sum_{t=1}^{T}\min\{1,\frac{1}{\lambda}\sigma^2_{u_t,t-1}(\boldsymbol{x}_t)\}\leq\log\det(\boldsymbol{I}_T+\lambda^{-1}\boldsymbol{K}_T).$$

Therefore, we combine above result with (6.22) and obtain

$$\sum_{t=1}^{T}\sigma_{u_t,t-1}(\boldsymbol{x}_t)\leq\sqrt{2T\alpha\|\boldsymbol{L}_\rho^{-1/2}\|_{\max}\cdot\log\det(\boldsymbol{I}_T+\lambda^{-1}\boldsymbol{K}_T)}$$

□

6.6.3.4 Proof of Lemma 6.6.5

Proof. We note that

$$\|\delta_t(\boldsymbol{x},u)\|^2_{\ell^2}=\|\phi((\boldsymbol{x},u))\|^2_{\ell^2}+\|\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\|^2_{\ell^2}-2\langle\phi((\boldsymbol{x},u)),\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\rangle_{\ell^2}$$

and we have

$$\begin{aligned}\|\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\|^2_{\ell^2}&=\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Phi}_t\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\\&=\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-1}K_t\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\\&=\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Sigma}_t\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)-\lambda\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-2}\boldsymbol{k}_t(\boldsymbol{x},u)\\&=\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)-\lambda\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-2}\boldsymbol{k}_t(\boldsymbol{x},u)\end{aligned}$$

and

$$\langle\phi((\boldsymbol{x},u)),\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)\rangle_{\ell^2}=\phi((\boldsymbol{x},u))^\top\boldsymbol{\Phi}_t^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u)=\boldsymbol{k}_t(\boldsymbol{x},u)^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u).$$

Putting above equalities together, we have

$$\begin{aligned}\|\delta_t(\boldsymbol{x},u)\|^2_{\ell^2} &= \|\phi((\boldsymbol{x},u))\|^2_{\ell^2} - \boldsymbol{k}_t(\boldsymbol{x},u)^\top \boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u) - \lambda \boldsymbol{k}_t(\boldsymbol{x},u)^\top \boldsymbol{\Sigma}_t^{-2}\boldsymbol{k}_t(\boldsymbol{x},u)\\ &= K((\boldsymbol{x},u),(\boldsymbol{x},u)) - \boldsymbol{k}_t(\boldsymbol{x},u)^\top \boldsymbol{\Sigma}_t^{-1}\boldsymbol{k}_t(\boldsymbol{x},u) - \lambda \boldsymbol{k}_t(\boldsymbol{x},u)^\top \boldsymbol{\Sigma}_t^{-2}\boldsymbol{k}_t(\boldsymbol{x},u)\\ &= \sigma^2_{u,t}(\boldsymbol{x}) - \lambda \boldsymbol{k}_t(\boldsymbol{x},u)^\top \boldsymbol{\Sigma}_t^{-2}\boldsymbol{k}_t(\boldsymbol{x},u)\\ &\leq \sigma^2_{u,t}(\boldsymbol{x})\end{aligned}$$

since $\boldsymbol{\Sigma}_t^{-1}$ is positive semindefinite.

□

6.6.3.5 Proof of Lemma 6.6.6

Proof. We first define

$$\boldsymbol{s}_t = \boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t = \sum_{s=1}^{t} \phi(\boldsymbol{x}_s, u_s)\epsilon_s.$$

Note that $\boldsymbol{s}_t$ is a martingale w.r.t $\mathcal{F}_t$.

Also we define a supermartingale

$$M_t(\boldsymbol{g}) = exp\big(\sum_{s=1}^{t} \frac{1}{\sigma}\langle \boldsymbol{g}, \boldsymbol{s}_t\rangle - \frac{1}{2}\|\boldsymbol{g}\|^2\big)$$

which has an alternative form

$$M_t(\boldsymbol{g}) = exp\big(\sum_{s=1}^{t} \frac{1}{\sigma}\langle \boldsymbol{g}, \phi(\boldsymbol{x}_s, u_s)\rangle\epsilon_s - \frac{1}{2}\|\boldsymbol{g}\|^2\big)$$

where $\boldsymbol{g}$ is the function vector with elements

We follow the approach from classical linear bandit (APS11), which is averaging $M_t(\boldsymbol{g})$ w.r.t a Gaussian distribution on $\boldsymbol{g}$. The key technical issue is the infinite dimension of the

function vector $\boldsymbol{g}$. We will first perform the truncated version which can precisely match the classical result. Let d be the dimension of the feature map. Our target is the obtain the limiting result when $d \to \infty$. Now assume $\boldsymbol{g}^d \sim \mathcal{N}(\mathbf{0}, \frac{1}{\lambda}\boldsymbol{I}_d)$, independent of everything else, and define

$$M_t^{(d)} = \mathbb{E}_{\boldsymbol{g}^d}[M_t(\boldsymbol{g}^d)] = \int M_t^{(d)}(\boldsymbol{g}) d\rho^d(\boldsymbol{g})$$

and by iterated expectation (i.e Fubini's theorem), we have

$$\mathbb{E}[M_t^{(d)} | \mathcal{F}_t] \leq M_{t-1}$$

which shows that M_t is a supermartingale.

Then we define $\Psi : \ell^2 \to \mathbb{R}^d$ as the truncation projection onto the first d coordinates: $\Psi_d \boldsymbol{\theta} = [\boldsymbol{\Theta}_1, \cdots, \boldsymbol{\Theta}_d]^\top$ for any $\boldsymbol{\theta} \in \ell^2$. We further denote

$$\boldsymbol{\Psi}_d \boldsymbol{\Phi}_t^\top = [\Psi_d \phi(\boldsymbol{x}_1, u_1), \cdots, \Psi_d \phi(\boldsymbol{x}_t, u_t)] \in \mathbb{R}^{d \times t}$$

and

$$\boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top = \boldsymbol{\Psi}_d \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t \boldsymbol{\Psi}_d.$$

We notices that

$$\frac{\det(\lambda \boldsymbol{I}_d)}{\det(\lambda \boldsymbol{I}_d + \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)} = \frac{1}{\det(\boldsymbol{I}_d + \lambda^{-1} \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)}$$

which leads to

$$M_t^{(d)} = \Big(\frac{\det(\lambda \boldsymbol{I}_d)}{\det(\lambda \boldsymbol{I}_d + \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)}\Big)^{1/2} \exp(\frac{1}{2\sigma^2} \|\boldsymbol{\Psi}_d \boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{(\lambda \boldsymbol{I}_d + \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)^{-1}})$$
$$= \det(\boldsymbol{I}_d + \lambda^{-1} \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)^{-1/2} \exp(\frac{1}{2\sigma^2} \|\boldsymbol{\Psi}_d \boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{(\lambda \boldsymbol{I}_d + \boldsymbol{\Psi}_d \boldsymbol{J}_t \boldsymbol{\Psi}_d^\top)^{-1}}).$$

Then taking the limit $d \to \infty$, we have the convergence martingale for $M_t^{(d)}$:

$$M_t = \det(\boldsymbol{I}_\infty + \lambda^{-1} \boldsymbol{J}_t)^{-1/2} \exp(\frac{1}{2\sigma^2} \|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{(\lambda \boldsymbol{I}_\infty + \boldsymbol{J}_t)^{-1}})$$
$$= \det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t)^{-1/2} \exp(\frac{1}{2\sigma^2} \|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{\Gamma_t^{-1}})$$

where the second step is from (Slyvestr) or Weinstein–Aronszajn identity. By Ville's inequality,

$$\mathbb{P}(\sup_{t=0,1,2,\cdots} M_t \geq \frac{1}{\delta}) \leq \mathbb{E}[M_0] \cdot \delta$$

and $M_0 = 1$. Thus we know that, with probability at least $1 - \delta$, for all $t = 0, 1, 2, \cdots$

$$\log(M_t) \leq \log(\frac{1}{\delta})$$

which leads to

$$-\frac{1}{2} \log\det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t) + \frac{1}{2\sigma^2} \|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{\Gamma_t^{-1}} \leq \log(\frac{1}{\delta}).$$

After re-arranging, we get

$$\|\boldsymbol{\Phi}_t \boldsymbol{\epsilon}_t\|^2_{\Gamma_t^{-1}} \leq 2\sigma^2 \log \frac{\sqrt{\det(\boldsymbol{I}_t + \lambda^{-1} \boldsymbol{K}_t)}}{\delta}.$$

which shows our result.

□

6.6.3.6 Proof of Lemma 6.6.4

Proof. Let us write $\boldsymbol{\Phi}_t = \boldsymbol{U}_t \boldsymbol{\Lambda}_t \boldsymbol{V}_t^\top$ as the singular value decomposition(SVD) of $\boldsymbol{\Phi}_t$. We have $\boldsymbol{\Lambda}_t = [\boldsymbol{\Lambda}_{1,t}, \boldsymbol{0}]$ where $\boldsymbol{\Lambda}_{1,t}$ is a $t \times t$ diagonal matrix with singular values of $\boldsymbol{\Phi}_t$. We also note that $\boldsymbol{\Sigma}_t \in \mathbb{R}^{t \times \infty}$ and $\boldsymbol{U}_t \in \mathbb{R}^{t \times t}$. We also have

$$\boldsymbol{J}_t = \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t = \boldsymbol{V}_t \begin{bmatrix} \boldsymbol{\Lambda}_{1,t}^2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{V}_t^\top$$

and similarly

$$\boldsymbol{K}_t = \boldsymbol{\Phi}_t \boldsymbol{\Phi}_t^\top = \boldsymbol{U}_t \boldsymbol{\Lambda}_{1,t}^2 \boldsymbol{U}_t^\top.$$

Then, we have

$$\boldsymbol{\Gamma}_t = \boldsymbol{V}_t \begin{bmatrix} \boldsymbol{\Lambda}_{1,t}^2 + \lambda \boldsymbol{I}_t & \boldsymbol{0} \\ \boldsymbol{0} & \lambda \boldsymbol{I}_\infty \end{bmatrix} \boldsymbol{V}_t^\top, \quad \boldsymbol{\Sigma}_t = \boldsymbol{U}_t (\boldsymbol{\Lambda}_{1,t}^2 + \lambda \boldsymbol{I}_t) \boldsymbol{U}_t^\top.$$

It is cleat to have the identity:

$$\boldsymbol{\Sigma}_t^{-1} \boldsymbol{\Phi}_t = \boldsymbol{\Phi}_t \boldsymbol{\Gamma}_t^{-1}$$

since both side equal $\boldsymbol{U}_t [\boldsymbol{D}_t, \boldsymbol{0}] \boldsymbol{V}_t^\top$ where $\boldsymbol{D}_t = \boldsymbol{\Lambda}_{1,t} (\boldsymbol{\Lambda}_{1,t}^2 + \lambda \boldsymbol{I}_t)^{-1}$, which is a diagonal matrix.

Next, we note that

$$\begin{aligned} \sigma_{u,t}^2(\boldsymbol{x}) &= K((\boldsymbol{x}, u), (\boldsymbol{x}, u)) - \boldsymbol{k}_t(\boldsymbol{x}, u)^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{k}_t(\boldsymbol{x}, u) \\ &= \phi(\boldsymbol{x}, u)^\top (\boldsymbol{I}_\infty - \boldsymbol{\Phi}_t^\top \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\Phi}_t) \phi(\boldsymbol{x}, u) \\ &= \phi(\boldsymbol{x}, u)^\top (\boldsymbol{I}_\infty - \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t \boldsymbol{\Gamma}_t^{-1}) \phi(\boldsymbol{x}, u) \end{aligned}$$

which is a norm of $\phi(\boldsymbol{x}, u)$ induced by matrix

$$\begin{aligned}
\boldsymbol{I}_\infty - \boldsymbol{\Phi}_t^\top \boldsymbol{\Phi}_t \boldsymbol{\Gamma}_t^{-1} &= \boldsymbol{I}_\infty - \boldsymbol{J}_t \boldsymbol{\Gamma}_t^{-1} \\
&= \boldsymbol{V}_t \begin{bmatrix} \lambda \boldsymbol{I}_t (\boldsymbol{\Lambda}_{1,t}^2 + \lambda \boldsymbol{I}_t)^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \lambda \boldsymbol{I}_\infty \end{bmatrix} \boldsymbol{V}_t^\top \\
&= \lambda \boldsymbol{V}_t \begin{bmatrix} (\boldsymbol{\Lambda}_{1,t}^2 + \lambda \boldsymbol{I}_t)^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_\infty \end{bmatrix} \boldsymbol{V}_t^\top \\
&= \lambda \boldsymbol{\Gamma}_t^{-1}.
\end{aligned}$$

Therefore, we have the other identity

$$\sigma_{u,t}^2(\boldsymbol{x}) = \lambda \|\phi(\boldsymbol{x}, u)\|_{\boldsymbol{\Gamma}_t^{-1}}^2.$$

□

6.6.4 Supplement to Experiments

This appendix provides full details of our synthetic environments, algorithm configurations, hyperparameter selection, implementation choices, ablations, and reporting protocol.

6.6.4.1 Synthetic Environments

Let $\mathcal{U} = \{1, \ldots, n\}$ denote users, $\mathcal{D} \subset \mathbb{R}^d$ the arm (context) space, and $m_t := |\mathcal{D}_t|$ the number of candidates shown at round t. We draw a global normalized context pool $\mathcal{D} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ with $\boldsymbol{x}^{(i)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ and $\boldsymbol{x}^{(i)} \leftarrow \boldsymbol{x}^{(i)}/\|\boldsymbol{x}^{(i)}\|$. At round t we present $\mathcal{D}_t$ by sampling m_t distinct items from $\mathcal{D}$ without replacement. One user u_t is served per round, drawn uniformly from $\mathcal{U}$ unless stated otherwise. Rewards are observed with additive noise $y_t = f(\boldsymbol{x}_t, u_t) + \epsilon_t$. We generate graphs, contexts, and ground-truth rewards under one linear regime (*Linear–GOB*) and two kernelized regimes (*Laplacian–Kernel* using GP draw and

representer draw).

User graph. We consider two graph random generators on $\mathcal{U}$. First random graph family is Erdős–Rényi (ER) random graphs: each (undirected) edge is present with probability p and weights $w_{ij} = 1$. We set $p = 0.2$ in our experiment. Second one is Radial basis function(RBF) random graphs: sample latent $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_q)$, set $w_{ij} = \exp(-\rho_L \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2^2)$, and sparsify by keeping edges with $w_{ij} \geq s$. We choose $s = 0.1$, $\rho_L = 0.1$ and $q = 4$ in our simulation.

Task Design. We design different level of the task. The simplest case is $(m, m_t, n, d, T) = (10, 5, 20, 5, 1000)$. This is a 10-arm bandit problem with 50% viewability at each round for all users. The medium level is $(m, m_t, n, d, T) = (20, 5, 20, 10, 3000)$ which leads to a 20-arm bandit problem with 25% viewability at each round for all users. We also have the toughest case using $(m, m_t, n, d, T) = (50, 5, 20, 20, 3000)$ which leads to a 50-arm bandit problem with 10% viewability at each round for all users. σ is set as 0.1 unless additional specification.

Regime 1: *Linear–GOB* (graph-smooth linear rewards). Sample initial user parameters $\boldsymbol{\Theta}_0 \in \mathbb{R}^{n \times d}$ with rows $\boldsymbol{\theta}_{0,i} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$. Enforce the graph homophily via Tikhonov smoothing(YE16):

$$\boldsymbol{\Theta} \;=\; \arg\min_{\tilde{\boldsymbol{\Theta}}} \; \|\tilde{\boldsymbol{\Theta}} - \boldsymbol{\Theta}_0\|_F^2 + \eta \,\mathrm{tr}(\tilde{\boldsymbol{\Theta}}^\top \boldsymbol{L} \tilde{\boldsymbol{\Theta}}) \;=\; (\boldsymbol{I}_n + \eta \boldsymbol{L})^{-1} \boldsymbol{\Theta}_0.$$

Thus $f(\boldsymbol{x}, u) = \boldsymbol{x}^\top \boldsymbol{\theta}_u$, where $\boldsymbol{\theta}_u$ is row u of $\boldsymbol{\Theta}$. The strength of the graph homophily η is set as 1.0 as default. But we also provide an ablation study on $\eta \in \{0.1, 1, 5, 1.0\}$.

Regime 2: *Laplacian-Kernel.* Our choice of the base kernel K_x over arms is *Squared Exponential* which are defined as

$$K_{\mathrm{SE}}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\ell^2)$$

where length-scale $\ell > 0$ and is set to be 1.0 in our experiment. Then we construct the *multi-user kernel* by the definition:

$$K((\boldsymbol{x}, u), (\boldsymbol{x}', u')) \;=\; [\boldsymbol{L}_\rho^{-1/2}]_{u,u'}\, K_x(\boldsymbol{x}, \boldsymbol{x}')$$

where we set $\rho = 0.01$ in our experiment.

Option A: *Laplacian-Kernel* with GP draw

We draw the joint values $\{f(\boldsymbol{x}, u)\}_{u \in \mathcal{U}, \boldsymbol{x} \in \mathcal{D}}$ from the zero-mean GP with covariance induced by K and fix f by interpolation on $\mathcal{D} \times \mathcal{U}$. Noise is $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.01 \cdot \text{range}(f)$.

Option B: *Laplacian-Kernel* with representer draw We consider the representer theorem for RKHS and sample the i.i.d. coefficients via $\alpha_{\boldsymbol{x},u} \sim \mathcal{N}(0, 1)$ on $\mathcal{D} \times \mathcal{U}$ and set

$$f(\boldsymbol{x}, u) \;=\; \sum_{u' \in \mathcal{U}, \boldsymbol{x}' \in \mathcal{D}} \alpha_{\boldsymbol{x},u}\, K\big((\boldsymbol{x}, u), (\boldsymbol{x}', u')\big).$$

6.6.4.2 Baselines

All methods face the *same* sequence $\{u_t, \mathcal{D}_t, \epsilon_t\}_{t=1}^T$ in each trial of each synthetic environment to ensure a fair comparison. Our experiment include the following baselines.

Per-User LinUCB(no graph).: We implement `Per-User LinUCB`, which ignores the whole graph and perform the linear bandit algorithm independently on each user.

Pooled LinUCBB(no graph).: We implement `Pooled LinUCB`, which ignores graph and personalization by treating the multi-user problem as a single agent bandit problem. Simply speaking, there is global linear UCB algorithm to solve the problem.

GP-UCB(no graph). We implement `GP-UCB`(CG17), which is the `IGP-UCB` from the previous study on GP and UCB (CG17). This is a kernelized baseline using K_x on arms only, ignoring the similarities across users (the Laplacian).

GoB.Lin. We implement `GoB.Lin`, which is the classical methods in gang-og-bandits problem (CGZ13). This is a Laplacian-regularized linear UCB algorithm on graph-whitened features (equivalent to `GraphUCB` with $\rho = 1$ i.e $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{L}$). The confidence scale in the algorithm is tuned from the table.

GraphUCB. We implement `GraphUCB`(YTD20), the Laplacian-regularized LinUCB. Also, the confidence scale in the algorithm is tuned from the table.

6.6.4.3 Centralized Protocol

At each t: sample $u_t \sim \mathrm{Unif}(\mathcal{U})$, present $\mathcal{D}_t$ (size m_t), select $\boldsymbol{x}_t \in \mathcal{D}_t$ per the algorithm, observe y_t, update our decision policy(model), and record $\Delta_t = \max_{\boldsymbol{x} \in \mathcal{D}_t} f(\boldsymbol{x}, u_t) - f(\boldsymbol{x}_t, u_t)$. Each configuration is repeated for R trials (final results use $R = 20$; preliminary/pilot tuning uses $R \in [5, 10]$).

6.6.4.4 Posterior Updates and Numerical Details

For GP-based methods we use a hybrid implementation, which is described as below.

Exact (Cholesky) phase: maintain $\boldsymbol{\Sigma}_t = K_t + \lambda \boldsymbol{I}$ and update via rank-one Cholesky for small t (cost $O(t^2)$ per step; initial inversion $O(t^3)$).

Recursive phase: switch to the rank-one recursions in (6.9), with $q_0 = K$ restricted to $\mathcal{D} \times \mathcal{U}$. This costs $O(nm)$ per update when applied to the whole grid $\mathcal{D} \times \mathcal{U}$.

By default we take $t_\star = \min\{1500,\ \lfloor n^{1/3} \rfloor m\}$ as the phase switch. We use Cholesky jitter 10^{-8}, clip negative variances to zero, and cache $K_x(\mathcal{D}, \mathcal{D})$. For large n we optionally apply graph spectral truncation $\boldsymbol{L}_\rho \approx \boldsymbol{U}_r \Lambda_r \boldsymbol{U}_r^\top$ (top-r eigenpairs), yielding $K \approx (\boldsymbol{U}_r \Lambda_r^{-1} \boldsymbol{U}_r^\top) \otimes K_x$.

6.6.4.5 Hyperparameters and Tuning

What is fixed across algorithms. For fairness, *base-kernel* hyperparameters are fixed inside each environment: the length-scale ℓ uses the median heuristic on $\mathcal{D}$, and the Laplacian ridge ρ is fixed.

What is tuned. Only the exploration scales are tuned by grid search on a pilot horizon ($T_{\text{pilot}} = 1500$ for medium/hard; $T_{\text{pilot}} = 1000$ for simple) using $R_{\text{pilot}} \in \{5, 10\}$:

Algorithm	Grid (pilot)
`LK-GP-UCB`, `GP-UCB`	$\beta \in \{0.5, 1, 2, 4\}$
`LK-GP-TS`	$\nu \in \{0.5, 1, 2, 4\}$
`GOB.Lin`, `GraphUCB`, `LinUCB` variants	$\alpha \in \{0.5, 1, 2, 4\}$

The best pilot setting (by mean pilot cumulative regret) is then *frozen* for the full-horizon evaluation. Noise/ridge λ in GP updates uses $\lambda \in \{\sigma^2, 0.5\sigma^2, 2\sigma^2\}$ on a small pilot if needed; otherwise $\lambda = \sigma^2$.

6.6.4.6 Ablations and Stress Tests

We report an ablation under the *medium, Laplacian-Kernel with GP Draw* environment (ER graph, fixed ℓ and ρ) on **Scalability in users (n):** $n \in \{20, 50, 100, 200\}$ with fixed (m, m_t, d, T) and graph generator. We plot final cumulative regret vs. n.

Part IV

Future Works and Conclusion

CHAPTER 7

Ongoing Projects and Future Works

7.1 Ongoing Projects

7.1.1 Graph Convolutional Logistic Bandit for Online Graph Classification

Problem View. Let $\mathcal{G}$ be a space of annotated graphs, where each element $G \in \mathcal{G}$ is represented as $G = (\boldsymbol{A}, \boldsymbol{X})$. Here, $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ denotes the adjacency matrix of an undirected graph with at most N nodes, while $\boldsymbol{X} \in \mathbb{R}^{N \times d}$ encodes node annotations or feature representations, where the i-th row of $\boldsymbol{X}$ corresponds to the d-dimensional feature vector of node i.

We consider a classification problem where, given an annotated graph $G \in \mathcal{G}$, the learner aims to predict its class label $Y \in [K]$. The key to optimal classification is the conditional distribution of Y given G, denoted by:

$$[\mu(G)]_k := \mathbb{P}(Y = k \mid G), \quad k \in [K].$$

Here, $\mu : \mathcal{G} \to \Delta^{K-1}$ maps an annotated graph to a probability vector over the K possible classes, where $\Delta^{K-1} := \{\boldsymbol{p} \in \mathbb{R}^K : \sum_i p_i = 1\}$ represents the K-dimensional probability simplex.

In this work, we investigate the *online* version of the classification problem, where at each round t, the learner observes $G_t \in \mathcal{G}$ and wishes to predict its label Y_t, given all the past observations. We formulate the problem as a K-armed bandit problem. Let $\mathcal{F}_t$ be the

history of interactions up to time t as $\mathcal{F}_t$, which is the information available to the learner at time t. Given G_t, the prediction for Y_t from learner can be thought of as taking an action a_t at time t, with $a_t \in [K]$. The learner then receives a bandit feedback as the reward signal, which is the indicator of correctly classifying the graph G_t, that is,

$$r_t = \mathbb{I}\{a_t = Y_t\}.$$

Let $\boldsymbol{\mu}_t = \mu(G_t)$ and simplify the notation with $\mu_{t,k} := [\boldsymbol{\mu}_t]_k$. Then,

$$\mathbb{E}_t[r_t] = \mathbb{P}_t(r_t = 1) = \big[\mathbb{P}(Y_t = \cdot \mid G_t)\big]_{a_t} = \mu_{t,a_t}$$

which implies that $r_t \mid \mathcal{F}_t \sim \mathsf{Ber}(\mu_{t,a_t})$, meaning that online graph classification can be viewed as a K-armed bandit with Bernoulli rewards, where the reward function $\mu(\cdot)$ determines the probabilities. Learning the classifier in online graph classification is equivalent to learning the reward function $\mu(\cdot)$ in the language of bandits. Note that this setting is a slightly special case of Bernoulli bandits in that $\mu(\cdot)$ maps to a probability simplex, enforcing that its elements sum to one, which is not necessary in the general cases.

A (randomized) policy $\pi(\cdot \mid G) \in \Delta_K$ induces $a_t \sim \pi(\cdot \mid G_t)$. The optimal action is $a_t^\star = \arg\max_k \mu_{t,k}$ and the cumulative regret is defined as $R_T = \sum_{t=1}^{T} \big(\mu_{t,a_t^\star} - \mu_{t,a_t}\big)$.

GCN-Logistic Bandit. We propose to learn the unknown reward function $\mu(\cdot)$ by fitting a Graph Convolutional Networks (GCN) as an embedding model. In each convolution layer, we use neighbor aggregation which is referred as a BLOCK operation. After iterations of aggregating, the representation of an entire graph is then obtained through a graph-level pooling which is referred to a READOUT operation. Formally, let $\mathcal{N}_i$ denote the neighborhood of node i (including i itself), and let $\boldsymbol{\phi}_i^{(0)} := \boldsymbol{X}_{i*}$ be the initial feature vector.

The GCN representation is defined recursively as:

$$\boldsymbol{\phi}_i^{(l)} = \text{BLOCK}^{(l)}(i) := \frac{1}{\sqrt{m}} \cdot \text{ReLU}(\boldsymbol{W}^{(l)} \sum_{j \in \mathcal{N}_i} \hat{c}_{ij} \cdot \boldsymbol{\phi}_j^{(l-1)}), \quad 1 \le l \le L,$$
$$\phi(G; \boldsymbol{W}) := \text{READOUT}(\{\boldsymbol{\phi}_i^{(L)}, i \in [N]\}) := \frac{1}{N} \sum_{i \in [N]} \boldsymbol{\phi}_i^{(L)}$$

Here, $\text{ReLU}(\cdot) = \max(\cdot, 0)$ is the Rectified Linear Unit (ReLU) activation function. $\hat{c}_{ij} := (\deg(i) + 1)^{-1/2} \cdot (\deg(j) + 1)^{-1/2}$ normalizes feature aggregation where $\deg(v)$ represents degree of the node v. m is the width of the neural network. The model parameters include $\boldsymbol{W} := (\boldsymbol{W}^{(l)})_{l \in [L]}$ with weight matrices $\boldsymbol{W}^{(l)} \in \mathbb{R}^{m \times m}$ for $1 < l < L$, while the first and last layers have dimensions $\boldsymbol{W}^{(1)} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{W}^{(L)} \in \mathbb{R}^{d \times m}$.

We consider the multinomial logistic bandit problem (AT21; LK24), a specialized instance of generalized linear bandits (KZS20). Concretely, we apply the softmax link function to estimate the probability vector (output of the reward function) and the predictions on the reward function is:

$$[\widehat{\mu}(G; \boldsymbol{\Theta})]_k = \frac{\exp(\phi(G; \boldsymbol{W})^\top \boldsymbol{\theta}_k)}{\sum_{i=1}^K \exp(\phi(G; \boldsymbol{W})^\top \boldsymbol{\theta}_i)}, \quad \forall k \in [K]$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_K^\top]^\top \in \mathbb{R}^{Kd}$ are the bandit parameters for the K classes and $\boldsymbol{\Theta} = (\boldsymbol{W}, \boldsymbol{\theta})$ represents the collection of learnable parameters. Inspired from neural linear bandit learning (ZM19; XWZ20), we treat the encoder output $\phi(G; \boldsymbol{W})$ as the logit and place uncertainty on the final linear layer $\boldsymbol{\theta}$. We denote $\boldsymbol{z}_t = \phi(G_t; \boldsymbol{W}_{t-1})$ and $\widehat{\mu}_{k,t} = [\widehat{\mu}(G_t; \boldsymbol{\Theta}_{t-1})]_k$. For each class k, maintain

$$\boldsymbol{V}_{k,t} = \lambda \boldsymbol{I}_m + \sum_{s \le t} \widehat{\mu}_{k,s}(1 - \widehat{\mu}_{k,s})\, \boldsymbol{z}_s \boldsymbol{z}_s^\top,$$

and update $\boldsymbol{V}_{k,t}^{-1}$ via Sherman–Morrison. This approximates the (class-k) Fisher curvature of the softmax likelihood, yielding a predictive variance $s_{k,t}(\boldsymbol{z}) = \sqrt{\boldsymbol{z}^\top \boldsymbol{V}_{k,t}^{-1} \boldsymbol{z}}$. We use UCB

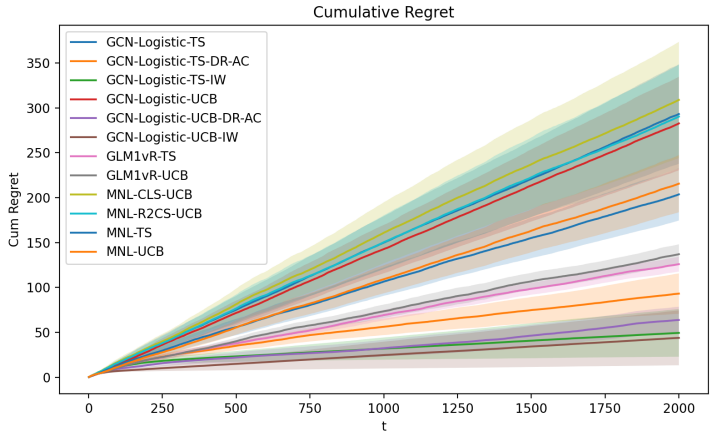

Figure 7.1: Cumulative Regret. Proposed `GCN-Logistic` algorithms have the best performance.

and TS as the bandit exploration strategy, applying on the logits. Formally, we define

$$\ell_{k,t}^{\text{ucb}} = \boldsymbol{\theta}_{k,t-1}^{\top} \boldsymbol{z}_t + \beta_t \cdot s_{k,t-1}(\boldsymbol{z}_t) \tag{7.1}$$

$$\ell_{k,t}^{\text{ts}} \sim \mathcal{N}(\boldsymbol{\theta}_{k,t-1}^{\top} \boldsymbol{z}_t,\ \nu^2 s_{k,t-1}^2(\boldsymbol{z}_t)). \tag{7.2}$$

At each round t, we update parameters $\boldsymbol{W}$ and $\boldsymbol{\theta}$ by minimizing the partial negative log-likelihood loss,

$$\mathcal{L}_t^{\text{PNLL}}(\boldsymbol{\Theta}) = -\Big[r_t \log[\widehat{\mu}(G_t; \boldsymbol{\Theta})]_{a_t} + (1 - r_t) \log\big(1 - [\widehat{\mu}(G_t; \boldsymbol{\Theta})]_{a_t}\big)\Big] + \frac{\lambda}{2} \|\boldsymbol{\Theta}\|_F^2.,$$

where the penalty is define as the sum of squared elements: $\|\boldsymbol{\Theta}\|_F^2 = \sum_{l=1}^{L} \|\boldsymbol{W}^{(l)}\|_F^2 + \|\boldsymbol{\theta}\|_2^2$. We use 3 Adam optimizer steps on $\mathcal{L}_t^{\text{PNLL}}(\boldsymbol{\Theta})$, backpropagating through the encoder. We proposed algorithms `GCN-Logistic-UCB` with decision rule $a_t = \arg\max_{k \in [K]} \ell_{k,t}^{\text{ucb}}$ and `GCN-Logistic-TS` with decision rule $a_t = \arg\max_{k \in [K]} \ell_{k,t}^{\text{ts}}$, using the estimated logit in (7.1) and (7.2).

Importance-Weighted Policy Learning. Naively training policy with supervised cross-entropy on bandit data is *biased*, because only the chosen action's outcome is revealed.

Importance-Weighted (IW) provides an *unbiased* stochastic gradient for the expected reward objective

$$J(\boldsymbol{\Theta}) \;=\; \mathbb{E}_{G\sim\mathcal{D}}\left[\sum_{k=1}^{K}\pi_{\boldsymbol{\Theta}}(k \mid G)\mathbb{P}(Y=k|G)\right] \tag{7.3}$$

where the policy in our design is $\pi_{\boldsymbol{\Theta}}(\cdot \mid G) = \widehat{\mu}(G;\boldsymbol{\Theta})$. If actions are sampled from a known logging distribution $\widehat{\mu}_t = [\widehat{\mu}_{1,t},\cdots,\widehat{\mu}_{K,t}]$, the single-step gradient estimator $\boldsymbol{g}_t^{\mathrm{IW}} = (r_t/\widehat{\mu}_{a_t,t})\,\nabla_{\boldsymbol{\Theta}}\log\pi_{\boldsymbol{\Theta}}(a_t \mid G_t)$ satisfies the unbiasedness for $\nabla_{\boldsymbol{\Theta}}J(\boldsymbol{\Theta})$ (up to a constant baseline). This yields **bandit-consistent** policy learning which optimizing the expected reward (7.3) using an unbiased bandit gradient with propensity $\widehat{\mu}_{a_t,t}$:

$$\mathcal{L}_t^{\mathrm{IW}}(\boldsymbol{\Theta}) = -\frac{r_t}{\widehat{\mu}_{a_t,t}}\log\pi_{\boldsymbol{\Theta}}(a_t \mid G_t) - \eta H(\pi_{\boldsymbol{\Theta}}(\cdot \mid G_t)), \tag{7.4}$$

where $H(\pi_{\boldsymbol{\Theta}}(\cdot \mid G_t))$ is the entropy of distribution $\pi_{\boldsymbol{\Theta}}(\cdot \mid G_t)$ and η is the regularization parameter for entropy. We proposed algorithms `GCN-Logistic-UCB-IW` and `GCN-Logistic-TS-IW` with decision rule in (7.1) and (7.2) and loss (7.4).

Doubly Robust Actor-Critic. While IW is unbiased, it may suffer from high variance when $\widehat{\mu}_{a_t,t}$ is small or rewards are sparse. Doubly Robust(DR) augments IW with a *critic* $\hat{\mu}_{\psi}(k \mid G)$ and enjoys two key properties: variance reduction and double robustness. Operationally, DR inherits the exploration/optimization decoupling of IW, adds a supervised learning signal for the critic, and often accelerates convergence for deep encoders (lower gradient noise, better credit assignment). Concretely, we train a critic $\widehat{\mu}_k(\boldsymbol{\psi};G) \in [0,1]$ to predict per-class correctness probabilities and form the DR actor loss

$$\mathcal{L}_t^{\mathrm{DR}}(\boldsymbol{\Theta}) = -\frac{r_t - \widehat{\mu}_{a_t}(\boldsymbol{\psi};G_t)}{\widehat{\mu}_{a_t,t}}\log\pi_{\boldsymbol{\Theta}}(a_t \mid G_t) - \sum_{k=1}^{K}\widehat{\mu}_k(\boldsymbol{\psi};G_t)\log\pi_{\boldsymbol{\Theta}}(k \mid G_t), \tag{7.5}$$

with critic update

$$\mathcal{L}_t^{\mathrm{critic}}(\boldsymbol{\Theta}) = \frac{1}{\widehat{\mu}_{a_t,t}}\mathrm{BCE}(\widehat{\mu}_{a_t}(\boldsymbol{\psi};G_t), r_t), \tag{7.6}$$

Algorithm	Accuracy	Std
GCN-Logistic-UCB-IW	0.4929	0.3131016705
GCN-Logistic-TS-IW	0.490025	0.2722978345
GCN-Logistic-UCB-DR-AC	0.482925	0.1506766254
GCN-Logistic-TS-DR-AC	0.46595	0.2307075206
GLM-TS	0.448975	0.08446520182
GLM-UCB	0.442975	0.1131839893
MNL-TS	0.411275	0.2990837531
MNL-UCB	0.401975	0.3264181717
GCN-Logistic-UCB	0.367425	0.5345457437
MNL-CLS-TS	0.364825	0.5932808051
GCN-Logistic-TS	0.3653	0.5680976217
MNL-CLS-UCB	0.35585	0.6636975821

Table 7.1: Accuracy on online 3-classes/arms task(K=3).

where BCE$(\cdot,\cdot)$ is the binary cross entropy for two discrete distribution. The DR gradient is unbiased if either propensities are correct (they are, by construction) or the critic is correctly specified; variance shrinks as the critic improves. We proposed algorithms `GCN-Logistic-UCB-DR-AC` and `GCN-Logistic-TS-DR-AC` with decision rule in (7.1) and (7.2) and loss (7.5) and (7.6).

Experiment. We experiment the six proposed algorithms `GCN-Logistic-*`, mentioned above, comparing to ε-greedy GCN (argmax with ε noise, for reference only), GLM-bandit on fixed graph features (e.g., WL kernel or degree histograms); Supervised Oracle (uses labels; upper bound, not available to learners) on synthetic data environment. Our synthetic generator is stochastic block or random geometric graphs with controllable *homophily*, *class margins* $\Delta = \mu_{(1)} - \mu_{(2)}$, and feature noise. We experiment on online 3-classes/arms task(K=3).

The encoder in our experiment is a 3 layer GCN with hidden dimension $m = 128$, dropout probability 0.2. The optimizer is AdamW using learning rate 2×10^{-3} for encoder, 1×10^{-3} for head and weight decay 10^{-4}, 1.0 gradient clip. Also, for the logits, we use $\lambda = 1$ and operate

played-class update by default. We tune confidence hyperparameter $\beta_t = c\sqrt{\log(1+t)}$ with $c \in [0.1, 2]$. For IW/DR knobs, we set entropy weight $\eta \in [0, 0.01]$, critic EMA target; replay window 1000k for IW/DR.

7.2 Future Works

Graph Neural Bandit with Random Graph Generative Models A promising direction is to make bandit policies *model-based* with respect to a random-graph generative assumption. We consider two complementary settings: (i) *attributes-given-structure* , where a fixed graph G is given and node/edge attributes (including rewards) arise from a probabilistic model tied to G; and (ii) *structure-first* , where a random graph $G \sim \mathcal{P}_G$ (e.g., SBM, random geometric, graphon) is drawn and attributes are subsequently generated conditional on G. The objective is twofold: first, derive UCB/TS-style algorithms whose estimators are the *Bayes* predictors under the chosen model; second, identify the corresponding graph neural architecture (spectral GNN, GCN, or MPNN) that is theoretically optimal for that model class.

We will incorporate the results that linear graph convolution or spectral GNN is the Bayes predictor under the Gasussian Markov random field (JB22) and build the bandit algoroithms on the bandit problems with graphs. Determining *which* GNN family is optimal follows from the generative assumptions. If the Bayes predictor is a low-degree polynomial in $\boldsymbol{L}$ (short correlation length), a first-order *GCN* is sufficient. If correlations are non-stationary, edge-typed, or heterophilous, the Bayes class cannot be expressed by a single shift-invariant filter; in this case a *message passing* (MPNN) family with edge-conditioned updates becomes appropriate. We will target at formalizing a model-to-architecture map using marginal likelihood or PAC-Bayes criteria to select between GNNs, and then lift the selection into UCB/TS with oracle-style regret bounds.

Bandit on Item-Users Graphs for Online Recommendation. A promising direction is to extend the multi-agent bandit on graphs to recommendation settings where both items and users/agents are jointly represented in a bipartite or knowledge graph. In such a graph, user nodes are connected to item nodes through interactions such as clicks, ratings, or purchases. Unlike classical contextual bandits that model each user–item interaction independently, the graph structure provides a natural way to capture collaborative signals and transfer knowledge across related users and items. Decision in this problem is to recommend an item to a user on the given graph. One avenue is to design bandit algorithms that adaptively learn representations on the item–user graph using graph neural networks or spectral methods, balancing exploration and exploitation while respecting the graph topology. For example, exploration strategies can be guided by structural uncertainty in the graph embeddings, or by leveraging homophily and community structures to generalize from observed interactions to unobserved ones. Such approaches would not only provide principled regret guarantees but also connect recommendation bandits more closely with modern graph learning techniques, paving the way for scalable and theoretically grounded online recommendation systems.

CHAPTER 8

Conclusion

Artificial intelligence is increasingly deployed in settings where decisions are made repeatedly, under uncertainty, and in the presence of relationships among entities such as customers connected by social ties, products linked by content similarity, sensors coupled by physical proximity, and molecules joined in interaction networks. Treating these relationships as *graphs* changes what "data efficiency" and "good exploration" mean in practice: information gathered for one entity can accelerate learning for its neighbors, while careless exploration can propagate errors across a network. This thesis adopts that perspective. It develops a graph centered view of sequential decision making and shows how graph structure, whether present in the data we act on or in the environment we operate within, can be turned into better decisions at scale. The result is a toolkit suited to recommendation, online advertising, marketplaces, scientific discovery, and other networked applications of AI that involve sequential decisions.

My first project centers on graph neural bandits with Thompson sampling (`GNN-TS`) in "graph as data" problems, where actions themselves are graphs or graph structured objects (for example, selecting a molecular graph, classifying a network snapshot, or choosing a subgraph for further measurement). The key insight is that modern GNN representations, when paired with principled uncertainty estimates, enable targeted exploration that respects the inductive biases of graph learning such as locality, message passing, and spectral smoothing, while avoiding the brittleness that purely optimistic (UCB style) heuristics can exhibit in sparse or noisy regimes. Empirically, `GNN-TS` improves early performance and sustains gains

as feedback accumulates, which matters in recommendation cold start, fraud detection on evolving transaction graphs, and scientific design loops where each experiment is costly. Conceptually, this work connects the success of GNNs in supervised settings with the demands of online decision making, showing how to turn strong representational priors into safer and faster exploration policies.

The second project focuses on "graph as structure" problems and introduces Laplacian Kernelized Bandits for multi user decision making on a known user graph. Here the graph is not the action; it is the medium through which learning should generalize, because users connected in a network often share preferences and ignoring that structure wastes data. By embedding the user graph in the learning rule using a kernel induced by the Laplacian and maintaining Gaussian process posteriors, the algorithms provide calibrated uncertainty over user item pairs and guide exploration that respects the structure. Beyond strong performance in both linear and nonlinear regimes, the contribution is conceptual: it offers a clean, unified way to inject homophily and neighborhood smoothing into bandit updates, clarifies when network sharing helps (and when it does not), and integrates smoothly with practical concerns such as partial visibility, cold start users, and scalable updates. These properties are directly relevant for large scale personalization systems, social platforms, and any service in which decisions for one person should learn from similar people while preserving individual nuance.

Building on these foundations, the thesis outlines ongoing and future directions with clear practical applications. One strand is a GCN logistic bandit for online multi class classification under bandit feedback, aimed at tasks like content moderation, risk triage, and real time quality control where labels are discrete, costly, and partially observed. The goal is to pair graph classifiers with calibrated exploration that respects operational constraints such as safety and latency and product goals such as fairness and coverage. We also note, briefly, two additional avenues for future work: model based graph neural bandits grounded in random graph generative models, and bandits on user item graphs for recommendation.

In summary, this thesis advances that graphs should be treated as a *first class inductive bias* in sequential decision making. On the "graph as data" side, `GNN-TS` shows how powerful representations and principled exploration can be fused to act effectively on structured objects. On the "graph as structure" side, Laplacian kernelization shows how to make personalization and multi agent learning data efficient, stable, and scalable by leveraging known relationships among decision units. Together, these contributions move graph based AI beyond static prediction toward reliable, high impact decision systems that learn while serving, adapt under shifting conditions, and do so with calibrated uncertainty and explicit use of structure.

Bibliography

[AAM24] Abhineet Agarwal, Anish Agarwal, Lorenzo Masoero, and Justin Whitehouse. "Mutli-Armed Bandits with Network Interference." *Advances in Neural Information Processing Systems*, **37**:36414–36437, 2024.

[ADH19] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. "On exact computation with an infinitely wide neural net." *Advances in neural information processing systems*, **32**, 2019.

[AG13] Shipra Agrawal and Navin Goyal. "Thompson sampling for contextual bandits with linear payoffs." In *International conference on machine learning*, pp. 127–135. PMLR, 2013.

[AL17] Marc Abeille and Alessandro Lazaric. "Linear thompson sampling revisited." In *Artificial Intelligence and Statistics*, pp. 176–184. PMLR, 2017.

[APS11] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. "Improved algorithms for linear stochastic bandits." *Advances in neural information processing systems*, **24**, 2011.

[AT21] Sanae Amani and Christos Thrampoulidis. "Ucb-based algorithms for multinomial logistic regression bandits." *Advances in Neural Information Processing Systems*, **34**:2913–2924, 2021.

[BAY21] Shaked Brody, Uri Alon, and Eran Yahav. "How attentive are graph attention networks?" *arXiv preprint arXiv:2105.14491*, 2021.

[BDD19] Marc Bellemare, Will Dabney, Robert Dadashi, Adrien Ali Taiga, Pablo Samuel Castro, Nicolas Le Roux, Dale Schuurmans, Tor Lattimore, and Clare Lyle. "A geometric perspective on optimal representations for reinforcement learning." *Advances in neural information processing systems*, **32**, 2019.

[BEL21] Sébastien Bubeck, Ronen Eldan, and Yin Tat Lee. "Kernel-based methods for bandit convex optimization." *Journal of the ACM (JACM)*, **68**(4):1–35, 2021.

[BM19] Alberto Bietti and Julien Mairal. "On the inductive bias of neural tangent kernels." *Advances in Neural Information Processing Systems*, **32**, 2019.

[BNS06] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples." *Journal of machine learning research*, **7**(11), 2006.

[CG17] Sayak Ray Chowdhury and Aditya Gopalan. "On kernelized multi-armed bandits." In *International Conference on Machine Learning*, pp. 844–853. PMLR, 2017.

[CGZ13] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. "A gang of bandits." *Advances in neural information processing systems*, **26**, 2013.

[CL11] Olivier Chapelle and Lihong Li. "An empirical evaluation of thompson sampling." *Advances in neural information processing systems*, **24**, 2011.

[CLR11] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. "Contextual bandits with linear payoff functions." In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214. JMLR Workshop and Conference Proceedings, 2011.

[CLZ21] Houshuang Chen, Shuai Li, Chihao Zhang, et al. "Understanding bandits with graph feedback." *Advances in Neural Information Processing Systems*, **34**:24659–24669, 2021.

[DBV16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering." *Advances in neural information processing systems*, **29**, 2016.

[DCK21] Yihan Du, Wei Chen, Yuko Kuroki, and Longbo Huang. "Collaborative pure exploration in kernel bandit." *arXiv preprint arXiv:2110.15771*, 2021.

[DMM20] Christoph Dann, Yishay Mansour, Mehryar Mohri, Ayush Sekhari, and Karthik Sridharan. "Reinforcement learning with feedback graphs." *Advances in Neural Information Processing Systems*, **33**:16868–16878, 2020.

[DSL22] Zhongxiang Dai, Yao Shu, Bryan Kian Hsiang Low, and Patrick Jaillet. "Sample-then-optimize batch neural thompson sampling." *Advances in Neural Information Processing Systems*, **35**:23331–23344, 2022.

[Dub20] Abhimanyu Dubey et al. "Kernel methods for cooperative multi-agent contextual bandits." In *International Conference on Machine Learning*, pp. 2740–2750. PMLR, 2020.

[EFH22] Emmanuel Esposito, Federico Fusco, Dirk van der Hoeven, and Nicolò Cesa-Bianchi. "Learning on the edge: Online learning with stochastic feedback graphs." *Advances in Neural Information Processing Systems*, **35**:34776–34788, 2022.

[FTC24] Yuting Feng, Vincent YF Tan, and Bogdan Cautis. "Influence Maximization via Graph Neural Bandits." In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 771–781, 2024.

[GH20] Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. "Constrained Bayesian optimization for automatic chemical design using variational autoencoders." *Chemical science*, **11**(2):577–586, 2020.

[GKK24] Quanquan Gu, Amin Karbasi, Khashayar Khosravi, Vahab Mirrokni, and Dongruo Zhou. "Batched neural bandits." *ACM/IMS Journal of Data Science*, **1**(1):1–18, 2024.

[GMG24a] Gianmarco Genalti, Marco Mussi, Nicola Gatti, Marcello Restelli, Matteo Castiglioni, and Alberto Maria Metelli. "Bridging Rested and Restless Bandits with Graph-Triggering: Rising and Rotting." *arXiv preprint arXiv:2409.05980*, 2024.

[GMG24b] Gianmarco Genalti, Marco Mussi, Nicola Gatti, Marcello Restelli, Matteo Castiglioni, Alberto Maria Metelli, et al. "Graph-triggered rising bandits." *PROCEEDINGS OF MACHINE LEARNING RESEARCH*, **235**:15351–15380, 2024.

[GWD18] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. "Automatic chemical design using a data-driven continuous representation of molecules." *ACS central science*, **4**(2):268–276, 2018.

[GYZ23] Yutian Gou, Jinfeng Yi, and Lijun Zhang. "Stochastic graphical bandits with heavy-tailed rewards." In *Uncertainty in Artificial Intelligence*, pp. 734–744. PMLR, 2023.

[GZ23] Xueping Gong and Jiheng Zhang. "Stochastic Graph Bandit Learning with Side-Observations." *arXiv preprint arXiv:2308.15107*, 2023.

[HLD21] Botao Hao, Tor Lattimore, and Wei Deng. "Information directed sampling for sparse linear bandits." *Advances in Neural Information Processing Systems*, **34**:16738–16750, 2021.

[HT14] Junya Honda and Akimichi Takemura. "Optimality of Thompson sampling for Gaussian bandits depends on priors." In *Artificial Intelligence and Statistics*, pp. 375–383. PMLR, 2014.

[HWW22] Mingguo He, Zhewei Wei, and Ji-Rong Wen. "Convolutional neural networks on graphs with chebyshev approximation, revisited." *Advances in neural information processing systems*, **35**:7264–7276, 2022.

[ICM22] Alexandra Iacob, Bogdan Cautis, and Silviu Maniu. "Contextual bandits for advertising campaigns: A diffusion-model independent approach." In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pp. 513–521. SIAM, 2022.

[IMB22] Tsuyoshi Idé, Keerthiram Murugesan, Djallel Bouneffouf, and Naoki Abe. "Targeted advertising on social networks using online variational tensor regression." *arXiv preprint arXiv:2208.10627*, 2022.

[JB22] Junteng Jia and Austin R Benson. "A unifying generative model for graph learning algorithms: Label propagation, graph convolutions, and combinations." *SIAM Journal on Mathematics of Data Science*, **4**(1):100–125, 2022.

[JBJ18] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. "Junction tree variational autoencoder for molecular graph generation." In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks." *Advances in neural information processing systems*, **31**, 2018.

[JMR19] Olivier Jeunen, Dmytro Mykhaylov, David Rohde, Flavian Vasile, Alexandre Gilotte, and Martin Bompaire. "Learning from bandit feedback: An overview of the state-of-the-art." *arXiv preprint arXiv:1909.08471*, 2019.

[KBK15] Jaya Kawale, Hung H Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. "Efficient Thompson Sampling for Online Matrix-Factorization Recommendation." *Advances in neural information processing systems*, **28**, 2015.

[KK22] Parnian Kassraie and Andreas Krause. "Neural contextual bandits without regret." In *International Conference on Artificial Intelligence and Statistics*, pp. 240–278. PMLR, 2022.

[KKB22] Parnian Kassraie, Andreas Krause, and Ilija Bogunovic. "Graph neural network bandits." *Advances in Neural Information Processing Systems*, **35**:34519–34531, 2022.

[KMK20] Tomáš Kocák, Rémi Munos, Branislav Kveton, Shipra Agrawal, and Michal Valko. "Spectral bandits." *The Journal of Machine Learning Research*, **21**(1):9003–9046, 2020.

[KSG19a] Branislav Kveton, Csaba Szepesvari, Mohammad Ghavamzadeh, and Craig Boutilier. "Perturbed-history exploration in stochastic linear bandits." *arXiv preprint arXiv:1903.09132*, 2019.

[KSG19b] Branislav Kveton, Csaba Szepesvari, Mohammad Ghavamzadeh, and Craig Boutilier. "Perturbed-history exploration in stochastic multi-armed bandits." *arXiv preprint arXiv:1902.10089*, 2019.

[KT19] Baekjin Kim and Ambuj Tewari. "On the optimality of perturbations in stochastic and adversarial multi-armed bandit problems." *Advances in Neural Information Processing Systems*, **32**, 2019.

[KVM14] Tomáš Kocák, Michal Valko, Rémi Munos, and Shipra Agrawal. "Spectral thompson sampling." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

[KXK20] Ksenia Korovina, Sailun Xu, Kirthevasan Kandasamy, Willie Neiswanger, Barnabas Poczos, Jeff Schneider, and Eric Xing. "Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations." In *International Conference on Artificial Intelligence and Statistics*, pp. 3393–3403. PMLR, 2020.

[KZL22] Fang Kong, Yichi Zhou, and Shuai Li. "Simultaneously learning stochastic and adversarial bandits with general graph feedback." In *International Conference on Machine Learning*, pp. 11473–11482. PMLR, 2022.

[KZS20] Branislav Kveton, Manzil Zaheer, Csaba Szepesvari, Lihong Li, Mohammad Ghavamzadeh, and Craig Boutilier. "Randomized exploration in generalized linear bandits." In *International Conference on Artificial Intelligence and Statistics*, pp. 2066–2076. PMLR, 2020.

[LK24] Kyeongyeol Lee, Ilmun Kim, et al. "Improved Confidence Sets for Multinomial Logistic Bandits." In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

[LLZ20] Chung-Wei Lee, Haipeng Luo, and Mengxiao Zhang. "A closer look at small-loss bounds for bandits with graph feedback." In *Conference on Learning Theory*, pp. 2516–2564. PMLR, 2020.

[LO24] Joongkyu Lee and Min-hwan Oh. "Nearly minimax optimal regret for multinomial logistic bandit." *Advances in Neural Information Processing Systems*, **37**:109003–109065, 2024.

[LS20] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[LSL16] Peter Landgren, Vaibhav Srivastava, and Naomi Ehrich Leonard. "On distributed cooperative decision-making in multiarmed bandits." In *2016 European Control Conference (ECC)*, pp. 243–248. IEEE, 2016.

[LTW20] Thodoris Lykouris, Eva Tardos, and Drishti Wali. "Feedback graph regret bounds for Thompson sampling and UCB." In *Algorithmic Learning Theory*, pp. 592–614. PMLR, 2020.

[LWW22] Chuanhao Li, Huazheng Wang, Mengdi Wang, and Hongning Wang. "Communication efficient distributed learning for kernelized contextual bandits." *Advances in Neural Information Processing Systems*, **35**:19773–19785, 2022.

[LZS18] Fang Liu, Zizhan Zheng, and Ness Shroff. "Analysis of thompson sampling for graphical bandits without the graphs." *arXiv preprint arXiv:1805.08930*, 2018.

[NGN21] Thanh Nguyen-Tang, Sunil Gupta, A Tuan Nguyen, and Svetha Venkatesh. "Offline neural contextual bandits: Pessimism, optimization and generalization." *arXiv preprint arXiv:2111.13807*, 2021.

[OI19] Min-hwan Oh and Garud Iyengar. "Thompson sampling for multinomial logit contextual bandits." *Advances in Neural Information Processing Systems*, **32**, 2019.

[QBH22] Yunzhe Qi, Yikun Ban, and Jingrui He. "Neural bandit with arm group graph." In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1379–1389, 2022.

[QBH23] Yunzhe Qi, Yikun Ban, and Jingrui He. "Graph neural bandits." In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1920–1931, 2023.

[RSP25] Alessio Russo, Yichen Song, and Aldo Pacchiano. "Pure exploration with feedback graphs." *arXiv preprint arXiv:2503.07824*, 2025.

[RTS18] Carlos Riquelme, George Tucker, and Jasper Snoek. "Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling." *arXiv preprint arXiv:1802.09127*, 2018.

[RV14] Daniel Russo and Benjamin Van Roy. "Learning to optimize via information-directed sampling." *Advances in neural information processing systems*, **27**, 2014.

[RVK18] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. "A tutorial on thompson sampling." *Foundations and Trends® in Machine Learning*, **11**(1):1–96, 2018.

[SBH13] Balazs Szorenyi, Róbert Busa-Fekete, István Hegedus, Róbert Ormándi, Márk Jelasity, and Balázs Kégl. "Gossip-based distributed stochastic bandit algorithms." In *International conference on machine learning*, pp. 19–27. PMLR, 2013.

[SKK09] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. "Gaussian process optimization in the bandit setting: No regret and experimental design." *arXiv preprint arXiv:0912.3995*, 2009.

[SNF13] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." *IEEE signal processing magazine*, **30**(3):83–98, 2013.

[TDD17] Aristide Tossou, Christos Dimitrakakis, and Devdatt Dubhashi. "Thompson sampling for stochastic bandits with graph feedback." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[TF23] Laura Toni and Pascal Frossard. "Online network source optimization with graph-kernel MAB." In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 242–258. Springer, 2023.

[Tho33] William R Thompson. "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples." *Biometrika*, **25**(3-4):285–294, 1933.

[TMR22] Parth Thaker, Mohit Malu, Nikhil Rao, and Gautam Dasarathy. "Maximizing and Satisficing in Multi-armed Bandits with Graph Information." *Advances in Neural Information Processing Systems*, **35**:2019–2032, 2022.

[UYA20] Sohini Upadhyay, Mikhail Yurochkin, Mayank Agarwal, Yasaman Khazaeni, and

Djallel Bouneffouf. "Graph Convolutional Network Upper Confident Bound." 2020.

[VBJ21] Sattar Vakili, Nacime Bouziani, Sepehr Jalali, Alberto Bernacchia, and Da-shan Shiu. "Optimal order simple regret for Gaussian process bandits." *Advances in Neural Information Processing Systems*, **34**:21202–21215, 2021.

[VCC17] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. "Graph attention networks." *arXiv preprint arXiv:1710.10903*, 2017.

[VKM13] Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. "Finite-time analysis of kernelised contextual bandits." *arXiv preprint arXiv:1309.6869*, 2013.

[VSL17] Sharan Vaswani, Mark Schmidt, and Laks Lakshmanan. "Horde of bandits using gaussian markov random fields." In *Artificial Intelligence and Statistics*, pp. 690–699. PMLR, 2017.

[WA24] Shuang Wu and Arash A Amini. "Graph Neural Thompson Sampling." *arXiv preprint arXiv:2406.10686*, 2024.

[Wei88] David Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules." *Journal of chemical information and computer sciences*, **28**(1):31–36, 1988.

[WKK20] Vikram Waradpande, Daniel Kudenko, and Megha Khosla. "Deep reinforcement learning with graph-based state representations." *arXiv preprint arXiv:2004.13965*, 2020.

[WLW19] Qingyun Wu, Zhige Li, Huazheng Wang, Wei Chen, and Hongning Wang. "Factorization bandits for online influence maximization." In *Proceedings of the 25th*

ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 636–646, 2019.

[WSK23a] Miles Wang-Henderson, Bartu Soyuer, Parnian Kassraie, Andreas Krause, and Ilija Bogunovic. "Graph Neural Bayesian Optimization for Virtual Screening." In *NeurIPS 2023 Workshop on Adaptive Experimental Design and Active Learning in the Real World*, 2023.

[WSK23b] Miles Wang-Henderson, Bartu Soyuer, Parnian Kassraie, Andreas Krause, and Ilija Bogunovic. "Graph Neural Network Powered Bayesian Optimization for Large Molecular Spaces." In *ICML 2023 Workshop on Structured Probabilistic Inference {\&} Generative Modeling*, 2023.

[WWL22] Shuang Wu, Chi-Hua Wang, Yuantong Li, and Guang Cheng. "Residual bootstrap exploration for stochastic linear bandit." In *Uncertainty in Artificial Intelligence*, pp. 2117–2127. PMLR, 2022.

[WYH20] Chi-Hua Wang, Yang Yu, Botao Hao, and Guang Cheng. "Residual bootstrap exploration for bandit algorithms." *arXiv preprint arXiv:2002.08436*, 2020.

[WZ22] Xiyuan Wang and Muhan Zhang. "How powerful are spectral graph neural networks." In *International conference on machine learning*, pp. 23341–23362. PMLR, 2022.

[XWZ20] Pan Xu, Zheng Wen, Handong Zhao, and Quanquan Gu. "Neural contextual bandits with deep representation and shallow exploration." *arXiv preprint arXiv:2012.01780*, 2020.

[YE16] Yael Yankelevsky and Michael Elad. "Dual graph regularized dictionary learning." *IEEE Transactions on Signal and Information Processing over Networks*, **2**(4):611–624, 2016.

[YJK19] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. "Graph transformer networks." *Advances in neural information processing systems*, **32**, 2019.

[YKW20] Tong Yu, Branislav Kveton, Zheng Wen, Ruiyi Zhang, and Ole J Mengshoel. "Graphical models meet bandits: A variational Thompson sampling approach." In *International Conference on Machine Learning*, pp. 10902–10912. PMLR, 2020.

[YTD20] Kaige Yang, Laura Toni, and Xiaowen Dong. "Laplacian-regularized graph bandits: Algorithms and theoretical analysis." In *International Conference on Artificial Intelligence and Statistics*, pp. 3133–3143. PMLR, 2020.

[YW20] Lin Yang and Mengdi Wang. "Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound." In *International Conference on Machine Learning*, pp. 10746–10756. PMLR, 2020.

[YWH23] Xinyang Yi, Shao-Chuan Wang, Ruining He, Hariharan Chandrasekaran, Charles Wu, Lukasz Heldt, Lichan Hong, Minmin Chen, and Ed H Chi. "Online matching: A real-time bandit system for large-scale recommendations." In *Proceedings of the 17th ACM Conference on Recommender Systems*, pp. 403–414, 2023.

[ZJ22] Xingyu Zhou and Bo Ji. "On kernelized multi-armed bandits with constraints." *Advances in neural information processing systems*, **35**:14–26, 2022.

[ZJL23] Tianpeng Zhang, Kasper Johansson, and Na Li. "Multi-armed bandit learning on a graph." In *2023 57th Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6. IEEE, 2023.

[ZLG20] Dongruo Zhou, Lihong Li, and Quanquan Gu. "Neural contextual bandits with ucb-based exploration." In *International Conference on Machine Learning*, pp. 11492–11502. PMLR, 2020.

[ZM19] Tom Zahavy and Shie Mannor. "Deep neural linear bandits: Overcoming catastrophic forgetting through likelihood matching." *arXiv preprint arXiv:1901.08612*, 2019.

[ZZL20] Weitong Zhang, Dongruo Zhou, Lihong Li, and Quanquan Gu. "Neural thompson sampling." *arXiv preprint arXiv:2010.00827*, 2020.