Robust Adaptive Decision Making: Bayesian Optimization and Beyond

Thèse N° 9147

Présentée le 25 janvier 2019

à la Faculté des sciences et techniques de l'ingénieur
Laboratoire de systèmes d'information et d'inférence
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

ILIJA BOGUNOVIC

Acceptée sur proposition du jury

Prof. E. Telatar, président du jury
Prof. V. Cevher, Prof. J. D. Haupt, directeurs de thèse
Prof. S. Jegelka, rapporteuse
Prof. A. Krause, rapporteur
Prof. M. Kapralov, rapporteur

2019

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

To my family

Acknowledgements

This thesis would not have been possible without the help and support of many people. First and foremost, I would like to express my gratitude to my advisor Volkan Cevher whose constant guidance and support have pushed me forward throughout my PhD. I really enjoyed my time in Volkan's lab and I learned so much from him. Thank you Volkan for introducing me to the interesting topics in machine learning and optimization, for your research enthusiasm, for many research discussions and ideas, for your patience, optimism and valuable advice.

I would also like to thank my co-advisor Jarvis Haupt for his kindness, valuable discussions and for hosting me during my short stay at the University of Minnesota.

I was honored to have Stefanie Jegelka, Michael Kapralov, Andreas Krause and Emre Telatar as members of my thesis committee. I am grateful for their time and valuable discussions. Moreover, I am thankful to Stefanie for hosting me at MIT during the fall semester in 2017, and for providing an excellent and welcoming working environment. I sincerely enjoyed and learned a lot from our collaboration. I would also like to thank Volkan for arranging and making this research visit possible. I am also grateful to Andreas Krause for our collaboration, many valuable discussions and for introducing me to the area of machine learning.

I have been fortunate to collaborate with a number of brilliant colleagues at EPFL and LIONS lab. A special thanks go to Jonathan Scarlett for all our amazing collaborations. I truly enjoyed working together and I learned so much from him. His work directly contributed to the content of this thesis, and without him, the work in this form would not have been possible. Thank you Jonathan for your friendship, for the countless inspiring research discussions, for taking the time to review this manuscript, and for your valuable and thorough feedback. I am also grateful to my co-authors Junyao Zhao and Slobodan Mitrovic whose efforts have directly contributed to the content of certain chapters in this thesis. Moreover, I am sincerely thankful to all my lab members for our collaborations and research discussions: Yen-Huan Li, Paul Rolland (who also worked out the French abstract of this thesis), Luca Baldassarre, Anastasios Kyrillidis, Baran Gozcu, Marwa El-Halabi, Alp Yurtsever, Kamal Parameswaran, Chen Liu, Fabian Lattore as well as the other previous and current members of LIONS. I have been fortunate to learn so much from you, and above all, I truly appreciate the time we spent together. I am also grateful to my lab-mate Hsieh Ya-Ping for many research discussions we had, and for being a great and supporting friend. Many thanks go to Gosia Baltaian, who was so helpful on many occasions and who made sure the lab operations were always running smoothly.

Of course, many thanks to all my friends in Lausanne for the great moments I had these couple of years: Milos Vasic, Ana Milicic, Nikola Kalentic, Stanko Novakovic, Rada Palmer, and many

others. Also, big thanks go to all my friends in Serbia for their constant support.

Last but certainly not least, I would like to thank my wife Jelena for all her love, support and encouragement. I am grateful to my parents Slavica and Dusan for all their love and support during all these years. Without them, this journey would not have been possible.

Abstract

The central task in many interactive machine learning systems can be formalized as the sequential optimization of a black-box function. Bayesian optimization (BO) is a powerful model-based framework for *adaptive* experimentation, where the primary goal is the optimization of the black-box function via sequentially chosen decisions. In many real-world tasks, it is essential for the decisions to be *robust* against, e.g., adversarial failures and perturbations, dynamic and time-varying phenomena, a mismatch between simulations and reality, etc. Under such requirements, the standard methods and BO algorithms become inadequate. In this dissertation, we consider four research directions with the goal of enhancing robust and adaptive decision making in BO and associated problems.

First, we study the related problem of level-set estimation (LSE) with Gaussian Processes (GPs). While in BO the goal is to find a maximizer of the unknown function, in LSE one seeks to find all "sufficiently good" solutions. We propose an efficient confidence-bound based algorithm that treats BO and LSE in a unified fashion. It is effective in settings that are non-trivial to incorporate into existing algorithms, including cases with pointwise costs, heteroscedastic noise, and multi-fidelity setting. Our main result is a general regret guarantee that covers these aspects.

Next, we consider GP optimization with robustness requirement: An adversary may perturb the returned design, and so we seek to find a robust maximizer in the case this occurs. This requirement is motivated by, e.g., settings where the functions during optimization and implementation stages are different. We propose a novel robust confidence-bound based algorithm. The rigorous regret guarantees for this algorithm are established and complemented with an algorithm-independent lower bound. We experimentally demonstrate that our robust approach consistently succeeds in finding a robust maximizer while standard BO methods fail.

We then investigate the problem of GP optimization in which the reward function varies with time. The setting is motivated by many practical applications in which the function to be optimized is not static. We model the unknown reward function via a GP whose evolution obeys a simple Markov model. Two confidence-bound based algorithms with the ability to "forget" about old data are proposed. We obtain regret bounds for these algorithms that jointly depend on the time horizon and the rate at which the function varies.

Finally, we consider the maximization of a set function subject to a cardinality constraint k in the case a number of items τ from the returned set may be removed. One notable application is in batch BO where we need to select experiments to run, but some of them can fail. Our focus is on the worst-case adversarial setting, and we consider both *submodular* (i.e., satisfies a natural notion of diminishing returns) and *non-submodular* objectives. We propose robust

algorithms that achieve constant-factor approximation guarantees. In the submodular case, the result on the maximum number of allowed removals is improved to $\tau = o(k)$ in comparison to the previously known $\tau = o(\sqrt{k})$. In the non-submodular case, we obtain new guarantees in the support selection and batch BO tasks. We empirically demonstrate the robust performance of our algorithms in these, as well as, in data summarization and influence maximization tasks.

Key words: Bayesian optimization, Bandit optimization, Gaussian process, Submodularity, Robust optimization, Regret bounds, Level-set estimation, Non-submodular optimization

Résumé

Dans de nombreux systèmes d'apprentissage interactifs, la tâche principale peut être réduite à l'optimisation séquentielle d'une certaine fonction. L'optimisation Bayesienne est un puissant algorithme basé sur un modèle, qui effectue des évaluations de manière adaptive, et dont le but est d'optimiser une fonction quelconque via une séquence de décisions. Dans de nombreuses applications, il est essentiel d'effectuer ces decisions de manière robuste, que ce soit envers des perturbations aléatoires ou non, des phénomènes temporels, ou encore un décalage entre des simulations et la réalité. En tenant compte de ces exigences, les méthodes standard et l'optimisation Bayesienne sont inadéquates. Dans cette thèse, nous considérons quatre directions de recherche dans le but d'améliorer la prise de décision adaptive et robuste dans le cadre de l'optimisation Bayesienne.

Dans un premier temps, nous étudions un problème similaire, qui est l'estimation des surfaces de niveau ("Level-set estimation" ou LSE en anglais) à l'aide de processus Gaussiens. Alors que le but de l'optimisation Bayesienne est de maximiser une fonction inconnue, le LSE a pour but de trouver toutes les solutions "suffisamment bonnes". Nous proposons un algorithme efficace basé sur des intervalles de confiance, et qui traite l'optimisation Bayesienne et LSE de manière unifiée. Cette algorithme est efficace dans des cas qui sont difficiles à traiter par des algorithmes existants, comme par exemple lorsque l'on inclut des coût ponctuels, que le bruit est non-uniforme, ou encore dans le cas de multi-fidélités. Dans notre théorème principal, nous prouvons une garantie théorique du regret prenant en compte ces aspects.

Dans un second temps, nous ajoutons une contrainte de robustesse à l'optimisation avec processus Gaussiens : un adversaire peut perturber chaque mesure, et nous cherchons donc un maximisateur robuste à ce genre de perturbation. Cette contrainte est utile, par exemple, dans les cas où les fonctions utilisées durant l'optimisation et l'implémentation sont différentes. Nous proposons pour cela un nouvel algorithme basé sur des intervalles de confiance robustes. Nous établissons également des garanties théoriques pour le regret, ainsi qu'une borne inférieure universelle, indépendante de l'algorithme. Nous démontrons expérimentalement que notre approche réussit constamment à trouver un maximisateur robuste, alors que l'algorithm standard d'optimisation Bayesienne échoue.

Nous nous intéressons ensuite au problème d'optimisation avec processus Gaussiens dans lequel la fonction à maximiser varie au cours du temps. Il existe en effet de nombreuses applications dans lesquelles l'objectif n'est pas statique. Nous modélisons pour cela la fonction a maximiser par un processus Gaussien dont l'évolution obéit un simple modèle Markovien. Nous proposons deux algorithmes basés sur des intervalles de confiance, avec la capacité d'"oublier"

les données trop vieilles. Nous obtenons des bornes pour le regret, qui dépendent à la fois du nombre d'evaluations, et de la vitesse à laquelle la fonction varie.

Enfin, nous considérons le problème de maximisation d'une fonction d'ensembles sous contrainte de cardinalité k, dans le cas où un nombre d'éléments τ de l'ensemble choisi peuvent être supprimés. Une application notable est l'optimisation Bayesienne groupée, où l'on doit sélectionner un certains nombre d'expériences à effectuer, mais certaines d'entre elles peuvent échouer. Nous nous concentrons sur le problème du pire cas, et considérons à la fois des fonctions sous-modulaires (c'est-à-dire qui satisfont une notion naturelle de rendement décroissant), et non sous-modulaires. Nous proposons des algorithms robustes qui fournissent des solutions avec facteur d'approximation constant. Dans le cas sous-modulaire, le nombre maximal de suppressions autorisées est amélioré à $\tau = o(k)$, en comparaison du résultat $\tau = o(\sqrt{k})$ précédemment connu. Dans le cas non sous-modulaire, nous obtenons de nouvelles garanties pour les tâches de sélection de support et d'optimisation Bayesienne groupée. Nous démontrons empiriquement l'aspect robuste de nos algorithmes dans ces tâches, ainsi que pour la synthèse de données et la maximisation d'influence.

Mots clés : optimisation Bayesienne, optimisation bandit-manchot, processus Gaussiens, sous-modularité, optimisation robuste, borne de regret, estimation des surfaces de niveau, optimisation non sous-modulaire

Contents

List of Figures

List of Tables

List of Algorithms

Bibliographic Note

This dissertation is based on the following publications:

- Ilija Bogunovic, Jonathan Scarlett and Volkan Cevher. "Time-Varying Gaussian Process Bandit Optimization". *International Conference on Artificial Intelligence and Statistics* (AISTATS), 2016 [BSC16].
- Ilija Bogunovic, Jonathan Scarlett, Andreas Krause and Volkan Cevher. "Truncated Variance Reduction: A Unified Approach to Bayesian Optimization and Level-Set Estimation". *Conference on Neural Information Processing Systems* (NIPS), 2016 [BSKC16].
- Ilija Bogunovic, Slobodan Mitrovic, Jonathan Scarlett and Volkan Cevher. "Robust Submodular Maximization: A Non-Uniform Partitioning Approach". *International Conference on Machine Learning (ICML)*, 2017 [BMSC17b].
- Ilija Bogunovic*, Junyao Zhao* and Volkan Cevher. "Robust Maximization of Non-Submodular Objectives". *International Conference on Artificial Intelligence and Statistics* (AISTATS), 2018 [BZC18].
- Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka and Volkan Cevher. "Adversarially Robust Optimization with Gaussian Processes". Accepted to *Conference on Neural Information Processing Systems* (NIPS), 2018 [BSJC18].

Other publications relevant to this disseration are:

- Ilija Bogunovic, Volkan Cevher, Jarvis Haupt and Jonathan Scarlett. "Active Learning of Self-concordant like Multi-index Functions". *International Conference on Acoustics, Speech and Signal Processing* (ICASSP), 2015 [BCHS15].
- Luca Baldassarre, Yen-Huan Li, Jonathan Scarlett, Baran Gözcü, Ilija Bogunovic and Volkan Cevher. "Learning-Based Compressive Subsampling". *IEEE Journal on Selected Topics in Signal Processing*, 2016 [BLS$^+$16].
- Ashkan Norouzi Fard, A. Bazzi, Marwa El Halabi, Ilija Bogunovic, Ya-Ping Hsieh and Volkan Cevher. "An Efficient Streaming Algorithm for the Submodular Cover Problem". *Conference on Neural Information Processing Systems* (NIPS), 2016 [NFBB$^+$16].

- Jonathan Scarlett, Ilija Bogunovic and Volkan Cevher. "Lower Bounds on Regret for Noisy Gaussian Process Bandit Optimization". *Conference on Learning Theory* (COLT), 2017 [SBC17].
- Ilija Bogunovic, Slobodan Mitrovic, Jonathan Scarlett and Volkan Cevher. "A Distributed Algorithm for Partitioned Robust Submodular Maximization". Inter. Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2017 [BMSC17a].
- Slobodan Mitrovic, Ilija Bogunovic, Ashkan Norouzi Fard, Jakub Tarnawski and Volkan Cevher. "Streaming Robust Submodular Maximization: A Partitioned Thresholding Approach". *Conference on Neural Information Processing Systems* (NIPS), 2017 [MBNF$^+$17].
- Paul Rolland, Jonathan Scarlett, Ilija Bogunovic and Volkan Cevher. "High Dimensional Bayesian Optimization via Additive Models with Overlapping Groups". *International Conference on Artificial Intelligence and Statistics* (AISTATS), 2018 [RSBC18].

1 Introduction

The past years have witnessed significant progress in technologies that are based on data-driven systems that interact with the environment, acquire information, reason and make decisions. Recent technological advances include the developments of the first program to defeat a Go world champion [SSS+17], agile robots that can learn complex behavior and operate in non-trivial environments [HSL+17] and systems for adaptive data center management that reduce energy costs [EG16], to name a few. Decision-making for self-driving cars has also generated significant improvement in various operative capabilities following in several prototypes already driving on our roads and streets [SAMR18]. In online advertising and recommendation systems, interactive machine learning systems are used to automatically generate a personalized recommendation to a user from a large pool of possibilities. Based on the users' feedback, these systems can integrate and utilize the newly acquired information and provide better recommendations for future interactions. Similar machine learning systems with human feedback are also used to infer what humans want, e.g., by being informed which of two proposed responses is preferred [CLB+17].

Interactive data-driven systems are also beginning to be introduced into other fields, e.g., medicine, astronomy, computational biology, chemistry, etc. New applications usually appear with more complex requirements, specific types of constraints, as well as ever-growing decision spaces. For instance, in chemical design, we seek to discover molecules that have some desirable properties, and which can be the key to the discovery of a new drug or material. However, the number of molecules with potential medical properties is enormous – estimated to be in between 10^{23} and 10^{60} [GBWD+18]. An additional challenge is the fact that decisions are usually *costly* (e.g., testing a new chemical compound requires running expensive simulations) while the resulting observations can often be *noisy* (e.g., the outcome of running the same experiment might vary). A significant challenge is how to automate the exploration of such and similar spaces and locate promising candidates quickly. Fortunately, the design space in many real-world problems is often *structured*, e.g., it is usually the case that similar molecules will have similar properties and similar movies will be rated similarly by the user. Hence, the key to tackling such problems lies in the ability of the systems to *adaptively* make decisions based on the previous observations and current model, with the goal of reducing the number of expensive interactions.

The greater need for adaptive data-driven systems that can perform in the real world tasks has also introduced important additional requirements. When it comes to a significant number of relevant techonologies, one of the requirements that is in focus is *robustness*. Failures, unpredictable or unstable performance of such systems can often lead to considerable social and economic consequences [Rec18]. As a result, it is important for the decisions they made to be robust in the case of adversarial attacks, dynamic effects and temporal variations, parameter perturbations, a mismatch between simulations and reality, etc. In various practical applications it is beneficial to go beyond a single best decision, but discover multiple sufficiently good backup solutions in the case of some of them result in failure. An essential question in all of these settings is how to make robust decisions while still being able to guarantee strong theoretical performance for our methods. In this dissertation, we study and address this general challenge by considering specific tasks and problem formulations.

The central task in many interactive machine learning systems can often be formalized as the iterative optimization of the unknown *black-box* function that represents some quantity of interest. The only way to learn about the unknown objective is through *bandit* feedback, i.e., point evaluations. For example, in recommender systems, the user's preferences are unknown, and we learn about them by iteratively recommending items and observing the feedback; in automatic chemical design, we regularly select molecules for testing, and we learn about their properties by querying the expert or by running specific simulations. Decision making in these settings corresponds to either adaptive experimentation with the goal of finding the best design, or balancing between *exploration* (i.e., acquiring new information about the unknown quantity) and *exploitation* (i.e., making decisions that are believed to be the best based on the previous interactions) with the goal of cumulative maximization of the importance of choices.

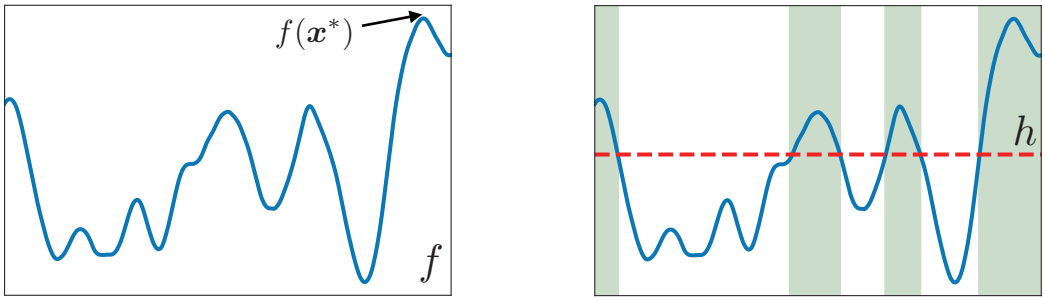
As mentioned before, we can benefit from the fact that the design space is structured in many real-world applications. A way to exploit this is to take the Bayesian perspective and assume a prior model of the unknown function. Typically, a *Gaussian process (GP)* [RW06] is used as a model when the central assumption is that adjacent observations should reveal information about each other [SS98]. In this dissertation, we consider *Bayesian optimization (BO)* – a powerful model-based framework for adaptive experimentation, where the primary goal is the optimization of the unknown function via sequentially chosen decisions and observations. Since its introduction, BO has been applied in different fields, e.g., recommender systems [VNDBK14], robotics [LWBS07], control and reinforcement learning [BSK16], environmental monitoring [SKKS10], preference learning [GDDL], combinatorial optimization [BP18] and many others. Perhaps, the most famous application is automatic hyperparameter tuning in machine learning, where BO is used to automatically select the best model and its associated hyperparameters [SLA12]. A great number of methods for BO (i.e. selection strategies that utilize the model to guide the sequential search) have been developed over time [SSW$^+$16]. One popular algorithm is GP-UCB [SKKS10], which builds confidence bounds around the unknown function and uses the *upper confidence bound* criterion to select its next decision. This idea comes from the seminal work [LR85] on the multi-armed bandit problem, where this strategy is used to balance between exploration and exploitation.

Despite a significant number of methods for Bayesian optimization and experimentation, numerous challenges remain. How can we perform adaptive experimentation in search for a design that not only maximizes the unknown quantity but is also robust against *adversarial perturbations*? How can one perform adaptive decision making to discover all *"sufficiently good"* design choices instead of finding a single best solution alone? In many of the applications, the unknown quantity is not static, but it varies with time. Besides the standard decision making dilemma of exploration vs. exploitation, how can we further balance between *forgetting* vs. *remembering* of the acquired data? Furthermore, how shall we choose which experiments to run (i.e., where to evaluate the unknown function) in the case of *point-wise costs* and *heteroscedastic noise* (i.e., when querying the unknown function at different points in the decision space leads to different costs and amount of noise in observations)? How can we automatically choose experiments in a *robust* way in the case some of them fail? These are some of the research questions that we consider in this dissertation with the goal of enhancing both robust and adaptive decision making in BO and related methods. Under most of these challenges, the standard BO methods become inadequate. To tackle these questions, we propose various *confdience-bound* based algorithms that, despite the lack of knowledge of the actual underlying function, are able to make decisions based on the previous interactions and their confidence in the model.

Finally, in many real-world tasks, it is of interest to choose a set of decisions in every interaction. However, what if some of the selected decisions/experiments fail? In many situations, we do not have a reasonable prior distribution on how failures happen, or we require robustness guarantees with a high level of certainty. In such case, protecting against worst-case failures is essential. In this dissertation, we address this question by formalizing it as the combinatorial optimization problem in which the goal is to choose a set of decisions (from a potentially large pool) that maximize some objective value so that if some decisions result in failure, the value degrades as little as possible. Finding a robust set of choices is not only of interest in adaptive experimentation and the case when experiments can fail, but it can be essential in many important machine learning applications. Some of the relevant problems where it is useful to select a robust set of decisions include feature selection [KED+17], influence maximization [KKT03], sensor placement [KSG08], data summarization [Mir17], when interpreting machine learning models [RSG16], just to name a few. In many of these, the objective set function satisfies the natural notion of diminishing returns – *submodularity*. In such cases and when decisions can fail, the standard greedy algorithm [NW78] that is near-optimal in the non-robust setting can perform arbitrarily badly. In this dissertation, we propose efficient algorithms for robust decision selection that address this issue by exploiting a very simple idea: The key is to select decisions that will admit a large objective value, but also a right number of redundant decisions that can *"cover for"* the important ones in the case they fail.

1.1 Contributions

In this dissertation, we explore four research challenges related to robust and adaptive decision making, with a particular focus on algorithms that provide strong theoretical guarantees:



(a) Unknown function f and its maximizer $\boldsymbol{x}^*$ (b) Level sets

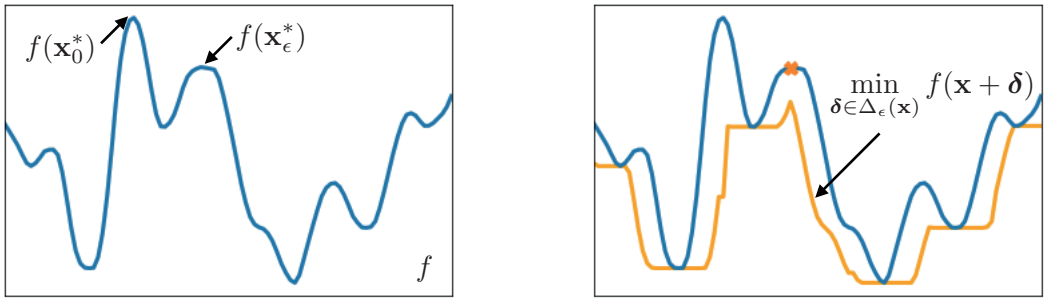
Figure 1.1: In BO the goal is to find $\boldsymbol{x}^*$ alone (Figure 1.1a), while in LSE (Figure 1.1b) one seeks to find all "sufficiently good" points, i.e., points for which $f(\boldsymbol{x})$ is above the given threshold h.

1. Designing a versatile and cost-effective method for Bayesian optimization and level-set estimation.

Bayesian optimization (BO) and level-set estimation (LSE) are related problems that are typically studied in isolation. Bayesian optimization [SSW+16] provides a powerful framework for automating experimentation and finds applications in robotics, environmental monitoring, and automated machine learning, etc. One seeks to find the maximum of an *unknown* reward function that is expensive to evaluate, based on a sequence of suitably-chosen decisions and their noisy observations. Level-set estimation (LSE) [GCHK13] is closely related to BO, but instead of seeking a maximizer, one seeks to classify the domain into points that lie above or below a certain threshold. Finding the best solution alone, as is the case in BO, might not be sufficient in applications where *robustness* or *diversity* of solutions is required. In such cases, LSE can be used to find all "sufficiently good" solutions (see Figure 1.1 for an illustration).

Popular methods for BO include expected improvement (EI), probability of improvement (PI), and Gaussian process upper confidence bound (GP-UCB) [SSW+16, SKKS10]. An algorithm for level-set estimation with GPs is given in [GCHK13], which keeps track of a set of unclassified points. These algorithms have theoretical guarantees and good computational complexity, but due to their *myopic* nature in considering no more than a single step into the future, it is unclear how to incorporate *pointwise costs* (i.e., a scenario in which different sampling locations are associated with different costs) and several other settings. In contrast, so-called *one-step lookahead* BO methods, entropy search (ES) [HS12], its predictive version (PES) [HLHG14] and minimum regret search (MRS) [Met16] permit versatility with respect to costs [SSA13], heteroscedastic noise [GWB97], and multi-task scenarios [SSA13]. Unlike the myopic algorithms, these are computationally expensive and, to our knowledge, no theoretical guarantees have been provided for these one-step lookahead algorithms. Hence, we seek to design a versatile, computationally efficient and cost-effective algorithm with rigorous theoretical guarantees.

In Chapter 3, we present an algorithm, truncated variance reduction (TruVaR), that addresses Bayesian optimization and level-set estimation with Gaussian processes in a *unified* fashion.



(a) Unknown function f (b) Robust objective

Figure 1.2: (a) A function f and its maximizer $\boldsymbol{x}_0^*$; (b) for the adversarial budget $\epsilon = 0.06$ and distance function $d(\boldsymbol{x}, \boldsymbol{x}') = |\boldsymbol{x} - \boldsymbol{x}'|$, the decision $\boldsymbol{x}_\epsilon^*$ that corresponds to the local "wider" maximum of f is the *optimal* ϵ*-stable* decision.

The algorithm greedily shrinks the total variance, up to a truncation threshold, within a set of potential maximizers (BO) or unclassified points (LSE), which is updated based on confidence bounds. TRUVAR is effective in important settings that are typically non-trivial to incorporate into the standard myopic algorithms, such as pointwise costs and heteroscedastic noise. On the other hand, compared to the one-step lookahead algorithms for BO, TRUVAR avoids the computationally expensive task of averaging over the posterior and/or measurements, and comes with rigorous theoretical guarantees for the two settings. Moreover, we present a new result in the *multi-fidelity* setting where, while sampling, one can select from a number of noise levels having associated costs. The cost-effectiveness of TRUVAR is demonstrated on both synthetic and real-world data sets, where it is able to outperform the competitor methods while incurring a significantly smaller sampling cost.

2. Designing a method for robust Bayesian optimization.

In many applications in which the goal is to optimize a black-box function, one is faced with forms of uncertainty that are not accounted for by standard algorithms. In robotics, the optimization is often performed via simulations, creating a mismatch between the model and the true function; in parameter tuning, the function is similarly mismatched due to limited training data; in recommendation systems, the underlying function is inherently time-varying, so the returned solution may become increasingly stale over time; the list goes on. We address these considerations by studying GP optimization with an additional requirement of *robustness*: The returned point may be perturbed by an adversary, and we require the function value to remain as high as possible even after this perturbation. This problem is of interest not only for attaining improved robustness to uncertainty but also for other related max-min optimization settings.

In general, for a given fixed perturbation budget, the function value of the global maximum after being adversarially perturbed might degrade significantly. We refer to the point that has the highest function value after being adversarially perturbed (in the worst-case sense) as *stable* optimal (see Figure 1.2). In such setting, all the BO strategies (e.g., [HS12, WJ17, SJ17a, RMGO18]) whose goal is to identify the global non-robust maximum can become inherently suboptimal.

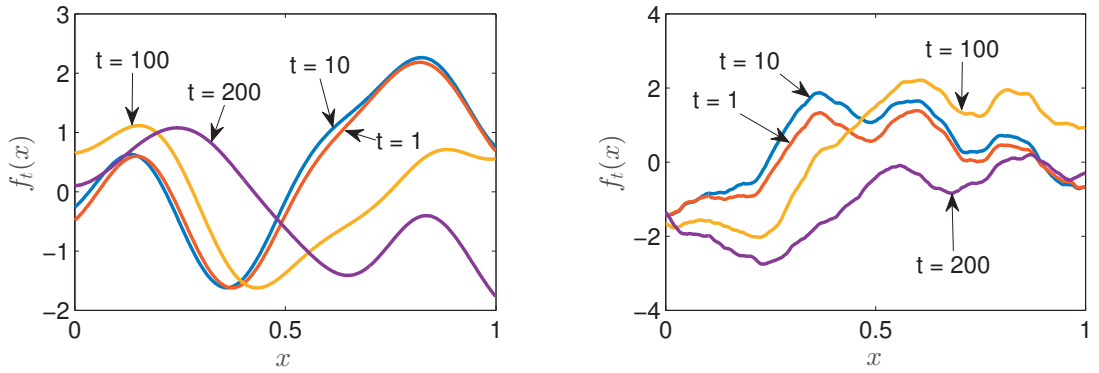


Figure 1.3: Two examples of time-varying reward functions. The location of the global maximum changes significantly at distant times.

In Chapter 4, we introduce a variant of GP optimization in which the returned solution is required to exhibit stability/robustness to an adversarial perturbation. We demonstrate the failures of standard BO algorithms, and we introduce a new algorithm StableOpt that overcomes these limitations. The algorithm is based on two distinct principles: *optimism in the face of uncertainty* when it comes to selecting where to sample next, and *pessimism in the face of uncertainty* when it comes to anticipating the perturbation of the selected point. We provide a novel theoretical analysis characterizing the number of samples required for StableOpt to attain a *near-optimal robust* solution, and we complement this with an algorithm-independent lower bound. We provide several important variations of our max-min optimization framework and theory, including the classical robust formulation, robust estimation, and group identification problems. We experimentally demonstrate a variety of potential applications of interest on real-world data sets, and we show that StableOpt consistently succeeds in finding a *stable maximizer* where several BO and baseline methods fail.

3. Can we design "no-regret" methods in the case of time-varying rewards?

In the previous two research challenges, the unknown objective function was assumed to be time-invariant, i.e., it was *static* and did not vary with time. However, in many practical applications, the function to be optimized varies with time: In sensor networks, measured quantities such as temperature undergo fluctuations; in recommender systems, the users' preferences may change according to external factors; similarly, financial markets are highly dynamic. Studying time-varying effects is of great importance in such applications.

In the time-invariant setting (i.e., classical setting), several different algorithms are known to be *"no-regret"* (e.g., [SKKS10]). Simply put, for a sufficiently large time horizon, these algorithms are guaranteed to find the global maximum. However, in the case of time-varying reward functions, the maximum function value and its location could change drastically throughout the time horizon. In such cases, the performance of these standard Bayesian optimization algorithms may deteriorate, since these continue to treat stale data as being equally important as fresh data.

Balancing between exploration and exploitation is not sufficient, and leads to the non-vanishing regret performance of the standard algorithms. This initiates the study of the additional *forgetting-remembering* dilemma and the development of algorithms that can exploit both spatial and temporal correlations present in the reward function.

In Chapter 5, we take a novel approach to handling time variations, modeling the reward function as a Gaussian process (GP) that varies according to a simple Markov model (as shown in Figure 1.3). We prove that no algorithm can attain vanishing regret for any fixed function variation rate. This motivates the study on how the regret *jointly* depends on the time horizon and the rate at which the reward function varies. We develop two algorithms based on the upper confidence bound strategy. The first, R-GP-UCB, completely forgets about the past data at regular intervals. The second, TV-GP-UCB, instead forgets about old data in a smooth fashion. Our main contribution comprises of novel regret bounds for these algorithms, providing an explicit characterization of the trade-off between the time horizon and the rate at which the function varies. We test the performance of the algorithms on both synthetic and real data and show that R-GP-UCB is computationally more efficient while the gradual forgetting of TV-GP-UCB performs favorably compared to the sharp resetting of R-GP-UCB. Moreover, both algorithms significantly outperform the standard BO algorithms, since they treat stale and fresh data equally.

4. How can we efficiently select a robust set of experiments/decisions that maximize our objective of interest, such that it degrades as little as possible if some of them fail?

In all the previous challenges, we considered the task of sequentially optimizing an unknown function from point evaluations that require performing costly experiments. When a batch of points is to be selected, this becomes the problem of subset selection, which we need to solve in every round. In case that experiments can fail, protecting against worst-case failures of expensive trials is of great importance. This challenge is not only associated with the robust experimentation, but it also arises in other important machine learning applications where the goal is to select a robust set of items that maximize some objective of interest. For instance, (i) in influence maximization problems, a subset of the chosen users may decide not to spread the word about a product; (ii) in summarization problems, a user may decide to remove some items from the summary due to their personal preferences; (iii) in the problem of sensor placement for outbreak detection, some of the sensors might fail; (iv) in the problem of feature selection, some selected features might be missing at the test time. In situations where one does not have a reasonable prior distribution on the elements removed, or where one requires robustness guarantees with a high level of certainty, protection against worst-case removals becomes essential.

The problem can be formalized as the robust set selection problem in which we want to select a set of items of size k to maximize some objective function such that the utility of the selected set degrades as little as possible after the worst-case adversarial failure (i.e., removal) of τ of them. We consider both the case when the objective function satisfies the natural notion of diminishing returns (i.e., adding an item to a set is "more beneficial" if it is done earlier than later in the selection process) – *submodularity* [KG12], and the case of *non-submodular* objectives [DK08].

We address the former case in Chapter 6. A constant-factor approximation guarantee was given in [OSU16] when the allowed number of removals is $\tau = o(\sqrt{k})$. We solve a key open problem raised therein, presenting a new and efficient Partitioned Robust (PRO) submodular maximization algorithm that achieves the same constant factor guarantee for more general $\tau = o(k)$. Our algorithm constructs partitions consisting of buckets with exponentially increasing sizes, and applies standard submodular optimization subroutines on the buckets to construct the robust solution. We numerically demonstrate the performance of PRO in data summarization [Mir17] and influence maximization [KKT03] tasks, demonstrating gains over both the non-robust greedy algorithm [NW78] and the algorithm of [OSU16].

We consider the case of non-submodular objectives in Chapter 7. We propose a simple and practical algorithm OBLIVIOUS-GREEDY, and prove the first constant-factor approximation guarantees for a broader class of non-submodular objectives. The obtained theoretical bounds also hold in the linear regime, i.e., when the number of allowed removals τ is linear in k. Our bounds depend on a few parameters including *submodularity ratio* and *inverse curvature*, that essentially, characterize how close the objective is to being submodular and approximate diminishing returns property of the objective, respectively. We provide a summary of these and other useful parameters that can be used to characterize any monotone set function, and prove some interesting relations between them.

The obtained relations allow us to provide bounds for these parameters for two important non-submodular objectives including the variance reduction objective and the one used in sparse support/feature selection problems. The variance reduction objective is the one used in the TRUVAR rule (introduced in Challenge 1). While this objective is not submodular in general, we show that constant factor guarantees can still be obtained and OBLIVIOUS-GREEDY can be used at every round in Bayesian optimization to select a robust set of experiments efficiently. By considering the sparse feature selection problem, we provide a novel connection between *strong convexity* and *weak supermodularity* (i.e., the property that tells how close the objective is to being supermodular). This result complements the one from [EKDN16] where the same connection is shown between strong convexity and *weak submodularity*. This result can potentially enlarge the number of applications where supermodular optimization and algorithms can be used. Finally, we empirically demonstrate the robust performance of our algorithm by considering these objectives. On various datasets, OBLIVIOUS-GREEDY consistently outperforms both robust and non-robust algorithms in terms of the robust objective value, but also when it comes to the generalization performance (in the feature selection problem) in the case of unseen data.

1.2 Organization of the Thesis

The structure of this dissertation is as follows:

- In Chapter 2, we provide a brief introduction to Bayesian optimization. We review the GP-UCB algorithm [SKKS10] and relevant theoretical results in BO. We proceed with a short review on submodular functions and optimization. Finally, we focus on a specific problem in robust submodular optimization and consider the previous work and algorithms.

- In Chapter 3, we study the unified and cost-effective approach to BO and LSE. We propose a new algorithm TRUVAR and investigate its theoretical guarantees both in BO and LSE. Other settings and their corresponding theoretical guarantees are also considered, including pointwise costs, heteroscedastic noise, multi-fidelity, and the non-Bayesian setting.

- In Chapter 4, we introduce the problem of adversarially robust Gaussian process optimization and propose a novel robust algorithm STABLEOPT. We study both theoretical and empirical performance of our algorithm in this and other robust settings including robust Bayesian optimization, robustness to unknown parameters, and group identification.

- In Chapter 5, we consider the problem of Gaussian process optimization when the unknown function varies with time. We propose new algorithms TV-GP-UCB and R-GP-UCB that can penalize the old data and study the setting-specific lower and upper regret bounds that jointly depend on the time horizon and rate of variation.

- In Chapter 6, we investigate the problem of robust submodular maximization in the presence of adversarial removals. We propose a novel robust algorithm PRO-GREEDY and address a key question from [OSU16], that is, whether the constant factor approximation guarantee therein can hold in the case of a greater number of allowed removals.

- In Chapter 7, we study the same problem as in Chapter 6 in the case when the objective is non-submodular. We present OBLIVIOUS-GREEDY and analyze its approximation guarantees in terms of the parameters that characterize set functions (e.g., submodularity ratio). We investigate the bounds for these parameters in different important applications.

- In Chapter 8, we review the main contributions of each chapter in this disseration, and provide various directions for future research.

We begin Chapters 3-7 by formally stating the considered problem followed by the main algorithm, theoretical results and experimental evaluations. Chapter-specific related work and main contributions are listed at the beginning of every chapter. In Chapters 3-7, we provide most of the proofs in the subappendices at the end of each chapter.

1.3 Notation

In this section, we outline definitions of the most commonly-used mathematical symbols in this disseration.

We use $\mathbb{R}$ to denote the set of real numbers, and $\mathbb{R}_+$ to denote the set of positive real numbers. The set of natural numbers is denoted by $\mathbb{N}$, the set of positive natural numbers by $\mathbb{N}_+$, and the set of integers by $\mathbb{Z}$.

We use bold symbols for vectors and matrices, with the latter being capitalized, e.g., $\boldsymbol{a}$ and $\boldsymbol{A}$. To denote i-th entry of a vector we use subscript a_i. We denote the transposes of $\boldsymbol{a}$ and $\boldsymbol{A}$ by $\boldsymbol{a}^T$ and $\boldsymbol{A}^T$, and if $\boldsymbol{A}$ is invertible, its inverse is denoted by $\boldsymbol{A}^{-1}$. The Euclidean norm of a vector $\boldsymbol{a}$ is denoted by $\|\boldsymbol{a}\|_2$, the sum of the absolute values of its entries by $\|\boldsymbol{a}\|_1$, and the maximum absolute value of its entries by $\|\boldsymbol{a}\|_\infty$. For a square $n \times n$ matrix $\boldsymbol{A}$, we use $\det(\boldsymbol{A})$ to denote its determinant and $\text{tr}(\boldsymbol{A})$ to denote its trace. We use $\boldsymbol{I}_n$ to denote identity matrix of size $n \times n$. For a vector $\boldsymbol{a}$, we write $\text{diag}(\boldsymbol{a})$ for a diagonal matrix containing the elements of vector $\boldsymbol{a}$ on the main diagonal. The inner product of two vectors is denoted with $\langle \cdot, \cdot \rangle$. For two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of size $n \times n$, the Hadamard product $\boldsymbol{A} \circ \boldsymbol{B}$ results in a matrix $\boldsymbol{C}$ of size $n \times n$, with elements given by $\boldsymbol{C}_{i,j} = (\boldsymbol{A})_{i,j}(\boldsymbol{B})_{i,j}$. For a matrix $\boldsymbol{A}$, we use $\|\boldsymbol{A}\|_F$ to denote its Frobenius norm.

Let $f(\cdot)$ and $g(\cdot)$ be two functions defined on some unbounded subset of the real numbers. We write $f(x) = O(g(x))$ if $|f(x)| \leq C|g(x)|$ for some C and sufficiently large x, and $f(x) = o(g(x))$ if $g(x) \neq 0$ and $\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$. Similarly, we write $f(x) = \Omega(g(x))$ if $|f(x)| \geq C'|g(x)|$ for some C' and sufficiently large x. We similarly use the notation $O^*(\cdot), \Omega^*(\cdot)$ to denote asymptotics up to logarithmic factors.

The symbol $\sim$ means "distributed according to". We use $\mathbb{P}[\cdot]$ to denote the probability of an event. The indicator function of an event E is denoted by $\mathbb{1}_E$. $\mathbb{E}[\cdot]$ and $\text{Var}[\cdot]$ are used for the expectation and variance of a random variable when the probability distribution is obvious from the context, and $\mathbb{E}_{q(x)}[f(x)]$, $\text{Var}_{q(x)}[f(x)]$ when $x \sim q(x)$. $\mathbb{E}_t[\cdot]$ is used to denote the expectation of a random variable with respect to its posterior distribution at time t. We use $H(X)$ to denote entropy, $H(X|Y)$ for conditional entropy, $I(X;Y)$ for mutual information and $D(P||Q)$ for KL divergence. We use Φ to denote standard normal CDF. We write GP to denote a Gaussian process distribution and $f \sim \text{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$ to denote that function f is distributed according to a Gaussian process with mean $\mu(\cdot)$ and kernel functions $k(\cdot, \cdot)$. After observing t data points, we use $\mu_t(\cdot)$ and $\sigma_t^2(\cdot)$ to denote the posterior mean and variance of a GP, and we use $\boldsymbol{K}_t$ to denote the kernel matrix for t noiseless observations.

For a set A, we write $|A|$ to denote its cardinality, and 2^A to denote its power set. For a function f defined on sets, $f(A|B)$ denotes the marginal gain of adding set A to set B, i.e. $f(A \cup B) - f(B)$. We use $A \setminus B$ to denote set difference. For an integer $k > 0$, we write $[k]$ for the set $\{1, \cdots, k\}$. The floor and ceiling functions are denoted by $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$, respectively. For a vector $\boldsymbol{a}$, we write $\text{supp}(\boldsymbol{a})$ for the set $\{i : x_i \neq 0\}$, and $\|\boldsymbol{a}\|_0$ for $|\text{supp}(\boldsymbol{a})|$.

2 Background Material

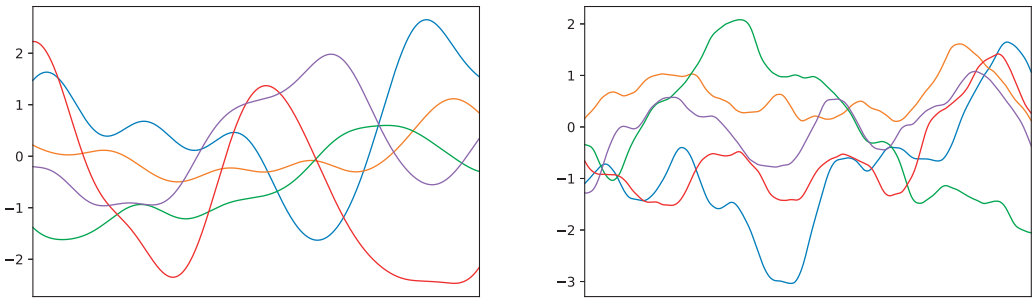
In this chapter, we present an overview of Gaussian process (GP) models that are frequently used in Bayesian optimization (BO). Various methods for adaptive experimentation are discussed together with the most common performance metrics. We provide a summary of the existing theoretical results in BO that are most relevant to the work presented in the subsequent chapters of this dissertation. In the second part of this chapter, we provide a short review of submodular functions and the relevant results in submodular optimization. Finally, we discuss the robust submodular formulation in which we need to choose a set of decisions under uncertainty that some of them might result in failure.

2.1 Gaussian Processes (GPs)

In this section, we provide a brief overview of Gaussian process models. A more comprehensive introduction can be found in [RW06, Section 2].

A Gaussian process over some input space D is a collection of dependent random variables $\{f(\boldsymbol{x})\}_{\boldsymbol{x}\in D}$ such that every finite subset of these random variables $\{f(\boldsymbol{x}_i)\}_{i=1}^{n}$, $n \in \mathbb{N}$ is jointly Gaussian. Any GP is fully specified by its mean $\mu : D \to \mathbb{R}$ and kernel (covariance) function $k : D \times D \to \mathbb{R}$, and hence it is denoted by $\mathrm{GP}(\mu(\cdot), k(\cdot,\cdot))$. Because GPs define a distribution over functions, they are frequently used in nonparmetric and nonlinear regression to model an unknown function. For $f \sim \mathrm{GP}(\mu(\cdot), k(\cdot,\cdot))$, we have $f : D \to \mathbb{R}$ and $f(\boldsymbol{x})$ is Gaussian for every $\boldsymbol{x} \in D$, with mean $\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$ and variance $k(\boldsymbol{x},\boldsymbol{x}) = \mathbb{E}[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))^2]$.

Typically, we use GPs as a prior model when the main assumption is that the function values of nearby inputs should reveal information about each other [SS98]. We can model versatile classes of functions by using different kernel functions (see Figure 2.1). An important class of kernels is the class of *stationary kernels*. These kernels are translation invariant, meaning that the dependence on $\boldsymbol{x}, \boldsymbol{x}'$ is only through their difference, i.e., $k(\boldsymbol{x},\boldsymbol{x}') = k(\boldsymbol{\tau})$ for $\boldsymbol{\tau} = \boldsymbol{x} - \boldsymbol{x}'$.



(a) Samples from $\mathrm{GP}(0, k_{\mathrm{SE}})$ (b) Samples from $\mathrm{GP}(0, k_{\mathrm{Mat}})$

Figure 2.1: Example of functions sampled from zero mean GP with SE and Matérn kernel. Different kernel functions can be used to model versatile classes of functions.

Two of the most commonly used stationary kernels in practice are squared exponential (SE)[1] and Matérn kernels:

$$k_{\mathrm{SE}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2l^2}\right), \tag{2.1}$$

$$k_{\mathrm{Mat}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|}{l}\right) J_\nu\left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|}{l}\right), \tag{2.2}$$

where l denotes the length-scale, $\nu > 0$ is an additional parameter that dictates the smoothness (the smaller this parameter is, the rougher the sampled functions are) and $J(\nu)$ and $\Gamma(\nu)$ denote the modified Bessel and Gamma functions, respectively ([RW06, Section 4.2.1]). Functions sampled from SE kernel are infinitely differentiable almost surely. In the case of the Matérn kernel, as its parameter $\nu \to \infty$ we recover the SE kernel. An overview of other kernel functions that can be interesting in practical applications can be found in [Duv14, Chapter 2]. When it comes to the mean function a usual assumption is that it is zero everywhere [RW06, Section 2.7]. We use $\mathrm{GP}(0, k)$ to denote a zero mean GP with kernel k.

Suppose $f \sim \mathrm{GP}(0, k)$, and suppose we sample f at $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t\}$ and observe $\{y_1, \ldots, y_t\}$ where for every $i \in [t]$:

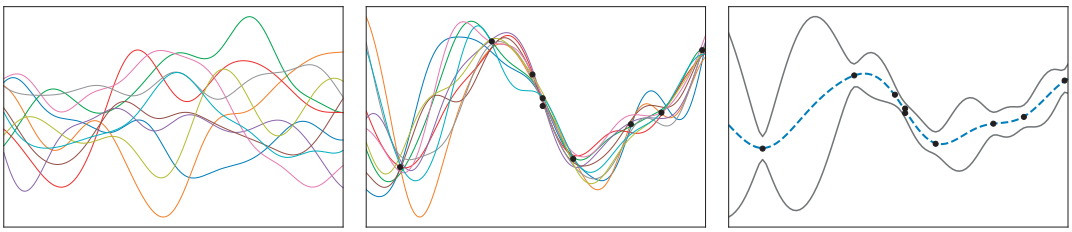
$$y_i = f(\boldsymbol{x}_i) + z_i \quad \text{and} \quad z_i \sim \mathcal{N}(0, \sigma^2).$$

By the GP property, for any $\boldsymbol{x} \in D$ with the corresponding value $f(\boldsymbol{x})$, it holds $\boldsymbol{y}_t = [y_1, \ldots, y_t]^T$ and $f(\boldsymbol{x})$ are jointly Gaussian given $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t\}$:

$$\begin{bmatrix} f(\boldsymbol{x}) \\ \boldsymbol{y}_t \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{k}_t(\boldsymbol{x})^T \\ \boldsymbol{k}_t(\boldsymbol{x}) & \boldsymbol{K}_t + \sigma^2 \boldsymbol{I}_t \end{bmatrix}\right), \tag{2.3}$$

Here, $\boldsymbol{k}_t(\boldsymbol{x})^T = \big[k(\boldsymbol{x}_1, \boldsymbol{x}), \ldots, k(\boldsymbol{x}_t, \boldsymbol{x})\big]$ and $\boldsymbol{K}_t = \big[k(\boldsymbol{x}_t, \boldsymbol{x}_{t'})\big]_{t,t'} \in \mathbb{R}^{t \times t}$ is the kernel matrix.

[1] This kernel is also known as Gaussian or Radial basis function (RBF) kernel.



(a) Samples from prior (b) Samples from posterior (c) Posterior uncertainty

Figure 2.2: An illustration of Bayesian posterior updates in GPs. In (a), we show samples from the GP prior. After receiving some (noisy) observations (black circles), posterior samples are illustrated in (b). In (c), we show the posterior mean prediction (dashed curve) plus and minus its 3 standard deviations (both obtained via (2.4)). We observe that the uncertainty shrinks around the observed points and is larger further away from observations.

By using the formula for the conditional distribution associated with a jointly Gaussian random vector [RW06, Appendix A], one can show that conditioned on the observed data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^t$, the posterior process is again GP with the posterior mean and variance:

$$\begin{aligned} \mu_t(\boldsymbol{x}) &= \boldsymbol{k}_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}_t, \\ \sigma_t^2(\boldsymbol{x}) &= k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}_t\right)^{-1} \boldsymbol{k}_t(\boldsymbol{x}) \end{aligned} \tag{2.4}$$

Therefore, for every $\boldsymbol{x} \in D$, conditioned on $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^t$, the posterior distribution of $f(\boldsymbol{x})$ is Gaussian, i.e., $\mathcal{N}(\mu_t(\boldsymbol{x}), \sigma_t^2(\boldsymbol{x}))$. The Bayesian posterior mechanics are illustrated in Figure 2.2. As can be observed in (2.4), the posterior mean and variance at some input can be computed analytically in closed form. This property, and together with the ability to describe uncertainty in the predictions, are the main reasons why GPs are frequently used in many machine learning applications. The algorithms and theory developed in later sections will heavily rely on these elegant features of GPs. In practice, Cholesky decomposition is used instead of directly inverting the kernel matrix in (2.4) (see Algorithm 2.1 in [RW06, Section 2.2] for more details).

2.2 Bayesian Optimization

Bayesian optimization (BO) refers to a sequential model-based approach for optimizing an *unknown* and usually *multimodal* reward function $f : D \to \mathbb{R}$ over some input space D, where a prior belief of the possible reward functions is prescribed. The unknown function is a *black-box*, meaning that it can be only observed through so called *bandit feedback*, i.e., point evaluations (no gradient information is available). We also assume that every function evaluation is *costly*. Since its introduction, BO has successfully been applied to numerous applications, including robotics [LWBS07], algorithm parameter tuning [SLA12], recommender systems [VNDBK14], environmental monitoring [SKKS10], and many more.

Algorithm 1 Bayesian Optimization Pseudocode

1: **for** $t = 1, 2, \ldots$ **do**
2: choose $\boldsymbol{x}_t$ by optimizing some acquisition (i.e., auxiliary) function $\vartheta(\cdot)$:

$$\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x}} \ \vartheta\left(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^{t-1}\right)$$

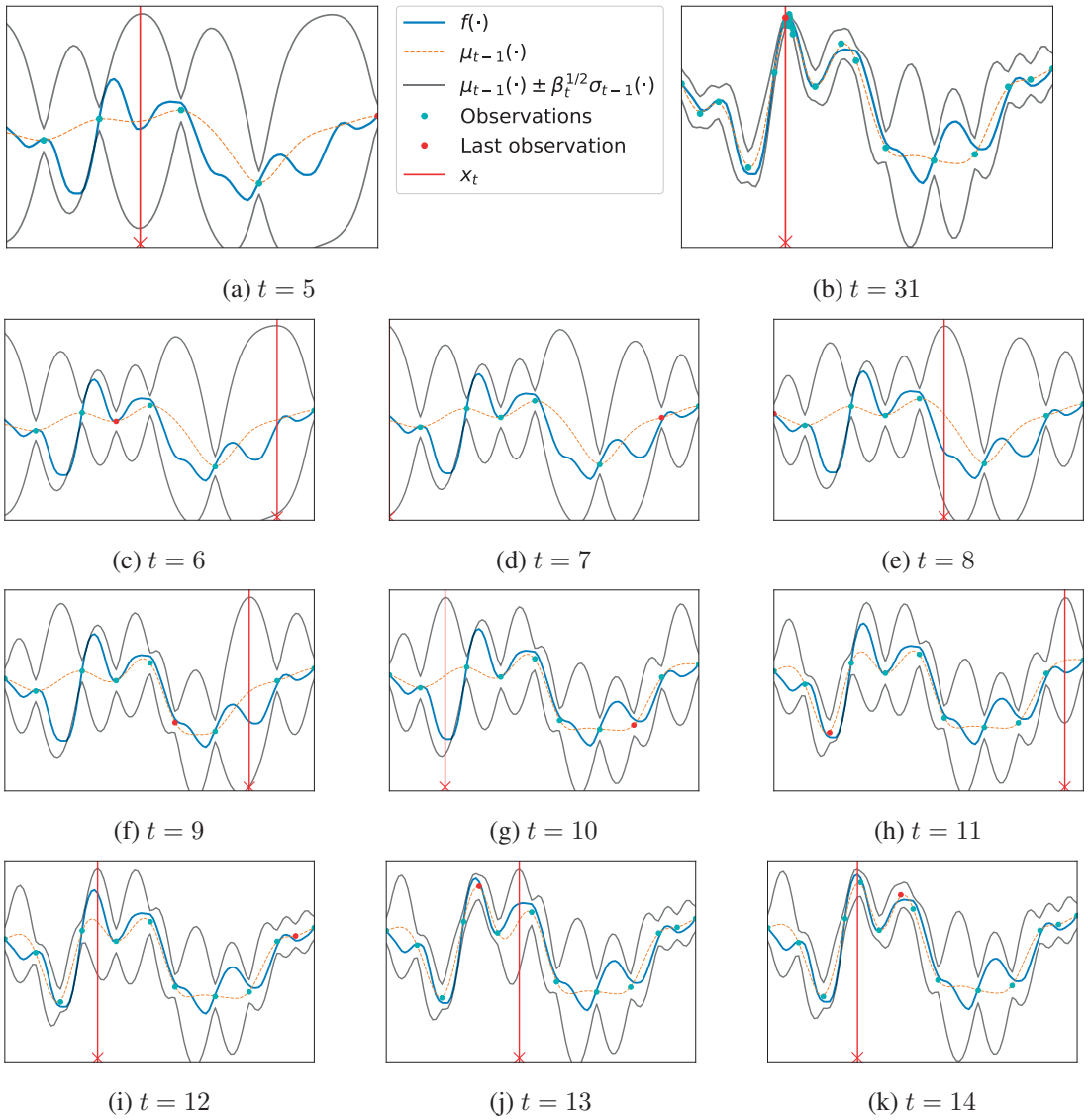
3: observe the objective function sample y_t at $\boldsymbol{x}_t$
4: augment data and update probabilistic model

The main idea in BO is to sequentially acquire data and refine the prescribed reward function model via Bayesian posterior updates. The overall procedure is given in Algorithm 1. We proceed by explaining the three main components: a *probabilistic model*, an *acquisition function* (i.e., auxiliary function) and a *performance metric*.

When it comes to the model, the most common option is to use a Gaussian process prior to model the unknown reward function, that is to assume $f \sim \mathrm{GP}(0, k)$. In practice, the parameters of the GP kernel function can be learned from the observed data via the marginal likelihood approach or sampling procedures (see [RW06, Section 5]). Given such model, the procedure develops in time steps (see Algorithm 1), where at every time step t, the unknown function is sampled at a point $\boldsymbol{x}_t$, and the corresponding observation y_t is received. By aggregating a new data pair $(\boldsymbol{x}_t, y_t)$, the model is updated via Bayesian posterior updates (outlined in (2.4)) by using all the previously collected data $\{\boldsymbol{x}_i, y_i\}_{i=1}^{t}$.

An acquisition function $\vartheta : D \to \mathbb{R}$, is a surrogate function that makes use of the posterior model to guide the sampling procedure. It is usually designed to balance between *exploration* and *exploitation*, i.e., it has high values where the uncertainty of the model is large (corresponding to exploration), and/or where the prediction of the model is high (exploitation). At every time step, the acquisition function is optimized and the resulting point $\boldsymbol{x}_t$ is the one where f is sampled at.

Acquisition functions are usually multimodal and potentially non-trivial to optimize. The central assumption in BO is that f is expensive and/or time-consuming to evaluate, so that the overhead that comes with the optimization of the acquisition function is significantly cheaper. For example, when tuning complex machine learning systems, one evaluation of f might correspond to training the system for a single hyperparameter configuration on a huge dataset, which might require days to finish. On the other hand, acquisition functions are defined on the posterior model, and hence to optimize them we do not require additional evaluations of f. An important practical aspect is the global optimization of the acquisition function when the domain is compact $D \subset \mathbb{R}^d$ (in contrast, when D is finite this task is easier). Various global optimization solvers are used for this problem in practice (see [SSW$^+$16, V.B] for more details). We note that in most of the theoretical results in BO, it is assumed that the kernel function is perfectly known and that the auxiliary optimizer finds the global maximum of the acquisition function (some of these assumptions are relaxed in, e.g., [WdF14],[WSJF14] and [SJ17a]). For more details on the practical aspects of BO, we refer the reader to the survey [SSW$^+$16].



(a) $t = 5$ (b) $t = 31$

(c) $t = 6$ (d) $t = 7$ (e) $t = 8$

(f) $t = 9$ (g) $t = 10$ (h) $t = 11$

(i) $t = 12$ (j) $t = 13$ (k) $t = 14$

Figure 2.3: Demo run of GP-UCB: We start GP-UCB after 5 samples are collected (see 2.3a). We observe in 2.3b that after some number of rounds the sampling is focused around the maximum. At every time step, GP-UCB selects a point with the highest upper confidence bound. We show some intermediate steps in 2.3c– 2.3k.

Methods for BO

One of the most popular acquisition functions is based on the so-called *principle of optimism in the face of uncertainty*. Simply put, despite the lack of knowledge in what $\boldsymbol{x}$ is the best, the idea is to construct an optimistic guess as to how good $f(\boldsymbol{x})$ for every $\boldsymbol{x} \in D$ is and pick $\boldsymbol{x}$ with the highest guess. This idea dates back to the seminal work [LR85] on the multi-armed bandit problem where the *upper confidence bound criterion* has been used to balance between exploration and exploitation. Following the same approach, the Gaussian Process Upper Confidence Bound

Algorithm 2 GP-UCB [SKKS10]

Input: Domain D, GP prior (μ_0, σ_0, k)

1: **for** $t = 1, 2, \ldots$ **do**

2: Choose

$$\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x} \in D} \ \mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2} \sigma_{t-1}(\boldsymbol{x})$$

3: Observe $y_t = f(\boldsymbol{x}_t) + z_t$

4: Update posterior according to (2.4) to obtain $\mu_t(\cdot)$ and $\sigma_t(\cdot)$

(GP-UCB) algorithm [SKKS10] chooses a point with the highest upper confidence bound,

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \ \underbrace{\mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2} \sigma_{t-1}(\boldsymbol{x})}_{\vartheta_{\mathrm{UCB}}(\boldsymbol{x})},$$

where β_t is a parameter controlling the level of exploration performed, $\mu_{t-1}(\cdot)$ and $\sigma_{t-1}(\cdot)$ are the posterior mean and variance computed via (2.4) based on the observations made prior to time step t. Intuitively, $\mu_{t-1}(\cdot)$ is the current estimate of f, while $\sigma_{t-1}(\cdot)$ is associated with its uncertainty. GP-UCB balances between exploitation and exploration: The first term $\mu_{t-1}(\cdot)$ encourages exploitation by favoring points with high posterior mean, while the second term $\beta_t^{1/2}\sigma_{t-1}(\cdot)$ is responsible for exploration, i.e. by favoring points with high level of uncertainty. This is illustrated in Figure 2.3, where a few iterations of GP-UCB together with the sampled points are shown. Finally, the exploration parameter β_t is set such that the true function f lies within the confidence bounds $\left[\mu_{t-1}(\cdot) \pm \beta_t^{1/2}\sigma_{t-1}(\cdot)\right]$ for every t with high probability (see Section 2.2.1 for details). We outline the complete pseudocode of GP-UCB in Algorithm 2.

Besides GP-UCB, other popular approaches include the following:

- **Probability of Improvement (PI):** This strategy was discovered very early in [Kus64]. It can be interpreted as favoring points that can improve upon the given target value ξ_t:

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \left\{ \mathbb{P}[f(\boldsymbol{x}) > \xi_t] := \Phi\left(\frac{\mu_{t-1}(\boldsymbol{x}) - \xi_t}{\sigma_{t-1}(\boldsymbol{x})} \right) \right\},$$

  where Φ is a standard normal CDF. Different heuristics exist that provide guidance on how ξ_t should be set. One way is to set ξ_t to the best-observed value so far. If it is not chosen "correctly", this causes PI to exploit aggressively and not find the maximizer [SSW+16].

- **Expected Improvement (EI):** EI [SSW+16] chooses

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \ \mathbb{E}_t[(f(\boldsymbol{x}) - \xi_t)\mathbb{1}_{\{f(\boldsymbol{x}) > \xi_t\}}],$$

  where $\mathbb{E}_t[\cdot]$ denotes averaging with respect to the posterior distribution at time t, and ξ_t is the parameter discussed in PI. Since the posterior is Gaussian, $f(\boldsymbol{x}) \sim \mathcal{N}(\mu_{t-1}(\boldsymbol{x}), \sigma_{t-1}^2(\boldsymbol{x}))$, the expectation can easily be expressed in closed form.

- **Entropy Search (ES):** The ES algorithm [HS12] seeks to minimize the uncertainty of the maximizer $\boldsymbol{x}^*$. It is given by the following rule:

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \; H[p_{t-1}(\boldsymbol{x}^*)] - \mathbb{E}_{p_{t-1|\boldsymbol{x}}(y)}[H[p_{t-1|(\boldsymbol{x},y)}(\boldsymbol{x}^*)]], \tag{2.5}$$

  where $H[p(\boldsymbol{x})] = \int_D p(\boldsymbol{x}) \log \frac{1}{p(\boldsymbol{x})} d\boldsymbol{x}$ denotes the differential entropy, $p_{t-1}(\boldsymbol{x}^*)$ and $p_{t-1|(\boldsymbol{x},y)}(\boldsymbol{x}^*)$ are the posterior distributions over the unknown maximizer $\boldsymbol{x}^*$ given all the sampling locations and their observations up to time t, and in the later case, given an additional data pair $(\boldsymbol{x}, y)$. The averaging is over the distribution $p_{t-1|\boldsymbol{x}}(y)$ of the random variable y, which is Gaussian with mean $\mu_{t-1}(\boldsymbol{x})$ and variance $\sigma_{t-1}(\boldsymbol{x})^2 + \sigma^2$. Since the exact evaluation of the ES rule is intractable, it is approximated using Monte Carlo techniques to average with respect to the posterior distribution and the observations.

The previously considered methods EI, PI and GP-UCB, are *myopic* in nature, i.e., they all select the currently best point and ignore its impact on the future selection steps. On the other hand, ES is a representative of the *one-step lookahead* methods, as it considers how the selection of a point $\boldsymbol{x}$ (together with its possible observations) would change the uncertainty of the maximizer $\boldsymbol{x}^*$. While one-step lookahead methods are computationally more expensive, they allow for some important settings (e.g., pointwise costs) that are typically non-trivial to incorporate into the myopic algorithms (see Chapter 3 for more details).

We note that recently several additional methods for BO have also been proposed, e.g. [FPD08, HLHG14, Met16, WJ17, RMGO18], etc. In this dissertation, we will mostly focus on the GP-UCB algorithm due to its simplicity and theoretical properties.

Regret

The third component in our modular view of BO is the performance metric. Same as in the *multi-armed bandit* literature [BCB12], we use *regret* to measure optimality of a sequence of decisions. Let $\boldsymbol{x}^*$ be the maximizer of the unknown function (it needs not be unique). We consider two widely considered performance metrics:

- **Cumulative regret**: Let $\boldsymbol{x}_t$ denote the point sampled by the algorithm at time t. At the end of T rounds, the cumulative regret incurred by the algorithm is given by

$$R_T = \sum_{t=1}^{T} (f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t)). \tag{2.6}$$

- **Simple regret**: At the end of T rounds, an additional point $\boldsymbol{x}^{(T)}$ is reported, and the simple regret incurred is:

$$r^{(T)} = f(\boldsymbol{x}^*) - f(\boldsymbol{x}^{(T)}). \tag{2.7}$$

Simple regret is a useful performance metric when the goal is to find the single best design.

For example, in the model selection task we want to discover the best set of hyperparameters to use as the input to our learning algorithm. It is also worth noting that the strategy for reporting $\boldsymbol{x}^{(T)}$ can be different from the strategy that is used to sample points at every time step. Cumulative regret, on the other hand, is typically of interest in applications where the relevance of decisions that we make at every time step is important (i.e., it encourages exploitation). For instance, when iteratively recommending items to a user, every recommendation should be up to the user's liking.

A desired asymptotic property of an algorithm is to be *no-regret*, which means:

$$\lim_{T\to\infty} \frac{R_T}{T} = 0,$$

implying that an algorithm should only incur sub-linear cumulative regret, i.e. $R_T = o(T)$. Simply put, this property means that for a sufficiently large time horizon the algorithm can discover an optimal point. If the reported $\boldsymbol{x}^{(T)}$ in the simple regret is obtained via

$$\boldsymbol{x}^{(T)} = \underset{\boldsymbol{x}\in\{\boldsymbol{x}_1,\dots,\boldsymbol{x}_T\}}{\arg\max}\ f(\boldsymbol{x}), \tag{2.8}$$

then $r^{(T)} \leq \frac{R_T}{T}$. Hence, in this case, the bounds on the average cumulative regret $\frac{R_T}{T}$ translate into the convergence rates for GP optimization. Note that the reporting strategy from (2.8), can only be executed by an algorithm in the noiseless setting where the true values of the unknown function are observed. In the setting where the observations are noisy, the used reporting strategies are slightly different. One popular option in such case is to report the point (among the sampled points) that has the highest posterior mean in the final round.

Both cumulative and simple regret are standard performance criteria used in the bandit optimization literature (e.g., [BCB12]). Cumulative regret is a measure used in the classical *multi-armed bandit problem (MAB)* to evaluate the performance of the exploration/exploitation algorithms. Simple regret is, on the other hand, the performance criteria used in the related problem of *best arm identification* [AB10, KCG16]. The links between these two notions of regret are studied in [MBS09], and the algorithms designed to optimize cumulative regret are frequently used to solve the best arm identification problem [JN14]. One perspective on BO is that it refers to the group of bandit optimization methods in which a prior distribution over the unknown reward function is assumed. In this dissertation, we sometimes refer to our problem as *Gaussian process bandit optimization* when the goal is to optimize the cumulative regret.

2.2.1 A Review of Theoretical Results in GP Optimization

In this section, we present the main theoretical results from [SKKS10], which are relevant to the work presented in this dissertation. We consider both Bayesian and non-Bayesian settings and their corresponding assumptions (Table 2.1). In the subsequent chapters, we will most often consider both settings for our problems.

Bayesian setting

In the Bayesian setting, we assume that the unknown function is random and distributed according to a GP, i.e., $f \sim \mathrm{GP}(0, k)$. This assumption (setting) is precisely the one considered in the previous section. Before stating the regret bounds for the GP-UCB algorithm in this setting, we introduce the quantity that characterizes the complexity of a GP optimization problem.

Assume $f \sim \mathrm{GP}(0, k)$ and $k(\boldsymbol{x}, \boldsymbol{x}) \leq 1$ for every $\boldsymbol{x} \in D$, i.e., the variance is bounded at every $\boldsymbol{x}$ in the input space D. The latter assumption holds for both SE and Mátern kernels, and in general, allows for regret bounds to be *scale-free*[2]. We consider the noisy setting, that is, when querying f at some point $\boldsymbol{x} \in D$, we receive a noisy observation $y = f(\boldsymbol{x}) + z$ where $z \sim \mathcal{N}(0, \sigma^2)$ is i.i.d. Gaussian noise.

Mutual information. Let $I(X; Y)$ denote the mutual information [CT01]. The informativeness of a set of sampling points $S = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T\}$ in the case of a GP model with Gaussian observational noise, is measured by the mutual information between f and corresponding observations $\boldsymbol{y}_S = (y_1, \cdots, y_T)$, and is given by:

$$I(\boldsymbol{y}_S; f) = I(\boldsymbol{y}_S; \boldsymbol{f}_S) = \frac{1}{2} \log \det \left(\boldsymbol{I}_T + \sigma^{-2} \boldsymbol{K}_T \right), \tag{2.9}$$

where $\boldsymbol{f}_S = (f(\boldsymbol{x}_1), \cdots, f(\boldsymbol{x}_T))$, $\boldsymbol{K}_T = [k(\boldsymbol{x}, \boldsymbol{x}')]_{\boldsymbol{x}, \boldsymbol{x}' \in S}$ is the kernel matrix of size $T \times T$, and $\boldsymbol{I}_T$ is the identity matrix of the corresponding size. Hence, the maximum amount of information that any set of T noisy samples $\boldsymbol{y}_S = (y_1, \cdots, y_T)$ can reveal about $f \sim \mathrm{GP}(0, k)$ is given by

$$\gamma_T = \max_{\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\} \subset D} \frac{1}{2} \log \det \left(\boldsymbol{I}_T + \sigma^{-2} \boldsymbol{K}_T \right), \tag{2.10}$$

We refer to γ_T as the *maximum information gain*. As shown in [SKKS10], when $D \subset \mathbb{R}^p$ is compact and convex, for the SE kernel, it holds

$$\gamma_T = O\left((\log T)^{p+1} \right), \tag{2.11}$$

and for the Matérn kernel with parameter ν, we have

$$\gamma_T = O^*\left(T^{\frac{p(p+1)}{2\nu + p(p+1)}} \right). \tag{2.12}$$

We will make use of these bounds ((2.11) and (2.12)) in the subsequent chapters, as the maximum information gain will also govern the obtained regret bounds.

Next, we state the regret bounds that hold for the GP-UCB algorithm in the Bayesian setting in the case of finite D.

Regret bounds. The following confidence lemma is the key for obtaining the regret bounds.

[2] If $k(\boldsymbol{x}, \boldsymbol{x}) \leq c$ for some constant $c \geq 1$ it can be reduced to the unit setting by dividing all samples by $\sqrt{c}$.

Setting	Bayesian	non-Bayesian
Assumption	$f \sim \mathrm{GP}(0, k)$	f is arbitrary with $\|f\|_k \leq B$

Table 2.1: Bayesian and non-Bayesian settings and their correponding assumptions. The kernel function k is typically assumed to be known in both settings.

Lemma 2.2.1 ([SKKS10]). *For each t, define $\beta_t = 2\log\frac{|D|t^2\pi^2}{6\delta}$. With probability at least $1-\delta$, we have for all $\boldsymbol{x}$ and t that $|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})$.*

Hence, the confidence parameter β_t is chosen to ensure that f remains with high probability within the confidence bounds over the run of the algorithm, i.e., $f(\boldsymbol{x}) \in \left[\mu_{t-1}(\boldsymbol{x}) \pm \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right]$ for every $\boldsymbol{x} \in D$ and $t \in \{1, \cdots, T\}$.

Theorem 2.2.1 ([SKKS10, Theorem 1]). *Let D be a finite set, $f \sim \mathrm{GP}(0, k)$ and $\delta \in (0, 1)$. When running* GP-UCB *with $\mathrm{GP}(0, k)$, known noise model, and β_t set according to Lemma 2.2.1, the following regret bound holds for every $T \geq 1$ with probability at least $1-\delta$:*

$$R_T \leq \sqrt{C_1 T \beta_T \gamma_T} \tag{2.13}$$

where $C_1 = \frac{8}{\log(1+\sigma^{-2})}$.

Under some regularity assumptions (see [SKKS10, Theorem 2]), the regret bound is of the same form (2.13) in the case $D \subset [0,1]^p (p \in \mathbb{N})$; in such case we run GP-UCB with $\beta_t \propto p\log(tp/\delta)$. By substituting the bounds obtained for γ_T (e.g., (2.11)) in the regret bound (2.13), it can be observed that GP-UCB attains sub-linear cumulative regret, i.e., $\lim_{T\to\infty}\frac{R_T}{T} = 0$.

Non-Bayesian Setting

Previously, we assumed that f is random and distributed according to $\mathrm{GP}(0, k)$ with some known kernel function k. In this section, we assume that the reward f is an arbitrary fixed function. Obtaining vanishing regret bounds for an arbitrary and unknown reward function is impossible in general, and hence, further assumptions are needed. The relation between GP kernel functions and the notion of the *reproducing kernel Hilbert space (RKHS)* [RW06, Section 6.1] motivates the smoothness assumption that we make in this section.

Assume that the input space D is compact, and the unknown function $f : D \to \mathbb{R}$ belongs to the RKHS $\mathcal{H}_k(D)$ of functions induced by the known kernel function $k : D \times D \to \mathbb{R}$, with $k(\boldsymbol{x}, \boldsymbol{x}) \leq 1$ for every $\boldsymbol{x} \in D$. Moreover, we assume that f has a small *RKHS norm* which, we will see below, enforces smoothness of f. Formally, we assume B is a known constant and $f \in \mathcal{F}_k(B)$ where

$$\mathcal{F}_k(B) = \{f \in \mathcal{H}_k(D) \,:\, \|f\|_k \leq B\}. \tag{2.14}$$

The RKHS $\mathcal{H}_k(D)$ is a Hilbert space of real functions uniquely determined by the kernel k (the opposite also holds), with an inner product $\langle \cdot, \cdot \rangle_k$ that obeys the reproducing property: $f(\boldsymbol{x}) = \langle f, k(\boldsymbol{x}, \cdot) \rangle$ for all $f \in \mathcal{H}_k(D)$. It also holds that for every $\boldsymbol{x} \in D$, $k(\boldsymbol{x}, \boldsymbol{x}')$ as a function of $\boldsymbol{x}'$ belongs to $\mathcal{H}_k(D)$. The induced RKHS norm, $\|f\|_k = \langle f, f \rangle_k^{1/2}$, measures the smoothness of every $f \in \mathcal{H}_k(D)$ with respect to the kernel function k. This can be observed by the following relation that holds for every $f \in \mathcal{H}_k(D)$ and $\boldsymbol{x}, \boldsymbol{x}' \in D$:

$$|f(\boldsymbol{x}) - f(\boldsymbol{x}')| = |\langle f, k(\boldsymbol{x}, \cdot) - k(\boldsymbol{x}', \cdot) \rangle_k| \leq \|f\|_k \|k(\boldsymbol{x}, \cdot) - k(\boldsymbol{x}', \cdot)\|_k,$$

which follows by the reproducing property and the Cauchy-Schwartz inequality. In the same way that different kernel functions dictate smoothness of GP samples (see Figure 2.1), they also lead to different penalization as measured by the corresponding RKHS norm $\| \cdot \|_k$.

In [SKKS10], it is shown that when f is in $\mathcal{F}_k(B)$, we can use the GP-UCB algorithm with the prior $\mathrm{GP}(0, k)$ as the input, and perform the same Bayesian posterior updates. In this case, we are running the algorithm with a misspecified[3] prior. The key is in the fact that the assumption $f \in \mathcal{F}_k(B)$ allows for setting the confidence parameter β_t to ensure that f remains within the confidence bounds (that are based on the GP posterior) with high probability during the run of the algorithm. Hence, in this setting as well, we can obtain the confidence lemma analogously to the Bayesian setting (Lemma 2.2.1) for appropriately set β_t.

Theorem 2.2.2 ([SKKS10, Theorem 3])**.** *Let D be a compact set and f is in $\mathcal{F}_k(B)$, and assume the observation noise is zero mean conditioned on the history and bounded by σ almost surely. Choose $\delta \in (0, 1)$ and run* GP-UCB *with* $\mathrm{GP}(0, k)$*, Gaussian noise model $\mathcal{N}(0, \sigma^2)$, and $\beta_t = (2B + 300\gamma_t \log^3(t/\delta))$. With probability at least $1 - \delta$ we have*

$$R_T \leq \sqrt{C_1 T \beta_T \gamma_T}, \tag{2.15}$$

for every $T \geq 1$, where $C_1 = \frac{8}{\log(1+\sigma^{-2})}$.

Compared to the Bayesian setting, the regret bound is of the same form (2.13) except for the change in the confidence parameter β_T. By substituting its value, the regret bound is

$$R_T = O^*\left(\sqrt{TB\gamma_T + T\gamma_T^2}\right). \tag{2.16}$$

In the case of the SE kernel, after replacing γ_T in (2.16) with its upper bound from (2.11), we observe that GP-UCB also attains sub-linear cumulative regret in this setting.

We conclude this section by mentioning that other aspects (e.g., robust, contextual and time-varying) of Bayesian optimization and its related problems (e.g., level-set estimation) will be overviewed in the subsequent relevant chapters.

[3] Because the misspecified prior is used, the authors in [SKKS10] also refer to this setting as *agnostic*.

2.3 A Review of (Robust) Submodular Maximization

We are motivated by the problem in which we seek to select a robust set of experiments in the case that some of them might result in failure. In this setting, we are uncertain about which experiments will fail, and hence, we need to make a robust decision when choosing which of them to run. This problem is combinatorial in nature, and while discrete combinatorial optimization problems frequently appear in machine learning, they are often NP-hard even to approximate. In the case of a set objective function that satisfies a natural notion of diminishing returns – *submodularity*, we can efficiently perform optimization in several different problem formulations and obtain strong theoretical guarantees.

Submodular objectives (to be defined shortly) appear in many applications of interest including influence maximization [KKT03], data summarization [Mir17], sensor networks [KSG08], active learning [WIB15], just to name a few. Another example is the mutual information $I(\boldsymbol{y}_S, \boldsymbol{f}_S)$ (*cf.* (2.9)), which we used in the previous section to measure the informativeness of a set of experiments S (sampling locations) in the case of a GP model and noisy observations. It turns out that this set function $f(S) = I(\boldsymbol{y}_S, \boldsymbol{f}_S)$ also satisfies submodularity [KG05a], which allows for efficient approximate optimization.

We start this section by introducing submodular functions, and then we look at two submodular optimization problems, namely, *submodular maximization* subject to a cardinality constraint and the *submodular covering* problem. An essential difference between this and the previous sections is that now our objective set function is known and we can evaluate it (i.e., obtain its true value) for any set of items. Furthermore, we introduce the robust submodular maximization problem, in which our goal is to protect against worst-case adversarial removals/failures [KMGG08].

Submodularity

Let V be a ground set of items with cardinality $|V| = n$, e.g., all the different experiments that we can choose to run. Consider a set function $f : 2^V \to \mathbb{R}$ defined on V that quantifies the utility of every set $S \subseteq V$. For example, this function can quantify how informative each set of experiments is with respect to the experimentally measured quantity. We use the following notation to denote the marginal gain in the function value due to adding the elements of a set $S \subseteq V$ to the set $P \subseteq V$:

$$f(S|P) := f(S \cup P) - f(P).$$

In the case that S is a singleton of the form $\{v\}$, we adopt the shorthand $f(v|S)$.

Submodular set functions satisfy a natural notion of diminishing returns. Intuitively, this means that the gain that we get in terms of the submodular objective value by adding an item earlier is at least as high as adding it later in the selection process. The following definition of submodular functions captures this intuition.

Definition 2.3.1 (Submodularity)**.** *A set function $f : 2^V \to \mathbb{R}$ is said to be* submodular *if for any sets $S \subseteq P \subseteq V$ and any element $v \in V \setminus Y$, it holds that*

$$f(S \cup \{v\}) - f(S) \geq f(P \cup \{v\}) - f(P). \tag{2.17}$$

This can equivalently be written as $f(v|S) \geq f(v|P)$. We refer to f as being *modular* in the case the previous definition holds with $f(v|S) = f(v|P)$. In this disseration, we will only consider *monotone* set functions meaning that for all sets $S \subseteq P \subseteq V$ we have $f(S) \leq f(P)$.

Submodular Maximization and Coverage

In this section, we consider the two related problems *submodular maximization* and *submodular covering*, both of which are known to be NP-hard in general [KG12].

In the first problem, the goal is to maximize a normalized[4] monotone submodular function $f(\cdot)$ subject to a cardinality constraint k. For a fixed k we want to find a set $S \subset V$ that solves

$$S \in \operatorname*{arg\,max}_{S \subseteq V, |S| \leq k} f(S). \tag{2.18}$$

For instance, in experimental design, this problem amounts to finding the set of k most informative experiments, and in influence maximization, it corresponds to finding k most influential users in the social network.

The following simple GREEDY algorithm starts with the empty set ($S_0 = \emptyset$), and then at every iteration adds the element with the highest maginal gain[5] $f(v|S)$, that is,

$$S_i = S_{i-1} \cup \{\operatorname*{arg\,max}_{v \in V} f(v|S)\}, \tag{2.19}$$

provides a constant factor approximation guarantee for the problem in (2.18). This finding is a celebrated result from [NWF78], which we formalize in the following theorem.

Theorem 2.3.1 ([NWF78])**.** *Let $f : 2^V \to \mathbb{R}_+$ be a monotone submodular function, and let* $\text{OPT}(k, V)$ *denote the optimal solution of size k, i.e.,*

$$\text{OPT}(k, V) \in \operatorname*{arg\,max}_{S \subseteq V, |S| \leq k} f(S).$$

Let S_k be the set of size k obtained via GREEDY (2.19)*. Then it follows,*

$$f(S_k) \geq (1 - 1/e) f(\text{OPT}(k, V)). \tag{2.20}$$

[4] We say a set function is normalized if $f(\emptyset) = 0$.

[5] We break ties arbitrarily in the case they occur.

S	$f(S)$	$\min_{v \in S} f(S \setminus v)$
$\emptyset$	0	0
$\{s_1\}$	n	0
$\{s_2\}$	ϵ	0
$\{s_3\}$	$n-1$	0
$\{s_1, s_2\}$	$n+\epsilon$	ϵ
$\{s_1, s_3\}$	n	$\mathbf{n-1}$
$\{s_2, s_3\}$	n	ϵ

Table 2.2: Function f used to demonstrate that GREEDY can perform arbitrarily badly.

In the case that we run GREEDY for some number of iterations $l \neq k$ ($l \in \mathbb{N}_+$), then the result from the previous theorem becomes [KG12]: $f(S_l) \geq (1-e^{-\frac{l}{k}})f(\text{OPT}(k, V))$. Furthermore, for general submodular objectives, it holds that no algorithm that is allowed to evaluate the objective at a polynomial number of sets cannot improve upon the obtained $(1 - 1/e)$-approximation factor [NWF78].

The second related problem that we consider is the *submodular covering problem* in which we seek to find a subset of the smallest size such that its utility is above some given threshold h. Formally, we wish to solve the following:

$$S \in \arg\min_{S \subseteq V} |S| \quad \text{subject to} \quad f(S) \geq h, \tag{2.21}$$

where h is some utility threshold in $[0, f(V)]$. In the budgeted version of the submodular covering problem [KG05b] we are given a cost function $c : 2^V \to \mathbb{R}_+$ and our goal is to instead minimize the cost $c(S)$ associated with the set S subject to $f(S) \geq h$. We consider this problem in Chapter 3 in the context of BO, where our objective is to select a set of most informative experiments (i.e., sampling points) but whose cost is as small as possible.

The following theorem characterizes the performance of the GREEDY algorithm when applied to the submodular covering problem in (2.21).

Theorem 2.3.2 ([Wol82])**.** *Consider a monotone submodular function $f : 2^V \to \mathbb{N}$. Given some threshold $h \in [0, f(V)]$, let $\text{OPT}_h \subseteq V$ denote the smallest size set such that $f(\text{OPT}_h) \geq h$. Let S be the first set obtained via* GREEDY *in* (2.19) *such that $f(S) \geq h$. In such case, the size of the solution set S can be bounded as follows:*

$$|S| \leq \left(1 + \ln \max_{v \in V} f(\{v\})\right) |\text{OPT}_h|.$$

We refer the reader to [KG12] for more details on submodular functions and optimization.

Robust Submodular Optimization

The following *robust* version of the problem in (2.18) was introduced in [KMGG08]:

$$\max_{S\subseteq V,|S|\leq k} \min_{E\subseteq S,|E|\leq \tau} f(S\setminus E), \tag{2.22}$$

where $f : 2^V \to \mathbb{R}_+$ is a monotone submodular function. We refer to τ as the *robustness parameter*, representing the size of the subset E that is removed from the selected set S. Our goal is to find a set S such that it is robust upon the *worst-case* adversarial removal of τ elements, i.e., after the removal, the objective value should remain as large as possible. For $\tau = 0$, the robust problem reduces to the non-robust submodular maximization problem in (2.18).

As mentioned before, this problem is of interest when we want to select a set of the most informative experiments that are costly to run and when some number of them might fail. In the case that we do not have any prior information on how experiments might fail, protecting against the worst-case failure becomes essential.

The GREEDY algorithm, which is near-optimal for the non-robust problem in (2.18) (Theorem 2.3.1) can perform arbitrarily badly for Problem (2.22). As an elementary example, let us fix $\epsilon \in [0, n-1)$ and $n \geq 0$, and consider the non-negative monotone submodular function given in Table 2.2. For $k = 2$, the GREEDY algorithm selects $\{s_1, s_2\}$. The set that maximizes $\min_{v\in S} f(S\setminus v)$ (i.e., $\tau = 1$) is $\{s_1, s_3\}$. For this set, $\min_{v\in\{s_1,s_2\}} f(\{s_1, s_2\}\setminus v) = n-1$, while for the greedy set the robust objective value is ϵ. As a result, the GREEDY algorithm can perform arbitrarily worse. In our experiments on real-world data sets (see Section 6.3), we further explore the empirical behavior of the greedy solution in the robust setting. Among other things, we observe that the greedy solution tends to be less robust when the objective value largely depends on the first few elements selected by the greedy rule.

In [KMGG08], the authors consider the following generalization of Problem (2.22):

$$\max_{S\subseteq V,|S|\leq k} \min_{i\in\{1,\ldots,n\}} f_i(S), \tag{2.23}$$

where $\{f_i\}_{i=1}^n$ is a finite set of normalized monotone submodular functions.[6] This problem is inapproximable in general however, the authors propose a bi-criteria approximation algorithm SATURATE which, when applied to Problem (2.22), returns a set of size $k(1+\Theta(\log(\tau k \log n)))$ and whose robust objective is at least as good as the optimal size-k set. SATURATE requires a number of function evaluations that is *exponential* in τ, making it very expensive to run even for small values.

In [OSU16], the authors consider the robust formulation in (2.22), and provide the first efficient algorithm with constant 0.387-factor approximation result, valid in the case that the maximum number of allowed removals (i.e., the budget of the worst-case adversary) is $\tau = o(\sqrt{k})$.

[6] Note that the minimum of submodular functions is not submodular in general [KG12].

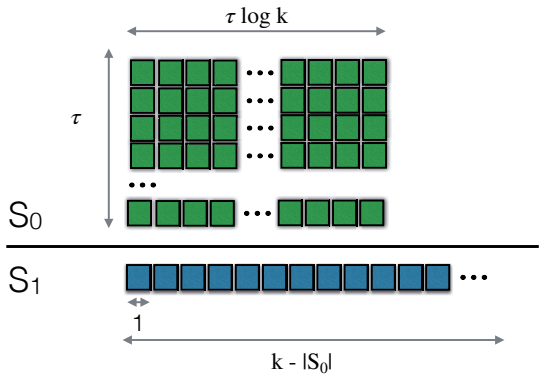


Figure 2.4: Illustration of the solution set $S = S_0 \cup S_1$ returned by the OSU algorithm. Each square represents a single element of the solution set (k elements in total), and each row corresponds to the elements selected in a single run of GREEDY. In the first τ runs of GREEDY, each solution is of size $\tau \log k$; the union of the selected elements corresponds to the set S_0. Finally, in the last run of GREEDY, which corresponds to S_1, the solution is of size $k - |S_0|$.

Further on, we will refer to the algorithm proposed therein via the authors' surnames as OSU. The algorithm uses GREEDY as a sub-routine $\tau + 1$ times. On each iteration, GREEDY is applied on the elements that are not yet selected on previous iterations, with these previously-selected elements ignored in the objective function. In the first τ runs, each greedy solution is of size $\tau \log k$, while in the last run, the solution is of size $k - \tau^2 \log k$ (what remains until the cardinality constraint k is satisfied). The union of all the obtained disjoint solutions leads to the final solution set. The illustration of the solution set whose total size is k is given in Figure 2.4. We will consider the problem in (2.22) further in Chapters 6 and 7, and propose new efficient algorithms.

More recently, in [KZK17], the authors have addressed the robust submodular maximization problem with adversarial removals by proposing the two-stage algorithm that selects more than k elements (i.e., it requires memory of size $O(k + \tau \log k/\delta^2)$) and achieves $1/2 - \delta$ approximation guarantee (in expectation). In order to produce the solution of size k from the larger selected set, their algorithm requires (as input in the second stage) the adversary to reveal the elements to be removed. Hence, this is a substantially different setting than the previously considered one from [OSU16]. Robust submodular maximization with respect to removals has also been considered in the streaming [MBNF$^+$17, MKK17], distributed [BMSC17a, KZK17] and sequential [TJP18] settings. In addition, robust versions of submodular maximization have been applied to influence maximization [HK16, CLT$^+$16] and budget allocation problems [SJ17b]. The work of [PBW$^+$16] considers the problem in (2.23) for different types of submodular constraints. In [HS17], the authors study the problem of maximizing a monotone submodular function under adversarial noise. We conclude this section by noting that very recently, a couple of different works have studied various related robust submodular problems [Udw17, Wil17, AHP$^+$17, CLSS17].

3 Versatile and Cost-effective Bayesian Optimization & Level-set Estimation

Bayesian optimization (BO) and level-set estimation (LSE) are related problems that are typically studied in isolation. In BO the goal is to find a maximizer of the unknown function alone, while in LSE one seeks to find all "sufficiently good" points. In this chapter, we present a new algorithm that addresses Bayesian optimization and level-set estimation with Gaussian processes in a *unified* fashion. The proposed algorithm is *cost-effective*, and unlike the standard methods, it comes with theoretical guarantees for the setting where sampling points incur different costs.

This chapter is based on the joint work with Jonathan Scarlett, Andreas Krause and Volkan Cevher [BSKC16].

3.1 Introduction

Bayesian optimization (BO) [SSW+16] provides a powerful framework for automating design problems, and finds applications in robotics, environmental monitoring, and automated machine learning, just to name a few. One seeks to find the maximum of an unknown reward function that is expensive to evaluate, based on a sequence of suitably-chosen points and noisy observations. Numerous BO algorithms have been presented previously; see Section 2.2 for an overview. Level-set estimation (LSE), e.g. [GCHK13], is closely related to BO, with the added twist that instead of seeking a maximizer, one seeks to classify the domain into points that lie above or below a certain threshold. This is of considerable interest in applications where solutions can "fail", as well as in applications such as environmental monitoring and sensor networks, allowing one to find all “sufficiently good” points rather than the best point alone.

In this chapter, we provide a unified treatment of the two via a new algorithm, *Truncated Variance Reduction* (TRUVAR), which enjoys theoretical guarantees, good computational complexity, and the versatility to handle important settings such as *pointwise costs*, *non-constant noise*, and *multi-task* scenarios. The main result of this chapter applies to the former two settings, and even in the fixed-noise and unit-cost case, we refine the existing bounds via a significantly improved dependence on the noise level.

3.1.1 Problem Statement

We seek to sequentially optimize an unknown reward function $f(\boldsymbol{x})$ over a finite domain D.[1] At time t, we query a single point $\boldsymbol{x}_t \in D$ and observe a noisy sample $y_t = f_t(\boldsymbol{x}_t) + z_t$, where $z_t \sim \mathcal{N}(0, \sigma^2(\boldsymbol{x}_t))$ for some known noise function $\sigma^2(\cdot) : D \to \mathbb{R}_+$. In general, some points may be noisier than others, in which case we have *heteroscedastic noise* [GWB97]. We associate with each point a *cost* according to some known cost function $c : D \to \mathbb{R}_+$. If both $\sigma^2(\cdot)$ and $c(\cdot)$ are set to be constant, then we recover the standard homoscedastic and unit-cost setting.

We model $f(\boldsymbol{x})$ as a Gaussian process (GP) [RW06] having mean zero and kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$, normalized so that $k(\boldsymbol{x}, \boldsymbol{x}) = 1$ for all $\boldsymbol{x} \in D$. The posterior distribution of f given the points and observations up to time t is again a GP, with the posterior mean and variance [GWB97]:

$$\mu_t(\boldsymbol{x}) = \boldsymbol{k}_t(\boldsymbol{x})^T \big(\boldsymbol{K}_t + \boldsymbol{\Sigma}_t\big)^{-1} \boldsymbol{y}_t \tag{3.1}$$

$$\sigma_t(\boldsymbol{x}, \boldsymbol{x})^2 = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}_t(\boldsymbol{x})^T \big(\boldsymbol{K}_t + \boldsymbol{\Sigma}_t\big)^{-1} \boldsymbol{k}_t(\boldsymbol{x}), \tag{3.2}$$

where $\boldsymbol{k}_t(\boldsymbol{x}) = \big[k(\boldsymbol{x}_i, \boldsymbol{x})\big]_{i=1}^t$, $\boldsymbol{K}_t = \big[k(\boldsymbol{x}_t, \boldsymbol{x}_{t'})\big]_{t,t'}$, and $\boldsymbol{\Sigma}_t = \operatorname{diag}(\sigma^2(\boldsymbol{x}_1), \ldots, \sigma^2(\boldsymbol{x}_t))$. Note, that the posterior update rule is the same as the one in (2.4), except that here due to the heteroscedastic noise assumption, $\sigma^2 \boldsymbol{I}_t$ is replaced with $\boldsymbol{\Sigma}_t$. We also let $\sigma^2_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}})$ denote the posterior variance of $\overline{\boldsymbol{x}}$ upon observing $\boldsymbol{x}$ along with $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{t-1}$.

We consider the standard goal in Bayesian optimization: After T time steps and for some accuracy value $\epsilon \geq 0$, the goal is to report a point $\boldsymbol{x}^{(T)}$ such that

$$f(\boldsymbol{x}^*) - f(\boldsymbol{x}^{(T)}) \leq \epsilon,$$

where $\boldsymbol{x}^* \in \arg\max_{\boldsymbol{x} \in D} f(\boldsymbol{x})$. We also refer to the difference $f(\boldsymbol{x}^*) - f(\boldsymbol{x}^{(T)})$ as simple regret $r^{(T)}$ (see Section 2.2), and we seek to characterize time T to ϵ-simple regret, i.e., $r^{(T)} \leq \epsilon$.

A related problem to BO is *level-set estimation* [GCHK13]. While in BO we seek to find a maximizer of the unknown function, in level-set estimation, we are interested in classifying the domain into points that lie above or below a given threshold h. This means that after T time steps and for some accuracy value $\epsilon \geq 0$, the goal is to report sets $H_T \subset D$ and $L_T \subset D$ such that for every $\boldsymbol{x} \in H_T$ and $\boldsymbol{x} \in L_T$ the following holds $f(\boldsymbol{x}) > h$ and $f(\boldsymbol{x}) < h$, respectively. Moreover, the value of every remaining unclassified point should be at most ϵ above or below h.

3.1.2 Related Work

Three popular techniques for Bayesian optimization are expected improvement (EI), probability of improvement (PI), and Gaussian process upper confidence bound (GP-UCB) [SSW+16, SKKS10], each of which chooses the point maximizing an acquisition function depending directly on the current posterior mean and variance (see Section 2.2 for more details on these methods).

[1] Extensions to continuous domains and non-Bayesian settings are also discussed in Section 3.6.

In [CBRV13], the GP-UCB-PE algorithm was presented for BO, choosing the highest-variance point within a set of potential maximizers that is updated based on confidence bounds. Another relevant BO algorithm is BaMSOO [WSJF14], which also keeps track of potential maximizers, but instead chooses points based on a global optimization technique called simultaneous online optimization (SOO). An algorithm for level-set estimation with GPs is given in [GCHK13], which keeps track of a set of unclassified points. These algorithms have theoretical guarantees and good computational complexity, but it is unclear how best to incorporate important phenomena such as pointwise costs and multi-task scenarios [SSA13]. The same is true for a heuristic for LSE known as Straddle [BS08].

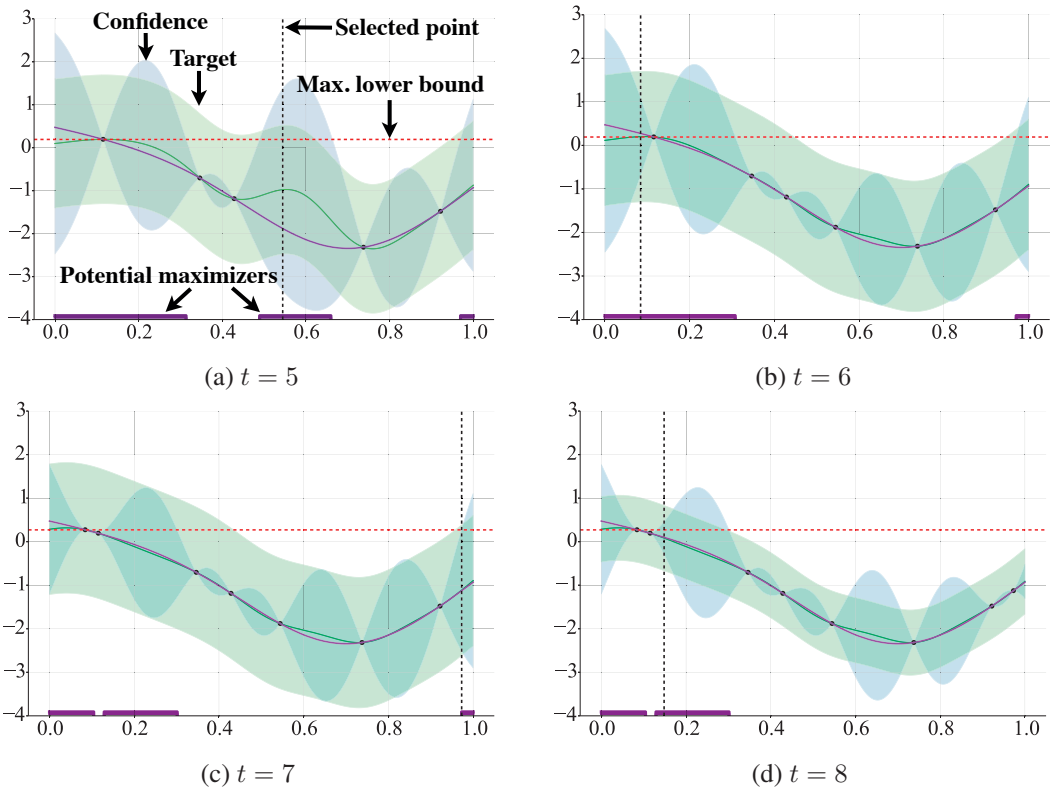
Entropy search (ES) [HS12] (Section 2.2) and its predictive version [HLHG14] choose points to reduce the uncertainty of the location of the maximum, doing so via a *one-step lookahead* of the posterior rather than directly using the current posterior. While this increases the computation, it also permits versatility with respect to costs [SSA13], heteroscedastic noise [GWB97], and multi-task scenarios [SSA13]. A recent approach called minimum regret search (MRS) [Met16] similarly performs a look-ahead, but instead chooses points to minimize the regret. To our knowledge, no theoretical guarantees have been provided for these one-step lookahead algorithms.

The multi-armed bandit (MAB) [BCB12] literature has developed alongside the BO literature, with the two often bearing similar concepts. The MAB literature is far too extensive to cover here, but we briefly mention some relevant variants. Extensive attention has been paid to the *best-arm identification* problem [JN14], and cost constraints have been incorporated in a variety of forms [MLG04]. The concept of "zooming in" to the optimal point has been explored [KSU08]. In general, the assumptions and analysis techniques in the MAB and BO literature are very different.

3.1.3 Contributions

The main contributions of this chapter are:

- In Section 3.3, we present the first unified analysis of BO and LSE via a new algorithm Truncated Variance Reduction (TruVaR). Similarly to ES and MRS, the algorithm performs a one-step lookahead that is highly beneficial in terms of versatility. However, unlike these previous works, our lookahead completely avoids the computationally expensive task of averaging over the posterior distribution and the observations.
- In contrast with the other one-step lookahead methods, ES and MRS, we provide theoretical bounds (Section 3.3) for TruVaR characterizing the *cost* required to achieve a certain accuracy in finding a near-optimal point in the case of BO or in classifying each point in the domain in the case of LSE.
- In the standard BO setting, we not only recover existing results in [GCHK13, CBRV13] (also in Theorem 2.2.1), but we also strengthen them in Section 3.4 via a significantly improved dependence on the noise level, with better asymptotics in the small noise limit.



(a) $t = 5$ (b) $t = 6$

(c) $t = 7$ (d) $t = 8$

Figure 3.1: An illustration of TRUVAR. In 3.1a, 3.1b, and 3.1c, three points within the set of potential maximizers M_t are selected in order to bring the confidence bounds to within the target range, and M_t shrinks during this process. In 3.1d, the target confidence width shrinks as a result of the last selected point bringing the confidence within M_t to within the previous target.

- In Section 3.5, we provide a novel result for the multi-fidelity setting in which the algorithm can choose the noise level (when sampling), each coming with an associated cost. We also extend our results to the *non-Bayesian* setting in Section 3.6, and show that our regret bounds in the case of the SE kernel nearly match the lower bounds from [SBC17].
- Finally, in Section 3.7, we compare TRUVAR to previous works on a variety of synthetic and real-world data sets, finding it to perform favorably in a diverse range of settings.

3.2 Truncated Variance Reduction Algorithm

3.2.1 TruVaR for Bayesian Optimization

Our algorithm is presented in Algorithm 3, making use of the updates described in Algorithm 4. TRUVAR keeps track of potential maximizers M_t. This is a set of points that still have a chance of being maximizers according to the confidence bounds at time t. For $t = 0$ this set is initialized to the whole domain $M_{(0)} = D$ (i.e., in the beginning, every point is a potential maximizer).

Algorithm 3 Truncated Variance Reduction (TRUVAR) for Bayesian Optimization [BSKC16]

Input: Domain D, GP prior (μ_0, σ_0, k), confidence bound parameters $\overline{\delta} > 0$, $r \in (0,1)$, $\{\beta_{(i)}\}_{i\geq 1}$, $\eta_{(1)} > 0$

1: Initialize the epoch number $i = 1$, target confidence $\eta_{(1)}$, potential maximizers $M_{(0)} = D$

2: **for** $t = 1, 2, \ldots$ **do**

3: Choose

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}\in D} \frac{\sum_{\overline{\boldsymbol{x}}\in M_{t-1}} \max\left\{\beta_{(i)}\sigma^2_{t-1}(\overline{\boldsymbol{x}}), \eta_{(i)}\right\} - \sum_{\overline{\boldsymbol{x}}\in M_{t-1}} \max\left\{\beta_{(i)}\sigma^2_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}}), \eta_{(i)}\right\}}{c(\boldsymbol{x})}.$$

4: Observe the noisy function sample y_t, and update according to Algorithm 4 to obtain M_t, μ_t, σ_t, lcb_t and ucb_t

5: **if** $\max_{\boldsymbol{x}\in M_t} \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}) \leq (1+\overline{\delta})\eta_{(i)}$ **then**

6: Increment i, set $\eta_{(i)} = r \times \eta_{(i-1)}$

Algorithm 4 BO Parameter Updates for TRUVAR [BSKC16]

Input: Selected points $\{\boldsymbol{x}_{t'}\}_{t'=1}^{t}$ and observation $\{y_{t'}\}_{t'=1}^{t}$, previous set M_{t-1}, parameter $\beta_{(i)}^{1/2}$.

1: Update μ_t and σ_t according to (3.1)–(3.2), and form the upper and lower confidence bounds:

$$\mathrm{ucb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) + \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}), \quad \mathrm{lcb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) - \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}).$$

2: Set

$$M_t = \left\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) \leq \max_{\overline{\boldsymbol{x}}\in M_{t-1}} \mathrm{lcb}_t(\overline{\boldsymbol{x}})\right\}.$$

In the later rounds it is updated based on the confidence bounds depending on constants $\beta_{(i)}$,

$$\mathrm{ucb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) + \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}), \quad \mathrm{lcb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) - \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}). \tag{3.3}$$

At time t, the set of potential maximizers is given by:

$$M_t = \left\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) \leq \max_{\overline{\boldsymbol{x}}\in M_{t-1}} \mathrm{lcb}_t(\overline{\boldsymbol{x}})\right\}. \tag{3.4}$$

The constraint $\boldsymbol{x} \in M_{t-1}$ in (3.4) ensures that set of potential maximizers is non-increasing over the course of the algorithm. Simply put, if a point has a smaller upper confidence bound than the highest known lower confidence bound, then such a point cannot be a maximizer (given the confidence bounds hold). The previously described updates are performed in Algorithm 4.

The algorithm proceeds in epochs (see Algorithm 3), where in each epoch i, it seeks to bring the *confidence* $\beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x})$ of points within the set M_t below a target value $\eta_{(i)}$ (hence, every epoch i can last different number of rounds that are indexed with t). TRUVAR does this by greedily minimizing the truncated sum of variances $\sum_{\overline{\boldsymbol{x}}\in M_{t-1}} \max\{\beta_{(i)}\sigma^2_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}}), \eta_{(i)}\}$ arising from choosing the point $\boldsymbol{x}$, along with a normalization and division by $c(\boldsymbol{x})$ to favor low-cost points.

Specifically, at every time step it chooses the point given by the following rule:

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \frac{\sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\left\{\beta_{(i)}\sigma^2_{t-1}(\overline{\boldsymbol{x}}),\ \eta_{(i)}\right\} - \sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\left\{\beta_{(i)}\sigma^2_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}}),\ \eta_{(i)}\right\}}{c(\boldsymbol{x})}. \tag{3.5}$$

We note that the first term $\sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\{\beta_{(i)}\sigma^2_{t-1}(\overline{\boldsymbol{x}}),\ \eta_{(i)}\}$ in the above rule is constant, i.e., it does not depend on $\boldsymbol{x}$. The truncation by $\eta_{(i)}$ in this decision rule means that once the confidence of a point is below the current target value, there is no preference in making it any lower (until the target is decreased). Once the confidence of every point in M_t is less than a factor $1 + \overline{\delta}$ above the target value, the target confidence is reduced according to a multiplication by $r \in (0, 1)$. An illustration of the process is given in Figure 3.1, with details in the caption.

The choices of $\beta_{(i)}$, $\overline{\delta}$, and r are discussed in Section 3.7. As with previous works, the kernel is assumed known in our theoretical results, whereas in practice it is typically learned from training data [SKKS10]. Characterizing the effect of model mismatch or online hyperparameter updates is an interesting direction for future work.

Some variants of our algorithm and theory are discussed in Section 3.7.3, including *pure* variance reduction, the batch setting [CBRV13], and a few others.

3.2.2 TruVaR for Level-Set Estimation

In this section, we present an algorithm for level-set estimation that uses the same truncated variance reduction acquisition rule (3.5). Our algorithm for level-set estimation is described in Algorithm 5, making use of the updates described in Algorithm 6. The algorithm accepts level set threshold h as an input. It keeps track of a sequence of unclassified points[2] M_t, representing points close to h. It also keeps track of the sets H_t and L_t, containing points believed to have function values above and below h, respectively.

The precise definitions of these sets are:

$$M_t = \left\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) \geq h \text{ and } \mathrm{lcb}_t(\boldsymbol{x}) \leq h\right\}, \tag{3.6}$$

$$H_t = H_{t-1} \cup \left\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{lcb}_t(\boldsymbol{x}) > h\right\}, \tag{3.7}$$

$$L_t = L_{t-1} \cup \left\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) < h\right\}. \tag{3.8}$$

The constraint $\boldsymbol{x} \in M_{t-1}$ in (3.6)–(3.8) ensures that M_t is non-increasing with respect to inclusion, and H_t and L_t are non-decreasing.

The precise performance criteria for the LSE setting is given in Definition 3.3.1 below; essentially, after spending a certain cost we report a classification (LSE), i.e. sets M_t, H_t and L_t.

[2] We use M_t to denote the set of unclassified points in LSE, while in BO it denotes the set of potential maximizers.

Algorithm 5 Truncated Variance Reduction (TRUVAR) for Level-Set Estimation [BSKC16]

Input: Domain D, GP prior (μ_0, σ_0, k), confidence bound parameters $\overline{\delta} > 0$, $r \in (0,1)$, $\{\beta_{(i)}\}_{i\geq 1}$, $\eta_{(1)} > 0$ and level-set threshold h

1: Initialize the epoch number $i = 1$, target confidence $\eta_{(1)}$, unclassified points $M_{(0)} = D$, $H_{(0)} = \emptyset$, $L_{(0)} = \emptyset$

2: **for** $t = 1, 2, \ldots$ **do**

3: Choose

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}\in D} \frac{\sum_{\overline{\boldsymbol{x}}\in M_{t-1}} \max\{\beta_{(i)}\sigma^2_{t-1}(\overline{\boldsymbol{x}}), \eta_{(i)}\} - \sum_{\overline{\boldsymbol{x}}\in M_{t-1}} \max\{\beta_{(i)}\sigma^2_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}}), \eta_{(i)}\}}{c(\boldsymbol{x})}.$$

4: Observe the noisy function sample y_t, and update according to Algorithm 6 to obtain μ_t, σ_t, lcb_t and ucb_t, as well as M_t,H_t and L_t

5: **if** $\max_{\boldsymbol{x}\in M_t} \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}) \leq (1+\overline{\delta})\eta_{(i)}$ **then**

6: Increment i, set $\eta_{(i)} = r \times \eta_{(i-1)}$.

Algorithm 6 LSE Parameter Updates for TRUVAR [BSKC16]

Input: Selected points and observations $\{\boldsymbol{x}_{t'}\}_{t'=1}^{t}$; $\{y_{t'}\}_{t'=1}^{t}$, previous sets M_{t-1}, H_{t-1}, L_{t-1}, parameter $\beta_{(i)}^{1/2}$, and level-set threshold h.

1: Update μ_t and σ_t according to (3.1)–(3.2), and form the upper and lower confidence bounds

$$\mathrm{ucb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) + \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}), \quad \mathrm{lcb}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) - \beta_{(i)}^{1/2}\sigma_t(\boldsymbol{x}).$$

2: Set

$$M_t = \big\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) \geq h \text{ and } \mathrm{lcb}_t(\boldsymbol{x}) \leq h\big\},$$

$$H_t = H_{t-1} \cup \big\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{lcb}_t(\boldsymbol{x}) > h\big\}, \quad L_t = L_{t-1} \cup \big\{\boldsymbol{x} \in M_{t-1} \,:\, \mathrm{ucb}_t(\boldsymbol{x}) < h\big\}.$$

3.3 Unified Approach to BO and LSE

In order to state the results for BO and LSE in a unified fashion, we define a notion of *ϵ-accuracy* for the two settings. That is, we define this term differently in the two scenarios, but then we provide theorems that simultaneously apply to both.

Definition 3.3.1. *After time step t of* TRUVAR*, we use the following terminology:*

- *For BO, the set M_t is* ϵ-accurate *if it contains all true maxima $\boldsymbol{x}^* \in \arg\max_{\boldsymbol{x}} f(\boldsymbol{x})$, and all of its points satisfy $f(\boldsymbol{x}^*) - f(\boldsymbol{x}) \leq \epsilon$.*
- *For LSE, the triplet (M_t, H_t, L_t) is* ϵ-accurate *if all points in H_t satisfy $f(\boldsymbol{x}) > h$, all points in L_t satisfy $f(\boldsymbol{x}) < h$, and all points in M_t satisfy $|f(\boldsymbol{x}) - h| \leq \frac{\epsilon}{2}$.*

In both cases, the cumulative cost *after time t is defined as $C_t = \sum_{t'=1}^{t} c(\boldsymbol{x}_{t'})$.*

We use $\frac{\epsilon}{2}$ in the LSE setting instead of ϵ since this creates a region of size ϵ where the function value lies, which is consistent with the BO setting.

3.3.1 General Result

Preliminaries

Suppose that the $\{\beta_{(i)}\}$ are chosen to ensure valid confidence bounds, i.e., $\mathrm{lcb}_t(\boldsymbol{x}) \leq f(\boldsymbol{x}) \leq \mathrm{ucb}_t(\boldsymbol{x})$ with high probability; see Theorem 3.3.1 below and its proof for such choices. In this case, we have after the i-th epoch of the algorithm that all points are either already discarded (BO) or classified (LSE), or are known to within some confidence $(1+\overline{\delta})\eta_{(i)}$. For the points with such confidence, we have $\mathrm{ucb}_t(\boldsymbol{x}) - \mathrm{lcb}_t(\boldsymbol{x}) \leq 2(1+\overline{\delta})\eta_{(i)}$, and hence

$$\mathrm{ucb}_t(\boldsymbol{x}) \leq \mathrm{lcb}_t(\boldsymbol{x}) + 2(1+\overline{\delta})\eta_{(i)} \leq f(\boldsymbol{x}) + 2(1+\overline{\delta})\eta_{(i)}, \tag{3.9}$$

and similarly $\mathrm{lcb}_t(\boldsymbol{x}) \geq f(\boldsymbol{x}) - 2(1+\overline{\delta})\eta_{(i)}$. This means that all points other than those within a gap of width $4(1+\overline{\delta})\eta_{(i)}$ must have been discarded or classified:

$$M_t \subseteq \left\{\boldsymbol{x} \,:\, f(\boldsymbol{x}) \geq f^* - 4(1+\overline{\delta})\eta_{(i)}\right\} =: \overline{M}_{(i)} \quad \text{(BO)} \tag{3.10}$$

$$M_t \subseteq \left\{\boldsymbol{x} \,:\, |f(\boldsymbol{x}) - h| \leq 2(1+\overline{\delta})\eta_{(i)}\right\} =: \overline{M}_{(i)} \quad \text{(LSE)} \tag{3.11}$$

Note that this is true regardless of the actual observations.

For a collection of points S, possibly containing duplicates, we write $c(S) = \sum_{\boldsymbol{x}\in S} c(\boldsymbol{x})$. Moreover, we denote the posterior variance upon observing the points up to time $t-1$ *and* the additional points in S by $\sigma_{t-1|S}(\overline{\boldsymbol{x}})$. Therefore, $c(\boldsymbol{x}) = c(\{\boldsymbol{x}\})$ and $\sigma_{t-1|\boldsymbol{x}}(\overline{\boldsymbol{x}}) = \sigma_{t-1|\{\boldsymbol{x}\}}(\overline{\boldsymbol{x}})$. The minimum cost (respectively, maximum cost) is denoted by $c_{\min} = \min_{\boldsymbol{x}\in D} c(\boldsymbol{x})$ (respectively, $c_{\max} = \max_{\boldsymbol{x}\in D} c(\boldsymbol{x})$).

Finally, we introduce the quantity

$$C^*(\xi, M) = \min\left\{c(S) \,:\, \max_{\overline{\boldsymbol{x}}\in M} \sigma_{0|S}(\overline{\boldsymbol{x}}) \leq \xi\right\}, \tag{3.12}$$

representing the minimum cost to achieve a posterior standard deviation of at most ξ within M.

Statement of Theorem

In all of our results, we make the following assumption.

Assumption 3.3.1. *The kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ is such that the variance reduction function*

$$\psi_{t,\boldsymbol{x}}(S) = \sigma_t^2(\boldsymbol{x}) - \sigma_{t|S}^2(\boldsymbol{x}) \tag{3.13}$$

is submodular (see (2.17)*) for any time t, and any points $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$ and $\boldsymbol{x}$.*

We refer the reader to Section 2.3 for a brief introduction on submodular functions. This assumption has been used in several previous works based on GPs, and sufficient conditions for its validity can be found in [DK08, Sec. 8]. We now state the following general guarantee.

Theorem 3.3.1. *Fix $\epsilon > 0$ and $\delta \in (0, 1)$, and suppose there exist values $\{C_{(i)}\}$ and $\{\beta_{(i)}\}$ s.t.*

$$C_{(i)} \geq C^*\left(\frac{\eta_{(i)}}{\beta_{(i)}^{1/2}}, \overline{M}_{(i-1)}\right) \log \frac{|\overline{M}_{(i-1)}|\beta_{(i)}}{\bar{\delta}^2 \eta_{(i)}^2} + c_{\max}, \text{ and} \tag{3.14}$$

and

$$\beta_{(i)} \geq 2 \log \frac{|D|\big(\sum_{i' \leq i} C_{(i')}\big)^2 \pi^2}{6\delta {c_{\min}}^2}. \tag{3.15}$$

Then if TRUVAR *is run with these choices of $\beta_{(i)}$ until the cumulative cost reaches*

$$C_\epsilon = \sum_{i\,:\,4(1+\bar{\delta})\eta_{(i-1)} > \epsilon} C_{(i)}, \tag{3.16}$$

then with probability at least $1 - \delta$, we have ϵ-accuracy.

While this theorem is somewhat abstract, it captures the fact that the algorithm improves when points having a lower cost and/or lower noise are available, since both of these lead to a smaller value of $C^*(\xi, M)$; the former by directly incurring a smaller cost, and the latter by shrinking the variance more rapidly. Next, we provide a proof of Theorem 3.3.1 that is based on the connection between TRUVAR rule and budgeted submodular covering problem (details on the standard version of this problem are given in Section 2.3). In the subsequent sections, we specialize this result to some important cases.

3.3.2 Proof of General Result

We begin by restating Lemma 2.2.1.

Lemma 2.2.1 ([SKKS10])**.** *For each t, define $\beta_t = 2 \log \frac{|D| t^2 \pi^2}{6\delta}$. With probability at least $1 - \delta$, we have for all $\boldsymbol{x}$ and t that $|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \beta_t^{1/2} \sigma_{t-1}(\boldsymbol{x})$.*

We conclude that in order for $\mu_t(\cdot) \pm \beta_{(i)}^{1/2} \sigma_t(\cdot)$ to provide valid confidence bounds, it suffices to ensure that $\beta_{(i)} \geq \beta_t$ for all t in epoch i. From (3.15), we see that this is true provided that the time taken to reach the end of the i-th epoch is at most $\frac{1}{c_{\min}} \sum_{i' \leq i} C_{(i')}$. Since $c_{\min}$ is the minimum pointwise cost, this holds provided that the cost incurred in epoch i is at most $C_{(i)}$. We therefore let $\beta_{(i)}$ be chosen accordingly (*cf.*, (3.15)), leaving us only to bound $C_{(i)}$ itself.

We connect TRUVAR with the following budgeted submodular covering problem:

$$\text{minimize}_S \;\; c(S) \quad \text{subject to} \;\; g_t(S) = g_{t,\max}. \tag{3.17}$$

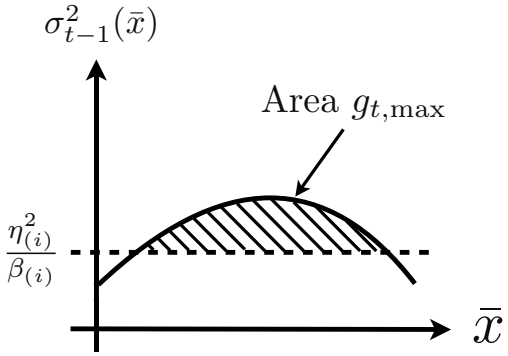


Figure 3.2: Illustration of the excess variance $g_{t,\max}$.

Here,

$$g_t(S) = \sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\left\{\sigma^2_{t-1}(\overline{\boldsymbol{x}}), \frac{\eta^2_{(i)}}{\beta_{(i)}}\right\} - \sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\left\{\sigma^2_{t-1|S}(\overline{\boldsymbol{x}}), \frac{\eta^2_{(i)}}{\beta_{(i)}}\right\}, \tag{3.18}$$

and $g_{t,\max}$ is the highest possible value[3] of $g_t(S)$ over arbitrarily large sets S, i.e., it is the value obtained once all of the summands in the second summation in (3.18) have saturated to $\frac{\eta^2_{(i)}}{\beta_{(i)}}$:

$$g_{t,\max} = \sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \left(\max\left\{\sigma^2_{t-1}(\overline{\boldsymbol{x}}), \frac{\eta^2_{(i)}}{\beta_{(i)}}\right\} - \frac{\eta^2_{(i)}}{\beta_{(i)}} \right) \tag{3.19}$$

$$= \sum_{\overline{\boldsymbol{x}} \in M_{t-1}} \max\left\{0, \sigma^2_{t-1}(\overline{\boldsymbol{x}}) - \frac{\eta^2_{(i)}}{\beta_{(i)}}\right\}. \tag{3.20}$$

We thus refer to $g_{t,\max}$ as the *excess variance*; see Figure 3.2 for an illustration. Note that each time instant t corresponds to a different function $g_t(S)$, and we are considering sets S of an arbitrary size even though our algorithm only chooses one point at each time instant. By our assumption on the submodularity of the variance reduction function, and the fact that taking the minimum with a constant[4] preserves submodularity [KG12], $g_t(S)$ is also submodular. It is also easily seen to be monotonically increasing, and normalized in the sense that $g_t(\emptyset) = 0$.

Our selection rule (3.5) at time t can now be interpreted as the first step in a greedy algorithm for solving the budgeted submodular optimization problem (3.17). To obtain performance guarantees, we make use of Lemma 2 of [KG05b] specialized to the special case when $|S| = 1$.

[3]Recall that S may contain duplicates, and these are counted multiple times accordingly in the definitions of both $c(S)$ and $g_t(S)$. More formally, all of our equations can be cast in terms of standard sets (without duplicates) by expanding D to $D \times \{1, \cdots, N\}$ for any integer N that is larger than the maximum number of points that can be chosen throughout the course of the algorithm.

[4]The minimum becomes a maximum after negation.

This result reads as follows in our own notation:

$$g_t(\{\boldsymbol{x}_t\}) \geq \frac{c(\boldsymbol{x}_t)}{c(S_t^*)} g_{t,\max}, \tag{3.21}$$

where S_t^* is an optimal solution to (3.17), and hence $g_t(S^*) = g_{t,\max}$. Here $\boldsymbol{x}_t$ is the point chosen greedily by our algorithm.

We now consider the behavior of the excess variance $g_{t,\max}$ in a single epoch, i.e., the duration of a single value of i (and hence $\eta_{(i)}$ and $\beta_{(i)}$) in the algorithm. We claim that for t and $t+1$ in the same epoch, we have

$$g_{t+1,\max} \leq g_{t,\max} - g_t(\{x_t\}). \tag{3.22}$$

To see this, we note from (3.18)–(3.19) that this would hold with equality if we were to have $M_t = M_{t-1}$, since by definition we have $\sigma_t^2(\overline{\boldsymbol{x}}) = \sigma_{t-1|\{\boldsymbol{x}_t\}}^2(\overline{\boldsymbol{x}})$. We therefore obtain (3.22) by recalling that the set M_{t-1} is decreasing with respect to inclusion, and noting from (3.20) that any given $g_{t,\max}$ can only decrease when M_{t-1} is smaller.

Combining (3.21)–(3.22) gives

$$g_{t+1,\max} \leq \left(1 - \frac{c(\boldsymbol{x}_t)}{c(S_t^*)}\right) g_{t,\max}. \tag{3.23}$$

Hence, by recursing and noting that $c(S_{t+1}^*) \leq c(S_t^*)$ due to the monotonicity of σ_t^2 in t, we obtain for t and $t+\ell$ in the same epoch that

$$\frac{g_{t+\ell,\max}}{g_{t,\max}} \leq \prod_{t'=t+1}^{t+\ell} \left(1 - \frac{c(\boldsymbol{x}_{t'})}{c(S_t^*)}\right) \tag{3.24}$$

$$\leq \exp\left(-\frac{\sum_{t'=t+1}^{t+\ell} c(\boldsymbol{x}_{t'})}{c(S_t^*)}\right), \tag{3.25}$$

where we have applied the inequality $1 - \alpha \leq e^{-\alpha}$. Moreover, the total cost incurred by choosing these points is precisely $\sum_{t=t_0+1}^{t_0+\ell} c(x_t)$. Thus, letting t_0 be the first time index in the epoch, we find that in order to remove all but a proportion γ of the initial excess variance $g_{t_0,\max}$, it suffices that the cost $C_{(i)}$ incurred in the epoch satisfies

$$C_{(i)} \geq c(S_{t_0}^*) \log \frac{1}{\gamma}. \tag{3.26}$$

Next, we observe that since the posterior Gaussian process variance is upper bounded by one due to the assumption $k(\boldsymbol{x}, \boldsymbol{x}) = 1$, the initial excess variance $g_{t_0,\max}$ is upper bounded by $g_{t_0,\max} \leq |M_{t_0}|$, the size of the set of potential maximizers at the *start* of the epoch. Recalling that the following holds $\overline{M}_{(i-1)} \supseteq M_{t_0}$ (see (3.10)–(3.11)), we also have $g_{t_0,\max} \leq |\overline{M}_{(i-1)}|$.

It follows that if we choose

$$\gamma = \frac{\overline{\delta}^2 \eta_{(i)}^2}{\beta_{(i)} |\overline{M}_{(i-1)}|}, \tag{3.27}$$

then removing all but a proportion γ of $g_{t_0,\max}$ also implies removing all but $\overline{\delta}^2 \frac{\eta_{(i)}^2}{\beta_{(i)}}$ of it. In other words, if at time t we have incurred a cost in the epoch satisfying (3.26) with γ as in (3.27), then we must have $g_{t,\max} \leq \overline{\delta}^2 \frac{\eta_{(i)}^2}{\beta_{(i)}}$.

Removing all of the excess variance would imply $\eta_{(i)}$-confidence at all points in M_t. In the worst case, the remaining excess variance $\overline{\delta}^2 \frac{\eta_{(i)}^2}{\beta_{(i)}}$ is concentrated entirely on a single point, in which case its confidence is upper bounded by $\sqrt{1+\overline{\delta}^2}\eta_{(i)}$, which is further upper bounded by $(1+\overline{\delta})\eta_{(i)}$ due to the identity $\sqrt{1+\alpha^2} \leq 1+\alpha$.

Combining these observations, we have that in the i-th epoch, upon incurring a cost of at least

$$C_{(i)} \geq c(S_{t_0}^*) \log \frac{|\overline{M}_{(i-1)}| \beta_{(i)}}{\overline{\delta}^2 \eta_{(i)}^2}, \tag{3.28}$$

we are guaranteed to have $(1+\overline{\delta})\eta_{(i)}$-confidence for all points in M_t. However, having such confidence is precisely the condition used in the algorithm to move onto the next epoch, and we conclude that the epoch must end by (or sooner than) the time that (3.28) holds. By definition, $c(S_{t_0}^*)$ is the smallest possible cost to uniformly shrink the posterior standard deviation within $M_{t_0} \subseteq \overline{M}_{(i-1)}$ down to $\frac{\eta_{(i)}^2}{\beta_{(i)}}$, thus coinciding with the definition in (3.12). Therefore, we can weaken (3.29) to

$$C_{(i)} \geq C^*\left(\frac{\eta_{(i)}}{\beta_{(i)}^{1/2}}, \overline{M}_{(i-1)}\right) \log \frac{|\overline{M}_{(i-1)}| \beta_{(i)}}{\overline{\delta} \eta_{(i)}^2}. \tag{3.29}$$

We are now in a position to check the conditions for ϵ-accuracy in Def. 3.3.1. In the case of BO, summing (3.29) over all of the epochs such that $4(1+\overline{\delta})\eta_{(i-1)} > \epsilon$ yields (3.16); recall from (3.10) that after any epoch i such that $4(1+\overline{\delta})\eta_{(i)} \leq \epsilon$, all points are at most ϵ-suboptimal. We also note that all true maxima must remain in M_t, due to the fact that we have chosen $\beta_{(i)}$ to ensure valid confidence bounds w.h.p., and we only ever discard points that are deemed suboptimal according to those bounds. For LSE, a similar conclusion follows from (3.11) by summing over all epochs such that $2(1+\overline{\delta})\eta_{(i-1)} > \frac{\epsilon}{2}$, which is exactly the same as $4(1+\overline{\delta})\eta_{(i-1)} > \epsilon$. Once again, all points in H_t and L_t are correct due to the validity of our confidence bounds. In both cases, we add an additional term $c_{\max}$ to the right-hand side of (3.29) to account for the fact that once the cost exceeds the right-hand side, it could exceed it by any amount up to $c_{\max}$.

3.4 Homoscedastic Noise and Unit-Cost Setting

In this section, we specialize our result (Theorem 3.3.1) to the homoscedastic and unit-cost setting, i.e. $\sigma^2(\boldsymbol{x}) = \sigma^2$ and $c(\boldsymbol{x}) = 1$. Recall the maximum mutual information definition from (2.10),

$$\gamma_T = \max_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_T} \frac{1}{2} \log\det\left(\boldsymbol{I}_T + \sigma^{-2}\boldsymbol{K}_T\right).$$

In Appendix 3.A.1, we provide a theorem with a condition for ϵ-accuracy of the form

$$T \geq O^*\left(\frac{C_1\gamma_T\beta_T}{\epsilon^2} + 1\right), \tag{3.30}$$

with $C_1 = \frac{1}{\log(1+\sigma^{-2})}$, thus matching existing bounds [GCHK13, CBRV13] up to logarithmic factors. In the following, we present a refined version that has a significantly better dependence on the noise level, thus exemplifying that a more careful analysis of our result in (3.14) can provide improvements over the standard bounding techniques.

Corollary 3.4.1. *Fix $\epsilon > 0$ and $\delta \in (0,1)$, define $\beta_T = 2\log\frac{|D|T^2\pi^2}{6\delta}$, and set $\eta_{(1)} = 1$ and $r = \frac{1}{2}$. There exist choices of $\beta_{(i)}$ (not depending on the time horizon T) such that we have ϵ-accuracy with probability at least $1-\delta$ once the following condition holds:*

$$T \geq \left(2\sigma^2\gamma_T\beta_T\frac{96(1+\overline{\delta})^2}{\epsilon^2} + C_1\gamma_T\beta_T\frac{6}{\sigma^2} + 2\left\lceil \log_2\frac{32(1+\overline{\delta})}{\epsilon\sigma}\right\rceil\right)\log\frac{16(1+\overline{\delta})^2|D|\beta_T}{\overline{\delta}^2\epsilon^2}, \tag{3.31}$$

where $C_1 = \frac{1}{\log(1+\sigma^{-2})}$. Hence, it suffices that

$$T \geq O^*\big(\frac{\sigma^2\gamma_T\beta_T}{\epsilon^2} + \frac{C_1\gamma_T\beta_T}{\sigma^2} + 1\big). \tag{3.32}$$

The choices $\eta_{(1)} = 1$ and $r = \frac{1}{2}$ are made for mathematical convenience, and a similar result follows for any other choices $\eta_{(1)} > 0$, $r \in (0,1)$, possibly with different constant factors.

As $\sigma^2 \to \infty$ (i.e., high noise), both of the above-mentioned bounds have noise dependence $O^*(\sigma^2)$, since $\log(1+\alpha^{-2}) = O(\alpha^{-2})$ as $\alpha \to \infty$. On the other hand, as $\sigma^2 \to 0$ (i.e., low noise), C_1 is logarithmic, and Corollary 3.4.1 is significantly better provided that $\epsilon \ll \sigma$.

3.5 Multi-fidelity Setting

In this section, we consider the setting that there is a domain of points D_0 that the reward function depends on, and alongside each point we can also *choose* a noise variance $\sigma^2(k)$ ($k = 1,\dots,K$) of the observation when sampling. Hence, $D = D_0 \times \{1,\cdots,K\}$. Lower noise variances incur a higher cost according to a cost function $c(k)$. The described setting corresponds to the scenario where we need to pay a higher cost to get a more precise (less noisy) observation.

Next, we present our main result in this setting:

Corollary 3.5.1. *For each $k = 1, \cdots, K$, let $T^*(k)$ denote the smallest value of T such that* (3.31) *holds with $\sigma^2(k)$ in place of σ^2, and with $\beta_T = 2\log\frac{|D|T^2{c_{\max}}^2\pi^2}{6\delta{c_{\min}}^2}$. Then, under the preceding setting, there exist choices of $\beta_{(i)}$ (not depending on T) such that we have ϵ-accuracy with probability at least $1-\delta$ once the cumulative cost reaches $\min_k c(k)T^*(k)$.*

This result roughly states that we obtain a bound as good as that obtained by sticking to *any fixed choice of noise level*. In other words, every choice of noise (and corresponding cost) corresponds to a different version of a BO or LSE algorithm (e.g., [GCHK13, CBRV13]), and our algorithm has a similar performance guarantee to the best among all of those. This is potentially useful in avoiding the need for running an algorithm once per noise level and *then* choosing the best-performing one. Moreover, we found numerically that beyond matching the best fixed noise strategy, we can *strictly improve* over it by mixing the noise levels; see Section 3.7.

3.6 Comparisons to Lower Bounds

In this section, we first discuss the extension of the result obtained in Section 3.4 (unit cost and homoscedastic setting) to the non-Bayesian setting (see Section 2.2.1), and then we compare the obtained bounds with the lower bounds from [SBC17].

In the previous sections, our focus was on the Bayesian setting, where f was considered to be random and distributed according to a Gaussian process with the specified kernel. In this section, we assume that the input space D is endowed with the positive definite kernel function $k(\cdot,\cdot)$ that is normalized to satisfy $k(\boldsymbol{x},\boldsymbol{x}) = 1$ for all $\boldsymbol{x} \in D$. The unknown function f is an arbitrary fixed function that has a bounded norm in the corresponding Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_k(D)$ induced by the kernel function, i.e. $f \in \mathcal{F}_k(B)$, where

$$\mathcal{F}_k(B) = \{f \in \mathcal{H}_k(D) \,:\, \|f\|_k \leq B\}.$$

The key to adapting TRUVAR to this setting is the following confidence lemma from [CG17]. It allows us to run TRUVAR with the GP prior $\mathrm{GP}(0,k)$ which is misspecified in this case.

Lemma 3.6.1 ([CG17])**.** *Fix $f \in \mathcal{F}_k(B)$, and consider the sampling model $y_t = f(\boldsymbol{x}_t) + z_t$ with $z_t \sim \mathcal{N}(0,\sigma^2)$ with independence between times. Under the choice*

$$\beta_t = \left(B + \sigma\sqrt{2(\gamma_{t-1} + \log\frac{e}{\xi})}\right)^2,$$

the following holds with probability at least $1-\xi$:

$$\mathrm{lcb}_{t-1}(\boldsymbol{x}) \leq f(\boldsymbol{x}) \leq \mathrm{ucb}_{t-1}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in D, \forall t \geq 1. \tag{3.33}$$

	Upper bound (this work)	Conj. upper bound (see Section 3.6)	Lower bound ([SBC17])
SE kernel Time to simple regret ϵ	$O^*\left(\frac{1}{\epsilon^2}\left(\log\frac{1}{\epsilon}\right)^{2p}\right)$	$O^*\left(\frac{1}{\epsilon^2}\left(\log\frac{1}{\epsilon}\right)^{p}\right)$	$\Omega\left(\frac{1}{\epsilon^2}\left(\log\frac{1}{\epsilon}\right)^{\frac{p}{2}}\right)$
Matérn kernel Time to simple regret ϵ	$O^*\left(\left(\frac{1}{\epsilon}\right)^{\frac{2(2\nu+p(p+1))}{2\nu-p(p+1)}}\right)$ (if $2\nu - p(p+1) > 0$)	$O^*\left(\left(\frac{1}{\epsilon}\right)^{2+\frac{p(p+1)}{\nu}}\right)$	$\Omega\left(\left(\frac{1}{\epsilon}\right)^{2+\frac{p}{\nu}}\right)$

Table 3.1: Summary of simple regret bounds for a fixed RKHS norm bound B and noise level σ^2.

Upon suitably changing the choice of $\beta_{(i)}$ according to the previous lemma in our algorithm, the validity of the confidence bounds is ensured, and the rest of the algorithm remains unchanged. We also need the following condition to hold for every epoch i (*cf.* Theorem 3.3.1)

$$\beta_{(i)} \geq \left(B + \sigma\sqrt{2(\gamma_{\bar{T}_{(i)}} + \log\frac{e}{\xi})}\right)^2, \tag{3.34}$$

where $\bar{T}_{(i)} = \sum_{i' \leq i} T_{(i)}$. Accordingly, in place of $\beta_T = 2\log\frac{|D|T^2\pi^2}{6\delta}$ (*cf.*, Corollary 3.4.1) we have

$$\beta_T = \left(B + \sigma\sqrt{2(\gamma_T + \log\frac{e}{\xi})}\right)^2. \tag{3.35}$$

The regret bound obtained by our algorithm in the non-Bayesian setting remains of the same form as the one obtained in the Bayesian setting in Section 3.4, i.e. time to ϵ-regret is $T \geq O^*\left(\frac{C_1\gamma_T\beta_T}{\epsilon^2} + 1\right)$, where β_T is now given in (3.35).

Our algorithm also extends easily to compact domains such as $[0,1]^p$. The main challenge is that the summations in (3.5) become integrals that need to be approximated numerically. The simplest way of doing this is to approximate the integrals by summations over a finite number of *representer points*, e.g., a grid of values that cover the domain sufficiently densely.

Hence, the time to ϵ simple regret for our algorithm in the non-Bayesian setting when $D = [0,1]^p$ (under assumption that we can exactly compute the acquisition rule) remains

$$T \geq O^*\left(\frac{C_1\gamma_T\beta_T}{\epsilon^2} + 1\right). \tag{3.36}$$

We proceed by considering the case that σ and B behave as $\Theta(1)$. For a fixed σ, and by substituting β_T with (3.35) in (3.36), we arrive at the condition for ϵ-optimality

$$\frac{T}{B\gamma_T + \gamma_T^2} \geq \frac{C}{\epsilon^2}, \tag{3.37}$$

where $C = O^*(1)$. By further substituting $\gamma_T = O((\log T)^{p+1})$ from (2.11) for the SE kernel and rearranging, the condition becomes of the form

$$T \geq C' \epsilon^{-2} \left(\log \frac{1}{\epsilon} \right)^{2p},$$

for some $C' = O^*(1)$. In comparison to the lower bound obtained in [SBC17], the two bounds coincide up to the factor of 2 vs. $\frac{1}{2}$ in the exponent (see Table 3.1).

For the Matérn kernel, by substituting

$$\gamma_T = O^* \left(T^{\frac{p(p+1)}{2\nu + p(p+1)}} \right)$$

from (2.12) in (3.37) and rearranging, the condition for ϵ- simple regret becomes

$$T \geq C'' \left(\frac{1}{\epsilon} \right)^{\frac{2(2\nu + p(p+1))}{2\nu - p(p+1)}},$$

where $C'' \in O^*(1)$. Moreover, for having a non-void condition on the simple regret we require $2\nu - p(p+1) > 0$.

The gap between the upper and lower bound obtained for the Matérn kernel is more significant (*cf.* Table 3.1) than in the case of the SE kernel. If (3.37) could be improved to

$$\frac{T}{B\gamma_T} \geq \frac{C'''}{\epsilon^2}, \tag{3.38}$$

for some $C''' = O^*(1)$, then this would lead to a non-void condition on the simple regret for all p and ν. We conjecture that such a bound could hold, similarly to the one obtained in the Bayesian setting (see the scallings in Corollary 3.4.1). This is an interesting open problem for future work. Next, we briefly discuss the resulting bounds if this conjecture was true. These bounds are summarized in the middle column of Table 3.1.

In the case of the squared exponential kernel, we would obtain the bound that would partially close the gap mentioned above on the factor of 2 in the exponent. For the Matérn kernel, we would obtain

$$T \geq O^* \left(\left(\frac{1}{\epsilon} \right)^{2 + p(p+1)/\nu} \right),$$

and hence, our bound would match the lower bound up to the replacement of p by $p(p+1)$.

3.7 Experimental Evaluation

We evaluate our algorithm in both the level-set estimation and Bayesian optimization settings.

Parameter choices: As with previous GP-based algorithms that use confidence bounds, our

theoretical choice of $\beta_{(i)}$ in TRUVAR is typically overly conservative. Therefore, instead of using (3.15) directly, we use a more aggressive variant with similar dependence on the domain size and time: $\beta_{(i)} = a \log(|D| t_{(i)}^2)$, where $t_{(i)}$ is the time at which the epoch starts, and a is a constant. Instead of the choice $a = 2$ dictated by (3.15), we set $a = 0.5$ for BO to avoid over-exploration. We found exploration to be more beneficial for LSE, and hence set $a = 1$ for this setting. We found TRUVAR to be robust with respect to the choices of the remaining parameters, and simply set $\eta_{(1)} = 1$, $r = 0.1$, and $\overline{\delta} = 0$ in all experiments (even though our theory requires $\overline{\delta} > 0$).

3.7.1 Level-set Estimation Experiments

For the LSE experiments, we use a common classification rule in all algorithms, classifying the points according to the posterior mean as $\hat{H}_t = \{\boldsymbol{x} : \mu_t(\boldsymbol{x}) \geq h\}$ and $\hat{L}_t = \{\boldsymbol{x} : \mu_t(\boldsymbol{x}) < h\}$. The classification accuracy is measured by the F_1-score (i.e., the harmonic mean of precision and recall) with respect to the true super- and sub-level sets.

We compare TRUVAR against the GP-based LSE algorithm [GCHK13] (GCHK), as well as the state-of-the-art straddle (STR) heuristic [BS08] and maximum variance rule (VAR). A short descriptions of these methods is as follows:

- The level-set estimation (LSE) algorithm [GCHK13] evaluates, at each iteration, the point that is not yet classified with the largest ambiguity:

  $$\boldsymbol{x}_t = \operatorname*{arg\,max}_{\boldsymbol{x} \in M_{t-1}} \min\{\mathrm{ucb}_t(\boldsymbol{x}) - h, h - \mathrm{lcb}_t(\boldsymbol{x})\},$$

  where ucb_t and lcb_t are defined as in (3.3) with a parameter β_t replacing $\beta_{(i)}$. Here, similarly to our algorithm, M_{t-1} is the set of points that have not yet been classified as having a value above or below the threshold h. We follow the recommendation in [GCHK13] of setting $\beta_t^{1/2} = 3$.

- The straddle (STR) heuristic [BS08] chooses

  $$\boldsymbol{x}_t = \operatorname*{arg\,max}_{\boldsymbol{x} \in D} 1.96\sigma_{t-1}(\boldsymbol{x}) - |\mu_{t-1}(\boldsymbol{x}) - h|,$$

  favoring high-uncertainty points that are expected to have function values closer to h.

- The maximum variance rule (VAR) simply chooses

  $$\boldsymbol{x}_t = \operatorname*{arg\,max}_{\boldsymbol{x} \in D} \sigma_{t-1}(\boldsymbol{x}).$$

Lake data (unit cost): We begin with a data set from the domain of environmental monitoring of inland waters, consisting of 2024 in situ measurements of chlorophyll concentration within a vertical transect plane, collected by an autonomous surface vessel in Lake Zürich [HPG+12]. As

in [GCHK13], our goal is to detect regions of high concentration. We evaluate each algorithm on a 50×50 grid of points, with the corresponding values coming from the GP posterior that was derived using the original data (see Figure 3.3d). We use the Matérn-5/2 ARD kernel, setting its hyperparameters by maximizing the likelihood on the second (smaller) available dataset. The level-set threshold h is set to 1.5.

In Figure 3.3a, we show the performance of the algorithms averaged over 100 different runs; here the randomness is only with respect to the starting point, as we are in the noiseless setting. We observe that in this unit-cost case, TRUVAR performs similarly to GCHK and STR. All methods outperform VAR, which is good for global exploration but less suited to level-set estimation.

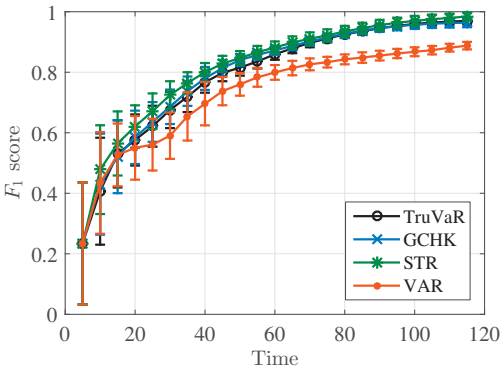
Lake data (non-unit cost): Next, we modify the above setting by introducing pointwise costs that are a function of the previous sampled point $\boldsymbol{x}'$, namely, $c_{\boldsymbol{x}'}(\boldsymbol{x}) = 0.25|x_1 - x_1'| + 4(|x_2| + 1)$, where x_1 is the vessel position and x_2 is the depth. Although we did not permit such a dependence on $\boldsymbol{x}'$ in our original setup, the algorithm itself remains unchanged. Our choice of cost penalizes the distance traveled $|x_1 - x_1'|$, as well as the depth of the measurement $|x_2|$. Since incorporating costs into existing algorithms is non-trivial, we only compare against the original version of the GCHK algorithm that ignores costs.

In Figure 3.3b , we see that TruVaR significantly outperform GCHK, achieving a higher F_1 score for a significantly smaller cost. The intuition behind this can be seen in Figures 3.3e and 3.3f, where we show the points sampled by TruVaR and GCHK in one run, connecting all pairs of consecutive points. GCHK is designed to pick few points, but since it ignores costs, the distance traveled is large. In contrast, by incorporating costs, TRUVAR travels small distances, often even staying in the same x_1 location to take measurements at multiple depths x_2.

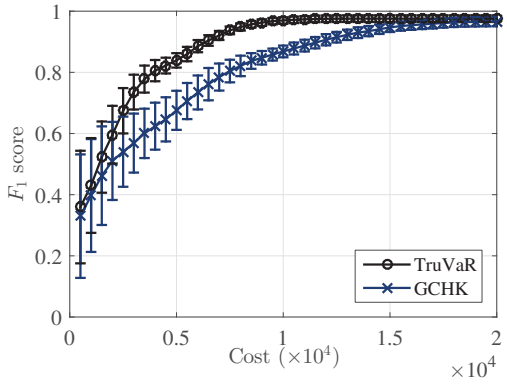
Synthetic data with multiple noise levels: In this experiment, we demonstrate Corollary 3.5.1 by considering the setting in which the algorithm can choose the sampling noise variance and incur the associated cost. We use a synthetic function sampled from a GP on a 50×50 grid with an isotropic squared exponential kernel having length scale $l = 0.1$ and unit variance, and set $h = 2.25$. Figure 3.4a plots the randomly-generated function that was used in this experiment. We use three different noise levels, $\sigma^2 \in \{10^{-6}, 10^{-3}, 0.05\}$, with corresponding costs $\{15, 10, 2\}$.

We run the GCHK algorithm separately for each of the three noise levels, while running TRUVAR as normal and allowing it to mix between the noise levels. The resulting F_1-scores are shown in Figure 3.3c. The best-performing version of GCHK changes throughout the time horizon, while TRUVAR is consistently better than all three.

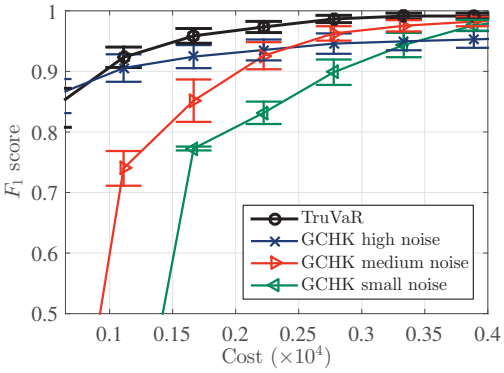
Figure 3.4b plots the average cost spent by the TRUVAR algorithm on each noise level by the end of the experiment. Again, we average the performance over 100 different experimental runs. We see that the cost is roughly equally distributed across the three levels. Specifically, we observed that TRUVAR initially chooses high noise levels in order to cheaply explore, and throughout the course of the experiments, it gradually switches to lower noise levels (despite



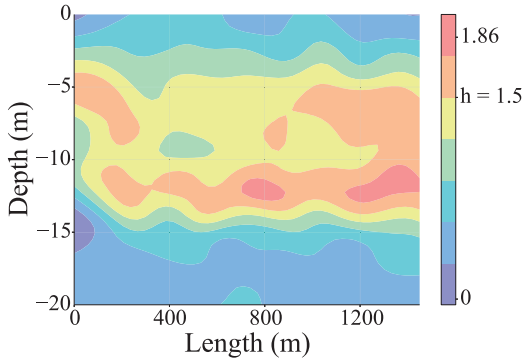
(a) Unit-cost, lake data



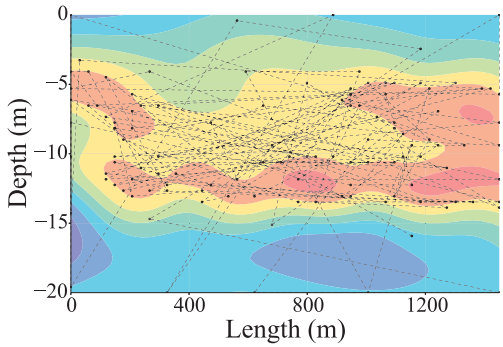
(b) Varying cost, lake data



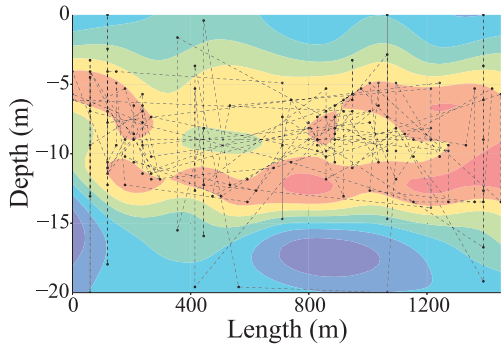
(c) Varying noise, synthetic data



(d) Inferred concentration function



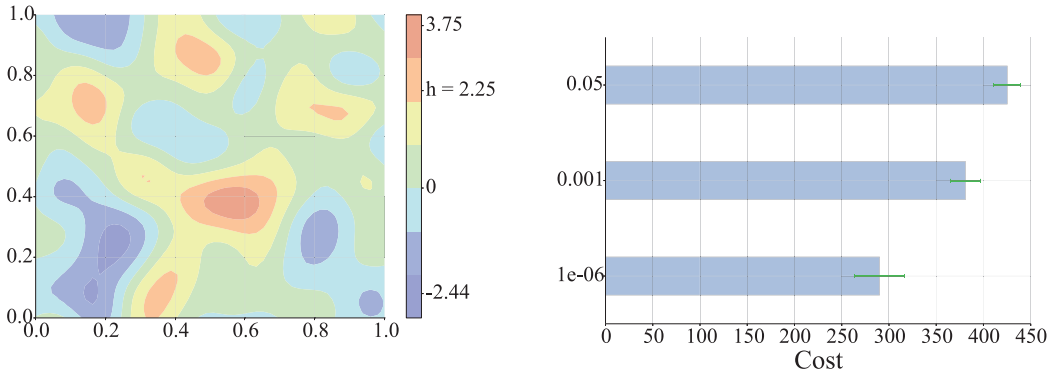
(e) Points chosen by LSE



(f) Points chosen by TruVaR

Figure 3.3: Experimental results for level-set estimation.

their high cost) in order to accurately determine the function values around the maximum. This is consistent with the behavior of the three version of GCHK, with $\sigma^2 = 10^{-6}$ performing well in the early stages, but $\sigma^2 = 0.05$ being preferable in the later stages.



(a) Threshold $h = 2.25$

(b) Cost paid per noise level

Figure 3.4: (a) Function used in synthetic level-set estimation experiments; (b) The amount of cost used by TRUVAR for each of the three noise levels..

3.7.2 Bayesian Optimization Experiments

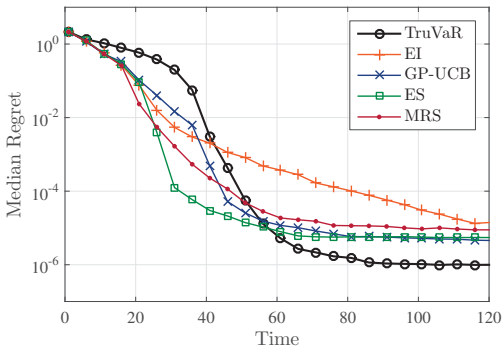
We now turn to the BO setting. We focus on the unit-cost case, since we already demonstrated the effects of costs in the LSE experiments.

Synthetic data: We first conduct a similar experiment as that in [HS12, Met16], generating 200 different test functions defined on $[0, 1]^2$. To generate a single test function, 200 points are chosen uniformly at random from $[0, 1]^2$, their function values are generated from a GP using an isotropic squared exponential kernel with length scale $l = 0.1$ and unit variance, and the resulting posterior mean forms the function on the whole domain $[0, 1]^2$. We subsequently assume that samples of this function are corrupted by Gaussian noise with $\sigma^2 = 10^{-6}$. For all algorithms considered, we evaluate the performance according to the regret of a single reported point, namely, the one having the highest posterior mean.

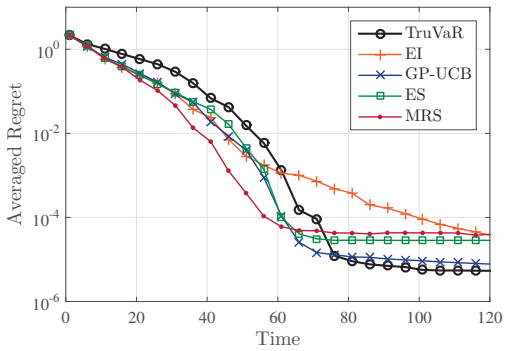
We compare the performance of TRUVAR against expected improvement (EI), GP-upper confidence bound (GP-UCB), entropy search (ES) (see Section 2.2 for the description of these methods). The exploration parameter β_t in GP-UCB is set according to the recommendation in [SKKS10] of dividing the theoretical value by five. In addition, we also compare against minimum regret search (MRS) whose acquisition functions is further outlined:

- The Minimum Regret Search (MRS) algorithm [Met16] also resembles ES, but works with the expected regret instead of the differential entropy. Monte Carlo techniques are used to average with respect to the posterior distribution and the measurements. The parameters for ES and MRS are set according to the recommendations given in [Met16, Section 5.1].

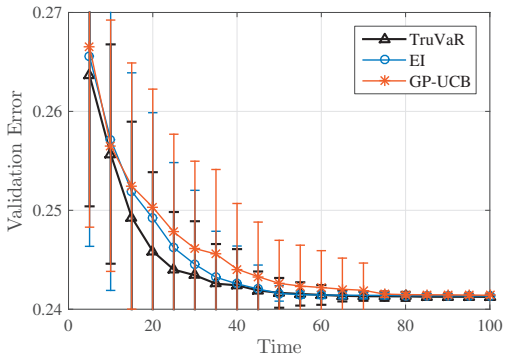
Figure 3.5a plots the median of the regret, and Figure 3.5b plots the mean after removing outliers (i.e., the best and worst 5% of the experimental runs). In the earlier rounds, both ES and MRS provide the best performance, while TRUVAR improves slowly due to the initial exploration.



(a) Synthetic, median



(b) Synthetic, outlier-adjusted mean



(c) SVM data

Figure 3.5: Experimental results for Bayesian optimization.

However, the regret of TRUVAR subsequently drops rapidly, giving the best performance in the later rounds after "zooming in" towards the maximum. GP-UCB generally performs well with the aggressive choice of β_t, despite previous works revealing it to perform poorly with the theoretical value.

Hyperparameter tuning data: We use the *SVM on grid* dataset, previously used in [SLA12]. A $24 \times 14 \times 4$ grid of hyperparameter configurations resulting in 1400 data points was pre-evaluated, forming the search space. The goal is to find a configuration with small validation error. We use a Matérn-5/2 ARD kernel, and re-learn its hyperparameters by maximizing the likelihood after sampling every 3 points. Since the hyperparameters are not fixed in advance, we replace M_{t-1} by D in (3.4) to avoid incorrectly ruling points out, allowing some removed points to be added again in later steps. Once the hyperparameters stop to vary significantly, the size of the set of potential maximizers decreases almost monotonically. Since we consider the noiseless setting here, we measure performance using the simple regret, i.e., the best point found so far.

Since the domain is discrete, we compare only against EI and GP-UCB. We again average over 100 random starting points, and plot the resulting validation error in Figure 3.5c. Even in this noiseless and unit-cost setting that EI and GP-UCB are suited to, we find that TRUVAR performs slightly better, giving a better validation error with smaller error bars.

3.7.3 Variations of the TRUVAR Algorithm

In this last section, we outline a couple of practical aspects and settings to which our algorithm TRUVAR can naturally be adapted. These include the following:

- **Efficiently Computing the Acquisition Function:** To compute the value of the acquisition function (3.5) for different $\boldsymbol{x} \in D$, we need to compute $\sigma^2_{t-1|\boldsymbol{x}}(M_{t-1}) \in \mathbb{R}^{|M_{t-1}|}$, i.e., the posterior variance of M_{t-1} upon observing $\boldsymbol{x}$ along with $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{t-1}$. Instead of computing it directly, it is computationally more efficient to compute $\sigma^2_{t-1|\boldsymbol{x}}(M_{t-1})$ as $\sigma^2_{t-1}(M_{t-1}) - \Delta\sigma^2_{t-1|\boldsymbol{x}}(M_{t-1})$. The only term that depends on $\boldsymbol{x}$ is $\Delta\sigma^2_{t-1|\boldsymbol{x}}(M_{t-1})$ and it can be computed, for a single $\boldsymbol{x} \in D$, as ([HS12]):

$$\Delta\sigma^2_{t-1|\boldsymbol{x}}(M_{t-1}) = \operatorname{diag}\big(\operatorname{Cov}_{t-1}(M_{t-1}, \boldsymbol{x})(\sigma^2 + \sigma^2_{t-1}(\boldsymbol{x}))^{-1}\operatorname{Cov}_{t-1}(\boldsymbol{x}, M_{t-1})\big), \quad (3.39)$$

  where

$$\sigma_{t-1}(\boldsymbol{x})^2 = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}_{t-1}(\boldsymbol{x})^T\big(\boldsymbol{K}_{t-1} + \boldsymbol{\Sigma}_{t-1}\big)^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}), \quad (3.40)$$

$$\operatorname{Cov}_{t-1}(M_{t-1}, \boldsymbol{x}) = \boldsymbol{k}(M_{t-1}, \boldsymbol{x}) - \boldsymbol{k}_{t-1}(M_{t-1})^T\big(\boldsymbol{K}_{t-1} + \boldsymbol{\Sigma}_{t-1}\big)^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x}). \quad (3.41)$$

  Here, $\boldsymbol{k}(M_{t-1}, \boldsymbol{x}) = \big[k(\bar{\boldsymbol{x}}_i, \boldsymbol{x})\big]_{i=1}^{|M_{t-1}|}$ and $\boldsymbol{k}_{t-1}(M_{t-1}) = \big[k(\bar{\boldsymbol{x}}_i, \boldsymbol{x}_j)\big]_{i,j=1}^{|M_{t-1}|,t-1}$ are in $\mathbb{R}^{|M_{t-1}|}$ and $\mathbb{R}^{|M_{t-1}|\times(t-1)}$, respectively. Given that we can precompute the Cholesky decomposition of the kernel matrix $\boldsymbol{K}_{t-1} + \boldsymbol{\Sigma}_{t-1}$, the term $\big(\boldsymbol{K}_{t-1} + \boldsymbol{\Sigma}_{t-1}\big)^{-1}\boldsymbol{k}_{t-1}(\boldsymbol{x})$ in both 3.40 and 3.41 can be computed in $O(t^2)$.

- **Non-monotonic M_t:** We have defined our sets M_t to become smaller on every time step. However, if $\beta_{(i)}$ is chosen aggressively, it may be preferable to replace M_{t-1} by D in (3.4)–(3.6), in which case some removed points may be added back in depending on how the posterior mean changes between steps. We take this approach in the real-world BO example of Section 3.7 in which the kernel hyperparameters are learned online, so as to avoid incorrectly ruling out points early on due to mismatched hyperparameters.

- **Avoiding computing the acquisition function everywhere:** We found that instead of computing the acquisition function at every point in D, limiting the selection to points in M_{t-1} does not harm the performance. To reduce the computation even further, one could adopt a strategy such as that proposed in [SSA14]: Take some relatively small number of points having the top GP-UCB or EI score, and then choose the point *in that restricted subset* having the highest score according to (3.5). In fact, the numerical results in Figure 3.5 suggest that this may not only reduce the computation, but also improve the performance in the very early rounds by making the algorithm *initially* behave more like GP-UCB or EI.

- **Pure variance reduction:** Setting $\eta_{(1)} = 0$ yields a *pure variance reduction* algorithm, which minimizes the total variance within M_t via a one-step lookahead. While our theory does not apply in this case, we found this choice to also work well in practice.

- **Implicit threshold for level-set estimation:** While we have focused on a threshold h for level-set estimation that is fixed in advance, one can easily incorporate the ideas of [GCHK13] to allow for an *implicit* threshold which is equal to some constant multiple of the function's maximum (which is random and unknown in advance).

- **Anticipating changes in M_t:** The acquisition function (3.5) computes the truncated variance reduction resulting from a one-step lookahead, but still sums over the previous set M_{t-1}. In order to make it more preferable to choose points that shrink M_t faster, it may be preferable to instead sum over $M_{t-1|\boldsymbol{x}}$, defined to be the updated set upon adding $\boldsymbol{x}$. The problem here is that such an update depend on the next posterior mean, whose update requires sampling f; however, one can get around this by replacing the observed value with the current mean when doing this one-step lookahead, and then using the true observed function sample only when $\boldsymbol{x}_t$ is actually chosen.

- **Batch Setting:** As we show in the proof of Theorem 3.3.1, our algorithm can be interpreted as performing the first step of a greedy submodular covering problem at each time step. This leads to a very natural extension to the batch setting, in which multiple points (say, k of them) are chosen at each time step: Simply perform the first k steps of the greedy covering algorithm during each batch.

3.A Proofs

3.A.1 Simplified Result for the Homoscedastic and Unit-Cost Setting

Since we are focusing on unit costs $c(\boldsymbol{x}) = 1$, the cost simply corresponds to the number of rounds T. To highlight this fact, we replace C^* in (3.12) by

$$T^*(\eta, M) = \min\Big\{|S| \;:\; \max_{\overline{\boldsymbol{x}} \in M} \sigma_{0|S}(\overline{\boldsymbol{x}}) \leq \xi\Big\}, \tag{3.42}$$

and similarly replace (3.14)–(3.16) by

$$T_{(i)} \geq T^*\left(\frac{\eta_{(i)}}{\beta_{(i)}^{1/2}}, \overline{M}_{(i-1)}\right) \log \frac{|\overline{M}_{(i-1)}|\beta_{(i)}}{\overline{\delta}^2 \eta_{(i)}^2} + 1 \tag{3.43}$$

$$\beta_{(i)} \geq 2 \log \frac{|\overline{M}_{(i-1)}|\big(\sum_{i' \leq i} T_{(i')}\big)^2 \pi^2}{6\delta} \tag{3.44}$$

$$T_\epsilon = \sum_{i \,:\, 4(1+\overline{\delta})\eta_{(i-1)} > \epsilon} T_{(i)}. \tag{3.45}$$

In this section, we prove the following as an application of Theorem 3.3.1.

Corollary 3.A.1. *Fix $\epsilon > 0$ and $\delta \in (0,1)$, define $\beta_T = 2\log \frac{|D|T^2\pi^2}{6\delta}$, and set $\eta_{(1)} = 1$ and $r = \frac{1}{2}$. There exist choices of $\beta_{(i)}$ (not depending on the time horizon T) such that we have ϵ-accuracy with probability at least $1-\delta$ once the following condition holds:*

$$T \geq \left(C_1 \gamma_T \beta_T \frac{96(1+\overline{\delta})^2}{\epsilon^2} + 2\left\lceil \log_2 \frac{8(1+\overline{\delta})}{\epsilon} \right\rceil\right) \log \frac{16(1+\overline{\delta})^2 |D| \beta_T}{\overline{\delta}^2 \epsilon^2} \tag{3.46}$$

$$= O^*\left(\frac{C_1 \gamma_T \beta_T}{\epsilon^2} + 1\right), \tag{3.47}$$

where $C_1 = \frac{1}{\log(1+\sigma^{-2})}$.

We bound the cardinality of S in (3.42) by considering a (suboptimal) procedure that greedily picks $\arg\max_{\boldsymbol{x} \in M} \sigma_t(\boldsymbol{x})$. We claim that after selecting k points according to this procedure to construct a set S_k, we have

$$\max_{\boldsymbol{x}} \sigma^2_{0|S_k}(\boldsymbol{x}) \leq C_1 \frac{\gamma_k}{k}, \tag{3.48}$$

where $C_1 = \frac{1}{\log(1+\sigma^{-2})}$. This is seen by writing

$$k \max_{\boldsymbol{x} \in M} \sigma^2_{0|S_k}(\boldsymbol{x}) = k\sigma^2_{0|S_k}(\boldsymbol{x}_k) \leq \sum_{j=1}^{k} \sigma^2_{0|S_j}(\boldsymbol{x}_j) \tag{3.49}$$

$$\leq \frac{1}{\log(1+\sigma^{-2})} \gamma_k. \tag{3.50}$$

We respectively used that $\boldsymbol{x}_k$ was chosen to maximize $\sigma_{0|S_k}$, that $\sigma_{0|S_i}(\boldsymbol{x}_i)$ always decreases as more points are chosen, and the bound on the sum of variances of the selected points of a GP from [SKKS10, Lemma 5.4].

Identifying k with T^*, and $\max_{\boldsymbol{x}\in M}\sigma_{0|S_k}(\boldsymbol{x})$ with $\xi = \frac{\eta}{\beta^{1/2}}$ (for some η and β to be specified), we obtain from (3.50) that

$$T^*\left(\frac{\eta}{\beta^{1/2}}, M\right) \leq \min\left\{T^* \,:\, T^* \geq \frac{C_1\gamma_{T^*}\beta}{\eta^2}\right\}. \tag{3.51}$$

Let $T^*_{(i)} := T^*\left(\frac{\eta_{(i)}}{\beta^{1/2}_{(i)}}, \overline{M}_{(i-1)}\right)$ be the value of T^* using the parameters $\eta = \eta_{(i)}$ and $\beta = \beta_{(i)}$ associated with epoch i. Letting $T = T_\epsilon$ denote the total time horizon, and using (3.45), we find that $\beta_{(i)}$ in (3.44) can be upper bounded by $2\log\frac{|D|T^2\pi^2}{6\delta}$, which is precisely β_T. By similarly using the monotonicity of γ_t, we obtain the condition

$$T^*_{(i)} \geq \frac{C_1\gamma_T\beta_T}{\eta^2_{(i)}} + 1, \tag{3.52}$$

where the addition of one is to account for possible rounding up to the nearest integer.

Substituting (3.52) into (3.43) gives the condition

$$T_{(i)} \geq \left(\frac{C_1\gamma_T\beta_T}{\eta^2_{(i)}} + 1\right)\log\frac{|\overline{M}_{(i-1)}|\beta_{(i)}}{\overline{\delta}^2\eta^2_{(i)}} + 1. \tag{3.53}$$

Since we are only considering values of i such that $4(1+\overline{\delta})\eta_{(i-1)} > \epsilon$, and since $\overline{M}_{(i-1)} \subseteq D$, we can upper bound the logarithm by $\log\frac{16(1+\overline{\delta})^2|D|\beta_{(i)}}{\overline{\delta}^2\epsilon^2} > 1$, and we can therefore weaken (3.53) to

$$T_{(i)} \geq \left(\frac{C_1\gamma_T\beta_T}{\eta^2_{(i)}} + 2\right)\log\frac{16(1+\overline{\delta})^2|D|\beta_{(i)}}{\overline{\delta}^2\epsilon^2}. \tag{3.54}$$

We also note that since $\eta_{(i)} = \eta_{(1)}r^{i-1}$, the condition $4(1+\overline{\delta})\eta_{(i-1)} > \epsilon$ is equivalent to

$$4(1+\overline{\delta})\eta_{(1)}r^{i-2} > \epsilon \tag{3.55}$$

$$\iff r^{i-2} > \frac{\epsilon}{4(1+\overline{\delta})\eta_{(1)}} \tag{3.56}$$

$$\iff i < 2 + \log_{1/r}\frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon} \tag{3.57}$$

$$\iff i \leq \left\lceil \log_{1/r}\frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon}\right\rceil + 1 \tag{3.58}$$

$$\iff i \leq \left\lceil \log_{1/r}\frac{4(1+\overline{\delta})\eta_{(1)}}{r\epsilon}\right\rceil, \tag{3.59}$$

where in the last line we used $\log_{1/r}\frac{1}{r} = 1$.

Summing (3.54) over all such i in accordance with (3.45), we obtain the condition

$$T \geq \left(C_1 \gamma_T \beta_T \sum_{i=1}^{\lceil \log_{1/r} \frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon} \rceil + 1} \frac{1}{\eta_{(i)}^2} + 2 \left\lceil \log_{1/r} \frac{4(1+\overline{\delta})\eta_{(1)}}{r\epsilon} \right\rceil \right) \log \frac{16(1+\overline{\delta})^2 |D| \beta_T}{\overline{\delta}^2 \epsilon^2}. \tag{3.60}$$

Finally, evaluating the summation gives

$$\begin{aligned}
\sum_{i=1}^{\lceil \log_{1/r} \frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon} \rceil + 1} \frac{1}{\eta_{(i)}^2} &= \sum_{i=1}^{\lceil \log_{1/r} \frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon} \rceil + 1} \frac{1}{\eta_{(1)}^2 r^{2(i-1)}} && (3.61) \\
&= \sum_{i=0}^{\lceil \log_{1/r} \frac{4(1+\overline{\delta})\eta_{(1)}}{\epsilon} \rceil} \frac{1}{\eta_{(1)}^2 r^{2i}} && (3.62) \\
&\leq \frac{1}{r^2(1-r^2)} \frac{16(1+\overline{\delta})^2}{\eta_{(1)}^2 \epsilon^2}, && (3.63)
\end{aligned}$$

where the last line follows from the identity $\sum_{i=0}^{\lceil \log_{1/r} A \rceil} \frac{1}{r^{2i}} \leq \frac{1}{r^2(1-r^2)} A^2$. Substituting $r = \frac{1}{2}$ and $\eta_{(1)} = 1$ concludes the proof; the former yields $\frac{1}{r^2(1-r^2)} = \frac{16}{3} \leq 6$.

3.A.2 Proof of Improved Noise Dependence (Corollary 3.4.1)

The bound in (3.50) is based on the inequality [SKKS10, Lemma 5.4]

$$\frac{\sigma_t^2}{\sigma^2} \leq C_1 \log \left(1 + \frac{\sigma_t^2}{\sigma^2} \right) \tag{3.64}$$

for $\sigma_t^2 \in [0, 1]$ (with $C_1 = \frac{\sigma^{-2}}{\log(1+\sigma^{-2})}$), which can be very loose when σ^2 is small.

Our starting point to improve the noise dependence is to note that the following holds under the more restrictive condition $\sigma_t^2 \leq \sigma^2$:

$$\begin{aligned}
\sigma_t^2 &= \sigma^2 \frac{\sigma_t^2}{\sigma^2} && (3.65) \\
&\leq 2\sigma^2 \log \left(1 + \frac{\sigma_t^2}{\sigma^2} \right), && (3.66)
\end{aligned}$$

where we have used the fact that $\alpha \leq 2\log(1+\alpha)$ for $\alpha \in [0, 1]$.

The idea that we pursue now is to use (3.66) in the epochs that are late enough to ensure that $\sigma_t^2 \leq \sigma^2$, and (3.50) in the earlier epochs. Since $(1+\overline{\delta})\eta_{(i)}$ represents the confidence level obtained after epoch i, and since $\beta_{(i)}^{1/2}\sigma_t$ represents the confidence level after time t, we find that in order to ensure $\sigma_t^2 \leq \sigma^2$ it suffices that the following condition holds $\frac{(1+\overline{\delta})^2 \eta_{(i)}^2}{\beta_{(i)}} \leq \sigma^2$.

Moreover, our choice of $\beta_{(i)}$ in (3.15) is always greater than one when $|D| \geq 2$ (which is a trivial condition), and hence we can simplify this condition to $\eta_{(i)}^2 \leq \sigma^2$, and write

$$\sum_i T_{(i)} \leq \sum_{i\,:\,\eta_{(i-1)}^2 > \sigma^2} T_{(i)}^{(C_1)} + \sum_i T_{(i)}^{(2\sigma^2)}, \tag{3.67}$$

where $T_{(i)}^{(C_1)}$ denotes bound on $T_{(i)}$ in (3.54) based on (3.50), and $T_{(i)}^{(C_1)}$ denotes the analogous bound based on (3.66) with $2\sigma^2$ in place of C_1. Similarly to (3.59), the first summation is over a subset of the range $i \leq \lceil \log_{1/r} \frac{\eta_{(1)}}{r\sigma} \rceil$, and it follows that the condition (3.60) may be replaced by

$$\begin{aligned} T \geq \Bigg(2\sigma^2\gamma_T\beta_T \sum_{i=1}^{\lceil \log_{1/r} \frac{\eta_{(1)}}{\sigma} \rceil + 1} \frac{1}{\eta_{(i)}^2} + 2\Big\lceil \log_{1/r} \frac{4(1+\bar{\delta})\eta_{(1)}}{r\epsilon} \Big\rceil\Bigg) \log \frac{16(1+\bar{\delta})^2|D|\beta_T}{\bar{\delta}^2\epsilon^2} \\ + \Bigg(C_1\gamma_T\beta_T \sum_{i\,:\,\eta_{(i-1)}^2 > \sigma^2} \frac{1}{\eta_{(i)}^2} + 2\Big\lceil \log_{1/r} \frac{\eta_{(1)}}{r\sigma} \Big\rceil\Bigg) \log \frac{16(1+\bar{\delta})^2|D|\beta_T}{\bar{\delta}^2\epsilon^2}. \end{aligned} \tag{3.68}$$

The first summation is handled in the same way as the previous subsection, and the second summation is upper bounded by writing

$$\sum_{i\,:\,\eta_{(i-1)}^2 > \sigma^2} \frac{1}{\eta_{(i)}^2} \leq \sum_{i=1}^{\lceil \log_{1/r} \frac{\eta_{(1)}}{\sigma} \rceil + 1} \frac{1}{\eta_{(1)}^2 r^{2(i-1)}} \tag{3.69}$$

$$\leq \frac{1}{r^2(1-r^2)\eta_{(1)}^2} \frac{1}{\sigma^2}, \tag{3.70}$$

where (3.69) follows since $\eta_{(i)} = \eta_{(1)} r^{i-1}$, and (3.70) follows in the same way as (3.63). Once again, setting $r = \frac{1}{2}$ and $\eta_{(1)} = 1$ concludes the proof, with the third term in (3.31) coming from the identity $2\lceil \log_2 \frac{8(1+\bar{\delta})}{\epsilon} \rceil + 2\lceil \log_2 \frac{2}{\sigma} \rceil \leq 2\lceil \log_2 \frac{32(1+\bar{\delta})}{\epsilon\sigma} \rceil$.

3.A.3 Proof for the Multi-fidelity setting (Corollary 3.5.1)

The proof follows the same arguments as those of Appendices 3.A.1 and 3.A.2, with C^* being upper bounded in K different ways, one for each possible noise level. The choice $\beta_T = 2\log \frac{|D|T^2 {c_{\max}}^2 \pi^2}{6\delta {c_{\min}}^2}$ arises as a simple upper bound to the right-hand side of (3.15) resulting from the fact that $\sum_{t=1}^{T} c(\boldsymbol{x}_t) \leq c_{\max} T$.

4 Robust Optimization with Gaussian Processes

We consider the problem of sequential Gaussian process optimization as in the previous chapter, but with an added *robustness requirement*: An adversary may perturb the returned point, and we require the function value to remain as high as possible even after this perturbation. This problem is motivated by, e.g., settings where the underlying functions during optimization and implementation stages are different. We show that standard BO algorithms do not exhibit the desired robustness properties, and give a novel confidence-bound based algorithm for this purpose.

This chapter is based on the joint work with Jonathan Scarlett, Stefanie Jegelka and Volkan Cevher [BSJC18].

4.1 Introduction

Gaussian processes (GP) provide a powerful means for sequentially optimizing a black-box function f that is costly to evaluate and for which noisy point evaluations are available. This approach has successfully been applied to numerous applications, including robotics [LWBS07], hyperparameter tuning [SLA12], recommender systems [VNDBK14], environmental monitoring [SKKS10], and more. In many such applications, one is faced with various forms of uncertainty that are not accounted for by standard algorithms. In robotics, the optimization is often performed via simulations, creating a mismatch between the assumed function and the true one; in hyperparameter tuning, the function is similarly mismatched due to limited training data; in recommendation systems and several other applications, the underlying function is inherently time-varying, so the returned solution may become increasingly stale over time; the list goes on.

In this chapter, we address these considerations by studying the GP optimization problem with an additional requirement of adversarial robustness: The returned point may be perturbed by an adversary, and we require the function value to remain as high as possible even after this perturbation. This problem is of interest not only for attaining improved robustness to uncertainty, but also for other related max-min optimization settings (see Section 4.3 for further discussion).

4.1.1 Problem Statement

Let f be an unknown reward function over a domain $D \subseteq \mathbb{R}^p$ for some dimension p. At time t, we query f at a single point $\boldsymbol{x}_t \in D$ and observe a noisy sample $y_t = f(\boldsymbol{x}_t) + z_t$, where $z_t \sim \mathcal{N}(0, \sigma^2)$. After T rounds, a recommended point $\boldsymbol{x}^{(T)}$ is returned. In contrast with the standard goal of making $f(\boldsymbol{x}^{(T)})$ as high as possible, we seek to find a point such that f remains high even after an adversarial perturbation; a formal description is given below.

We consider the non-Bayesian setting (*cf.* Section 2.2.1) first, while in Section 4.3 we show that our main results also hold in the Bayesian setting. We assume that D is endowed with the positive definite kernel function $k(\cdot, \cdot)$, and f has a bounded norm in the corresponding Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_k(D)$. Specifically, we assume that $f \in \mathcal{F}_k(B)$, where

$$\mathcal{F}_k(B) = \{f \in \mathcal{H}_k(D) \,:\, \|f\|_k \leq B\}, \tag{4.1}$$

and $\|f\|_k$ is the RKHS norm in $\mathcal{H}_k(D)$. It is well-known (e.g., see [SKKS10, CG17]) that this assumption permits the construction of confidence bounds via Gaussian process (GP) methods; recall Lemma 3.6.1. We assume that the kernel is normalized to satisfy $k(\boldsymbol{x}, \boldsymbol{x}) = 1$ for all $x \in D$. Two commonly-considered kernels are squared exponential (SE) and Matérn that we provide in (2.2). Given a sequence of decisions $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t\}$ and corresponding noisy observations $\{y_1, \cdots, y_t\}$, the posterior distribution under a GP prior is also Gaussian, with the mean and variance given in (2.4).

We proceed to define our optimization goal. Let $d(\boldsymbol{x}, \boldsymbol{x}')$ be a function mapping $D \times D \to \mathbb{R}$, and let ϵ be a constant known as the *stability parameter*. For each point $\boldsymbol{x} \in D$, we define a set

$$\Delta_\epsilon(\boldsymbol{x}) = \big\{\boldsymbol{x}' - \boldsymbol{x} \,:\, \boldsymbol{x}' \in D \;\text{ and }\; d(\boldsymbol{x}, \boldsymbol{x}') \leq \epsilon\big\}. \tag{4.2}$$

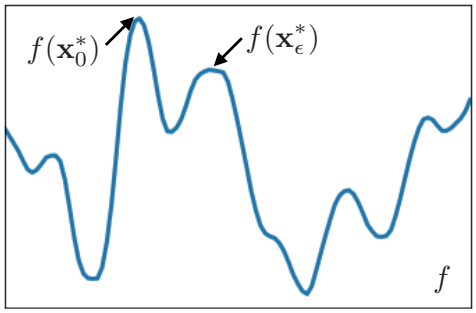
One can interpret this as the set of perturbations of $\boldsymbol{x}$ such that the newly obtained point $\boldsymbol{x}'$ is within a "distance" ϵ of $\boldsymbol{x}$. While we refer to $d(\cdot, \cdot)$ as the distance function throughout this chapter, we allow it to be a general function, and not necessarily a distance in the mathematical sense. As we exemplify in Section 4.4, the parameter ϵ might be naturally specified as part of the application, or might be better treated as a parameter that can be tuned for the purpose of the overall learning goal.

We define an ϵ-*stable optimal input* to be any $\boldsymbol{x}^*_\epsilon$ satisfying

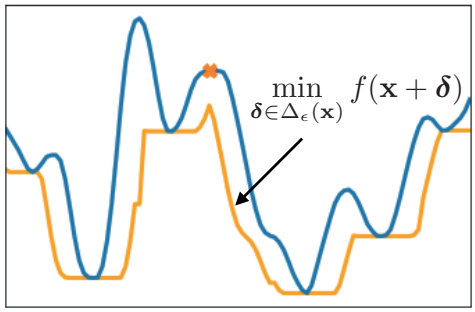
$$\boldsymbol{x}^*_\epsilon \in \operatorname*{arg\,max}_{\boldsymbol{x} \in D} \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x} + \boldsymbol{\delta}). \tag{4.3}$$

Our goal is to report a point $\boldsymbol{x}^{(T)}$ that is stable in the sense of having low ϵ-*regret*, defined as

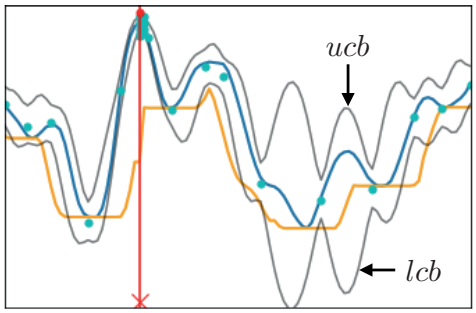
$$r_\epsilon(\boldsymbol{x}) = \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x}^*_\epsilon)} f(\boldsymbol{x}^*_\epsilon + \boldsymbol{\delta}) - \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x} + \boldsymbol{\delta}). \tag{4.4}$$



(a) Unknown function f



(b) Robust objective



(c) Points selected by GP-UCB

Figure 4.1: (a) A function f and its maximizer $\boldsymbol{x}_0^*$; (b) for $\epsilon = 0.06$ and $d(\boldsymbol{x}, \boldsymbol{x}') = |\boldsymbol{x} - \boldsymbol{x}'|$, the decision that corresponds to the local "wider" maximum of f is the *optimal ϵ-stable* decision; (c) GP-UCB selects a point that nearly maximizes f, but is strictly suboptimal in the ϵ-stable sense.

Note that once $r_\epsilon(\boldsymbol{x}) \leq \eta$ for some accuracy value $\eta \geq 0$, it follows that

$$\min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x}+\boldsymbol{\delta}) \geq \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x}_\epsilon^*)} f(\boldsymbol{x}_\epsilon^*+\boldsymbol{\delta}) - \eta. \tag{4.5}$$

We assume that $d(\cdot,\cdot)$ and ϵ are known, i.e., they are specified as part of the problem.

As a running example, we consider that $d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|$ for some norm $\|\cdot\|$ (e.g., ℓ_2-norm), in which case achieving low ϵ-regret amounts to favoring *broad peaks* instead of narrow ones, particularly for higher ϵ; see Figure 4.1 for an illustration. In Section 4.3, we discuss how our framework also captures a variety of other interesting max-min optimization settings.

Failure of Classical Methods

Various algorithms have been developed for achieving small regret in the standard GP optimization problem. A prominent example is GP-UCB (Section 2.2), which chooses

$$\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x}\in D} \ \mathrm{ucb}_{t-1}(\boldsymbol{x}), \qquad \mathrm{ucb}_{t-1}(\boldsymbol{x}) := \mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}). \tag{4.6}$$

This algorithm is guaranteed to achieve sublinear cumulative regret with high probability[1], for a suitably chosen β_t (see Theorem 2.2.1). While this is useful when $\boldsymbol{x}_\epsilon^* = \boldsymbol{x}_0^*$, in general for a given fixed $\epsilon \neq 0$, these two decisions might not coincide, and hence $\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x}_0^*)} f(\boldsymbol{x}_0^* + \boldsymbol{\delta})$ can be significantly smaller than $\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x}_\epsilon^*)} f(\boldsymbol{x}_\epsilon^* + \boldsymbol{\delta})$. A visual example is given in Figure 4.1c, where the selected point of GP-UCB for $t = 20$ is shown. This point nearly maximizes f, but it is strictly suboptimal in the ϵ-stable sense. The same limitation applies to other BO strategies (e.g., [HS12, HLHG14, BSKC16, WJ17, SJ17a, RMGO18]) whose goal is to identify the global non-robust maximum $\boldsymbol{x}_0^*$. In Section 4.4, we will see that more advanced baseline strategies also perform poorly when applied to our problem.

4.1.2 Related Work

Numerous algorithms have been developed for GP optimization in recent years [SKKS10, HS12, HLHG14, BSKC16, WJ17, SJ17a, RMGO18]. Beyond the standard setting, important extensions have been considered including batch sampling [DKB14, GDHL16, CBRV13, AFF10], contextual and time-varying settings [KO11, BSC16], safety requirements [SGBK15], and high dimensional settings [KSP15, WLJK17, RSBC18], just to name a few.

Various forms of robustness in GP optimization have been considered previously. A prominent example is that of outliers [MCTM18], in which certain function values are highly unreliable; however, this is a separate issue from the one considered in this chapter, since in [MCTM18] the returned point does not undergo any perturbation. Another related recent work is [BN17], which assumes that the *sampled points* (rather than the returned one) are subject to uncertainty. In addition to this difference, the uncertainty in [BN17] is random rather than adversarial, which is complementary but distinct from our work. The same is true of a setting called *unscented Bayesian optimization* in [NMCBJ16]. Moreover, no theoretical results are given in [BN17, NMCBJ16].

Our problem formulation is also related to other works on non-convex robust optimization, particularly those of *Bertsimas et al.* [BNT10b, BNT10a]. In these works, a stable design $\boldsymbol{x}$ is sought that solves $\min_{\boldsymbol{x} \in D} \max_{\boldsymbol{\delta} \in \mathcal{U}} f(\boldsymbol{x} + \boldsymbol{\delta})$. Here, $\boldsymbol{\delta}$ resides in some uncertainty set $\mathcal{U}$, and represents the perturbation against which the design $\boldsymbol{x}$ needs to be protected. Related problems have also recently been considered in the context of adversarial training (e.g. [SND18]). Compared to these works, our work bears the crucial difference that the objective function is *unknown*, and we can only learn about it through noisy point evaluations (i.e. bandit feedback).

Other works, such as [CLSS17, Wil17, SWJ18, KMGG08, BMSC17b], have considered robust optimization problems of the following form: For a given set of objectives $\{f_1, \dots, f_m\}$ find $\boldsymbol{x}$ achieving $\max_{\boldsymbol{x} \in D} \min_{i \in [m]} f_i(\boldsymbol{x})$. We discuss variations of our algorithm for this type of formulation in Section 4.3.

[1] In this discussion, we take $d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2$, so that $\epsilon = 0$ recovers the standard non-stable regret from Section 2.2

4.1.3 Contributions

The main contributions of this chapter are:

- In Section 4.1.1, a variant of GP optimization is introduced in which the returned solution is required to exhibit robustness to an adversarial perturbation.
- We demonstrate the failures of standard algorithms, and in Section 4.2, we introduce a new algorithm StableOpt that overcomes these limitations.
- We provide a novel theoretical analysis characterizing the number of samples required for StableOpt to attain a near-optimal robust solution, and we complement this with an algorithm-independent lower bound (Sections 4.2.1 and 4.2.2).
- In Section 4.3, we provide several important variations of our max-min optimization framework and theory, including connections and comparisons to previous works.
- In Section 4.4, we experimentally demonstrate a variety of potential applications of interest on real-world data sets, and we show that StableOpt consistently succeeds in finding a stable maximizer where several baseline methods fail.

4.2 Stable Algorithm and Theory

Our proposed algorithm, StableOpt, is described in Algorithm 7, and makes use of the following confidence bounds depending on a parameter β_t (*cf.*, Lemma 3.6.1 below):

$$\mathrm{ucb}_{t-1}(\boldsymbol{x}) := \mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \quad \mathrm{lcb}_{t-1}(\boldsymbol{x}) := \mu_{t-1}(\boldsymbol{x}) - \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}). \tag{4.7}$$

We refer to the point

$$\tilde{\boldsymbol{x}}_t = \operatorname*{arg\,max}_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} \mathrm{ucb}_{t-1}(\boldsymbol{x}+\boldsymbol{\delta}) \tag{4.8}$$

as the one having the highest "stable" upper confidence bound. However, the queried point is not $\tilde{\boldsymbol{x}}_t$, but instead $\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t$, where $\boldsymbol{\delta}_t \in \Delta_\epsilon(\tilde{\boldsymbol{x}}_t)$ is chosen to minimize the *lower* confidence bound. As a result, the algorithm is based on two distinct principles: (i) optimism in the face of uncertainty when it comes to selecting $\tilde{\boldsymbol{x}}_t$; (ii) pessimism in the face of uncertainty when it comes to anticipating the perturbation of $\tilde{\boldsymbol{x}}_t$. The first of these is inherent to existing algorithms such as GP-UCB, whereas the second is unique to the stable GP optimization problem. An example illustration of StableOpt's execution (sampling) is given in Figure 4.2.

We have left the final reported point $\boldsymbol{x}^{(T)}$ unspecified in Algorithm 7, as there are numerous reasonable choices. The simplest choice is to simply return $\boldsymbol{x}^{(T)} = \tilde{\boldsymbol{x}}_T$, but in our theory and experiments, we will focus on $\boldsymbol{x}^{(T)}$ equaling the point in $\{\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_T\}$ with the highest lower confidence bound.

Algorithm 7 STABLEOPT [BSJC18]

Input: Domain D, GP(μ_0, σ_0, k), confidence bound parameters $\{\beta_t\}$, stability parameter ϵ, distance function $d(\cdot, \cdot)$

1: **for** $t = 1, 2, \ldots, T$ **do**

2: Set

$$\tilde{\boldsymbol{x}}_t = \arg\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} \mathrm{ucb}_{t-1}(\boldsymbol{x} + \boldsymbol{\delta}).$$

3: Set $\boldsymbol{\delta}_t = \arg\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta})$

4: Sample $\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t$, and observe $y_t = f(\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t) + z_t$

5: Update μ_t, σ_t, ucb_t and lcb_t according to (2.4) and (4.7), by including $\{(\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t, y_t)\}$

6: **end for**

4.2.1 Upper Bound on Regret

As in the previous chapters, our analysis makes use of the maximum information gain under t noisy measurements:

$$\gamma_t = \max_{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t} \frac{1}{2} \log\det(\boldsymbol{I}_t + \sigma^{-2} \boldsymbol{K}_t). \tag{2.10}$$

STABLEOPT depends on the exploration parameter β_t, which determines the width of the confidence bounds. In our main result, we set β_t according to Lemma 3.6.1 which we recall here:

Lemma 3.6.1 ([CG17])**.** *Fix $f \in \mathcal{F}_k(B)$, and consider the sampling model $y_t = f(\boldsymbol{x}_t) + z_t$ with $z_t \sim \mathcal{N}(0, \sigma^2)$ with independence between times. Under the choice*

$$\beta_t = \left(B + \sigma \sqrt{2(\gamma_{t-1} + \log \frac{e}{\xi})} \right)^2,$$

the following holds with probability at least $1 - \xi$:

$$\mathrm{lcb}_{t-1}(\boldsymbol{x}) \leq f(\boldsymbol{x}) \leq \mathrm{ucb}_{t-1}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in D, \forall t \geq 1. \tag{3.33}$$

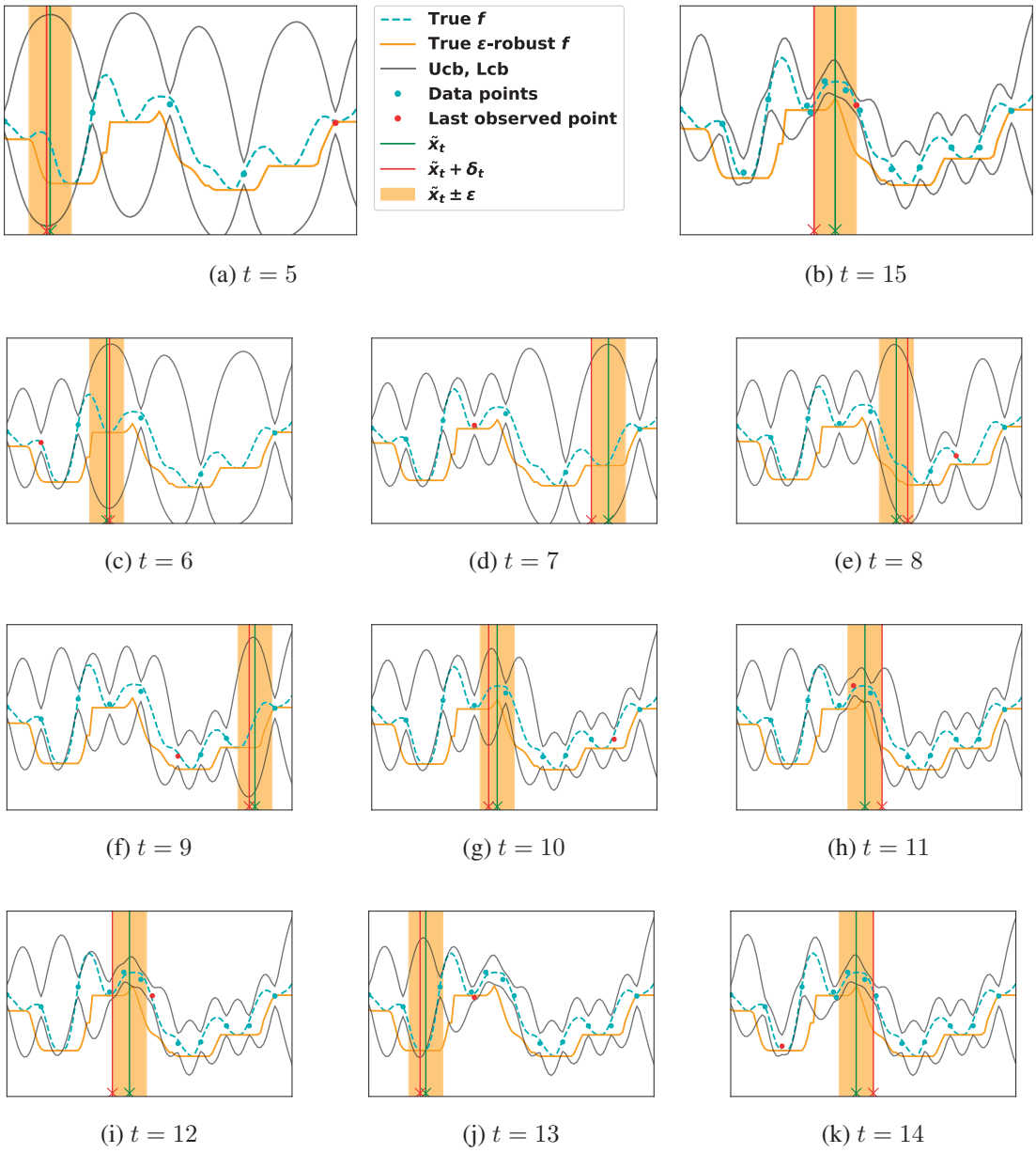
The following theorem upper bounds the performance of STABLEOPT under a suitable choice of the recommended point $\boldsymbol{x}^{(T)}$.

Theorem 4.2.1. (Upper Bound) *Fix ϵ, $\eta > 0$, $B > 0$, $T \in \mathbb{Z}_+$, and $\xi \in (0, 1)$, and a distance function $d(\boldsymbol{x}, \boldsymbol{x}')$, and suppose that*

$$\frac{T}{\beta_T \gamma_T} \geq \frac{C_1}{\eta^2},$$

where $C_1 = 8/\log(1 + \sigma^{-2})$. For any $f \in \mathcal{F}_k(B)$, STABLEOPT *with β_t set as in Lemma 3.6.1 achieves $r_\epsilon(\boldsymbol{x}^{(T)}) \leq \eta$ after T rounds with probability at least $1 - \xi$, where*

$$\boldsymbol{x}^{(T)} = \tilde{\boldsymbol{x}}_{t^*}, \quad t^* = \arg\max_{t=1,\ldots,T} \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}). \tag{4.9}$$



(a) $t = 5$ (b) $t = 15$

(c) $t = 6$ (d) $t = 7$ (e) $t = 8$

(f) $t = 9$ (g) $t = 10$ (h) $t = 11$

(i) $t = 12$ (j) $t = 13$ (k) $t = 14$

Figure 4.2: An execution of StableOpt on the running example. Figures 4.2a and 4.2b give an example of the selection procedure of StableOpt at two different time steps. We observe that after $t = 15$ steps, $\tilde{\boldsymbol{x}}_t$ obtained in Eq. 4.8 corresponds to $\boldsymbol{x}^*_\epsilon$. The intermediate steps are presented in the subsequent rows.

Proof. Recall that $\tilde{\boldsymbol{x}}_t$ is the point computed by StableOpt in (4.8) at time t, and that $\boldsymbol{\delta}_t$ corresponds to the perturbation obtained in StableOpt (Line 3) at time t. In the following, we condition on the event in Lemma 3.6.1 holding true, meaning that ucb_t and lcb_t provide valid confidence bounds as per (3.33). As stated in the lemma, this holds with probability at least $1 - \xi$.

By the definition of ϵ-instant regret, we have

$$\begin{aligned}
r_\epsilon(\tilde{\boldsymbol{x}}_t) &= \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x}+\boldsymbol{\delta}) - \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x}_t)} f(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}) \\
&\le \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x}+\boldsymbol{\delta}) - \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}) && (4.10)\\
&= \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x}+\boldsymbol{\delta}) - \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) && (4.11)\\
&\le \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} \mathrm{ucb}_{t-1}(\boldsymbol{x}+\boldsymbol{\delta}) - \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) && (4.12)\\
&= \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{ucb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}) - \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) && (4.13)\\
&\le \mathrm{ucb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) - \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) && (4.14)\\
&= 2\beta_t^{1/2}\sigma_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t), && (4.15)
\end{aligned}$$

where (4.10) and (4.12) follow from Lemma 3.6.1, (4.11) follows since $\boldsymbol{\delta}_t$ minimizes lcb_{t-1} by definition, (4.13) follows since $\tilde{\boldsymbol{x}}_t$ maximizes the robust upper confidence bound by definition, (4.14) follows by upper bounding the minimum by the specific choice $\boldsymbol{\delta}_t \in \Delta_\epsilon(\boldsymbol{x}_t)$, and (4.15) follows since the upper and lower confidence bounds are separated by $2\beta_t^{1/2}\sigma_{t-1}(\cdot)$ according to their definitions in (4.7).

In fact, the analysis from (4.10) to (4.15) shows that the following *pessimistic estimate* of $r_\epsilon(\tilde{\boldsymbol{x}}_t)$ is upper bounded by $2\beta_t^{1/2}\sigma_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t)$:

$$\overline{r}_\epsilon(\tilde{\boldsymbol{x}}_t) = \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x}+\boldsymbol{\delta}) - \min_{\boldsymbol{\delta}\in\Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{lcb}_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}). \tag{4.16}$$

Unlike $r_\epsilon(\tilde{\boldsymbol{x}}_t)$, the algorithm has the required knowledge to identify the value of $t \in \{1,\dots,T\}$ with the smallest $\overline{r}_\epsilon(\tilde{\boldsymbol{x}}_t)$. Specifically, the first term on the right-hand side of (4.16) does not depend on t, so the smallest $\overline{r}_\epsilon(\tilde{\boldsymbol{x}}_t)$ is achieved by $\boldsymbol{x}^{(T)}$ defined in (4.9). Since the minimum is upper bounded by the average, it follows that

$$\begin{aligned}
r_\epsilon(\boldsymbol{x}^{(T)}) &\le \overline{r}_\epsilon(\boldsymbol{x}^{(T)}) && (4.17)\\
&\le \frac{1}{T}\sum_{t=1}^{T} 2\beta_t^{1/2}\sigma_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) && (4.18)\\
&\le \frac{2\beta_T^{1/2}}{T}\sum_{t=1}^{T}\sigma_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t), && (4.19)
\end{aligned}$$

where (4.19) uses the monotonicity of β_T. Next, we claim that for

$$2\sum_{t=1}^{T}\sigma_{t-1}(\tilde{\boldsymbol{x}}_t+\boldsymbol{\delta}_t) \le \sqrt{C_1 T\gamma_T}, \tag{4.20}$$

where $C_1 = 8/\log(1+\sigma^{-2})$. In fact, this is a special case of the well-known result from [SKKS10].

This result, i.e., [SKKS10, Lemma 5.4],[2] upper bounds the sum of posterior standard deviations of sampled points in terms of the information gain γ_T (recall that in round t, StableOpt samples at location $\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t$). Combining (4.19)–(4.20) and re-arranging, we deduce that after T satisfies $\frac{T}{\beta_T \gamma_T} \geq \frac{C_1}{\eta^2}$, the ϵ-instant regret is at most η, thus completing the proof. □

This result holds for general kernels, and for both finite and continuous D. Our analysis bounds function values according to the confidence bounds in Lemma 3.6.1 analogously to GP-UCB [SKKS10], but also addresses the non-trivial challenge of characterizing the perturbations $\boldsymbol{\delta}_t$.

Theorem 4.2.1 can be made more explicit by substituting bounds on γ_T; in particular, we have $\gamma_T = O((\log T)^{p+1})$ for the SE kernel, and $\gamma_T = O(T^{\frac{p(p+1)}{2\nu+p(p+1)}} \log T)$ for the Matérn-ν kernel (see Section 2.2.1). The former yields $T = O^*\big(\frac{1}{\eta^2}\big(\log \frac{1}{\eta}\big)^{2p}\big)$ in Theorem 4.2.1 for constant B, σ^2, and ϵ (where $O^*(\cdot)$ hides dimension-independent log factors), which we will shortly see nearly matches an algorithm-independent lower bound.

4.2.2 Lower Bound on Regret

Establishing lower bounds under general kernels and input domains is an open problem even in the non-robust setting. Accordingly, the following theorem focuses on a more specific setting than the upper bound: We let the input domain be $[0,1]^p$ for some dimension p, and we focus on the SE and Matérn kernels. In addition, we only consider the case that $d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2$, though extensions to other norms (e.g., ℓ_1 or ℓ_∞) follow immediately from the proof.

Theorem 4.2.2. (Lower Bound) *Let $D = [0,1]^p$ for some dimension p, and set the distance function $d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2$. Fix $\epsilon \in \big(0, \frac{1}{2}\big)$, $\eta \in \big(0, \frac{1}{2}\big)$, $B > 0$, and $T \in \mathbb{Z}_+$. Suppose there exists an algorithm that, for any $f \in \mathcal{F}_k(B)$, reports a point $\boldsymbol{x}^{(T)}$ achieving ϵ-regret $r_\epsilon(\boldsymbol{x}^{(T)}) \leq \eta$ after T rounds w.p. at least $1 - \xi$. Then, provided that $\frac{\eta}{B}$ and ξ are sufficiently small, we have the following:*

1. *For $k = k_{\text{SE}}$, it is necessary that $T = \Omega\big(\frac{\sigma^2}{\eta^2}\big(\log \frac{B}{\eta}\big)^{p/2}\big)$,*
2. *For $k = k_{\text{Matérn}}$, it is necessary that $T = \Omega\big(\frac{\sigma^2}{\eta^2}\big(\frac{B}{\eta}\big)^{p/\nu}\big)$.*

Here we assume that the stability parameter ϵ, dimension p, target probability ξ, and kernel parameters l, ν are fixed (i.e., not varying as a function of the parameters T, η, σ and B).

The proof is based on constructing a finite subset of "difficult" functions in $\mathcal{F}_k(B)$ and applying lower bounding techniques from the multi-armed bandit literature, also making use of several auxiliary results from the non-robust setting [SBC17]. More specifically, the functions in the restricted class consist of narrow negative "valleys" that the adversary can perturb the reported point into, but that are hard to identify until a large number of samples have been taken.

[2] More precisely, [SKKS10, Lemma 5.4] alongside an application of the Cauchy-Schwarz inequality as in [SKKS10].

For constant σ^2 and B, the condition for the SE kernel simplifies to $T = \Omega\big(\frac{1}{\eta^2}\big(\log\frac{1}{\eta}\big)^{p/2}\big)$, thus nearly matching the upper bound $T = O^*\big(\frac{1}{\eta^2}\big(\log\frac{1}{\eta}\big)^{2p}\big)$ of StableOpt established above. In the case of the Matérn kernel, more significant gaps remain between the upper and lower bounds; similar gaps remain even in the standard (non-robust) setting as shown in Section 3.6.

4.3 Other Robust Settings and Variations of StableOpt

While the above problem formulation seeks robustness within an ϵ-ball corresponding to the distance function $d(\cdot,\cdot)$, our algorithm and theory apply to a variety of seemingly distinct settings. We outline a few such settings here; in Appendix 4.A, we give details of their derivations.

Robust Bayesian optimization. Algorithm 7 and Theorem 4.2.1 extend readily to the Bayesian case ($f \sim \mathrm{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$). Since the proof of Theorem 4.2.1 is based on confidence bounds, the only change required is selecting β_t to be that used for the Bayesian setting in [SKKS10]. As a result, our framework also captures the problem of *adversarially robust BO*. All of the variations below apply to both the Bayesian and non-Bayesian settings.

Robustness to unknown parameters. Consider the scenario where an unknown function $f : D \times \Theta \to \mathbb{R}$ has a bounded RKHS norm under some composite kernel $k((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}'))$. Some special cases ([KO11]) include

$$k((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}')) = k(\boldsymbol{x}, \boldsymbol{x}') + k(\boldsymbol{\theta}, \boldsymbol{\theta}') \quad \text{and} \quad k((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}')) = k(\boldsymbol{x}, \boldsymbol{x}')k(\boldsymbol{\theta}, \boldsymbol{\theta}').$$

The posterior mean $\mu_t(\boldsymbol{x}, \boldsymbol{\theta})$ and variance $\sigma_t^2(\boldsymbol{x}, \boldsymbol{\theta})$ conditioned on the previous observations $(\boldsymbol{x}_1, \boldsymbol{\theta}_1, y_1), ..., (\boldsymbol{x}_{t-1}, \boldsymbol{\theta}_{t-1}, y_{t-1})$ are computed analogously to (2.4).

A robust optimization formulation in this setting is to seek $\boldsymbol{x}$ that solves

$$\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\theta} \in \Theta} f(\boldsymbol{x}, \boldsymbol{\theta}), \tag{4.21}$$

i.e., we seek to find a configuration $\boldsymbol{x}$ that is robust against any possible parameter vector $\boldsymbol{\theta} \in \Theta$. Potential applications of this setup include the following:

- In areas such a robotics, we may have numerous parameters to tune (given by $\boldsymbol{x}$ and $\boldsymbol{\theta}$ collectively), but when it comes to implementation, some of them (i.e., only $\boldsymbol{\theta}$) become out of our control. Hence, we need to be robust against whatever values they may take.
- We wish to tune hyperparameters in order to make an algorithm work simultaneously for a number of distinct data types that bear some similarities. The data types are represented by $\boldsymbol{\theta}$, and we can choose the data type to our liking during the optimization stage.

StableOpt can be used to solve (4.21); we maintain $\boldsymbol{\theta}_t$, and modify the main steps to

$$\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\theta} \in \Theta} \mathrm{ucb}_{t-1}(\boldsymbol{x}, \boldsymbol{\theta}) \quad \text{and} \quad \boldsymbol{\theta}_t \in \arg\min_{\boldsymbol{\theta} \in \Theta} \mathrm{lcb}_{t-1}(\boldsymbol{x}_t, \boldsymbol{\theta}). \tag{4.22}$$

At each time step, StableOpt receives a noisy observation $y_t = f(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + z_t$, which is used with $(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$ for computing the posterior. As explained in Appendix 4.A, the guarantee $r_\epsilon(\boldsymbol{x}^{(T)}) \leq \eta$ in Theorem 4.2.1 is replaced by

$$\min_{\boldsymbol{\theta}\in\Theta} f(\boldsymbol{x}^{(T)}, \boldsymbol{\theta}) \geq \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\theta}\in\Theta} f(\boldsymbol{x}, \boldsymbol{\theta}) - \eta.$$

Robust estimation. Continuing with the consideration of a composite kernel on $(\boldsymbol{x}, \boldsymbol{\theta})$, we consider the following practical problem variant proposed in [BNT10b]. Let $\bar{\boldsymbol{\theta}} \in \Theta$ be an estimate of the true problem coefficient $\boldsymbol{\theta}^* \in \Theta$. Since, $\bar{\boldsymbol{\theta}}$ is an estimate, the true coefficient satisfies $\boldsymbol{\theta}^* = \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}}$, where $\boldsymbol{\delta}_{\boldsymbol{\theta}}$ represents uncertainty in $\bar{\boldsymbol{\theta}}$. Often, practitioners solve $\max_{\boldsymbol{x}\in D} f(\boldsymbol{x}, \bar{\boldsymbol{\theta}})$ and ignore the uncertainty. As a more sophisticated approach, we let $\Delta_\epsilon(\bar{\boldsymbol{\theta}}) = \left\{\boldsymbol{\theta} - \bar{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \text{ and } d(\bar{\boldsymbol{\theta}}, \boldsymbol{\theta}) \leq \epsilon\right\}$, and consider the following robust problem formulation: $\max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}_{\boldsymbol{\theta}}\in\Delta_\epsilon(\bar{\boldsymbol{\theta}})} f(\boldsymbol{x}, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}})$. For the given estimate $\bar{\boldsymbol{\theta}}$, the main steps of StableOpt in this setting are

$$\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}_{\boldsymbol{\theta}}\in\Delta_\epsilon(\bar{\boldsymbol{\theta}})} \mathrm{ucb}_{t-1}(\boldsymbol{x}, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}}) \quad \text{and} \quad \boldsymbol{\delta}_{\boldsymbol{\theta},t} \in \arg\min_{\boldsymbol{\delta}_{\boldsymbol{\theta}}\in\Delta_\epsilon(\bar{\boldsymbol{\theta}})} \mathrm{lcb}_{t-1}(\boldsymbol{x}_t, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}}), \quad (4.23)$$

and the noisy observations take the form $y_t = f(\boldsymbol{x}_t, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta},t}) + z_t$. The guarantee $r_\epsilon(\boldsymbol{x}^{(T)}) \leq \eta$ in Theorem 4.2.1 is replaced by

$$\min_{\boldsymbol{\delta}_{\boldsymbol{\theta}}\in\Delta_\epsilon(\bar{\boldsymbol{\theta}})} f(\boldsymbol{x}^{(T)}, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}}) \geq \max_{\boldsymbol{x}\in D} \min_{\boldsymbol{\delta}_{\boldsymbol{\theta}}\in\Delta_\epsilon(\bar{\boldsymbol{\theta}})} f(\boldsymbol{x}, \bar{\boldsymbol{\theta}} + \boldsymbol{\delta}_{\boldsymbol{\theta}}) - \eta.$$

Robust group identification. In some applications, it is natural to partition D into disjoint subsets, and search for the subset with the highest worst-case function value (see Section 4.4 for a movie recommendation example). Letting $\mathcal{G} = \{G_1, \ldots, G_k\}$ denote the groups that partition the input space, the robust optimization problem is given by $\max_{G\in\mathcal{G}} \min_{\boldsymbol{x}\in G} f(\boldsymbol{x})$, i.e., the algorithm reports a group $G^{(T)}$. The main steps of StableOpt are given by

$$G_t \in \arg\max_{G\in\mathcal{G}} \min_{\boldsymbol{x}\in G} \mathrm{ucb}_{t-1}(\boldsymbol{x}) \quad \text{and} \quad \boldsymbol{x}_t \in \arg\min_{\boldsymbol{x}\in G_t} \mathrm{lcb}_{t-1}(\boldsymbol{x}), \quad (4.24)$$

and the observations are of the form $y_t = f(\boldsymbol{x}_t) + z_t$ as usual. The guarantee $r_\epsilon(\boldsymbol{x}^{(T)}) \leq \eta$ in Theorem 4.2.1 is replaced by

$$\min_{\boldsymbol{x}\in G^{(T)}} f(\boldsymbol{x}) \geq \max_{G\in\mathcal{G}} \min_{\boldsymbol{x}\in G} f(\boldsymbol{x}) - \eta.$$

The preceding variations of StableOpt, as well as their resulting variations of Theorem 4.2.1, follow by substituting certain (unconventional) choices of $d(\cdot, \cdot)$ and ϵ into Algorithm 7 and Theorem 4.2.1, with $(\boldsymbol{x}, \boldsymbol{\theta})$ in place of $\boldsymbol{x}$ where appropriate. The details are given in Section 4.A.

4.4 Experimental Evaluation

In this section, we experimentally validate the performance of StableOpt by comparing against several baselines. Each algorithm that we consider may distinguish between the *sampled point* (i.e., the point that produces the noisy observation y_t) and the *reported point* (i.e., the point believed to be near-optimal in terms of ϵ-stability). For StableOpt, as described in Algorithm 7, the sampled point is $\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta}_t$, and the reported point after time t is the one in $\{\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_t\}$ with the highest value of $\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\tilde{\boldsymbol{x}}_t)} \mathrm{lcb}_t(\tilde{\boldsymbol{x}}_t + \boldsymbol{\delta})$.[3] In addition, we consider the following baselines:

- GP-UCB. We consider GP-UCB to be a good representative of the wide range of existing methods that search for the non-robust global maximum.
- MaxiMin-GP-UCB. We consider a natural generalization of GP-UCB in which, at each time step, the sampled and reported point are both given by

$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} \mathrm{ucb}_{t-1}(\boldsymbol{x} + \boldsymbol{\delta}).$$

- Stable-GP-Random. The sampling point $\boldsymbol{x}_t$ at every time step is chosen uniformly at random, while the reported point at time t is chosen to be the point among the sampled points $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_t\}$ according to the same rule as the one used for StableOpt.
- Stable-GP-UCB. The sampled point is given by the GP-UCB rule, while the reported point is again chosen in the same way as in StableOpt.

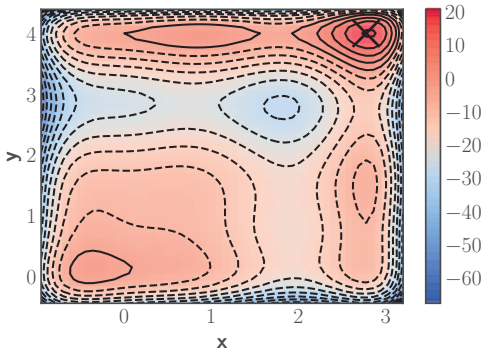
As observed in e.g. [SKKS10, BSKC16]), the theoretical choice of β_t is overly conservative. We therefore adopt a constant value of $\beta_t^{1/2} = 2.0$ in each of the above methods, which we found to provide a suitable exploration/exploitation trade-off for each of the above algorithms.

Given a reported point $\boldsymbol{x}^{(t)}$ at time t, the performance metric is the ϵ-regret $r_\epsilon(\boldsymbol{x}^{(t)})$ given in (4.4). Two observations are in order: (i) All the baselines are heuristic approaches for our problem, and they do not have guarantees in terms of near-optimal stability; (ii) We do not compare against other standard BO methods, as their performance is comparable to that of GP-UCB; in particular, they suffer from exactly the same pitfalls described at the end of Section 4.1.1.

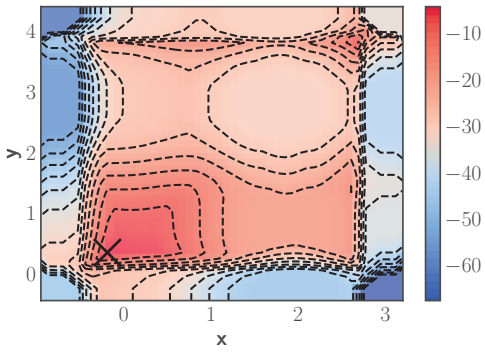
Synthetic function. We consider the synthetic function from [BNT10b] (see Figure 4.3a), given by

$$\begin{aligned} f_{\mathrm{poly}}(x, y) = &-2x^6 + 12.2x^5 - 21.2x^4 - 6.2x + 6.4x^3 + 4.7x^2 - y^6 + 11y^5 \\ &- 43.3y^4 + 10y + 74.8y^3 - 56.9y^2 + 4.1xy + 0.1y^2x^2 - 0.4y^2x - 0.4x^2y. \end{aligned} \tag{4.25}$$

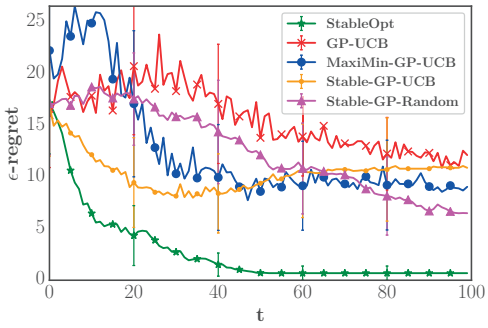
[3] This is slightly different from Theorem 4.2.1, which uses the confidence bound $\mathrm{lcb}_{\tau-1}$ for $\boldsymbol{x}_\tau$ instead of adopting the common bound lcb_t. We found the latter to be more suitable when the kernel hyperparameters are updated online, whereas Theorem 4.2.1 assumes a known kernel. Theorem 4.2.1 can be adapted to use lcb_t alone by intersecting the confidence bounds at each time instant so that they are monotonically shrinking [GCHK13].



(a) $f_{\text{poly}}(x, y)$



(b) $g_{\text{poly}}(x, y)$



(c) ϵ-regret

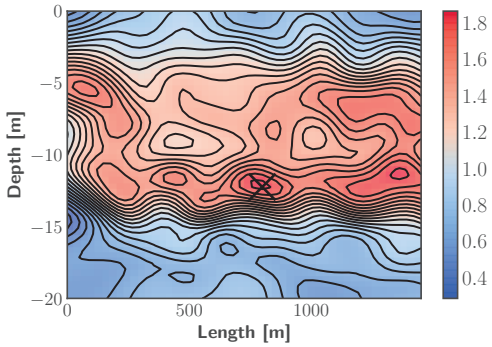
Figure 4.3: Synthetic function from [BNT10b] (in (a)), counterpart with worst-case perturbations (in (b)), and the performance (in (c)). StableOpt significantly outperforms the baselines.

The decision space is a uniformly spaced grid of points in $((-0.95, 3.2), (-0.45, 4.4))$ of size 10^4. There exist multiple local maxima, and the global maximum is at $(x_f^*, y_f^*) = (2.82, 4.0)$, with $f_{\text{poly}}(x_f^*, y_f^*) = 20.82$. Similarly as in [BNT10b], given stability parameters $\boldsymbol{\delta} = (\delta_x, \delta_y)$, where $\|\boldsymbol{\delta}\|_2 \leq 0.5$, the robust optimization problem is

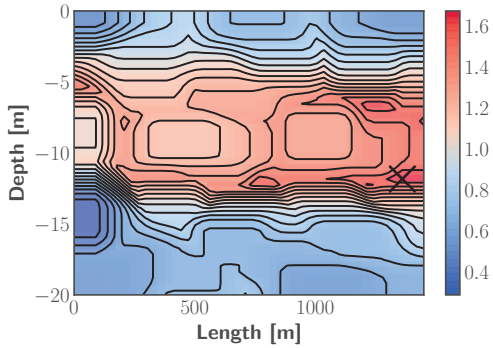
$$\max_{(x,y)\in D} g_{\text{poly}}(x, y), \qquad g_{\text{poly}}(x, y) := \min_{(\delta_x,\delta_y)\in\Delta_{0.5}(x,y)} f_{\text{poly}}(x - \delta_x, y - \delta_y).$$

A plot of g_{poly} is shown in Fig. 4.3b. The global maximum is attained at $(x_g^*, y_g^*) = (-0.195, 0.284)$ and $g_{\text{poly}}(x_g^*, y_g^*) = -4.33$, and the inputs maximizing f yield $g_{\text{poly}}(x_f^*, y_f^*) = -22.34$.

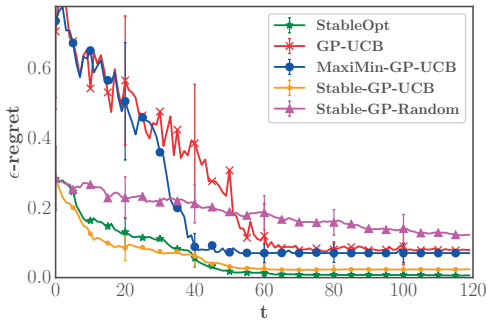
We initialize the above algorithms by selecting 10 uniformly random inputs (x, y), keeping those points the same for each algorithm. The kernel adopted is a SE ARD kernel. We randomly sample 500 points whose function value is above -15.0 to learn the GP hyperparameters via maximum likelihood, and then run the algorithms with these hyperparameters. The observation noise standard deviation is set to 0.1, and the number of sampling rounds is $T = 100$. We repeat the experiment 100 times and show the average performance in Figure 4.3c. We observe that StableOpt significantly outperforms the baselines. In the later rounds, the baselines report points that are close to the global optimizer, which is suboptimal with respect to the ϵ-regret.



(a) Chlorophyll concentration



(b) Robust objective



(c) ϵ-regret

Figure 4.4: Experiment on the Zürich lake dataset; In the later rounds StableOpt is the only method that reports a near-optimal ϵ-stable point.

Lake data. We consider an application regarding environmental monitoring of inland waters, using a data set containing 2024 in situ measurements of chlorophyll concentration within a vertical transect plane, collected by an autonomous surface vessel in Lake Zürich. This dataset was considered in the previous chapter to detect regions of high concentration. The goal was to locate all regions whose concentration exceeds a pre-defined threshold.

Here we consider a different goal: We seek to locate a region of a given size such that the concentration throughout the region is as high as possible (in the max-min sense). This is of interest in cases where high concentration only becomes relevant when it is spread across a sufficiently wide area. We consider rectangular regions with different pre-specified lengths in each dimension:

$$\Delta_{\epsilon_D,\epsilon_L}(\boldsymbol{x}) = \{\boldsymbol{x}' - \boldsymbol{x} \,:\, \boldsymbol{x}' \in D,\ |x_D - x'_D| \leq \epsilon_D \cap |x_L - x'_L| \leq \epsilon_L\}, \tag{4.26}$$

where $\boldsymbol{x} = (x_D, x_L)$ and $\boldsymbol{x}' = (x'_D, x'_L)$ indicate the depth and length, and we denote the stability parameters by (ϵ_D, ϵ_L). This corresponds to $d(\cdot,\cdot)$ being a weighted ℓ_∞-norm.

We evaluate each of the algorithms on a 50×50 grid of points, with the corresponding values coming from the Gaussian process posterior that was derived using the originally collected data.

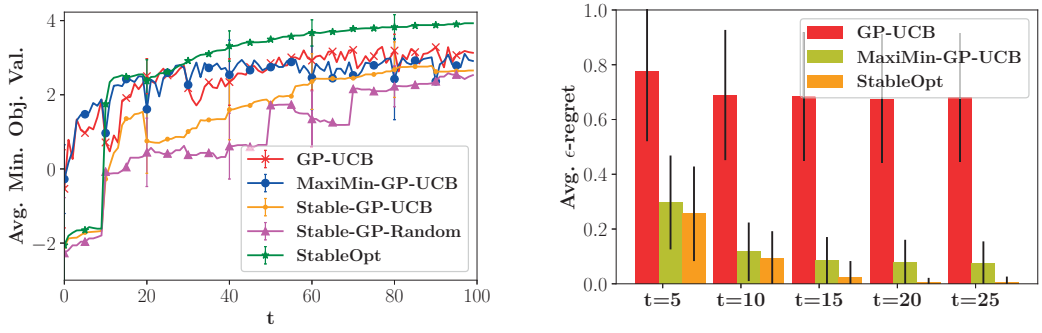


Figure 4.5: Robust robot pushing experiment (Left) and MovieLens-100K experiment (Right)

We use the Matérn-5/2 ARD kernel, setting its hyperparameters by maximizing the likelihood on a second (smaller) available dataset. The parameters ϵ_D and ϵ_L are set to 1.0 and 100.0, respectively. The stability requirement changes the global maximum and its location, as can be observed in Figure 4.4. The number of sampling rounds is $T = 120$, and each algorithm is initialized with the same 10 random data points and corresponding observations. The performance is averaged over 100 different runs, where every run corresponds to a different random initialization. In this experiment, STABLE-GP-UCB achieves the smallest ϵ-regret in the early rounds, while in the later rounds STABLEOPT is the only method that reports a near-optimal ϵ-stable point.

Robust robot pushing. We consider the deterministic version of the robot pushing objective from [WJ17], with publicly available code.[4] The goal is to find a good pre-image for pushing an object to a target location. The 3-dimensional function takes as input the robot location (r_x, r_y) and pushing duration r_t, and outputs $f(r_x, r_y, r_t) = 5 - d_{\text{end}}$, where d_{end} is the distance from the pushed object to the target location. Here, D is continuous: $r_x, r_y \in [-5, 5]$ and $r_t \in [1, 30]$.

We consider a twist on this problem in which there is uncertainty regarding the precise target, so one seeks a set of input parameters that is robust against a number of different potential locations. In the simplest case, the number of such locations is finite, meaning we can model this problem as $\boldsymbol{r} \in \arg\max_{\boldsymbol{r} \in D} \min_{i \in [m]} f_i(\boldsymbol{r})$, where each f_i corresponds to a different target location, and $[m] = \{1, \ldots, m\}$. This is a special case of (4.21) with a finite set Θ of cardinality m.

In our experiment, we use $m = 2$. Our goal is to find an input $\boldsymbol{r}$ that is robust against two different target locations. The first one is uniform over the domain, and the second is uniform over the ℓ_1-ball centered at the first target location with radius $r = 2.0$. We initialize each algorithm with one random sample from each f_i. We run each method for $T = 100$, and for a reported point $\boldsymbol{r}_t$ at time t, we compare the methods in terms of the robust objective $\min_{i \in [m]} f_i(\boldsymbol{r}_t)$. We perform a fully Bayesian treatment of the hyperparameters, sampling every 10 rounds[5] as in [HLHG14, WJ17]. We average over 30 random pairs of $\{f_1, f_2\}$ and report the results in Fig. 4.5. STABLEOPT noticeably outperforms its competitors except in some of the early rounds.

[4] https://github.com/zi-w/Max-value-Entropy-Search

[5] We note that the apparent discontinuities in certain curves are a result of the hyperparmeter re-estimation.

Group movie recommendation. Our goal in this task is to recommend a group of movies to a user such that *every* movie in the group is to their liking. We use the MovieLens-100K dataset, which consists of 1682 movies and 943 users. The data takes the form of an incomplete matrix $\mathbf{R}$ of ratings, where $R_{i,j}$ is the rating of movie i given by the user j. To impute the missing rating values, we apply non-negative matrix factorization with $p = 15$ latent factors. This produces a feature vector for each movie $\boldsymbol{m}_i \in \mathbb{R}^p$ and user $\boldsymbol{u}_j \in \mathbb{R}^p$. We use 10% of the user data for training, in which we fit a Gaussian distribution $P(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. For a given user $\boldsymbol{u}_j$ in the test set, $P(\boldsymbol{u})$ is considered to be a prior, and the objective is given by $f_j(\boldsymbol{m}_i) = \boldsymbol{m}_i^T \boldsymbol{u}_j$, corresponding to a GP with a linear kernel. We cluster the movie feature vectors into $k = 80$ groups, i.e., $\mathcal{G} = \{G_1, \ldots, G_k\}$, via the k-means algorithm. Similarly to (4.24), the robust optimization problem for a given user j is:

$$\max_{G \in \mathcal{G}} g_j(G), \qquad g_j(G) = \min_{\boldsymbol{m}_i \in G} f_j(\boldsymbol{m}_i). \tag{4.27}$$

That is, for the user with feature vector $\boldsymbol{u}_j$, our goal is to find the group of movies to recommend such that the entire collection of movies is robust with respect to the user's preferences.

In this experiment, we compare StableOpt against GP-UCB and MaxiMin-GP-UCB. We report the ϵ-regret given by $g_j(G^*) - g_j(G^{(t)})$ where G^* is the maximizer of (4.27), and $G^{(t)}$ is the reported group after time t. Since we are reporting groups rather than points, the baselines require slight modifications: At time t, GP-UCB selects the movie $\boldsymbol{m}_t$ (i.e., asks for the user's rating of it) and reports the group $G^{(t)}$ to which $\boldsymbol{m}_t$ belongs. MaxiMin-GP-UCB reports $G^{(t)} \in \arg\max_{G \in \mathcal{G}} \min_{\boldsymbol{m} \in G} \mathrm{ucb}_{t-1}(\boldsymbol{m})$ and then selects $\boldsymbol{m}_t \in \arg\min_{\boldsymbol{m} \in G^{(t)}} \mathrm{ucb}_{t-1}(\boldsymbol{m})$. Finally, StableOpt reports a group in the same way as MaxiMin-GP-UCB, but selects $\boldsymbol{m}_t$ analogously to (4.24). In Figure 4.5, we show the average ϵ-regret, where the average is taken over 500 different test users. In this experiment, the average ϵ-regret decreases rapidly after only a small number of rounds. Among the three methods, StableOpt is the only one that finds the optimal movie group.

4.A Details on Variations from Section 4.3

We claim that the StableOpt variations and theoretical results outlined in Section 4.3 are in fact special cases of Algorithm 7 and Theorem 4.2.1, despite being seemingly quite different. The idea behind this claim is that Algorithm 7 and Theorem 4.2.1 allow for the "distance" function $d(\cdot, \cdot)$ to be completely arbitrary, so we may choose it in rather creative/unconventional ways.

In more detail, we have the following:

- For the unknown parameter setting $\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\theta} \in \Theta} f(\boldsymbol{x}, \boldsymbol{\theta})$, we replace $\boldsymbol{x}$ in the original setting by the concatenated input $(\boldsymbol{x}, \boldsymbol{\theta})$, and set

  $$d((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}')) = \|\boldsymbol{x} - \boldsymbol{x}'\|_2.$$

  If we then set $\epsilon = 0$, we find that the input $\boldsymbol{x}$ experiences no perturbation, whereas $\boldsymbol{\theta}$ may be perturbed arbitrarily, thereby reducing (4.3) to

  $$\max_{\boldsymbol{x} \in D} \min_{\boldsymbol{\theta} \in \Theta} f(\boldsymbol{x}, \boldsymbol{\theta})$$

  as desired.

- For the robust estimation setting, we again use the concatenated input $(\boldsymbol{x}, \boldsymbol{\theta})$. To avoid overloading notation, we let $d_0(\boldsymbol{\theta}, \boldsymbol{\theta}')$ denote the distance function (applied to $\boldsymbol{\theta}$ alone) adopted for this case in Section 4.3. We set

  $$d((\boldsymbol{x}, \boldsymbol{\theta}), (\boldsymbol{x}', \boldsymbol{\theta}')) = \begin{cases} d_0(\boldsymbol{\theta}, \boldsymbol{\theta}') & \boldsymbol{x} = \boldsymbol{x}' \\ \infty & \boldsymbol{x} \neq \boldsymbol{x}'. \end{cases}$$

  Due to the second case, the input $\boldsymbol{x}$ experiences no perturbation, since doing so would violate the distance constraint of ϵ. We are left with $\boldsymbol{x} = \boldsymbol{x}'$ and $d_0(\boldsymbol{\theta}, \boldsymbol{\theta}') \leq \epsilon$, as required.

- For the grouped setting $\max_{G \in \mathcal{G}} \min_{\boldsymbol{x} \in G} f(\boldsymbol{x})$, we adopt the function

  $$d(\boldsymbol{x}, \boldsymbol{x}') = \mathbf{1}\{\boldsymbol{x} \text{ and } \boldsymbol{x}' \text{ are in different groups}\},$$

  and set $\epsilon = 0$. Considering the formulation in (4.3), we find that any two inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$ yield the same ϵ-stable objective function, and hence, reporting a point $\boldsymbol{x}$ is equivalent to reporting its group G. As a result, (4.3) reduces to the desired formulation

  $$\max_{G \in \mathcal{G}} \min_{\boldsymbol{x} \in G} f(\boldsymbol{x}).$$

The variations of StableOpt in (4.22)–(4.24), as well as the corresponding theoretical results outlined in Section 4.3, follow immediately by substituting the respective choices of $d(\cdot, \cdot)$ and ϵ above into Algorithm 7 and Theorem 4.2.1. Note that in the first two examples, the definition of γ_t in (2.10) is modified to take the maximum over not only $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t$, but also $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_t$.

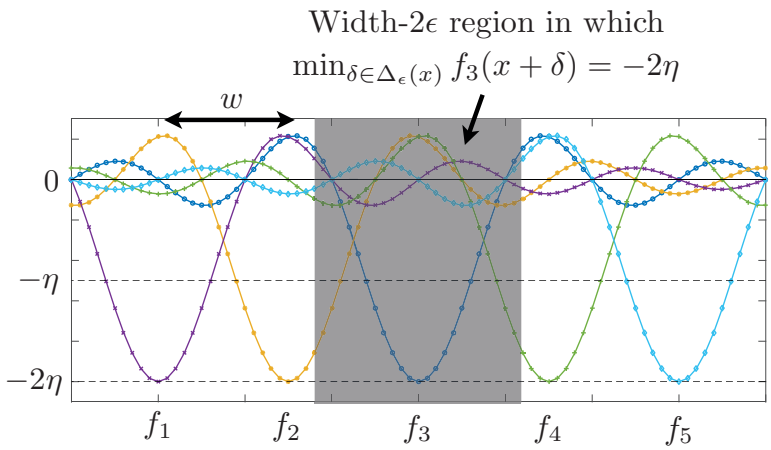


Figure 4.6: Illustration of functions $f_1, \ldots, f_5$ equal to a common function shifted by various multiples of a given parameter w. In the ϵ-stable setting, there is a wide region (shown in gray for the dark blue curve f_3) within which the perturbed function value equals -2η.

4.B Proofs

4.B.1 Lower Bound (Proof of Theorem 4.2.2)

Our lower bounding analysis builds heavily on that of the non-robust optimization setting with $f \in \mathcal{F}_k(B)$ studied in [SBC17], but with important differences. Roughly speaking, the analysis of [SBC17] is based on the difficulty of finding a very narrow "bump" of height 2η in a function whose values are mostly close to zero. In the ϵ-stable setting, however, even the points around such a bump will be adversarially perturbed to another point whose function value is nearly zero. Hence, all points are essentially equally bad.

To overcome this challenge, we consider the reverse scenario: Most of the function values are still nearly zero, but there exists a narrow *valley* of depth -2η. This means that every point within an ϵ-ball around the function minimizer will be perturbed to the point with value -2η. Hence, a constant fraction of the volume is still 2η-suboptimal, and it is impossible to avoid this region with high probability unless the time horizon T is sufficiently large. An illustration is given in Figure 4.6, with further details below. We now proceed with the formal proof.

Preliminaries

Recall that we are considering an arbitrary given (deterministic) Gaussian process optimization algorithm. More precisely, such an algorithm consists of a sequence of decision functions that return a sampling location $\boldsymbol{x}_t$ based on $y_1, \ldots, y_{t-1}$, and an additional decision function that reports the final point $\boldsymbol{x}^{(T)}$ based on $y_1, \ldots, y_T$. The points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}$ (or $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$) do not need to be treated as additional inputs to these functions, since $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1})$ is a deterministic function of $(y_1, \ldots, y_{t-1})$.

We first review several useful results and techniques from [SBC17]:

- We lower bound the worst-case ϵ-regret within $\mathcal{F}_k(B)$ by the ϵ-regret averaged over a suitably-designed finite collection $\{f_1, \ldots, f_M\} \subset \mathcal{F}_k(B)$ of size M.

- We choose each $f_m(\boldsymbol{x})$ to be a shifted version of a common function $g(\boldsymbol{x})$ on $\mathbb{R}^p$. Specifically, each $f_m(\boldsymbol{x})$ is obtained by shifting $g(\boldsymbol{x})$ by a different amount, and then cropping to $D = [0,1]^p$. For our purposes, we require $g(\boldsymbol{x})$ to satisfy the following properties:

  1. The RKHS norm in $\mathbb{R}^p$ is bounded, $\|g\|_k \leq B$;
  2. We have (i) $g(\boldsymbol{x}) \in [-2\eta, 2\eta]$ with minimum value $g(0) = -2\eta$, and (ii) there is a "width" w such that $g(\boldsymbol{x}) > -\eta$ for all $\|\boldsymbol{x}\|_\infty \geq w$;
  3. There are absolute constants $h_0 > 0$ and $\zeta > 0$ such that $g(\boldsymbol{x}) = \frac{2\eta}{h_0} h\big(\frac{\boldsymbol{x}\zeta}{w}\big)$ for some function $h(\boldsymbol{z})$ that decays faster than any finite power of $\|\boldsymbol{z}\|_2^{-1}$ as $\|\boldsymbol{z}\|_2 \to \infty$.

  Letting $g(\boldsymbol{x})$ be such a function, we construct the M functions by shifting $g(\boldsymbol{x})$ so that each $f_m(\boldsymbol{x})$ is centered on a unique point in a uniform grid, with points separated by w in each dimension. Since $D = [0,1]^p$, one can construct

$$M = \left\lfloor \left(\frac{1}{w}\right)^p \right\rfloor \tag{4.28}$$

  such functions. We will use this construction with $w \ll 1$, so that there is no risk of having $M = 0$, and in fact M can be assumed larger than any desired absolute constant.

- It is shown in [SBC17] that the above properties[6] can be achieved with

$$M = \left\lfloor \left(\frac{r\sqrt{\log \frac{B(2\pi l^2)^{p/4} h(0)}{2\eta}}}{\zeta \pi l} \right)^p \right\rfloor. \tag{4.29}$$

  in the case of the SE kernel, and with

$$M = \left\lfloor \left(\frac{B c_3}{\eta}\right)^{p/\nu} \right\rfloor, \tag{4.30}$$

  in the case of the Matérn kernel, where

$$c_3 := \left(\frac{r}{\zeta}\right)^\nu \cdot \left(\frac{c_2^{-1/2}}{2(8\pi^2)^{(\nu+p/2)/2}} \right), \tag{4.31}$$

  and where $c_2 > 0$ is an absolute constants. Note that these values of M amount to choosing w in (4.28), and the assumption of sufficiently small $\frac{\eta}{B}$ in the theorem statement ensures that $M \gg 1$ (or equivalently $w \ll 1$) as stated above.

[6] Here $g(\boldsymbol{x})$ plays the role of $-g(\boldsymbol{x})$ in [SBC17] due to the discussion at the start of this appendix, but otherwise the construction is identical.

- Property 2 above ensures that the "robust" function value $\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x})$ equals -2η for any $\boldsymbol{x}$ whose ϵ-neighborhood includes the minimizer $\boldsymbol{x}_{\min}$ of f, while being $-\eta$ or higher for any input whose entire ϵ-neighborhood is separated from $\boldsymbol{x}_{\min}$ by at least w. For $w \ll 1$ and $\epsilon < 0.5$, a point of the latter type is guaranteed to exist, which implies

$$r_\epsilon(\boldsymbol{x}) \geq \eta \tag{4.32}$$

  for any $\boldsymbol{x}$ whose ϵ-neighborhood includes $\boldsymbol{x}_{\min}$.

In addition, we introduce the following notation, also used in [SBC17]:

- The probability density function of the output sequence $\boldsymbol{y} = (y_1, \ldots, y_T)$ when the underlying function is f_m is denoted by $P_m(\boldsymbol{y})$. We also define $f_0(\boldsymbol{x}) = 0$ to be the zero function, and define $P_0(\boldsymbol{y})$ analogously for the case that the optimization algorithm is run on f_0. Expectations and probabilities (with respect to the noisy observations) are similarly written as $\mathbb{E}_m$, $\mathbb{P}_m$, $\mathbb{E}_0$, and $\mathbb{P}_0$ when the underlying function is f_m or f_0. On the other hand, in the absence of a subscript, $\mathbb{E}[\cdot]$ and $\mathbb{P}[\cdot]$ are taken with respect to the noisy observations *and* the random function f drawn uniformly from $\{f_1, \ldots, f_M\}$ (recall that we are lower bounding the worst case by this average).

- Let $\{\mathcal{R}_m\}_{m=1}^M$ be a partition of the domain into M regions according the above-mentioned uniform grid, with f_m taking its minimum value of -2η in the centre of $\mathcal{R}_m$. Let j_t be the index at time t such that $\boldsymbol{x}_t$ falls into $\mathcal{R}_{j_t}$; this can be thought of as a quantization of $\boldsymbol{x}_t$.

- Define the maximum (absolute) function value within a given region $\mathcal{R}_j$ as

$$\overline{v}_m^j := \max_{\boldsymbol{x} \in \mathcal{R}_j} |f_m(\boldsymbol{x})|, \tag{4.33}$$

  and the maximum KL divergence to P_0 within the region as

$$\overline{D}_m^j := \max_{\boldsymbol{x} \in \mathcal{R}_j} D(P_0(\cdot|\boldsymbol{x}) \| P_m(\cdot|\boldsymbol{x})), \tag{4.34}$$

  where $P_m(y|\boldsymbol{x})$ is the distribution of an observation y for a given selected point x under the function f_m, and similarly for $P_0(y|x)$.

- Let $N_j \in \{0, \ldots, T\}$ be a random variable representing the number of points from $\mathcal{R}_j$ that are selected throughout the T rounds.

Next, we present several useful lemmas. The following well-known change-of-measure result, which can be viewed as a form of Le Cam's method, has been used extensively in both discrete and continuous bandit problems.

Lemma 4.B.1. [ACBFS98, p. 27] *For any function $a(\boldsymbol{y})$ taking values in a bounded range $[0, A]$,*

we have

$$\left|\mathbb{E}_m[a(\boldsymbol{y})] - \mathbb{E}_0[a(\boldsymbol{y})]\right| \leq A\, d_{\mathrm{TV}}(P_0, P_m) \tag{4.35}$$

$$\leq A\, \sqrt{D(P_0 \| P_m)}, \tag{4.36}$$

where $d_{\mathrm{TV}}(P_0, P_m) = \frac{1}{2} \int_{\mathbb{R}^T} |P_0(\boldsymbol{y}) - P_m(\boldsymbol{y})|\, d\boldsymbol{y}$ *is the total variation distance.*

We briefly remark on some slight differences here compared to [ACBFS98, p. 27]. There, only $\mathbb{E}_m[a(\boldsymbol{y})] - \mathbb{E}_0[a(\boldsymbol{y})]$ is upper bounded in terms of $d_{\mathrm{TV}}(P_0, P_m)$, but one easily obtains the same upper bound on $\mathbb{E}_0[a(\boldsymbol{y})] - \mathbb{E}_m[a(\boldsymbol{y})]$ by interchanging the roles of P_0 and P_m. The step (4.36) follows from Pinsker's inequality, $d_{\mathrm{TV}}(P_0, P_m) \leq \sqrt{\frac{D(P_0\|P_m)}{2}}$, and by upper bounding $\frac{1}{\sqrt{2}} \leq 1$ to ease the notation.

The following result simplifies the divergence term in (4.36).

Lemma 4.B.2. [SBC17, Eq. (44)] *Under the preceding definitions, we have*

$$D(P_0 \| P_m) \leq \sum_{j=1}^{M} \mathbb{E}_0[N_j] \overline{D}_m^j. \tag{4.37}$$

The following well-known property gives a formula for the KL divergence between two Gaussians.

Lemma 4.B.3. [SBC17, Eq. (36)] *For* P_1 *and* P_2 *being Gaussian with means* (μ_1, μ_2) *and a common variance* σ^2*, we have*

$$D(P_1 \| P_2) = \frac{(\mu_1 - \mu_2)^2}{2\sigma^2}. \tag{4.38}$$

Finally, we have the following technical result regarding the "needle-in-haystack" type function constructed above.

Lemma 4.B.4. [SBC17, Lemma 7] *The functions* $\{f_m\}_{m=1}^M$ *corresponding to* (4.29)–(4.30) *are such that the quantities* $\overline{v}_m^j$ *satisfy* $\sum_{m=1}^M (\overline{v}_m^j)^2 = O(\eta^2)$ *for all* j*.*

Analysis of the average ϵ-stable regret

Let $J_{\mathrm{bad}}(m)$ be the set of j such that all $\boldsymbol{x} \in \mathcal{R}_j$ yield $\min_{\boldsymbol{\delta} \in \Delta_\epsilon(\boldsymbol{x})} f(\boldsymbol{x} + \boldsymbol{\delta}) = -2\eta$ when the true function is f_m, and define $\mathcal{R}_{\mathrm{bad}}(m) = \cup_{j \in J_{\mathrm{bad}}(m)} \mathcal{R}_j$. By the ϵ-regret lower bound in (4.32),

we have

$$\mathbb{E}_m[r_\epsilon(\boldsymbol{x}^{(T)})] \geq \eta \mathbb{P}_m[\boldsymbol{x}^{(T)} \in \mathcal{R}_{\text{bad}}(m)] \tag{4.39}$$

$$\geq \eta\left(\mathbb{P}_0[\boldsymbol{x}^{(T)} \in \mathcal{R}_{\text{bad}}(m)] - \sqrt{D(P_0 \| P_m)}\right) \tag{4.40}$$

$$\geq \eta\left(\mathbb{P}_0[\boldsymbol{x}^{(T)} \in \mathcal{R}_{\text{bad}}(m)] - \sqrt{\sum_{j=1}^{M} \mathbb{E}_0[N_j]\overline{D}_m^j}\right), \tag{4.41}$$

where (4.40) follows from Lemma 4.B.1 with $a(\boldsymbol{y}) = \mathbf{1}\{x^{(T)} \in \mathcal{R}_{\text{bad}}(m)\}$ and $A = 1$ (recall that $\boldsymbol{x}^{(T)}$ is a function of $\boldsymbol{y} = (y_1, \ldots, y_T)$), and (4.41) follows from Lemma 4.B.2. Averaging over m uniform in $\{1, \ldots, M\}$, we obtain

$$\mathbb{E}[r_\epsilon(\boldsymbol{x}^{(T)})] \geq \eta\left(\frac{1}{M}\sum_{m=1}^{M} \mathbb{P}_0[\boldsymbol{x}^{(T)} \in \mathcal{R}_{\text{bad}}(m)] - \frac{1}{M}\sum_{m=1}^{M}\sqrt{\sum_{j=1}^{M} \mathbb{E}_0[N_j]\overline{D}_m^j}\right). \tag{4.42}$$

We proceed by bounding the two terms separately.

- We first claim that
$$\frac{1}{M}\sum_{m=1}^{M} \mathbb{P}_0[\boldsymbol{x}^{(T)} \in \mathcal{R}_{\text{bad}}(m)] \geq C_1 \tag{4.43}$$
for some $C_1 > 0$. To show this, it suffices to prove that any given $\boldsymbol{x}^{(T)} \in D$ is in at least a constant fraction of the $\mathcal{R}_{\text{bad}}(m)$ regions, of which there are M. This follows from the fact that the ϵ-ball centered at
$$\boldsymbol{x}_{m,\min} = \arg\min_{\boldsymbol{x} \in D} f_m(\boldsymbol{x})$$
takes up a constant fraction of the volume of D, where the constant depends on both the stability parameter ϵ and the dimension p. A small caveat is that because the definition of $\mathcal{R}_{\text{bad}}$ insists that the *every* point in the region $\mathcal{R}_j$ is within distance ϵ of $\boldsymbol{x}_{m,\min}$, the left-hand side of (4.43) may be slightly below the relevant ratio of volumes above. However, since Theorem 4.2.2 assumes that $\frac{\epsilon}{B}$ is sufficiently small, the choices of M in (4.29) and (4.30) ensure that M is sufficiently large for this "quantization" effect to be negligible.

- For the second term in (4.42), we claim that
$$\frac{1}{M}\sum_{m=1}^{M}\sqrt{\sum_{j=1}^{M} \mathbb{E}_0[N_j]\overline{D}_m^j} \leq C_2\frac{\eta}{\sigma}\sqrt{\frac{T}{M}} \tag{4.44}$$

for some $C_2 > 0$. To see this, we write

$$\begin{aligned}
&\frac{1}{M}\sum_{m=1}^{M}\sqrt{\sum_{j=1}^{M}\mathbb{E}_0[N_j]\overline{D}_m^j} \\
&\quad= O\Big(\frac{1}{\sigma}\Big)\cdot\frac{1}{M}\sum_{m=1}^{M}\sqrt{\sum_{j=1}^{M}\mathbb{E}_0[N_j](\overline{v}_m^j)^2} && (4.45)\\
&\quad\le O\Big(\frac{1}{\sigma}\Big)\cdot\sqrt{\frac{1}{M}\sum_{m=1}^{M}\sum_{j=1}^{M}\mathbb{E}_0[N_j](\overline{v}_m^j)^2} && (4.46)\\
&\quad= O\Big(\frac{1}{\sigma}\Big)\cdot\sqrt{\frac{1}{M}\sum_{j=1}^{M}\mathbb{E}_0[N_j]\Big(\sum_{m=1}^{M}(\overline{v}_m^j)^2\Big)} && (4.47)\\
&\quad= O\Big(\frac{\eta}{\sqrt{M}\sigma}\Big)\cdot\sqrt{\sum_{j=1}^{M}\mathbb{E}_0[N_j]} && (4.48)\\
&\quad= O\Big(\frac{\sqrt{T}\eta}{\sqrt{M}\sigma}\Big), && (4.49)
\end{aligned}$$

where (4.45) follows since the divergence $D(P_0(\cdot|\boldsymbol{x})\|P_m(\cdot|\boldsymbol{x}))$ associated with a point $\boldsymbol{x}$ having value $v(\boldsymbol{x})$ is $\frac{v(\boldsymbol{x})^2}{2\sigma^2}$ (*cf.*, (4.38)), (4.46) follows from Jensen's inequality, (4.48) follows from Lemma 4.B.4, and (4.49) follows from $\sum_j N_j = T$.

Substituting (4.43) and (4.44) into (4.42), we obtain

$$\mathbb{E}[r_\epsilon(\boldsymbol{x}^{(T)})] \ge \eta\Big(C_1 - C_2\frac{\eta}{\sigma}\sqrt{\frac{T}{M}}\Big), \tag{4.50}$$

which implies that the regret is lower bounded by $\Omega(\eta)$ unless $T = \Omega\big(\frac{M\sigma^2}{\eta^2}\big)$. Substituting M from (4.29) and (4.30), we deduce that the conditions on T in the theorem statement are necessary to achieve average regret $\mathbb{E}[r_\epsilon(\boldsymbol{x}^{(T)})] = O(\eta)$ with a sufficiently small implied constant.

From average to high-probability regret

Recall that we are considering functions whose values lie in the range $[-2\eta, 2\eta]$, implying that $r_\epsilon(\boldsymbol{x}^{(T)}) \le 4\eta$. Letting T_η be the lower bound on T derived above for achieving average regret $O(\eta)$ (i.e., we have $\mathbb{E}[r_\epsilon^{(T_\eta)}] = \Omega(\eta)$), it follows from the reverse Markov inequality (i.e., Markov's inequality applied to the random variable $4\epsilon - r_\epsilon^{(T_\eta)}$) that

$$\mathbb{P}[r_\epsilon(\boldsymbol{x}^{(T_\eta)}) \ge c\eta] \ge \frac{\Omega(\eta) - c\eta}{4\eta - c\eta} \tag{4.51}$$

for any $c > 0$ sufficiently small for the numerator and denominator to be positive. The right-hand side is lower bounded by a constant for any such c, implying that the probability of achieving ϵ-regret at most $c\eta$ cannot be arbitrarily close to one. By renaming η as η', it follows that in order to achieve some target ϵ-stable regret η' with probability sufficiently close to one, a lower bound of the same form as the average regret bound holds. In other words, the conditions on T in the theorem statement remain necessary also for the high-probability regret.

We emphasize that Theorem 4.2.2 concerns the high-probability regret when "high probability" means *sufficiently close to one* as a function of ϵ, p, and the kernel parameters (but still constant with respect to T and η). We do not claim a lower bound under any particular *given* success probability (e.g., η-optimality with probability at least $\frac{3}{4}$).

5 Gaussian Process Optimization with Time-Varying Reward Function

In the previous chapters, we considered the problems in which the objective function was time-invariant, i.e., the unknown reward function was *static* and did not vary with time. However, in many practical applications, the function to be optimized is not static. In the time-varying setting, the performance of standard algorithms may deteriorate, since these continue to treat stale data as being equally important as fresh data. In this chapter, we consider the Bayesian optimization with bandit feedback, adopting a formulation that allows for the reward function to *vary with time*.

This chapter is based on the joint work with Jonathan Scarlett and Volkan Cevher [BSC16].

5.1 Introduction

In the previous chapters, we discussed the theory and methods for Bayesian/bandit optimization problems, where one seeks to sequentially select a sequence of points to optimize an unknown reward function from noisy samples [SKKS10, BCB12, LR85]. Such problems have numerous applications, including sensors networks, recommender systems, and finance. A key challenge is to rigorously trade-off between *exploration*, i.e., learning the behavior of the function across the whole domain, and *exploitation*, i.e., selecting points that have previously given high rewards.

In the vast majority of applications, the function to be optimized is not static, but varies with time: In sensor networks, measured quantities such as temperature undergo fluctuations; in recommender systems, the users' preferences may change according to external factors; similarly, financial markets are highly dynamic. In such cases, the performance of standard algorithms may deteriorate, since these treat old data as being equally important as new data. The development of algorithms and theory to handle time variations is therefore crucial. In this chapter, we take a novel approach to handling time variations, modeling the reward function as a Gaussian process (GP) that varies according to a simple Markov model. We propose algorithms based on the upper confidence bound strategy with the ability to forget about old data. Furthermore, we analyze their theoretical performance and evaluate them on various time-varying datasets.

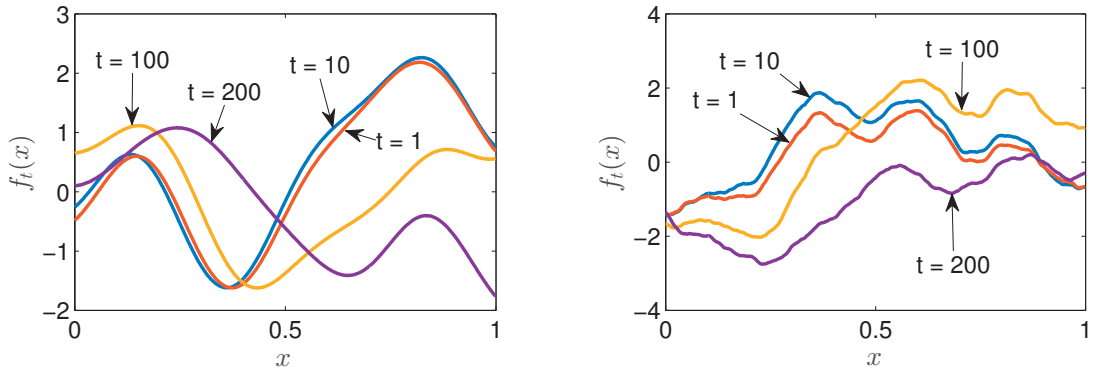


Figure 5.1: Examples of GP functions when $\epsilon = 0.01$: (Left) SE kernel ($l = 0.2$); (Right) Matérn kernel ($\nu = 1.5$). Note that the location of the maximum changes significantly at distant times.

5.1.1 Problem Statement

We seek to sequentially optimize an unknown reward function f_t over a compact, convex subset $D \subset \mathbb{R}^p$.[1] At time t, we can interact with f_t only by querying at some point $\boldsymbol{x}_t \in D$, after which we observe a noisy observation $y_t = f_t(\boldsymbol{x}_t) + z_t$, where $z_t \sim \mathcal{N}(0, \sigma^2)$. We assume that the noise realizations at different time instants are independent. The goal is to maximize the reward via a suitable trade-off between exploration and exploitation. This problem is ill-posed for arbitrary reward functions even in the time-invariant setting, and it is thus necessary to introduce suitable smoothness assumptions. We model the reward function as a sample from a Gaussian process $f \sim \mathrm{GP}(0, k)$, where its smoothness is dictated by the choice of kernel function k.

Let $k : D \times D \to \mathbb{R}_+$ be a kernel function, and let $\mathrm{GP}(\mu, k)$ be a Gaussian process [RW06] with mean $\mu \in \mathbb{R}^p$ and kernel k. As in the previous chapters, we assume bounded variance: $\forall \boldsymbol{x} \in D, k(\boldsymbol{x}, \boldsymbol{x}) \leq 1$. Two common kernels are squared exponential and Matérn given in (2.2).

Letting $g_1, g_2, \ldots$ be independent random functions on D with $g_i \sim \mathrm{GP}(0, k)$, the reward functions are modeled as follows:

$$f_1(\boldsymbol{x}) = g_1(\boldsymbol{x}) \tag{5.1}$$

$$f_{t+1}(\boldsymbol{x}) = \sqrt{1-\epsilon}\, f_t(\boldsymbol{x}) + \sqrt{\epsilon}\, g_{t+1}(\boldsymbol{x}) \quad \forall t \geq 2, \tag{5.2}$$

where $\epsilon \in [0, 1]$ quantifies how much the function changes after every time step. If $\epsilon = 0$ then we recover the standard time-invariant model [SKKS10], whereas if $\epsilon = 1$ then the reward functions are independent between time steps. Importantly, for any choice of ϵ we have for all t that $f_t \sim \mathrm{GP}(0, k)$. See Figure 5.1 for an illustration.

From a practical perspective, this model has the desirable property of only having one additional hyperparameter ϵ compared to the standard GP model, thus facilitating the learning process.

[1] Finite domains were also handled in the time-invariant setting (see Theorem 2.2.1), and all of our upper bounds have counterparts for such scenarios that are in fact simpler to obtain compared to the compact case.

It serves as a suitable model for reward functions that vary at a steady rate, though we will see numerically in Section 5.4 that the resulting algorithms are also effective more generally.

As noted in *regression* studies in [VVLGS12, VVSLG12], our model is equivalent to a *spatiotemporal* kernel model with temporal kernel $(1-\epsilon)^{|t_1-t_2|/2}$. We expect our techniques to apply similarly to other temporal kernels, particularly *stationary* kernel functions that depend only on the time difference $|t_1 - t_2|$, but we focus on (5.1)–(5.2) for concreteness. Spatiotemporal kernels can also be considered in the contextual bandit setting [KO11], but to our knowledge, no regret bounds have been given that explicitly characterize the dependence on the function's rate of variation, as is done in our main result.

Let $\boldsymbol{x}_t^*$ denote a maximizer of f_t at time t, i.e., $\boldsymbol{x}_t^* = \arg\max_{\boldsymbol{x}} f_t(\boldsymbol{x})$, and suppose that our choice at time t is $\boldsymbol{x}_t$. Then the *instantaneous regret* we incur at time t is $r_t = f_t(\boldsymbol{x}_t^*) - f_t(\boldsymbol{x}_t)$. We are interested in minimizing the *cumulative regret* $R_T = \sum_{t=1}^{T} r_t$.

These definitions coincide with those for the time-invariant setting (*cf.* Eq. (2.6)) when $\epsilon = 0$. Note that we do not aim to merely compete with fixed strategies, but instead to track the maximum of f_t for all t. In our setting, a notion of regret based on competing with a fixed strategy would typically lead to a negative cumulative regret. In other words, *all fixed strategies perform poorly*.

In time-invariant scenarios, as well as several time-varying ones, algorithms are typically designed to achieve *sublinear regret*. In our setting, we will show that for *fixed* ϵ, the regret R_T must in fact be $\Omega(T)$ (in Theorem 5.3.1). Intuitively, this is because if the function changes significantly at each time step, one cannot expect to track its maximum to arbitrary precision. However, we emphasize that what is really of interest is the *joint* dependence of R_T on T and ϵ, and we thus seek regret bounds of the form $O^*(T\psi(\epsilon))$ for some function $\psi(\epsilon)$ that vanishes as $\epsilon \to 0$. Our approach is analogous to Slivkins and Upfal [SU08], who considered another time-varying setting with unavoidable $\Omega(T)$ regret for any fixed function variation parameter, and focused on the behavior in the implied constant in the limit as that parameter vanishes.

For the squared exponential and Matérn kernels, we obtain regret bounds of the form $O^*(T\epsilon^\alpha)$ for some $\alpha > 0$ (*cf.*, Corollary 5.3.1), which can be viewed as being sublinear whenever $\epsilon = O(T^{-c})$ for some $c > 0$. We observe that when $c < 1$, the correlation between $f_1(\boldsymbol{x})$ and $f_T(\boldsymbol{x})$ is negligible, meaning that the corresponding maximum may (and typically will) change drastically over the duration of the time horizon, e.g., see Figure 5.1.

Limitations of GP-UCB

As in the previous chapters, we recall GP-UCB (Section 2.2) that works in the time-invariant setting. At every round the selected point maximizes a function of the form $\mu_{t-1}(\boldsymbol{x}) + \sqrt{\beta_t}\sigma_{t-1}(\boldsymbol{x})$. Given the previous samples $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$ and corresponding observations $\boldsymbol{y}_t = [y_1, \ldots, y_t]$, μ_{t-1} and σ_{t-1} of the time-invariant GP $f(\boldsymbol{x})$ are given in (2.4). Intuitively, one seeks points with a high mean to favor exploitation, but with a high standard deviation to favor exploration.

In the time-invariant setting, GP-UCB is known to achieve sublinear regret under mild assumptions (see Theorem 2.2.1). The problem with using it in our setting is that it treats all of the samples as being equally important, whereas according to our model, the samples become increasingly stale with time. In Section 5.2, we present our algorithms that account for this fact.

5.1.2 Related Work

In the time-invariant setting, a wide variety of works have made use of upper confidence bound (UCB) algorithms, where the selected point maximizes a linear combination of the posterior mean and standard deviation. In particular, Srivinas *et al.* [SKKS10] provided regret bounds for the GP-UCB algorithm that we outline in Theorem 2.2.1, and several extensions were given subsequently, including the contextual [KO11] and high-dimensional [DKC13, SSZA14, WZH$^+$13] settings.

While the study of time-varying models is limited in the GP setting, several such models have been considered in the multi-armed bandit (MAB) setting. Perhaps the most well-known one is the adversarial setting [LR85, BCB12, BDKP15], where one typically seeks to compete with the best fixed strategy. Rewards modeled by Markov chains have been considered under the categories of *restless bandits* [Whi88, BNM00, OR12, SU08], where the reward for each arm changes at each time step, and *rested bandits* [TL12, LLZ13], where only the pulled arm changes.

Two further related works are those of [SU08], who studied a MAB problem with varying rewards based on Brownian motion, and Besbes *et al.*[BGZ14], who considered a general MAB setting with time-varying rewards subject to a total budget in the amount of change allowed. Both [SU08] and [BGZ14] demonstrate the need for a *forgetting-remembering trade-off* arising from the fact that using the information from more samples may decrease the variance of the function estimates, while older information may be stale and hence misleading. Both papers present strategies in which the algorithm is reset at regular intervals in order to discard stale data. This is shown to be optimal in the worst case for the function class considered in [BGZ14], whereas in [SU08] it is shown that simple resetting strategies can be suboptimal in more specific scenarios, and alternative approaches are presented.

In contrast to GP-based settings such as ours, the setups of [SU08] and [BGZ14] consider finite action spaces, and assume independence between the rewards associated with different arms. Thus, observing the reward of one arm does not reveal any information about the other ones, and the algorithms are designed to exploit temporal correlations, but not spatial correlations.

5.1.3 Contributions

The main contributions of this chapter are:

- In Section 5.2, we introduce two algorithms for addressing the fundamental trade-offs that are inherent in the problem formulation: (i) trading off exploration with exploitation;

(ii) differentiating between stale and fresh data in the presence of time variations; (iii) exploiting spatial and temporal correlations present in the reward function.

- Our main results present regret bounds (Section 5.3), first for general kernels and then for the SE and Matérn kernels, that explicitly characterize the trade-off between the time horizon and the rate at which the function varies. Their proofs require novel techniques to handle difficulties arising from the time variations, such as the maximum function value and its location changing drastically throughout the duration of the time horizon.
- In Section 5.3, we provide an algorithm-independent lower bound on the cumulative regret.
- We demonstrate the utility of our model and algorithms on both synthetic and real-world data in Section 5.4.

5.2 Algorithms for Time-Varying Rewards

In this section, we present two algorithms that simultaneously balance between exploration and exploitation as well as forgetting and remembering. We first introduce an algorithm R-GP-UCB that takes a conceptually simple approach to handling the forgetting-remembering trade-off, namely, running the GP-UCB algorithm within blocks of size N, and applying resetting at the start of each block. Some insight on how to choose N is given by our bounds in the following section. The pseudo-code is shown in Algorithm 8.

Algorithm 8 GP-UCB with Resetting (R-GP-UCB) [BSC16]

Input: Domain D, GP prior (μ_0, σ_0, k), block size N

for $t = 1, 2...$ **do**
  if $t \bmod N = 1$ **then**
    Reset $\mu_{t-1}(\boldsymbol{x}) = \mu_0(\boldsymbol{x})$ and $\sigma_{t-1}(\boldsymbol{x}) = \sigma_0(\boldsymbol{x})$ for each $\boldsymbol{x}$
  Choose $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in D} \mu_{t-1}(\boldsymbol{x}) + \sqrt{\beta_t}\sigma_{t-1}(\boldsymbol{x})$
  Sample $y_t = f_t(\boldsymbol{x}_t) + z_t$
  Perform Bayesian update as in (2.4), using only the samples $\{\boldsymbol{x}_t\}$ and $\{y_t\}$ obtained since the most recent reset, to obtain μ_t and σ_t

Our second algorithm, TV-GP-UCB, instead forgets in a "smooth" fashion, by using a posterior update rule obtained via the time-varying model (5.1)–(5.2). In analogy with (2.4), the mean and variance of f_t given the previous samples $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_t)$ and corresponding observations $\boldsymbol{y}_t = [y_1, \dots, y_t]$ are given by

$$\tilde{\mu}_{t+1}(\boldsymbol{x}) := \widetilde{\boldsymbol{k}}_t(\boldsymbol{x})^T \big(\widetilde{\boldsymbol{K}}_t + \sigma^2 \boldsymbol{I}_t\big)^{-1} \boldsymbol{y}_t \tag{5.3}$$

$$\tilde{\sigma}^2_{t+1}(\boldsymbol{x}, \boldsymbol{x}') := k(\boldsymbol{x}, \boldsymbol{x}) - \widetilde{\boldsymbol{k}}_t(\boldsymbol{x})^T \big(\widetilde{\boldsymbol{K}}_t + \sigma^2 \boldsymbol{I}_t\big)^{-1} \widetilde{\boldsymbol{k}}_t(\boldsymbol{x}), \tag{5.4}$$

where $\widetilde{\boldsymbol{K}}_t = \boldsymbol{K}_t \circ \boldsymbol{D}_t$ with $\boldsymbol{D}_t = \big[(1-\epsilon)^{|i-j|/2}\big]_{i,j=1}^T$, and $\widetilde{\boldsymbol{k}}_t(\boldsymbol{x}) = \boldsymbol{k}_t(\boldsymbol{x}) \circ \boldsymbol{d}_t$ where $\boldsymbol{d}_t = \big[(1-\epsilon)^{(T+1-i)/2}\big]_{i=1}^T$. Here $\circ$ is the Hadamard product, and $\boldsymbol{I}_k$ is the identity matrix.

The derivation of (5.3)–(5.4) is given in Appendix 5.A. Using these updates, the TV-GP-UCB algorithm is given in Algorithm 9. The idea is that the older a sample is, the smaller the value in the corresponding entries of $\boldsymbol{d}_t$ and $\boldsymbol{D}_t$ defined following (5.4), and hence the less it contributes to the final values of $\tilde{\mu}_t(\boldsymbol{x})$ and $\tilde{\sigma}_t(\boldsymbol{x})$. This algorithm can in fact be considered a special case of contextual GP-UCB [KO11] with a spatio-temporal kernel, but our analysis (Section 5.3) goes far beyond that of [KO11] in order to explicitly characterize the dependence on T and ϵ.

Algorithm 9 Time-Varying GP-UCB (TV-GP-UCB) [BSC16]

Input: Domain D, GP prior ($\tilde{\mu}_0$, $\tilde{\sigma}_0$, k) and parameter ϵ

1: **for** $t = 1, 2...$ **do**

2: Choose $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}\in D} \tilde{\mu}_{t-1}(\boldsymbol{x}) + \sqrt{\beta_t}\tilde{\sigma}_{t-1}(\boldsymbol{x})$

3: Sample $y_t = f_t(\boldsymbol{x}_t) + z_t$

4: Perform Bayesian update as in (5.3)–(5.4) to obtain $\tilde{\mu}_t$ and $\tilde{\sigma}_t$

As it is presented above, TV-GP-UCB has an identical computational complexity to GP-UCB, i.e. the complexity of the sequential Bayesian update is $O(T^2)$ [ORR$^+$08]. R-GP-UCB is less complex, since the matrix operations are on matrices of size N rather than the overall time horizon T. In practice, however, one could further modify TV-GP-UCB to improve the efficiency by occasionally resetting and/or discarding stale data [ORR$^+$08].

5.3 Time-varying Regret Bounds

In this section, we provide our main theoretical upper and lower bounds on the regret. We assume throughout this section that hyperparameters are known, i.e. both spatial kernel hyperparameters and ϵ; in the numerical section we will address real-world problems where these are unknown.

5.3.1 Preliminary Definitions and Results

Smoothness Assumptions: Each of our results below will assume that the kernel k is such that a (strict) subset of the following statements hold for some (a_i, b_i) and all $L \geq 0$:

$$\mathbb{P}\left[\sup_{\boldsymbol{x}\in D} \left|f(\boldsymbol{x})\right| > L\right] \leq a_0 e^{-(L/b_0)^2} \tag{5.5}$$

$$\mathbb{P}\left[\sup_{\boldsymbol{x}\in D} \left|\frac{\partial f}{\partial x^{(j)}}\right| > L\right] \leq a_1 e^{-(L/b_1)^2}, \quad j = 1, \ldots, p \tag{5.6}$$

$$\mathbb{P}\left[\sup_{\boldsymbol{x}\in D} \left|\frac{\partial^2 f}{\partial x^{(j_1)} \partial x^{(j_2)}}\right| > L\right] \leq a_2 e^{-(L/b_2)^2}, \quad j_1, j_2 = 1, \ldots, p, \tag{5.7}$$

where $f \sim \mathrm{GP}(0, k)$.

Assumption (5.5) is mild, since $f(\boldsymbol{x})$ is Gaussian and thus has exponential tails. Assumption (5.6) was used in [SKKS10], and ensures that the behavior of the GP is not too erratic. It is satisfied for SE, as well as the Matérn kernel with $\nu > 2$ [SKKS10], though for other kernels (e.g., Ornstein-Uhlenbeck) it can fail. Assumption (5.7) is used only for our lower bound; it is again satisfied by the SE kernel, as well as the Matérn kernel with $\nu > 4$.

Mutual Information: It was shown in Section 2.2.1 that a key quantity governing the regret bounds of GP-UCB in the time-invariant setting is the mutual information

$$I(\boldsymbol{f}_T; \boldsymbol{y}_T) = \frac{1}{2} \log \det \left(\boldsymbol{I}_T + \sigma^{-2} \boldsymbol{K}_T \right), \tag{5.8}$$

where

$$\boldsymbol{f}_T := (f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_T))$$

in the case of the time-invariant GP f. The corresponding maximum over any set of points $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_T)$ is given by

$$\gamma_T := \max_{\boldsymbol{x}_1, \dots, \boldsymbol{x}_T} I(\boldsymbol{f}_T; \boldsymbol{y}_T). \tag{5.9}$$

In our setting, the analogous quantities are as follows:

$$\tilde{I}(\boldsymbol{f}_T; \boldsymbol{y}_T) = \frac{1}{2} \log \det \left(\boldsymbol{I}_T + \sigma^{-2} \widetilde{\boldsymbol{K}}_T \right), \tag{5.10}$$

$$\tilde{\gamma}_T := \max_{\boldsymbol{x}_1, \dots, \boldsymbol{x}_T} \tilde{I}(\boldsymbol{f}_T; \boldsymbol{y}_T), \tag{5.11}$$

where

$$\boldsymbol{f}_T := \boldsymbol{f}_T(\boldsymbol{x}_T) = (f_1(\boldsymbol{x}_1), \dots, f_T(\boldsymbol{x}_T)).$$

While these take the same form as (5.8)–(5.9), they can behave significantly differently when $\epsilon > 0$. The time-varying versions are typically much higher due to the fact $\boldsymbol{f}_T$ represents the points of T different random functions, as opposed to a single function at T different points.

Algorithm-Independent Lower Bound: The following result gives an asymptotic lower bound for any bandit optimization algorithm under fairly mild assumptions, expressed in terms of the time horizon T and parameter ϵ.

Theorem 5.3.1. *Suppose that the kernel is such that $f \sim \mathrm{GP}(0, k)$ is almost surely twice continuously differentiable and satisfies* (5.6)–(5.7) *for some (a_1, b_1, a_2, b_2). Then, any GP bandit optimization algorithm incurs expected regret $\mathbb{E}[R_T] = \Omega(T\epsilon)$.*

The proof (in Appendix 5.C.4) reveals that this result holds true even in the *full information*[2] (as opposed to bandit) setting, and is based on the fact that at each time step, there is a non-zero probability that the maximum value and its location change by an amount proportional to ϵ. As discussed above, this lower bound motivates the study of the *joint* dependence on the regret of T and ϵ, and in particular, the highest possible constant α such that the regret behaves as $O^*(T\epsilon^\alpha)$.

[2] In the full information setting, we get to observe the entire function from the previous time step.

5.3.2 General Upper Bounds

We now present our general bounds on the algorithms from Section 5.2 whose proofs can be found in Appendix 5.C.1 and 5.C.2. The regret bounds are of a similar form, but we will shortly apply these to specific kernels and find that the bounds for TV-GP-UCB yield better scaling laws.

General Bounds: The following theorems provide regret bounds for R-GP-UCB and TV-GP-UCB, respectively. We will simplify them below to obtain scaling laws for specific kernels.

Theorem 5.3.2. *Let the domain $D \subset [0,r]^p$ be compact and convex, and suppose that the kernel is such that $f \sim \mathrm{GP}(0,k)$ is almost surely continuously differentiable and satisfies* (5.5)–(5.6) *for some (a_0, b_0, a_1, b_1). Fix $\delta \in (0,1)$, and set*

$$\beta_T = 2\log\frac{2\pi^2 T^2}{3\delta} + 2p\log\left(rpbT^2\sqrt{\log\frac{2pa\pi^2T^2}{3\delta}}\right). \tag{5.12}$$

Defining $C_1 = 8/\log(1+\sigma^{-2})$, R-GP-UCB satisfies the following after T time steps:

$$R_T \leq \sqrt{C_1 T\beta_T\left(\frac{T}{N}+1\right)\gamma_N} + 2 + T\psi_T(N,\epsilon) \tag{5.13}$$

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

$$\psi_T(N,\epsilon) := \sqrt{\beta_T\left(3\sigma^{-2}+\sigma^{-4}\right)N^3\epsilon} + \left(\sigma^{-2}+\sigma^{-4}\right)N^3\epsilon(2+b_0)\sqrt{\log\frac{2(1+a_0)\pi^2T^2}{3\delta}}. \tag{5.14}$$

The proof of Theorem 5.3.2 departs from regular time-invariant proofs such as [SKKS10] in the sense that the posterior updates (2.4) assumed by the algorithm differ from the true posterior described by (5.3)–(5.4), thus requiring a careful handling of the effect of the mismatch.

Theorem 5.3.3. *Let the domain $D \subset [0,r]^p$ be compact and convex, and suppose that the kernel is such that $f \sim \mathrm{GP}(0,k)$ is almost surely continuously differentiable and satisfies* (5.6) *for some (a_1, b_1). Fix $\delta \in (0,1)$, and set*

$$\beta_T = 2\log\frac{\pi^2T^2}{2\delta} + 2p\log\left(rpbT^2\sqrt{\log\frac{pa\pi^2T^2}{2\delta}}\right). \tag{5.15}$$

Defining $C_1 = 8/\log(1+\sigma^{-2})$, TV-GP-UCB satisfies the following after T time steps:

$$R_T \leq \sqrt{C_1 T\beta_T\tilde{\gamma}_T} + 2 \tag{5.16}$$

$$\leq \sqrt{C_1 T\beta_T\left(\frac{T}{\tilde{N}}+1\right)\left(\gamma_{\tilde{N}} + \tilde{N}^3\epsilon\right)} + 2 \tag{5.17}$$

with probability at least $1-\delta$, where (5.17) *holds for any $\tilde{N} \in \{1,\dots,T\}$.*

The step in (5.16) is obtained using techniques similar to those of [SKKS10, KO11], whereas the step in (5.17) is non-trivial and new. This step is key to our analysis, bounding the maximum mutual information $\tilde{\gamma}_T$ for the time varying case in terms of the analogous quantity $\gamma_{\tilde{N}}$ from the time-invariant setting. The idea in doing this is to split the block $\{1, \dots, T\}$ into smaller blocks of size $\tilde{N}$ within which the overall variation in f_t is not too large. This is in contrast with R-GP-UCB (and [BGZ14]), where the algorithm takes the block length N as a parameter and explicitly resets the algorithm every N time steps. For TV-GP-UCB, the length $\tilde{N}$ is only introduced as a tool in the analysis.

Applications to Specific Kernels: Specializing the above results to the squared exponential and Matérn kernels, using the corresponding bounds on γ_N from [SKKS10], and optimizing N as a function of T and ϵ, we obtain the following.

Corollary 5.3.1. *Under the conditions of Theorems 5.3.2 and 5.3.3, we have for any fixed p:*

1. *For the squared exponential kernel, $R_T = O^*(\max\{\sqrt{T}, T\epsilon^{1/8}\})$ for R-GP-UCB with $N = \Theta(\min\{T, \epsilon^{-1/4}\})$, and $R_T = O^*(\max\{\sqrt{T}, T\epsilon^{1/6}\})$ for TV-GP-UCB.*
2. *Consider the Matérn kernel with parameter $\nu > 2$, and set $c = \frac{p(p+1)}{2\nu+p(p+1)} \in (0, 1)$. We have $R_T = O^*(\max\{\sqrt{T^{1+c}}, T\epsilon^{\frac{1}{2}\frac{1-c}{4-c}}\})$ for R-GP-UCB with $N = \Theta(\min\{T, \epsilon^{-\frac{1}{4-c}}\})$, and $R_T = O^*(\max\{\sqrt{T^{1+c}}, T\epsilon^{\frac{1}{2}\frac{1-c}{3-c}}\})$ for TV-GP-UCB.*

The proof is given in Appendix 5.C.3. Note that, upon substituting $\epsilon = 0$, the preceding $O^*(\cdot)$ terms are dominated by the first terms in the maxima, and the bounds for both algorithms reduce to those obtained by GP-UCB in Theorem 2.2.1. In the case that ϵ vanishes more slowly (e.g., as $1/\sqrt{T}$), the regret bounds for TV-GP-UCB are strictly better than those of R-GP-UCB. The worsened bounds for the latter arise due to the above-mentioned mismatch in the update rules.

For both kernels, the optimized block length N of R-GP-UCB increases as ϵ decreases; this is to be expected, as it means that older samples are more correlated with the present function. We also observe that N increases as the function becomes smoother (by increasing ν for the Matérn kernel, or by switching from Matérn to squared exponential).

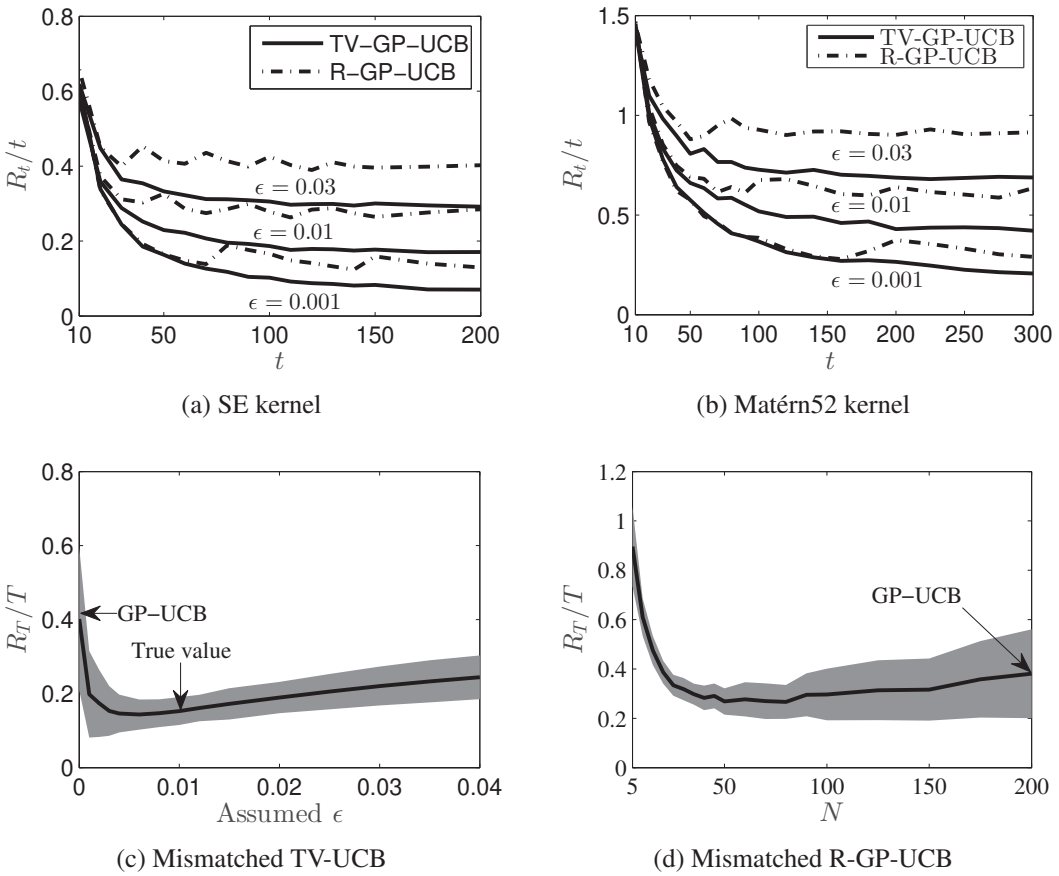
5.4 Experimental Evaluation

In this section, we test our algorithms on both synthetic and real data, as well as studying the effect of mismatch with respect to the algorithm parameters ϵ and N.

Practical considerations: While (5.12) and (5.15) give explicit choices for β_t, these usually tend to be too conservative in practice. For good empirical performance, we rely only on the *scaling* $\beta_t = O(p \log t)$ dictated by these choices, letting $\beta_t = c_1 \log(c_2 t)$ (similarly to [SKKS10, KSP15], for example). We found $c_1 = 0.8$ and $c_2 = 4$ to be suitable for trading off exploration and exploitation, and we therefore use these in all of our synthetic experiments.

Our theoretical analysis assumed that we know the hyperparameters of both spatial and temporal kernel. Having perfect knowledge of ϵ and other hyperparameters is typically not realistic. The GP perspective allows us to select them in a principled way by maximizing the GP marginal likelihood [RW06]. In our real-world experiments below, we select ϵ in such manner, using the approach from [VVLGS12], outlined in Appendix 5.B. In our synthetic experiments, we consider both the cases of perfect and imperfect knowledge of ϵ.

Baseline Comparisons: We are not aware of any algorithms other than those in Section 5.2 that exploit both spatial and temporal correlations. In both our synthetic and real-data experiments, we found it crucial to handle both of these in order to obtain reasonable values for the cumulative regret, thus drastically limiting the number of reasonable baselines. Nevertheless, we also consider GP-UCB (which exploits spatial but not temporal correlations), and in the real-world experiments, we consider a completely random selection (thus corresponding to a choice that we should hope to beat significantly).



(a) SE kernel

(b) Matérn52 kernel

(c) Mismatched TV-UCB

(d) Mismatched R-GP-UCB

Figure 5.2: Numerical performance of upper confidence bound algorithms on synthetic data.

5.4.1 Synthetic Data

We consider a two-dimensional setting and quantize the decision space $D = [0, 1]^2$ into the grid of 50×50 equally-spaced points. We generate our data according to the time-varying model (5.2), considering both the squared exponential and Matérn kernels. For the former we set $l = 0.2$, and for the latter we set $\nu = 2.5$ and $l = 0.2$. We set the sampling noise variance σ^2 to 0.01, i.e., 1% of the signal variance.

Matched Case: We first consider the case that the algorithm parameters are "matched". Specifically, the parameter ϵ for TV-GP-UCB is the true parameter for the model, and the parameter N for R-GP-UCB is chosen in accordance with Corollary 5.3.1: $N = \lceil \min\left\{T, 12\epsilon^{-1/4}\right\} \rceil$ for the squared exponential kernel, and $N = \lceil \min\left\{T, 24\epsilon^{-\frac{1}{4-c}}\right\} \rceil$ for the Matérn kernel, where the constants were found via cross-validation.

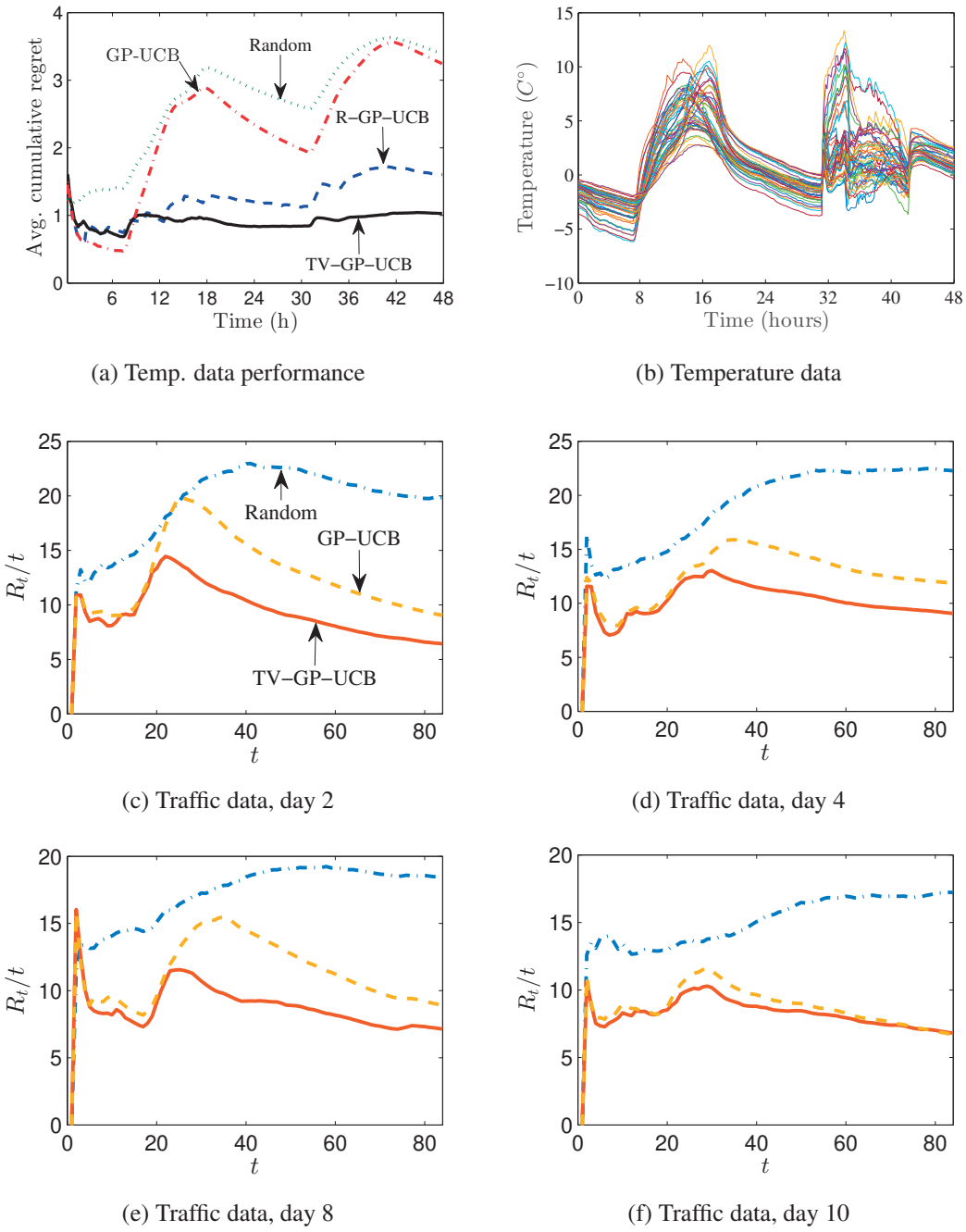
In Figures 5.2a and 5.2b, we show the average regret $\frac{R_T}{T}$ of TV-GP-UCB and R-GP-UCB for $\epsilon \in \{0.001, 0.01, 0.03\}$. For each time shown, we average the performance over 200 independent trials. We observe that for all values of ϵ and for both kernel functions, TV-GP-UCB outperforms R-GP-UCB, which is consistent with the theoretical bounds we obtained in the previous section. Furthermore, we see that the curves for R-GP-UCB have an oscillatory behavior, resulting from the fact that one tends to incur more regret just after a reset is done. In contrast, the curves for TV-GP-UCB are more steady, since the algorithm forgets in a "smooth" fashion.

Mismatch and Robustness: We consider the stability of TV-GP-UCB when there is mismatch between the true ϵ and the one used in TV-GP-UCB. We focus on the SE kernel, and we set $\epsilon = 0.01$ and $T = 200$. From Figure 5.2c, we see that the performance of TV-GP-UCB is robust with respect to the mis-specification of ϵ. In particular, the increase in regret as ϵ is increasingly over-estimated is slow. In contrast, while slightly under-estimating ϵ is not harmful, the regret increases rapidly beyond a certain point. In particular, using 0 instead of the true ϵ corresponds to simply running the standard GP-UCB algorithm, and gives the worst performance within the range shown. Note that the shaded area corresponds to a standard deviation from the mean.

Next, we study R-GP-UCB on the same model to determine the robustness with respect to the choice of N; the results are shown in Figure 5.2d. Values of N that are too small are problematic, since the algorithm resets too frequently. While the *mean* of the regret is robust with respect to increasing N, we observe that the corresponding standard deviation also steadily increases. GP-UCB is again recovered as a special case, corresponding to $N = T$.

5.4.2 Real Data

We use temperature data collected from 46 sensors deployed at Intel Research Berkeley. The dataset contains 5 days of measurements collected at 10-minute intervals. The goal of the spatiotemporal monitoring problem (see [KO11] for details) is to activate a sensor at every time step that reports a high temperature. Hence, f_t consists of the set of all sensor reportings at time t.



(a) Temp. data performance

(b) Temperature data

(c) Traffic data, day 2

(d) Traffic data, day 4

(e) Traffic data, day 8

(f) Traffic data, day 10

Figure 5.3: Numerical performance of upper confidence bound algorithms on real data.

A single sensor is activated every 10 minutes., and the regret is measured as the temperature difference between reporting of the activated sensor and the one that reports the maximum temperature at that particular time. Fig. 5.3b plots each of the 46 functions with respect to time.

As a base comparison, we consider a method that simply picks the sensors uniformly at random.

We also consider the standard GP-UCB algorithm (Algorithm 2), even though it is unsuitable here since the reward function is varying with time.[3] Although it is not shown, we note that the REXP3 algorithm [BGZ14] (Exp3 with resetting) performed comparably to GP-UCB for this data set, suffering from the fact that it does not exploit correlations between the sensors.

We use the first three days of measurements for learning our algorithms' parameters. First, we compute the empirical covariance matrix from these days and use it as the kernel matrix in all of the algorithms. Next, using the same three training days, we obtain $\epsilon = 0.03$ by maximizing the marginal likelihood [VVLGS12], and we obtain $N = 15$ by cross-validation. The algorithms are run on the final two days of the data. The results ($c_1 = 0.8, c_2 = 0.4, \sigma^2 = 0.5$ or 5% of the signal variance) are shown in Figure 5.3a. We observe that GP-UCB performs well for a short time, but then starts to suffer from the stale data, eventually becoming barely better than a random guess. Once again, TV-GP-UCB improves over R-GP-UCB, with the gap generally increasing over the duration of the experiment.

Next, we use traffic speed data from 357 sensors deployed along the highway I-880 South (California). The dataset contains one month of measurements, where 84 measurements were made on every day in between 6 AM and 11 AM. Our goal is to identify the least congested part of the highway by tracking the point of maximum speed. We use two thirds of the dataset to compute the empirical covariance matrix (and set it as the kernel matrix), and to learn ϵ by maximizing the marginal likelihood for all the training days [VVLGS12], treating each day as being statistically independent. The last 10 days were used for testing. Due to the small time horizon $T = 84$ in comparison to the number of sensors, we restrict the domain to contain 50 sensors, chosen randomly from the 357. Our results ($\epsilon = 0.04$, $\sigma^2 = 5.0$ or 5% of the signal variance, $T = 84$, $c_1 = 0.2$, $c_2 = 0.4$) were averaged over 20 different initially activated sensors.

In Figure 5.3, in final two columns, we show the outcome of the experiment for 4 testing days. TV-GP-UCB outperforms GP-UCB on most testing days, with the two being comparable for a few of the days (e.g., see Figure 5.3f). The latter situation arises when the indices of the best sensors do not change drastically over the time horizon, which is not always the case. In general, both algorithms suffer a large regret when sensors that were reporting high speeds suddenly change and start to report small speeds. However, TV-GP-UCB recovers more quickly from this compared to GP-UCB, due to its forgetting mechanism.

Note that we have omitted R-GP-UCB from this experiment, since we found it to be unsuitable due to the small time horizon. Moreover, this is the same reason that GP-UCB performs reasonably, unlike the temperature sensor example. Essentially, GP-UCB suffers more with a longer time horizon due to the larger amount of stale data.

[3] In [SKKS10], the same data was used to test GP-UCB in a different way; in each experiment, the function $f(x)$ was taken to be the set of temperatures at a single time.

5.A TV Posterior Updates

Here we derive the posterior update rules for the time-varying setting via a suitable adaptation of the derivation for the time-invariant setting [RW06]. Observe from (5.1)–(5.2) that each function f_t depends only on the functions g_i for $i \leq t$. By a simple recursion, we readily obtain for all t, j and $\boldsymbol{x}$ that

$$\mathrm{Cov}[f_t(\boldsymbol{x}), f_{t+j}(\boldsymbol{x}')] = (1-\epsilon)^{j/2}\mathbb{E}[f_t(\boldsymbol{x})f_t(\boldsymbol{x}')] = (1-\epsilon)^{j/2}k(\boldsymbol{x},\boldsymbol{x}'). \tag{5.18}$$

Hence, and since each output y_t equals the corresponding function sample $f_t(\boldsymbol{x}_t)$ plus additive Gaussian noise z_t with variance σ^2, the joint distribution between the previous outputs $\boldsymbol{y}_t = (y_1, \ldots, y_t)$ (corresponding to the points $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$) and the next function value $f_{t+1}(\boldsymbol{x})$ is

$$\begin{bmatrix} \boldsymbol{y} \\ f_{t+1}(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \widetilde{\boldsymbol{K}}_t + \sigma^2\boldsymbol{I}_t & \widetilde{\boldsymbol{k}}_t(\boldsymbol{x}) \\ \widetilde{\boldsymbol{k}}_t(\boldsymbol{x})^T & k(\boldsymbol{x},\boldsymbol{x}) \end{bmatrix}\right) \tag{5.19}$$

using the definitions in the proposition statement. Using the formula for the conditional distribution associated with a jointly Gaussian random vector [RW06, App. A], we find that $f_{t+1}(\boldsymbol{x})$ is conditionally Gaussian with mean $\tilde{\mu}_t(\boldsymbol{x})$ and variance $\tilde{\sigma}_t(\boldsymbol{x})^2$, as was to be shown.

5.B Learning Time-Varying Parameter via Maximum-Likelihood

In this section, we provide an overview of how ϵ can be learned from training data in a principled manner; the details can be found in [VVSLG12, Section 4.3] and [RW06, Section 5]. Throughout this appendix, we assume that the kernel matrix is parametrized by a set of hyperparameters θ (e.g., $\theta = (\nu, l)$ for the Matérn kernel), σ and ϵ.

Let $\bar{\boldsymbol{y}}$ be a vector of observations such that the i-th entry is observed at time t_i as a result of sampling the function f_{t_i} at location $\boldsymbol{x}_i$. Note that there will typically be many indices i sharing common values of t_i, since in the training data we often have multiple samples at each time. Under our time-varying GP model, the marginal log-likelihood of $\bar{\boldsymbol{y}}$ given the hyperparameters is

$$\log p(\bar{\boldsymbol{y}}|\theta,\sigma,\epsilon) = -\frac{1}{2}\bar{\boldsymbol{y}}^T(\bar{\boldsymbol{K}} \circ \bar{\boldsymbol{D}} + \sigma^2\boldsymbol{I})^{-1}\bar{\boldsymbol{y}} - \frac{1}{2}\log|\bar{\boldsymbol{K}} \circ \bar{\boldsymbol{D}} + \sigma^2\boldsymbol{I}| - \frac{n}{2}\log(2\pi) \tag{5.20}$$

where $(\bar{\boldsymbol{K}})_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $(\bar{\boldsymbol{D}})_{ij} = (1-\epsilon)^{|t_i - t_j|/2}$. To set the hyperparameters by maximizing the marginal likelihood, we can use the partial derivatives with respect to the hyperparameters:

$$\frac{\partial \log p(\bar{\boldsymbol{y}}|\theta,\sigma,\epsilon)}{\partial\epsilon} = \frac{1}{2}\mathrm{tr}\big((\boldsymbol{\alpha}\boldsymbol{\alpha}^T - (\bar{\boldsymbol{K}} \circ \bar{\boldsymbol{D}} + \sigma^2\boldsymbol{I})^{-1})(\bar{\boldsymbol{K}} \circ \bar{\boldsymbol{D}}')\big), \tag{5.21}$$

where $\boldsymbol{\alpha} = (\bar{\boldsymbol{K}} \circ \bar{\boldsymbol{D}} + \sigma^2\boldsymbol{I})^{-1}\bar{\boldsymbol{y}}$, and $(\bar{\boldsymbol{D}}')_{ij} = -v(1-\epsilon)^{v-1}$ with $v = |t_i - t_j|/2$.

We can now fit ϵ and the other hyperparameters by optimizing the marginal likelihood on the training data, e.g. by using an optimization algorithm from the family of quasi-Newton methods.

5.C Proofs

5.C.1 Analysis of TV-GP-UCB (Theorem 5.3.3)

We recall the following alternative form for the mutual information (see (5.10)) from [SKKS10, Lemma 5.3], which extends immediately to the time-varying setting:

$$\tilde{I}(\boldsymbol{f}_T; \boldsymbol{y}_T) = \frac{1}{2}\sum_{t=1}^{T} \log\big(1 + \sigma^{-2}\tilde{\sigma}^2_{t-1}(\boldsymbol{x}_t)\big). \tag{5.22}$$

Proof of 5.16

The initial steps of the proof follow similar ideas to [SKKS10], but with suitable modifications to handle the fact that we have a different function f_t at each time instant. A key difficulty is in subsequently bounding the maximum mutual information in the presence of time variations, which is done in the following subsection.

We first fix a discretization $D_t \subset D \subseteq [0, r]^p$ of size $(\tau_t)^p$ satisfying

$$\|\boldsymbol{x} - [\boldsymbol{x}]_t\|_1 \leq rp/\tau_t, \quad \forall \boldsymbol{x} \in D, \tag{5.23}$$

where $[\boldsymbol{x}]_t$ denotes the closest point in D_t to $\boldsymbol{x}$. A uniformly-spaced grid suffices to ensure this.

We now fix a constant $\delta > 0$ and an increasing sequence of positive constants $\{\pi_t\}_{t=1}^{\infty}$ satisfying $\sum_{t\geq 1} \pi_t^{-1} = 1$ (e.g. $\pi_t = \pi^2 t^2/6$), and condition on three high-probability events:

1. First, if $\beta_t \geq 2\log\frac{3\pi_t}{\delta}$ then the selected points $\{\boldsymbol{x}_t\}_{t=1}^{T}$ satisfy the confidence bounds

   $$|f_t(\boldsymbol{x}_t) - \tilde{\mu}_{t-1}(\boldsymbol{x}_t)| \leq \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t), \quad \forall t \tag{5.24}$$

   with probability at least $1 - \frac{\delta}{3}$. To see this, we note that conditioned on the outputs $(y_1, \ldots, y_{t-1})$, the sampled points $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$ are deterministic, and

   $$f_t(\boldsymbol{x}_t) \sim \mathcal{N}(\tilde{\mu}_{t-1}(\boldsymbol{x}_t), \tilde{\sigma}^2_{t-1}(\boldsymbol{x}_t)).$$

   An $\mathcal{N}(\mu, \sigma^2)$ random variable is within $\sqrt{\beta}\sigma$ of μ with probability at least $1 - e^{-\beta/2}$, and hence our choice of β_t ensures that the event corresponding to time t in (5.24) occurs with probability at least $\frac{\delta}{3\pi_t}$. Taking the union bound over t establishes the claim.

2. By the same reasoning with an additional union bound over $\boldsymbol{x} \in D_t$, if $\beta_t \geq 2\log\frac{3|D_t|\pi_t}{\delta}$,

   $$|f_t(\boldsymbol{x}) - \tilde{\mu}_{t-1}(\boldsymbol{x})| \leq \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}), \quad \forall t, \boldsymbol{x} \in D_t \tag{5.25}$$

   holds with probability at least $1 - \frac{\delta}{3}$.

3. Finally, we claim that setting $L_t = b\sqrt{\log(3pa\pi_t/\delta)}$ yields

$$|f_t(\boldsymbol{x}) - f_t(\boldsymbol{x}')| \leq L_t \|\boldsymbol{x} - \boldsymbol{x}'\|_1, \quad \forall t, \boldsymbol{x} \in D, \boldsymbol{x}' \in D \tag{5.26}$$

with probability at least $1 - \frac{\delta}{3}$. To see this, we note that by the assumption in (5.6) and the union bound over $j = 1, \ldots, p$, the event corresponding to time t in (5.26) holds with probability at least $1 - pae^{-L_t^2/b^2} = \frac{\delta}{3\pi_t}$. Taking the union bound over t establishes the claim.

Again applying the union bound, all three of (5.24)–(5.26) hold with probability at least $1 - \delta$. We henceforth condition on each of them occurring.

Combining (5.23) with (5.26) yields for all $\boldsymbol{x}$ that

$$|f_t(\boldsymbol{x}) - f_t([\boldsymbol{x}]_t)| \leq L_t rp/\tau_t \tag{5.27}$$

$$= b\sqrt{\log(3pa\pi_t/\delta)}rp/\tau_t, \tag{5.28}$$

and hence choosing $\tau_t = rpbt^2\sqrt{\log(2pa\pi_t/\delta)}$ yields

$$|f_t(\boldsymbol{x}) - f_t([\boldsymbol{x}]_t)| \leq 1/t^2. \tag{5.29}$$

Note that this choice of τ_t yields $|D_t| = (\tau_t)^p = (rpbt^2\sqrt{\log(3pa\pi_t/\delta)})^p$. In order to satisfy both lower bounds on β_t stated before (5.24) and (5.25), it suffices to take higher of the two (i.e., the second), yielding

$$\beta_t = 2\log(3\pi_t/\delta) + 2p\log(rpbt^2\sqrt{\log(3pa\pi_t/\delta)}). \tag{5.30}$$

This coincides with (5.15) upon setting $\pi_t = \pi^2 t^2/6$.

Substituting (5.29) into (5.25) and applying the triangle inequality, we find the maximizing point $\boldsymbol{x}_t^*$ at time t satisfies

$$|f_t(\boldsymbol{x}_t^*) - \tilde{\mu}_{t-1}([\boldsymbol{x}_t^*]_t)| \leq \beta_t^{1/2}\tilde{\sigma}_{t-1}([\boldsymbol{x}_t^*]_t) + 1/t^2. \tag{5.31}$$

Thus, we can bound the instantaneous regret as follows:

$$r_t = f_t(\boldsymbol{x}_t^*) - f_t(\boldsymbol{x}_t) \tag{5.32}$$

$$\leq \tilde{\mu}_{t-1}([\boldsymbol{x}_t^*]_t) + \beta_t^{1/2}\tilde{\sigma}_{t-1}([\boldsymbol{x}_t^*]_t) + 1/t^2 - f_t(\boldsymbol{x}_t) \tag{5.33}$$

$$\leq \tilde{\mu}_{t-1}(\boldsymbol{x}_t) + \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t) + 1/t^2 - f_t(\boldsymbol{x}_t) \tag{5.34}$$

$$\leq 2\beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t) + 1/t^2, \tag{5.35}$$

where in (5.33) we used (5.31), (5.34) follows since the function $\tilde{\mu}_{t-1}(\boldsymbol{x}) + \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x})$ is maximized at $\boldsymbol{x}_t$ by the definition of the algorithm, and (5.35) follows from (5.24).

Finally, we bound the cumulative regret as

$$R_T = \sum_{t=1}^{T} r_t \leq \sum_{t=1}^{T} \left(2\beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t) + 1/t^2\right) \tag{5.36}$$

$$\leq \sqrt{T\sum_{t=1}^{T} 4\beta_t \tilde{\sigma}_{t-1}^2(\boldsymbol{x}_t)} + 2 \tag{5.37}$$

$$\leq \sqrt{C_1 T \beta_T \tilde{\gamma}_T} + 2, \tag{5.38}$$

where (5.37) follows using $\sum_{t=1}^{\infty} 1/t^2 = \pi^2/6 \leq 2$ the fact that $\|\boldsymbol{z}\|_1 \leq \sqrt{T}\|\boldsymbol{z}\|_2$ for any vector $\boldsymbol{z} \in \mathbb{R}^T$. Equation (5.38) is proved using following steps from [SKKS10, Lemma 5.4], which we include for completeness:

$$\sum_{t=1}^{T} 4\beta_t \tilde{\sigma}_{t-1}^2(\boldsymbol{x}_t) \leq 4\beta_T\sigma^2 \sum_{t=1}^{T} \sigma^{-2}\tilde{\sigma}_{t-1}^2(\boldsymbol{x}_t) \tag{5.39}$$

$$\leq 4\beta_T\sigma^2 \sum_{t=1}^{T} C_2 \log\left(1 + \sigma^{-2}\tilde{\sigma}_{t-1}^2(\boldsymbol{x}_t)\right) \tag{5.40}$$

$$\leq C_1 \beta_T \tilde{\gamma}_T, \tag{5.41}$$

where (5.39) follows since β_t is increasing in T, (5.40) holds with $C_2 = \sigma^{-2}/\log(1+\sigma^{-2})$ using the identity $z^2 \leq C_2 \log(1+z^2)$ for $z^2 \in [0, \sigma^{-2}]$ (note also that the following holds $\sigma^{-2}\tilde{\sigma}_{t-1}^2(\boldsymbol{x}_t) \leq \sigma^{-2}k(\boldsymbol{x}_t, \boldsymbol{x}_t) \leq \sigma^{-2}$), and (5.41) follows from the definitions of C_1 and $\tilde{\gamma}_T$, along with the alternative form for the mutual information in (5.22).

Proof of 5.17

It remains to show that

$$\tilde{\gamma}_T \leq \left(\frac{T}{\tilde{N}} + 1\right)\left(\gamma_{\tilde{N}} + \tilde{N}^3\epsilon\right) \tag{5.42}$$

under the definitions in (5.10)–(5.11). Recall that $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$ are the points of interest and

$$\boldsymbol{f}_T = (f_1(\boldsymbol{x}_1), \ldots, f_T(\boldsymbol{x}_T))$$

are the corresponding function values, and $\boldsymbol{y}_T = (y_1, \ldots, y_T)$ contains the corresponding noisy observations with

$$y_i = f_i(\boldsymbol{x}_i) + z_i.$$

At a high level, we bound the mutual information with time variations in terms of the corresponding quantity for the time-invariant case [SKKS10] by splitting the time steps $\{1, \ldots, T\}$ into $\frac{T}{\tilde{N}}$ blocks of length $\tilde{N}$, such that within each block the function f_i does not vary significantly. We assume for the time being that $T/\tilde{N}$ is an integer, and then handle the general case.

Using the chain rule for mutual information and the fact that the noise sequence $\{z_i\}$ is independent, we have [CT01, Lemma 7.9.2]

$$\tilde{I}(\boldsymbol{f}_T; \boldsymbol{y}_T) \leq \sum_{i=1}^{T/\tilde{N}} \tilde{I}(\boldsymbol{f}_{\tilde{N}}^{(i)}; \boldsymbol{y}_{\tilde{N}}^{(i)}), \tag{5.43}$$

where $\boldsymbol{y}_{\tilde{N}}^{(i)} = (y_{\tilde{N}(i-1)+1}, \ldots, y_{\tilde{N}i})$ contains the measurements in the i-th block, and $\boldsymbol{f}_{\tilde{N}}^{(i)}$ is defined analogously. Maximizing both sides over $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$, we obtain

$$\tilde{\gamma}_T \leq \frac{T}{\tilde{N}} \tilde{\gamma}_{\tilde{N}}. \tag{5.44}$$

We are left to bound $\tilde{\gamma}_{\tilde{N}}$. To this end, we write the relevant covariance matrix as

$$\widetilde{\boldsymbol{K}}_{\tilde{N}} = \boldsymbol{K}_{\tilde{N}} \circ \boldsymbol{D}_{\tilde{N}} = \boldsymbol{K}_{\tilde{N}} + \boldsymbol{A}_{\tilde{N}}, \tag{5.45}$$

where

$$\boldsymbol{A}_{\tilde{N}} := \boldsymbol{K}_{\tilde{N}} \circ \boldsymbol{D}_{\tilde{N}} - \boldsymbol{K}_{\tilde{N}} \tag{5.46}$$

$$= \boldsymbol{K}_{\tilde{N}} \circ (\boldsymbol{D}_{\tilde{N}} - \boldsymbol{1}_{\tilde{N}}) \tag{5.47}$$

and $\boldsymbol{1}_{\tilde{N}}$ is the $\tilde{N} \times \tilde{N}$ matrix of ones. Observe that the (i,j)-th entry of $\boldsymbol{D}_{\tilde{N}} - \boldsymbol{1}_{\tilde{N}}$ has absolute value $1 - (1-\epsilon)^{\frac{|i-j|}{2}}$, which is upper bounded for all $\epsilon \in [0,1]$ by $\epsilon|i-j|$.[4] Hence, and using the fact that each entry of $\boldsymbol{K}_{\tilde{N}}$ lies in the range $[0,1]$, we obtain the following bound on the Frobenius norm:

$$\|\boldsymbol{A}_{\tilde{N}}\|_F^2 \leq \sum_{i,j} (i-j)^2 \epsilon^2 \tag{5.48}$$

$$= \frac{1}{6} \tilde{N}^2 (\tilde{N}^2 - 1) \epsilon^2 \tag{5.49}$$

$$\leq \tilde{N}^4 \epsilon^2, \tag{5.50}$$

where (5.49) is a standard double summation formula. We will use this inequality to bound $\tilde{\gamma}_N$ via Mirsky's theorem, which is given as follows.

Lemma 5.C.1. (Mirsky's theorem [HJ12, Cor. 7.4.9.3]) *For any matrices $\boldsymbol{U}_{\tilde{N}}$ and $\boldsymbol{V}_{\tilde{N}}$, and any unitarily invariant norm $|||\cdot|||$, we have*

$$\left|\left|\left|\mathrm{diag}(\lambda_1(\boldsymbol{U}_{\tilde{N}}), \ldots, \lambda_{\tilde{N}}(\boldsymbol{U}_{\tilde{N}})) - \mathrm{diag}(\lambda_1(\boldsymbol{V}_{\tilde{N}}), \ldots, \lambda_{\tilde{N}}(\boldsymbol{V}_{\tilde{N}}))\right|\right|\right| \leq \left|\left|\left|\boldsymbol{U}_{\tilde{N}} - \boldsymbol{V}_{\tilde{N}}\right|\right|\right|, \tag{5.51}$$

where $\lambda_i(\cdot)$ is the i-th largest eigenvalue.

[4] For $|i-j| \geq 2$, this follows since the function of interest is concave, passes through the origin, and has derivative $\frac{|i-j|}{2} \leq |i-j|$ there. For $k = 1$, the statement follows by observing that equality holds for $\epsilon \in \{0, 1\}$, and noting that the function of interest is convex.

Using this lemma with $\boldsymbol{U}_{\tilde{N}} = \boldsymbol{K}_{\tilde{N}} + \boldsymbol{A}_{\tilde{N}}$, $\boldsymbol{V}_{\tilde{N}} = \boldsymbol{K}_{\tilde{N}}$, and $|||\cdot||| = \|\cdot\|_F$, and making use of (5.50), we find that $\lambda_i(\boldsymbol{K}_{\tilde{N}} + \boldsymbol{A}_{\tilde{N}}) = \lambda_i(\boldsymbol{K}_{\tilde{N}}) + \Delta_i$ for some $\{\Delta_i\}_{i=1}^{\tilde{N}}$ satisfying $\sum_{i=1}^{\tilde{N}} \Delta_i^2 \leq \tilde{N}^4\epsilon^2$. We thus have

$$\tilde{\gamma}_{\tilde{N}} = \sum_{i=1}^{\tilde{N}} \log\left(1 + \lambda_i(\boldsymbol{K}_{\tilde{N}} + \boldsymbol{A}_{\tilde{N}})\right) \tag{5.52}$$

$$= \sum_{i=1}^{\tilde{N}} \log\left(1 + \lambda_i(\boldsymbol{K}_{\tilde{N}}) + \Delta_i\right) \tag{5.53}$$

$$\leq \gamma_{\tilde{N}} + \sum_{i=1}^{\tilde{N}} \log(1 + \Delta_i) \tag{5.54}$$

$$\leq \gamma_{\tilde{N}} + \tilde{N}\log(1 + \tilde{N}^2\epsilon) \tag{5.55}$$

$$\leq \gamma_{\tilde{N}} + \tilde{N}^3\epsilon, \tag{5.56}$$

where (5.54) follows from the inequality $\log(1+a+b) \leq \log(1+a)+\log(1+b)$ for non-negative a and b (and the definition in (5.8)), (5.55) follows since a simple analysis of the optimality conditions of

$$\text{maximize} \quad \sum_{i=1}^{\tilde{N}} \log(1 + \Delta_i) \qquad \text{subject to} \qquad \sum_{i=1}^{\tilde{N}} \Delta_i^2 \leq \tilde{N}^4\epsilon^2 \tag{5.57}$$

reveals that the maximum is achieved when all of the Δ_i are equal to $\tilde{N}^2\epsilon$, and (5.56) follows from the inequality $\log(1 + a) \leq a$.

Recalling that we are considering the case that $T/\tilde{N}$ is an integer, we obtain (5.17) by combining (5.44) and (5.56). In the general case, we simply use the fact that $\tilde{\gamma}_T$ is increasing in T by definition, hence leading to the addition of one in (5.17).

5.C.2 Analysis of R-GP-UCB (Theorem 5.3.2)

Parts of the proof of Theorem 5.3.2 overlap with that of Theorem 5.3.3; we focus primarily on the key differences. First, overloading the notation from the TV-GP-UCB analysis, we let $\tilde{\mu}_t(\boldsymbol{x})$ and $\tilde{\sigma}_t(\boldsymbol{x})$ be defined as in (5.3)–(5.4), but using *only the samples since the previous reset in the R-GP-UCB algorithm*, and similarly for $\boldsymbol{k}_t$, $\widetilde{\boldsymbol{k}}_t$, $\boldsymbol{d}_t$, and so on. Thus, for example, the dimension of $\boldsymbol{k}_t$ is at most the length N between resets, and the entries of $\boldsymbol{D}_t$ are no smaller than $(1-\epsilon)^{N/2}$. Note that the time-invariant counterparts $\mu_t(\boldsymbol{x})$ and $\sigma_t(\boldsymbol{x})$ (computed using $\boldsymbol{k}$ and $\boldsymbol{K}$ in place of $\widetilde{\boldsymbol{k}}$ and $\widetilde{\boldsymbol{K}}$) are used in the algorithm, thus creating a mismatch that must be properly handled.

Recall the definitions of the discretization D_t (whose cardinality is again set to τ_t^p for some τ_t), the corresponding quantization function $[\boldsymbol{x}]_t$, and the constants π_t. We now condition on four (rather than three) high probability events:

- Setting $\beta_t = 2\log\frac{4|D_t|\pi_t}{\delta}$, the same arguments as those leading to (5.24)–(5.25) reveal

$$|f_t(\boldsymbol{x}_t) - \tilde{\mu}_{t-1}(\boldsymbol{x}_t)| \leq \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t) \quad \forall t \geq 1 \tag{5.58}$$

$$|f_t(\boldsymbol{x}) - \tilde{\mu}_{t-1}(\boldsymbol{x})| \leq \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}) \quad \forall t \geq 1, \boldsymbol{x} \in D_t \tag{5.59}$$

  with probability at least $1 - \frac{\delta}{2}$. Note that in proving these claims we only condition on the observations since the last reset, rather than all of the points since $t = 1$.

- Using the same argument as (5.26), the assumption in (5.6) implies that

$$\Big|\frac{\partial f_t(\boldsymbol{x})}{\partial x^{(j)}}\Big| \leq L_t := b_1\sqrt{\log\frac{4da_1\pi_t}{\delta}} \quad \forall t \geq 1, \boldsymbol{x} \in D, j \in \{1,\dots,d\} \tag{5.60}$$

  with probability at least $1 - \frac{\delta}{4}$.

- We claim that the assumption in (5.5) similarly implies that

$$\big|y_t\big| \leq \tilde{L}_t := (2+b_0)\sqrt{\log\frac{4(1+a_0)\pi_t}{\delta}} \quad \forall t \geq 1, \boldsymbol{x} \in D \tag{5.61}$$

  with probability at least $1 - \frac{\delta}{4}$. To see this, we first note that $\mathbb{P}\big[|z_t| \leq L\big] \leq e^{-L^2/2}$ since $z_t \sim \mathcal{N}(0,1)$, and by a standard bound on the standard normal tail probability. Combining this with (5.5) and noting that $|y_t| \leq |f_t(\boldsymbol{x}_t)| + |z_t|$, we find that the probability $\mathbb{P}\big[|y_t| > 2L\big]$ is upper bounded by $e^{-L^2/2} + a_0 e^{-(L/b_0)^2}$, which in turn is upper bounded by $(1+a_0)e^{-(L/(2+b_0))^2}$. Choosing $L = \tilde{L}_t/2$ and applying the union bound, we obtain (5.61).

By the union bound, all four of (5.58)–(5.61) hold with probability at least $1-\delta$.

As in the TV-GP-UCB proof, we set $\tau_t = rpt^2L_t$, thus ensuring that $|f_t(\boldsymbol{x}) - f_t([\boldsymbol{x}]_t)| \leq \frac{1}{t^2}$ for all $\boldsymbol{x} \in D$.

Defining

$$\Delta_t^{(\mu)} := \sup_{\boldsymbol{x}\in D} |\tilde{\mu}_t(\boldsymbol{x}) - \mu_t(\boldsymbol{x})| \tag{5.62}$$

$$\Delta_t^{(\sigma)} := \sup_{\boldsymbol{x}\in D} |\tilde{\sigma}_t(\boldsymbol{x}) - \sigma_t(\boldsymbol{x})| \tag{5.63}$$

to be the maximal errors between the true and the mismatched posterior updates, we have:

$$\begin{aligned}
r_t &= f_t(\boldsymbol{x}_t^*) - f_t(\boldsymbol{x}_t) && (5.64)\\
&\leq f_t([\boldsymbol{x}_t^*]_t) - f_t(\boldsymbol{x}_t) + \frac{1}{t^2} && (5.65)\\
&\leq \tilde{\mu}_{t-1}([\boldsymbol{x}_t^*]_t) + \beta_t^{1/2}\tilde{\sigma}_{t-1}([\boldsymbol{x}_t^*]_t) - \tilde{\mu}_{t-1}(\boldsymbol{x}_t) + \beta_t^{1/2}\tilde{\sigma}_{t-1}(\boldsymbol{x}_t) + \frac{1}{t^2} && (5.66)\\
&\leq \mu_{t-1}([\boldsymbol{x}_t^*]_t) + \beta_t^{1/2}\sigma_{t-1}([\boldsymbol{x}_t^*]_t) - \mu_{t-1}(\boldsymbol{x}_t) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + 2\Delta_t^{(\mu)} + 2\beta_t^{1/2}\Delta_t^{(\sigma)} + \frac{1}{t^2} && (5.67)\\
&\leq 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t) + 2\Delta_t^{(\mu)} + 2\beta_t^{1/2}\Delta_t^{(\sigma)} + \frac{1}{t^2}, && (5.68)
\end{aligned}$$

where (5.65) follows in the same way as (5.33), (5.66) follows from (5.58)–(5.59), (5.67) follows from the definitions in (5.62)–(5.63), and (5.68) follows from the choice of $\boldsymbol{x}_t$ in the algorithm.

The key remaining step is to characterize $\Delta_t^{(\mu)}$ and $\Delta_t^{(\sigma)}$. Our findings are summarized in the following lemma.

Lemma 5.C.2. *Conditioned on the event in* (5.61)*, we have* $\Delta_t^{(\mu)} \leq \big(\sigma^{-2} + \sigma^{-4}\big)N^3\epsilon\tilde{L}_t$ *and* $\Delta_t^{(\sigma)} \leq \big(3\sigma^{-2} + \sigma^{-4}\big)N^3\epsilon$ *almost surely.*

This lemma implies Theorem 5.3.2 upon substitution into (5.68), setting $\pi_t = \pi^2 t^2/6$, and following the steps from (5.36) onwards. In the remainder of the section, we prove the lemma. The claims on $\Delta_t^{(\mu)}$ and $\Delta_t^{(\sigma)}$ are proved similarly; we focus primarily on the latter since it is the (slightly) more difficult of the two.

The subsequent analysis applies for arbitrary values of t and $\boldsymbol{x}$, so we use the shorthands $\boldsymbol{k} := \boldsymbol{k}_t(\boldsymbol{x})$, $\boldsymbol{K} := \boldsymbol{K}_t(\boldsymbol{x})$, $\widetilde{\boldsymbol{k}} := \widetilde{\boldsymbol{k}}_t(\boldsymbol{x})$, $\widetilde{\boldsymbol{K}} := \widetilde{\boldsymbol{K}}_t$ and $\boldsymbol{I} := \boldsymbol{I}_t$. We first use the definition in (5.4) and the triangle inequality to write

$$\begin{aligned}
&|\tilde{\sigma}_t(\boldsymbol{x})^2 - \sigma_t(\boldsymbol{x})^2|\\
&\quad = \big|\widetilde{\boldsymbol{k}}^T(\widetilde{\boldsymbol{K}} + \sigma^2\boldsymbol{I})^{-1}\widetilde{\boldsymbol{k}} - \boldsymbol{k}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}\big| && (5.69)\\
&\quad \leq \big|\widetilde{\boldsymbol{k}}^T(\widetilde{\boldsymbol{K}} + \sigma^2\boldsymbol{I})^{-1}\widetilde{\boldsymbol{k}} - \widetilde{\boldsymbol{k}}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\widetilde{\boldsymbol{k}}\big| + \big|\widetilde{\boldsymbol{k}}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\widetilde{\boldsymbol{k}} - \boldsymbol{k}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}\big| && (5.70)\\
&\quad := T_1 + T_2. && (5.71)
\end{aligned}$$

We proceed by bounding T_1 and T_2 separately, starting with the latter.

Set $\boldsymbol{M} := (\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}$ for brevity. By expanding the quadratic function $(\widetilde{\boldsymbol{k}}-\boldsymbol{k})^T\boldsymbol{M}(\widetilde{\boldsymbol{k}}-\boldsymbol{k})^T$, grouping the terms appearing in T_2, and applying the triangle inequality, we obtain

$$T_2 \le 2|\boldsymbol{k}^T\boldsymbol{M}(\widetilde{\boldsymbol{k}}-\boldsymbol{k})| + |(\widetilde{\boldsymbol{k}}-\boldsymbol{k})^T\boldsymbol{M}(\widetilde{\boldsymbol{k}}-\boldsymbol{k})|. \tag{5.72}$$

We upper bound each of these terms of the form $\boldsymbol{a}^T\boldsymbol{M}\boldsymbol{b}$ by $\|\boldsymbol{a}\|_2\|\boldsymbol{M}\|_{2\to2}\|\boldsymbol{b}\|_2$, where $\|\boldsymbol{M}\|_{2\to2}$ is the spectral norm. By definition, λ is an eigenvalue of $\boldsymbol{K}$ if and only if $\frac{1}{\lambda+\sigma^2}$ is an eigenvalue of $\boldsymbol{M}$; since $\boldsymbol{K}$ is positive semi-definite, it follows that $\|\boldsymbol{M}\|_{2\to2} \le \frac{1}{\sigma^2}$. We also have $\|\boldsymbol{k}\|_2^2 \le N$ since the entries of $\boldsymbol{k}$ lies in $[0,1]$, and $\|\widetilde{\boldsymbol{k}}-\boldsymbol{k}\|_2^2 \le N^3\epsilon^2$ since the absolute values of the entries of $\widetilde{\boldsymbol{k}}-\boldsymbol{k}$ are upper bounded by $N\epsilon$ by the argument following (5.47). Combining these, we obtain

$$T_2 \le 2\sigma^{-2}N^2\epsilon + \sigma^{-2}N^3\epsilon^2. \tag{5.73}$$

To bound T_1, we use the following inequality for positive definite matrices $\boldsymbol{U},\boldsymbol{V}$ and any unitarily invariant norm $|||\cdot|||$ [Bha97, Lemma X.1.4]:

$$\big|\big|\big|(\boldsymbol{U}+\boldsymbol{I})^{-1} - (\boldsymbol{U}+\boldsymbol{V}+\boldsymbol{I})^{-1}\big|\big|\big| \le \big|\big|\big|\boldsymbol{I} - (\boldsymbol{V}+\boldsymbol{I})^{-1}\big|\big|\big|. \tag{5.74}$$

Specializing to the spectral norm, multiplying through by $\frac{1}{\sigma^2}$, and choosing $\boldsymbol{U} = \frac{1}{\sigma^2}\boldsymbol{K}$ and $\boldsymbol{V} = \frac{1}{\sigma^2}(\widetilde{\boldsymbol{K}}-\boldsymbol{K})$, we obtain

$$\big\|(\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1} - (\widetilde{\boldsymbol{K}}+\sigma^2\boldsymbol{I})^{-1}\big\|_{2\to2} \le \big\|\sigma^{-2}\boldsymbol{I} - (\widetilde{\boldsymbol{K}}-\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}\big\|_{2\to2}. \tag{5.75}$$

Next, λ is an eigenvalue of the matrix $\widetilde{\boldsymbol{K}}-\boldsymbol{K}$ if and only if $\sigma^{-2} - \frac{1}{\lambda+\sigma^2}$ is an eigenvalue of $\sigma^{-2}\boldsymbol{I} - (\widetilde{\boldsymbol{K}}-\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}$. Writing $\sigma^{-2} - \frac{1}{\lambda+\sigma^2} = \sigma^{-2}\big(1 - \frac{1}{\lambda/\sigma^2+1}\big) \le \sigma^{-4}\lambda$, it follows that the right-hand side of (5.75) is upper bounded by $\sigma^{-4}\|\widetilde{\boldsymbol{K}}-\boldsymbol{K}\|_{2\to2}$. Using (5.50) (observe that $\boldsymbol{A}_N = \widetilde{\boldsymbol{K}}-\boldsymbol{K}$) and the fact that the spectral norm is upper bounded by the Frobenius norm, we obtain $\|\widetilde{\boldsymbol{K}}-\boldsymbol{K}\|_{2\to2} \le N^2\epsilon$. Substituting into (5.75), we conclude that the matrix $\boldsymbol{M}' := (\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1} - (\widetilde{\boldsymbol{K}}+\sigma^2\boldsymbol{I})^{-1}$ has a spectral norm which is upper bounded by $\sigma^{-4}N^2\epsilon$. Finally, T_1 can be written as $\widetilde{\boldsymbol{k}}^T\boldsymbol{M}'\widetilde{\boldsymbol{k}}$, and since $\|\widetilde{\boldsymbol{k}}\|_2^2 \le N$ (since each entry of $\widetilde{\boldsymbol{k}}$ lies in $[0,1]$), we obtain

$$T_1 \le \sigma^{-4}N^3\epsilon. \tag{5.76}$$

Combining (5.73) and (5.76) and crudely writing $N^2\epsilon \le N^3\epsilon$ and $N^3\epsilon^2 \le N^3\epsilon$, we obtain

$$|\tilde{\sigma}_t(\boldsymbol{x})^2 - \sigma_t(\boldsymbol{x})^2| \le \big(3\sigma^{-2}+\sigma^{-4}\big)N^3\epsilon, \tag{5.77}$$

and hence, applying the inequality $(a-b)^2 \le |a^2-b^2|$, we obtain

$$\Delta_t^{(\sigma)} \le \sqrt{\big(3\sigma^{-2}+\sigma^{-4}\big)N^3\epsilon}. \tag{5.78}$$

To characterize $\Delta_t^{(\mu)}$, we write the following analog of (5.70):

$$|\tilde{\mu}_t(\boldsymbol{x})^2 - \mu_t(\boldsymbol{x})^2|$$
$$\leq \left|\widetilde{\boldsymbol{k}}^T(\widetilde{\boldsymbol{K}} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} - \widetilde{\boldsymbol{k}}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}\right| + \left|\widetilde{\boldsymbol{k}}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} - \boldsymbol{k}^T(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}\right| \tag{5.79}$$
$$:= T_1 + T_2. \tag{5.80}$$

Following the same arguments as those above, and noting from (5.61) that $\|\boldsymbol{y}\|_2^2 \leq N\tilde{L}_N^2$, we obtain

$$T_1 \leq \sigma^{-2}N^2\epsilon\tilde{L}_N \tag{5.81}$$
$$T_2 \leq \sigma^{-4}N^3\epsilon\tilde{L}_N, \tag{5.82}$$

and hence

$$\Delta_t^{(\mu)} \leq \left(\sigma^{-2} + \sigma^{-4}\right)N^3\epsilon\tilde{L}_N. \tag{5.83}$$

5.C.3 Applications to Specific Kernels (Corollary 5.3.1)

In this section, we let $\mathcal{I}_T(z)$ denote the integer in $\{1, \dots, T\}$ which is closest to $z \in \mathbb{R}$. We focus primarily on the proof for TV-GP-UCB, since the proof for R-GP-UCB is essentially identical.

We begin with the squared exponential kernel. From (2.11), we have that the maximum mutual information $\gamma_{\tilde{N}} = O(p\log\tilde{N}) = O^*(1)$, and we thus obtain

$$\left(\frac{T}{\tilde{N}} + 1\right)\left(\gamma_{\tilde{N}} + N^3\epsilon\right) = O^*\left(\left(\frac{T}{\tilde{N}} + 1\right)(1 + \tilde{N}^3\epsilon)\right). \tag{5.84}$$

Setting $\tilde{N} = \mathcal{I}_T(\epsilon^{-1/3})$, we find that this behaves as $O^*(T\epsilon^{1/3})$ when $\epsilon \geq \frac{1}{T^3}$, and as $O^*(1)$ when $\epsilon < \frac{1}{T^3}$ (and hence $\tilde{N} = T$). Substitution into Theorem 5.3.3 yields the desired result.

For the Matérn kernel, we have from (2.12) that $\gamma_{\tilde{N}} = O(\tilde{N}^c\log\tilde{N}) = O^*(\tilde{N}^c)$ where $c = \frac{p(p+1)}{2\nu + p(p+1)}$, and we thus obtain

$$\left(\frac{T}{\tilde{N}} + 1\right)\left(\gamma_{\tilde{N}} + \tilde{N}^3\epsilon\right) = O^*\left(\left(\frac{T}{\tilde{N}} + 1\right)(\tilde{N}^c + \tilde{N}^3\epsilon)\right). \tag{5.85}$$

Setting $\tilde{N} = \mathcal{I}_T(\epsilon^{-\frac{1}{3-c}})$, we find that this behaves as $O^*(T\epsilon^{\frac{1-c}{3-c}})$ when $\epsilon \geq \frac{1}{T^{3-c}}$, and as $O^*(T^c)$ when $\epsilon < \frac{1}{T^{3-c}}$ (and hence $\tilde{N} = T$). Substitution into Theorem 5.3.3 yields the desired result.

For R-GP-UCB, the arguments are analogous using Theorem 5.3.2 in place of Theorem 5.3.3, with N playing the role of $\tilde{N}$. We set $N = \mathcal{I}_T(\epsilon^{-1/4})$ for the squared exponential kernel and $N = \mathcal{I}_T(\epsilon^{-\frac{1}{4-c}})$ for the Matérn kernel.

5.C.4 Lower Bound (Theorem 5.3.1)

We obtain a lower bound on the regret of *any* algorithm by considering the optimal algorithm for a genie-aided setting. Specifically, suppose that at time t, the entire function f_{t-1} is known perfectly. We claim that the optimal strategy, in the sense of minimizing the expected regret, is to choose $\boldsymbol{x}_t$ to be any maximizer of f_{t-1}. This can be seen by noting that minimizing the regret $r_t = f_t(\boldsymbol{x}_t^*) - f_t(\boldsymbol{x}_t)$ is equivalent to maximizing the function value $f_t(\boldsymbol{x}_t)$, since $f_t(\boldsymbol{x}_t^*)$ is unaffected by the choice of $\boldsymbol{x}_t$. Then, conditioned on the entire function f_{t-1}, the next value $f_t(\boldsymbol{x})$ is distributed as $\mathcal{N}(\sqrt{1-\epsilon} f_{t-1}(\boldsymbol{x}), \epsilon)$, and clearly the optimal strategy is to choose the point that maximizes the mean. We proceed by lower bounding the regret incurred by such a scheme. Recall that for each t, both f_t and g_t are distributed as $\mathrm{GP}(0, k)$. Thus, (5.6) and (5.7) hold for all such functions.

We let ∇f denote the gradient vector of a function f, and let $\nabla^2 f$ denote the Hessian matrix. For the time being, we condition on the previous function f_{t-1}, the selected point $\boldsymbol{x}_t$ (i.e., the maximizer of f_{t-1}) and the innovation function g_t satisfying the following events for some positive constants L and η:

$$\mathcal{A}_1 := \left\{ \left| \frac{\partial^2 f_{t-1}(\boldsymbol{x})}{\partial x^{(j_1)} \partial x^{(j_2)}} \right| \le L, \ \ \forall j_1, j_2, \boldsymbol{x} \right\} \tag{5.86}$$

$$\mathcal{A}_2 := \left\{ \left| \frac{\partial^2 g_t(\boldsymbol{x})}{\partial x^{(j_1)} \partial x^{(j_2)}} \right| \le L, \ \ \forall j_1, j_2, \boldsymbol{x} \right\} \tag{5.87}$$

$$\mathcal{A}_3 := \left\{ \frac{\sqrt{\epsilon} \left| \frac{\partial g_t(\boldsymbol{x})}{\partial x^{(j)}} \right|}{2L\sqrt{p}} \le \eta, \ \ \forall j \right\} \tag{5.88}$$

$$\mathcal{A}_4 := \big\{ d(\boldsymbol{x}_t, B) \ge \eta \big\}, \tag{5.89}$$

where $d(\boldsymbol{x}_t, B) := \min_{\boldsymbol{x} \in B} \|\boldsymbol{x}_t - \boldsymbol{x}\|_2$ is the distance of $\boldsymbol{x}_t$ to the closest point on the boundary B of the compact domain D. Observe that for any fixed η, $\mathbb{P}[\mathcal{A}_i]$ can be made arbitrarily close to one for $i = 1, 2, 3$ by choosing L sufficiently large. Moreover, we have $\mathbb{P}[\mathcal{A}_4] > 0$ for sufficiently small η, since otherwise the maximum of $f \sim \mathrm{GP}(0, k)$ would be on the boundary of the domain D with probability one. Applying the union bound, we conclude that the event $\mathcal{A} := \mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{A}_3 \cap \mathcal{A}_4$ occurs with strictly positive probability for suitable chosen η and L.

We fix an arbitrary vector $\boldsymbol{v}$ with $\|\boldsymbol{v}\|_2 = 1$ and a constant $\delta > 0$, and note that the regret r_t at time t an be lower bounded as follows provided that $\boldsymbol{x}_t + \boldsymbol{v}\delta \in D$:

$$r_t = \max_{\boldsymbol{x}} f_t(\boldsymbol{x}) - f_t(\boldsymbol{x}_t) \tag{5.90}$$

$$\ge f_t(\boldsymbol{x}_t + \boldsymbol{v}\delta) - f_t(\boldsymbol{x}_t) \tag{5.91}$$

$$= \sqrt{1-\epsilon}\big(f_{t-1}(\boldsymbol{x}_t + \boldsymbol{v}\delta) - f_{t-1}(\boldsymbol{x}_t)\big) + \sqrt{\epsilon}\big(g_t(\boldsymbol{x}_t + \boldsymbol{v}\delta) - g_t(\boldsymbol{x}_t)\big) \tag{5.92}$$

$$= \sqrt{1-\epsilon}\,\frac{1}{2}\delta^2 \boldsymbol{v}^T \big[\nabla^2 f_{t-1}(\boldsymbol{x}_t + \boldsymbol{v}\delta_f)\big]\boldsymbol{v} + \sqrt{\epsilon}\big(\delta \boldsymbol{v}^T \nabla g_t(\boldsymbol{x}_t) + \frac{1}{2}\delta^2 \boldsymbol{v}^T \big[\nabla^2 g_t(\boldsymbol{x}_t + \boldsymbol{v}\delta_g)\big]\boldsymbol{v}\big). \tag{5.93}$$

Eq. (5.92) follows by substituting the update equations in (5.1)–(5.2), and (5.93) holds for some $\delta_f \in [0, \delta]$ and $\delta_g \in [0, \delta]$ by a second-order Taylor expansion; note that $\nabla f_{t-1}(\boldsymbol{x}_t) = 0$ since $\boldsymbol{x}_t$ maximizes f_{t-1}, whose peak is away from the boundary of the domain by (5.89).

We choose the unit vector $\boldsymbol{v}$ to have the same direction as the gradient vector $\nabla g_t(\boldsymbol{x}_t)$, so that $\delta \boldsymbol{v}^T \nabla g_t(\boldsymbol{x}_t) = \delta \|\nabla g_t(\boldsymbol{x}_t)\|_2$. By (5.86)–(5.87), the entries of $\nabla^2 f_{t-1}(\boldsymbol{x}_t + \boldsymbol{v}\delta_f)$ and $\nabla^2 g_t(\boldsymbol{x}_t + \boldsymbol{v}\delta_g)$ are upper bounded by L, and thus a standard inequality between the entry-wise ℓ_∞ norm and the spectral norm reveals that the latter is upper bounded by Lp. This, in turn, implies that $\boldsymbol{v}^T \big[\nabla^2 f_{t-1}(\boldsymbol{x}_t + \boldsymbol{v}\delta_f)\big]\boldsymbol{v}$ and $\boldsymbol{v}^T \big[\nabla^2 g_t(\boldsymbol{x}_t + \boldsymbol{v}\delta_g)\big]\boldsymbol{v}$ are upper bounded by Lp, and hence we have the following

$$r_t \geq \sqrt{\epsilon}\delta \|\nabla g_t(\boldsymbol{x}_t)\|_2 - \frac{1}{2} Lp\delta^2 \big(\sqrt{1+\epsilon} + \sqrt{\epsilon}\big) \tag{5.94}$$

$$\geq \sqrt{\epsilon}\delta \|\nabla g_t(\boldsymbol{x}_t)\|_2 - Lp\delta^2, \tag{5.95}$$

where we have used $\sqrt{1+\epsilon} + \sqrt{\epsilon} \leq 2$. By differentiating with respect to δ, it is easily verified that the right-hand side is maximized by $\delta = \frac{\sqrt{\epsilon}\|\nabla g_t(\boldsymbol{x}_t)\|_2}{2Lp}$. This choice is seen to be valid (i.e., it yields $\boldsymbol{x}_t + \boldsymbol{v}\delta$ still in the domain) by (5.88)–(5.89) and the fact that $\|\boldsymbol{z}\|_2 \leq \sqrt{p}\|\boldsymbol{z}\|_\infty$ for $\boldsymbol{z} \in \mathbb{R}^p$, and we obtain

$$r_t \geq \frac{\epsilon \|\nabla g_t(\boldsymbol{x}_t)\|_2^2}{4Lp}. \tag{5.96}$$

It follows that the expectation of r_t is lower bounded by

$$\mathbb{E}[r_t] \geq \mathbb{P}[\mathcal{A}]\mathbb{E}[r_t|\mathcal{A}] \tag{5.97}$$

$$\geq \mathbb{P}[\mathcal{A}] \frac{\epsilon\, \mathbb{E}\big[\|\nabla g_t(\boldsymbol{x}_t)\|_2^2 \,|\, \mathcal{A}\big]}{4Lp} \tag{5.98}$$

$$= \Theta(\epsilon), \tag{5.99}$$

where (5.97) follows since $r_t \geq 0$ almost surely, and (5.99) follows since $\mathbb{E}\big[\|\nabla g_t(\boldsymbol{x}_t)\|_2^2 \,|\, \mathcal{A}\big] > 0$ by a simple proof by contradiction: The expectation can only equal zero if its (non-negative) argument is zero almost surely, but if that were the case then the unconditional distribution of $\|\nabla g_t(\boldsymbol{x}_t)\|_2^2$ would satisfy $\mathbb{P}[\|\nabla g_t(\boldsymbol{x}_t)\|_2^2 = 0] > \mathbb{P}[\mathcal{A}]$, which is impossible since the entries of $\nabla g_t(\boldsymbol{x}_t)$ are Gaussian [RW06, Sec. 9.4] and hence have zero probability of being exactly zero.

Finally, using (5.99), the average cumulative regret satisfies $\mathbb{E}[R_T] = \sum_{i=1}^T \mathbb{E}[r_T] = \Omega(T\epsilon)$.

6 Robust Submodular Maximization in the Presence of Adversarial Removals

In this chapter, we consider another robust problem formulation in which our goal is to choose a set of decisions to maximize some objective of interest, in the case that some of the selected decisions might result in failure. Formally, we study the problem of maximizing a monotone submodular function subject to a cardinality constraint k, with the added twist that a number of items τ from the returned set may be removed. We focus on the worst-case setting considered in [OSU16], in which a constant-factor approximation guarantee was given for $\tau = o(\sqrt{k})$. We solve a key open problem raised therein, presenting a new Partitioned Robust (PRo) submodular maximization algorithm that achieves the same guarantee for more general $\tau = o(k)$.

This chapter is based on the joint work with Slobodan Mitrovic, Jonathan Scarlett and Volkan Cevher [BMSC17b].

6.1 Introduction

Discrete optimization problems arise frequently in machine learning, and are often NP-hard even to approximate. In the case of a set function exhibiting *submodularity*, one can efficiently perform maximization subject to cardinality constraints with a $\left(1 - \frac{1}{e}\right)$-factor approximation guarantee. Applications include influence maximization [KKT03], document summarization [LB11], sensor placement [KG07], and active learning [KG12], just to name a few.

In many applications, one requires *robustness* in the decision (i.e., solution set) returned by the algorithm, in the sense that the objective value degrades as little as possible when some elements of the set are removed. For instance, (i) when deciding to run expensive experiments to learn about a quantity of interest, some number of the selected experiments might fail; (ii) in influence maximization problems, a subset of the chosen users may decide not to spread the word about a product; (ii) in summarization problems, a user may choose to remove some items from the summary due to their personal preferences; (iii) in the problem of sensor placement for outbreak detection, some of the sensors might fail. In situations where one does not have a reasonable prior distribution on the elements removed, protecting against worst-case removals becomes important.

Algorithm	Max. Robust.	Cardinality	Oracle Evals.	Approx.
SATURATE[KMGG08]	Arbitrary	$k + \Theta(k \log(k))$	exponential in τ	1.0
OSU [OSU16]	$o(\sqrt{k})$	k	$O(nk)$	0.387
PRO-GREEDY	$o(k)$	k	$O(nk)$	0.387

Table 6.1: Algorithms for robust monotone submodular optimization with a cardinality constraint. Our algorithm PRO-GREEDY is efficient and allows for greater robustness.

This setting results in the robust submodular function maximization problem, in which we seek to return a set of cardinality k that is robust with respect to the worst-case removal of τ elements.

6.1.1 Problem Statement

In this chapter, we study the robust problem formulation introduced in Section 2.3. Here, we recall this problem and its main ingredients.

Let V be a ground set with cardinality $|V| = n$, and let $f : 2^V \to \mathbb{R}_+$ be a monotone submodular set function defined on V. The function f is said to be submodular if for any sets $S \subseteq P \subseteq V$ and any element $e \in V \setminus Y$, it holds that

$$f(S \cup \{e\}) - f(S) \geq f(P \cup \{e\}) - f(P),$$

and is monotone if for any sets $S \subseteq P \subseteq V$ we have $f(S) \leq f(P)$. First, we recall the problem of maximizing a monotone submodular function subject to a cardinality constraint (Section 2.3):

$$\max_{S \subseteq V, |S| \leq k} f(S). \tag{2.18}$$

A celebrated result of [NWF78] shows that a simple greedy algorithm that starts with an empty set and then iteratively adds elements with highest marginal gain provides a $(1 - 1/e)$-approximation.

In this chapter, we consider the following robust version[1] of the previous problem:

$$\max_{S \subseteq V, |S| \leq k} \min_{E \subseteq S, |E| \leq \tau} f(S \setminus E)$$

We refer to τ as the *robustness parameter*, representing the size of the subset E that is removed from the selected set S. Our goal is to find a set S such that it is robust upon the worst possible removal of τ elements, i.e., after the removal, the objective value should remain as large as possible. More information on the previous and related work can be found in Section 2.3.

[1] For $\tau = 0$, the robust problem reduces to the one in (2.18).

6.1.2 Contributions

The main contributions of this chapter are:

- In this chapter, we solve a key open problem posed in [OSU16], namely, whether a constant-factor approximation guarantee is possible for general $\tau = o(k)$, as opposed to only $\tau = o(\sqrt{k})$. We answer this question in the affirmative, providing a new Partitioned Robust (PRO) submodular maximization algorithm (in Section 6.2.1) that attains a constant-factor approximation guarantee; see Table 6.1 for comparison of different algorithms for robust monotone submodular optimization with a cardinality constraint. More details on these algorithms can be found in Section 2.3.
- Achieving this result requires novelty both in the algorithm and its analysis: While our algorithm bears some similarity to that of [OSU16] (Fig. 2.4), it uses a new structure in which the constructed set is arranged into partitions consisting of buckets whose sizes increase exponentially with the partition index. As shown in Section 6.2.4, a key step in our analysis provides a recursive relationship between the objective values attained by buckets appearing in adjacent partitions.
- In addition to the above contributions, in Section 6.3 we provide the first empirical study beyond what is demonstrated for $\tau = 1$ in [KMGG08]. We demonstrate several scenarios in which our algorithm outperforms both the GREEDY algorithm and the algorithm of [OSU16].

6.1.3 Applications

In this section, we provide several examples of applications where the robustness of the solution is favorable. The objective functions in these applications are non-negative, monotone and submodular, and are used in our numerical experiments in Section 6.3.

Robust influence maximization. The goal in the influence maximization problem is to find a set of k nodes (i.e., a targeted set) in a network that maximizes some measure of influence. For example, this problem appears in viral marketing, where companies wish to spread the word of a new product by targeting the most influential individuals in a social network. Due to poor incentives or dissatisfaction with the product, for instance, some of the users from the targeted set might make the decision not to spread the word about the product.

For many of the existing diffusion models used in the literature (e.g., see [KKT03]), given the targeted set S, the expected number of influenced nodes at the end of the diffusion process is a monotone and submodular function of S [HK16]. For simplicity, we consider a basic model in which all of the neighbors of the users in S become influenced, as well as those in S itself.

More formally, we are given a graph $G = (V, E)$, where V stands for nodes and E are the edges. For a set S, let $\mathcal{N}(S)$ denote all of its neighboring nodes, and let $R_S \subseteq S$ represent the

users that decide not to spread the word. The goal is to solve the robust *dominating set problem*, i.e., to find a set of nodes S of size k that maximizes

$$\min_{|R_S|\leq\tau, R_S\subseteq S} |(S \setminus R_S) \cup \mathcal{N}(S \setminus R_S)|. \tag{6.1}$$

Robust personalized image summarization. In the personalized image summarization problem, a user has a collection of images, and the goal is to find k images that are representative of the collection. After being presented with a solution, the user might decide to remove a certain number of images from the representative set due to various reasons (e.g., bad lighting, motion blur, etc.). Hence, our goal is to find a set of images that remain good representatives of the collection even after the removal of some number of them.

One popular way of finding a representative set in a massive dataset is via exemplar based clustering, i.e., by minimizing the sum of pairwise dissimilarities between the exemplars S and the elements of the data set V. This problem can be posed as a submodular maximization problem subject to a cardinality constraint; *cf.*, [LBZK16].

Here, we are interested in solving the robust summarization problem, i.e., we want to find a set of images S of size k that maximizes

$$\min_{|R_S|\leq\tau, R_S\subseteq S} f(\{e_0\}) - f((S \setminus R_S) \cup \{e_0\}), \tag{6.2}$$

where e_0 is a reference element and

$$f(S) = \frac{1}{|V|} \sum_{v\in V} \min_{s\in S} d(s, v)$$

is the *k-medoid* loss, and where $d(s, v)$ measures the dissimilarity between images s and v.

Further potential applications not covered here include robust sensor placement [KMGG08], robust protection of networks [Bog12], robust feature selection [GR06] and robust batch Bayesian Optimization [DKB14], some of which we will consider in the subsequent chapters.

6.2 Algorithm and its Guarantees

6.2.1 The Algorithm

Our algorithm, which we call the Partitioned Robust (PRo) submodular maximization algorithm, is presented in Algorithm 10. As the input, we require a non-negative monotone submodular function $f : 2^V \to \mathbb{R}_+$, the ground set of elements V, and an optimization subroutine $\mathcal{A}$. The subroutine $\mathcal{A}(k', V')$ takes a cardinality constraint k' and a ground set of elements V'. Below, we describe the properties of $\mathcal{A}$ that are used to obtain approximation guarantees.

The output of the algorithm is a set $S \subseteq V$ of size k that is robust against the worst-case removal of τ elements. The returned set consists of two sets S_0 and S_1 (Figure 6.1). S_1 is obtained by running the subroutine $\mathcal{A}$ on $V \setminus S_0$ (i.e., ignoring S_0), and is of size $k - |S_0|$. Hence, the set S_1 to which we refer as the *non-robust* part of the solution is near-optimal on $V \setminus S_0$.

Algorithm 10 Partitioned Robust Submodular optimization algorithm (PRO) [BMSC17b]

Input: Set V, k, τ, $\eta \in \mathbb{N}_+$, algorithm $\mathcal{A}$
Output: Set $S \subseteq V$ such that $|S| \leq k$

$S_0, S_1 \leftarrow \emptyset$
for $i \leftarrow 0$ **to** $\lceil \log \tau \rceil$ **do**
    for $j \leftarrow 1$ **to** $\lceil \tau / 2^i \rceil$ **do**
        $B_j \leftarrow \mathcal{A}\left(2^i \eta,\ (V \setminus S_0)\right)$
        $S_0 \leftarrow S_0 \cup B_j$
$S_1 \leftarrow \mathcal{A}\left(k - |S_0|,\ (V \setminus S_0)\right)$
$S \leftarrow S_0 \cup S_1$
return S

We refer to the set S_0 as the *robust part* of the solution set S. It consists of $\lceil \log \tau \rceil + 1$ partitions, where every partition $i \in \{0, \cdots, \lceil \log \tau \rceil\}$ consists of $\lceil \tau / 2^i \rceil$ buckets B_j, $j \in \{1, \cdots, \lceil \tau / 2^i \rceil\}$. In partition i, every bucket contains $2^i \eta$ elements, where $\eta \in \mathbb{N}_+$ is a parameter that is arbitrary for now; we use $\eta = \log^2 k$ in our asymptotic theory, but our numerical studies indicate that even $\eta = 1$ works well in practice. Each bucket B_j is created afresh by using the subroutine $\mathcal{A}$ on $V \setminus S_{0,\text{prev}}$, where $S_{0,\text{prev}}$ contains all elements belonging to the previous buckets. The following proposition bounds the cardinality of S_0, and is proved in Appendix 6.A.1.

Proposition 6.2.1. *Fix $k \geq \tau$ and $\eta \in \mathbb{N}_+$. The size of the robust part S_0 constructed in Algorithm 10 is $|S_0| = \sum_{i=0}^{\lceil \log \tau \rceil} \lceil \tau / 2^i \rceil 2^i \eta \leq 3\eta\tau(\log k + 2)$.*

This proposition reveals that the feasible values of τ (i.e., those with $|S_0| \leq k$) can be as high as $O\big(\frac{k}{\eta \tau}\big)$. We will later set $\eta = O(\log^2 k)$, thus permitting all $\tau = o(k)$ up to a few logarithmic factors. In contrast, we recall that the algorithm OSU proposed in [OSU16] adopts a simpler approach where a robust set is used consisting of τ buckets of equal size $\tau \log k$ (Figure 2.4), thereby only permitting the scaling $\tau = o(\sqrt{k})$.

We provide the following intuition as to why PRO succeeds despite having a smaller size for S_0 compared to the algorithm given in [OSU16]. First, by the design of the partitions, there always exists a bucket in partition i that at most 2^i items are removed from. The bulk of our analysis is devoted to showing that the union of these buckets yields a sufficiently high objective value. While the earlier buckets have a smaller size, they also have a higher objective value per item due to diminishing returns, and our analysis quantifies and balances this trade-off. Similarly, our analysis quantifies the trade-off between how much the adversary can remove from the (typically large) set S_1 and the robust part S_0.

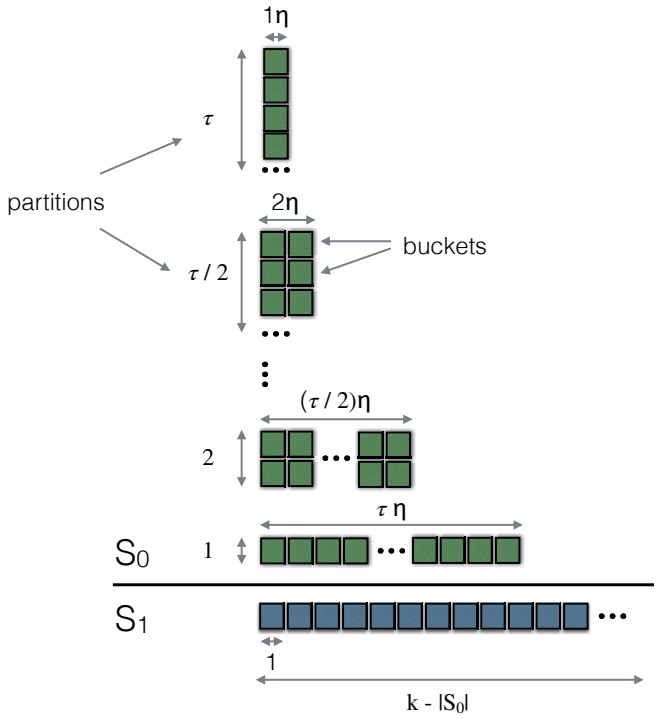


Figure 6.1: Illustration of the set $S = S_0 \cup S_1$ returned by PRO. The size of $|S_1|$ is $k - |S_0|$, and the size of $|S_0|$ is given in Prop 6.2.1. Every partition in S_0 contains the same number of elements (up to rounding).

6.2.2 Subroutine and Assumptions

PRO accepts a subroutine $\mathcal{A}$ as the input. We consider a class of algorithms that satisfy the β-*iterative property*, defined below. We assume that the algorithm outputs the final set in some specific order $(v_1, \ldots, v_k)$, and we refer to v_i as the i-*th output element*.

Definition 6.2.1. *Consider a normalized monotone submodular set function f on a ground set V, and an algorithm $\mathcal{A}$. Given any set $T \subseteq V$ and size k, suppose that $\mathcal{A}$ outputs an ordered set $(v_1, \ldots, v_k)$ when applied to T, and define $\mathcal{A}_i(T) = \{v_1, \ldots, v_i\}$ for $i \leq k$. We say that $\mathcal{A}$ satisfies the* β-iterative property *if*

$$f(\mathcal{A}_{i+1}(T)) - f(\mathcal{A}_i(T)) \geq \frac{1}{\beta} \max_{v \in T} f(v|\mathcal{A}_i(T)). \tag{6.3}$$

We recall that the notation $f(v|\mathcal{A}_i(T))$ stands for $f(\mathcal{A}_i(T) \cup \{v\}) - f(\mathcal{A}_i(T))$. Intuitively, (6.3) states that in every iteration, the algorithm adds an element whose marginal gain is at least a $1/\beta$ fraction of the maximum marginal. This necessarily requires that $\beta \geq 1$.

Examples. Besides the classic greedy algorithm, which satisfies (6.3) with $\beta = 1$, a good candidate for our subroutine is THRESHOLDING-GREEDY [BV14], which satisfies the β-iterative prop-

erty with $\beta = 1/(1-\epsilon)$. This decreases the number of function evaluations to $O(n/\epsilon \log n/\epsilon)$.

The STOCHASTIC-GREEDY [MBK+15] algorithm is another potential subroutine candidate that can be used. While it is unclear whether this algorithm satisfies the β-iterative property, it requires an even smaller number of function evaluations, namely, it requires $O(n \log 1/\epsilon)$. We will see in Section 6.3 that PRO performs well empirically when used with this subroutine. We henceforth refer to PRO used along with its appropriate subroutine as PRO-GREEDY, PRO-THRESHOLDING-GREEDY, etc.

Properties. Throughout the chapter, we use $\mathrm{OPT}(k, V)$ to denote the following optimal set of size k in the case of *non-robust* submodular maximization:

$$\mathrm{OPT}(k, V) \in \operatorname*{arg\,max}_{S \subseteq V, |S|=k} f(S).$$

The following lemma generalizes a classical property of the greedy algorithm [NWF78, KG12] (provided in Section 2.3) to the class of algorithms satisfying the β-iterative property.

Lemma 6.2.1. *Consider a normalized monotone submodular function $f : 2^V \to \mathbb{R}_+$ and an algorithm $\mathcal{A}(T)$, $T \subseteq V$, that satisfies the β-iterative property in* (6.3)*. Let $\mathcal{A}_l(T)$ denote the set returned by the algorithm $\mathcal{A}(T)$ after l iterations. Then for all $k, l \in \mathbb{N}_+$*

$$f(\mathcal{A}_l(T)) \geq \left(1 - e^{-\frac{l}{\beta k}}\right) f(\mathrm{OPT}(k, T)). \tag{6.4}$$

We will also make use of the following property, which is implied by the β-iterative property.

Proposition 6.2.2. *Consider a submodular set function $f : 2^V \to \mathbb{R}_+$ and an algorithm $\mathcal{A}$ that satisfies the β-iterative property for some $\beta \geq 1$. Then, for any $T \subseteq V$ and element $e \in V \setminus \mathcal{A}(T)$ we have that*

$$f(e|\mathcal{A}(T)) \leq \beta \frac{f(\mathcal{A}(T))}{k}. \tag{6.5}$$

Intuitively, (6.5) states that the marginal gain of any non-selected element cannot be more than β times the average objective value of the selected elements. Proofs of Lemma 6.2.1 and Proposition 6.2.2 can be found in Appendix 6.A.3 and 6.A.2, respectively.

6.2.3 Main Result: Approximation Guarantee

For the robust maximization problem, we let $\mathrm{OPT}(k, V, \tau)$ denote the optimal set:

$$\mathrm{OPT}(k, V, \tau) \in \operatorname*{arg\,max}_{S \subseteq V, |S|=k} \min_{E \subseteq S, |E| \leq \tau} f(S \setminus E).$$

Moreover, for a set S, we let E^*_S denote the minimizer

$$E^*_S \in \operatorname*{arg\,min}_{E\subseteq S, |E|\leq\tau} f(S \setminus E). \tag{6.6}$$

With these definitions in place, the main theoretical result of this chapter is as follows.

Theorem 6.2.1. *Let f be a normalized monotone submodular function, and let $\mathcal{A}$ be a subroutine satisfying the β-iterative property. For a given budget k and parameters $2 \leq \tau \leq \frac{k}{3\eta(\log k+2)}$ and $\eta \geq 4(\log k + 1)$,* PRO *returns a set S of size k such that*

$$f(S \setminus E^*_S) \geq \frac{\frac{\eta}{5\beta^3\lceil\log\tau\rceil+\eta}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right)}{1 + \frac{\eta}{5\beta^3\lceil\log\tau\rceil+\eta}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right)} f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}), \tag{6.7}$$

where E^*_S and $E^*_{\mathrm{OPT}(k,V,\tau)}$ are defined as in (6.6)*.*

In addition, if $\tau = o\left(\frac{k}{\eta \log k}\right)$ and $\eta \geq \log^2 k$, then we have the following as $k \to \infty$:

$$f(S \setminus E^*_S) \geq \left(\frac{1 - e^{-1/\beta}}{2 - e^{-1/\beta}} + o(1)\right) f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}). \tag{6.8}$$

In particular, the PRO-GREEDY *algorithm achieves an asymptotic approximation factor of at least 0.387, and* PRO-THRESHOLDING-GREEDY *with the input parameter ϵ achieves an asymptotic approximation factor of at least $0.387(1 - \epsilon)$.*

This result solves an open problem raised in [OSU16], namely, whether a constant-factor approximation guarantee can be obtained for $\tau = o(k)$ as opposed to only $\tau = o\big(\sqrt{k}\big)$. In the asymptotic limit, our constant factor of 0.387 for the greedy subroutine matches that of [OSU16], but our algorithm permits significantly "higher robustness" in the sense of allowing larger τ values. To achieve this, we require novel proof techniques, which we now outline.

6.2.4 High-level Overview of the Analysis

The proof of Theorem 6.2.1 is provided in Appendix 6.A.4. Here we provide a high-level overview of the main challenges.

Let E denote a cardinality-τ subset of the returned set S that is removed. By the construction of the partitions, it is easy to verify that each partition i contains a bucket from which at most 2^i items are removed. We denote these by $B_0, \ldots, B_{\lceil\log\tau\rceil}$, and write $E_{B_i} := E \cap B_i$. Moreover, we define $E_0 := E \cap S_0$ and $E_1 := E \cap S_1$.

We establish the following lower bound on the final objective function value:

$$f(S \setminus E) \geq \max \left\{ f(S_0 \setminus E_0), f(S_1) - f(E_1 | (S \setminus E)), f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \right) \right\}. \tag{6.9}$$

The arguments to the first and third terms are trivially seen to be subsets of the solution set S after the adversarial removal E, i.e., $S \setminus E$, and the second term represents the utility of the set S_1 subsided by the utility of the elements removed from S_1.

The first two terms above are easily lower bounded by convenient expressions via submodular and the β-iterative property. The bulk of the proof is dedicated to bounding the third term. To do this, we establish the following recursive relations with suitably-defined "small" values of α_j:

$$f\left(\bigcup_{i=0}^{j} (B_i \setminus E_{B_i}) \right) \geq \left(1 - \frac{1}{1 + \frac{1}{\alpha_j}} \right) f(B_j) \tag{6.10}$$

$$f\left(E_{B_j} \;\middle|\; \bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i}) \right) \leq \alpha_j f\left(\bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i}) \right). \tag{6.11}$$

Intuitively, the first equation shows that the objective value from buckets $i = 0, \ldots, j$ *with removals* cannot be too much smaller than the value in bucket j *without removals*, and the second one shows that the loss in bucket j due to the removals is at most a small fraction of the objective value from buckets $0, \ldots, j-1$. The proofs of both the base case of the induction and the inductive step make use of submodularity properties and the β-iterative property (*cf.*, Def. 6.2.1).

Once the suitable lower bounds are obtained for the terms in (6.9), we can show that as the second term increases, the third term decreases, and accordingly lower bound their maximum by the value obtained when the two are equal. A similar balancing argument is then applied to the resulting term and the first term in (6.9). Finally, the condition $\tau \leq \frac{k}{3\eta(\log k + 2)}$ follows directly from Proposition 6.2.1; namely, it is a sufficient condition for $|S_0| \leq k$, as is required by PRO.

6.3 Experimental Evaluation

In this section, we numerically validate the performance of PRO and the claims given in the preceding sections. In particular, we compare our algorithm against the OSU algorithm proposed in [OSU16] on different datasets and corresponding objective functions (see Table 6.2). We demonstrate matching or improved performance in a broad range of settings, as well as observing that PRO can be implemented with larger values of τ, corresponding to a greater robustness. Moreover, we show that for certain real-world data sets, the classic GREEDY algorithm can perform badly for the robust problem. We do not compare against SATURATE [KMGG08], due to its high computational cost for even a small τ.

Dataset	n	dimension	f
Tiny-10k	10 000	3074	Exemplar
Tiny-50k	50 000	3074	Exemplar
CM-Molecules	7211	276	Exemplar
Network	# nodes	# edges	f
ego-Facebook	4039	88 234	DomSet
ego-Twitter	81 306	1 768 149	DomSet

Table 6.2: Datasets and corresponding objective functions.

Setup. Given a solution set S of size k, we measure the performance in terms of the minimum objective value upon the worst-case removal of τ elements, i.e. $\min_{E \subseteq S, |E| \leq \tau} f(S \setminus E)$. Unfortunately, for a given solution set S, finding such a set E is an instance of the submodular minimization problem with a cardinality constraint,[2] which is known to be NP-hard with polynomial approximation factors [SF11]. Hence, in our experiments, we only implement the optimal "adversary" (i.e., removal of items) for small to moderate values of τ and k, for which we use a fast C++ implementation of branch-and-bound.

Despite the difficulty in implementing the optimal adversary, we observed in our experiments that the *greedy adversary*, which iteratively removes elements to reduce the objective value as much as possible, has a similar impact on the objective compared to the optimal adversary for the data sets considered. Hence, we also provide a larger-scale experiment in the presence of a greedy adversary. Throughout, we write OA and GA to abbreviate the optimal adversary and greedy adversary, respectively.

In our experiments, the size of the robust part of the solution set (i.e., $|S_0|$) is set to τ^2 and $\tau \log \tau$ for OSU and PRO, respectively. That is, we set $\eta = 1$ in PRO, and similarly ignore constant and logarithmic factors in OSU, since both appear to be unnecessary in practice. We show both the "raw" objective values of the solutions, as well as the objective values after the removal of τ elements. In all experiments, we implement GREEDY using the LAZY-GREEDY implementation given in [Min78].

The objective functions shown in Table 6.2 are given in Section 6.1.3. For the exemplar objective function, we use $d(s, v) = \|s - v\|^2$, and let the reference element e_0 be the zero vector. Instead of using the whole set V, we approximate the objective by considering a smaller random subset of V for improved computational efficiency. Since the objective is additively decomposable and bounded, standard concentration bounds (e.g., the Chernoff bound) ensure that the empirical mean over a random subsample can be made arbitrarily accurate.

Data sets. We consider the following datasets, along with the objectives given in Section 6.1.3:

[2] This can be seen by noting that for submodular f and any $E \subseteq X \subseteq V$, $f'(E) = f(X \setminus E)$ remains submodular.

- EGO-FACEBOOK: This network data consists of social circles (or friends lists) from Facebook forming an undirected graph with 4039 nodes and 88234 edges.

- EGO-TWITTER: This dataset consists of 973 social circles from Twitter, forming a directed graph with 81306 nodes and 1768149 edges. Both EGO-FACEBOOK and EGO-TWITTER were used previously in [ML14].

- TINY10K and TINY50K: We used two Tiny Images data sets of size $10k$ and $50k$ consisting of images each represented as a 3072-dimensional vector [TFF08]. Besides the number of images, these two datasets also differ in the number of classes that the images are grouped into. We shift each vector to have zero mean.

- CM-MOLECULES: It consists of 7211 small organic molecules, each represented as a 276 dimensional vector. Each vector is obtained by processing the molecule's *Coulomb* matrix representation [Rup15]. We shift and normalize each vector to zero mean and unit norm.

Results. In the first set of experiments, we compare PRO $-$ GREEDY (written using the shorthand PRO-GR in the legend) against GREEDY and OSU on the EGO-FACEBOOK and EGO-TWITTER datasets. In this experiment, the dominating set selection objective in (6.1) is considered. Figure 2 (a) and (c) show the results before and after the worst-case removal of $\tau = 7$ elements for different values of k. In Figure 2 (b) and (d), we show the objective value for fixed $k = 50$ and $k = 100$, respectively, while the robustness parameter τ is varied.

GREEDY achieves the highest raw objective value, followed by PRO-GREEDY and OSU. However, after the worst-case removal, PRO-GREEDY-OA outperforms both OSU-OA and GREEDY-OA. In Figure 2 (a) and (b), GREEDY-OA performs poorly due to a high concentration of the objective value on the first few elements selected by GREEDY. While OSU requires $k \geq \tau^2$, PRO only requires $k \geq \tau \log \tau$, and hence it can be run for larger values of τ (e.g., see Figure 2 (b) and (c)). Moreover, in Figure 2 (a) and (b), we can observe that although PRO uses a smaller number of elements to build the robust part of the solution set, it has better robustness in comparison with OSU.

In the second set of experiments, we perform the same type of comparisons on the TINY10 and CM-MOLECULES datasets. The exemplar based clustering in (6.2) is used as the objective function. In Figure 2 (e) and (h), the robustness parameter is fixed to $\tau = 7$ and $\tau = 6$, respectively, while the cardinality k is varied. In Figure 2 (f) and (h), the cardinality is fixed to $k = 100$ and $k = 50$, respectively, while the robustness parameter τ is varied.

Again, GREEDY achieves the highest objective value. On TINY10, GREEDY-OA (Figure 2 (e) and (f)) has a large gap between the raw and final objective, but it still slightly outperforms PRO-GREEDY-OA. This demonstrates that GREEDY can work well in some cases, despite failing in others. We observed that it succeeds here because the objective value is relatively more uniformly spread across the selected elements. On the same dataset, PRO-GREEDY-OA outperforms

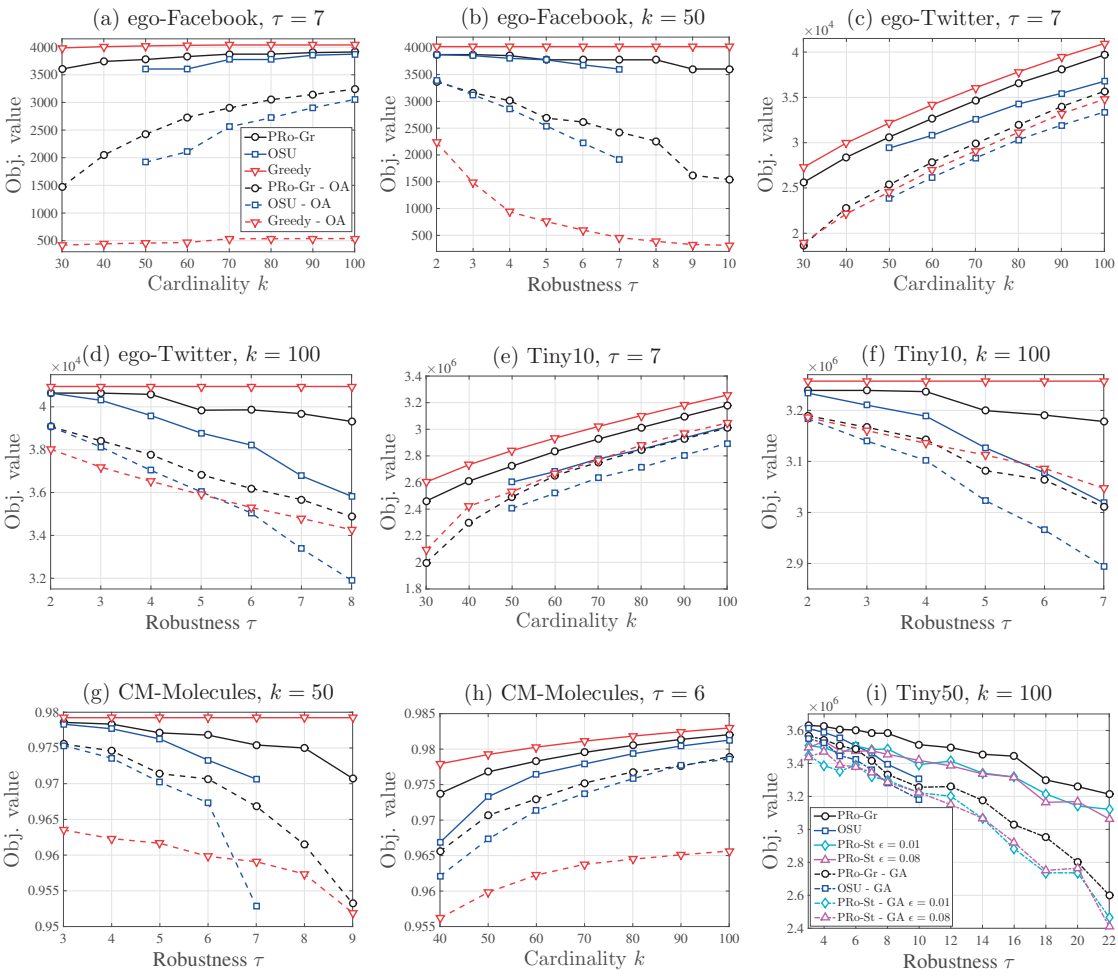


Figure 6.2: Numerical comparisons of the algorithms PRO-GREEDY, GREEDY and OSU, and their objective values PRO-OA, OSU-OA and GREEDY-OA once τ elements are removed. Figure (i) shows the performance on the larger scale experiment where both GREEDY and STOCHASTIC-GREEDY are used as subroutines in PRO.

OSU-OA. On our second dataset CM-MOLECULES (Figure 2 (g) and (h)), PRO-GREEDY-OA achieves the highest robust objective value, followed by OSU-OA and GREEDY-OA.

In our final experiment (see Figure 2 (i)), we compare the performance of PRO $-$ GREEDY against two instances of PRO $-$ STOCHASTIC-GREEDY with $\epsilon = 0.01$ and $\epsilon = 0.08$ (shortened to PRO-ST in the legend), seeking to understand to what extent using the more efficient stochastic subroutine impacts the performance. We also show the performance of OSU. In this experiment, we fix $k = 100$ and vary τ. We use the greedy adversary instead of the optimal one, since the latter becomes computationally challenging for larger values of τ.

In Figure 2 (i), we observe a slight decrease in the objective value in the case of PRO $-$

STOCHASTIC-GREEDY due to the stochastic optimization. On the other hand, the gaps between the robust and non-robust solutions remain similar, or even shrink. Overall, we observe that at least in this example, the stochastic subroutine does not compromise the quality of the solution too significantly, despite having a lower computational complexity.

6.A Proofs

6.A.1 Proof of Proposition 6.2.1

We have

$$\begin{aligned}
|S_0| &= \sum_{i=0}^{\lceil \log \tau \rceil} \lceil \tau/2^i \rceil 2^i \eta \\
&\leq \sum_{i=0}^{\lceil \log \tau \rceil} \left(\frac{\tau}{2^i} + 1 \right) 2^i \eta \\
&\leq \eta(\lceil \log \tau \rceil + 1)(\tau + 2^{\lceil \log \tau \rceil}) \\
&\leq 3\eta\tau(\lceil \log \tau \rceil + 1) \\
&\leq 3\eta\tau(\log k + 2).
\end{aligned}$$

6.A.2 Proof of Proposition 6.2.2

Recalling that $\mathcal{A}_j(T)$ denotes a set constructed by the algorithm after j iterations, we have

$$\begin{aligned}
f(\mathcal{A}_j(T)) - f(\mathcal{A}_{j-1}(T)) &\geq \frac{1}{\beta} \max_{e \in T} f(e|\mathcal{A}_{j-1}(T)) \\
&\geq \frac{1}{\beta} \max_{e \in T} f(e|\mathcal{A}_k(T)) \\
&\geq \frac{1}{\beta} \max_{e \in T \setminus \mathcal{A}_k(T)} f(e|\mathcal{A}_k(T)), \qquad (6.12)
\end{aligned}$$

where the first inequality follows from the β-iterative property (6.3), and the second inequality follows from $\mathcal{A}_{j-1}(S) \subseteq \mathcal{A}_k(S)$ and the submodularity of f.

Continuing, we have

$$\begin{aligned}
f(\mathcal{A}_k(T)) &= \sum_{j=1}^{k} f(\mathcal{A}_j(T)) - f(\mathcal{A}_{j-1}(T)) \\
&\geq \frac{k}{\beta} \max_{e \in T \setminus \mathcal{A}_k(T)} f(e|\mathcal{A}_k(T)),
\end{aligned}$$

where the last inequality follows from (6.12).

By rearranging, we have for any $e \in T \setminus \mathcal{A}_k(T)$ that

$$f(e|\mathcal{A}_k(T)) \leq \beta \frac{f(\mathcal{A}_k(T))}{k}.$$

6.A.3 Proof of Lemma 6.2.1

Recalling that $A_j(T)$ denotes the set constructed after j iterations when applied to T, we have

$$\begin{aligned}\max_{e\in T\setminus A_{j-1}(T)} f(e|A_{j-1}(T)) &\geq \frac{1}{k}\sum_{e\in \mathrm{OPT}(k,T)\setminus A_{j-1}(T)} f(e|A_{j-1}(T)) \\ &\geq \frac{1}{k} f(\mathrm{OPT}(k,T)|A_{j-1}(T)) \\ &\geq \frac{1}{k}\big(f(\mathrm{OPT}(k,T)) - f(A_{j-1}(T))\big), \end{aligned} \tag{6.13}$$

where the first line holds since the maximum is lower bounded by the average, the line uses submodularity, and the last line uses monotonicity.

By combining the β-iterative property with (6.13), we obtain

$$\begin{aligned} f(\mathcal{A}_j(T)) - f(\mathcal{A}_{j-1}(T)) &\geq \frac{1}{\beta}\max_{e\in T\setminus A_{j-1}(T)} f(e|A_{j-1}(T)) \\ &\geq \frac{1}{k\beta}\big(f(\mathrm{OPT}(k,T)) - f(A_{j-1}(T))\big). \end{aligned}$$

By rearranging, we obtain

$$f(\mathrm{OPT}(k,T)) - f(A_{j-1}(T)) \leq \beta k\big(f(\mathcal{A}_j(T)) - f(\mathcal{A}_{j-1}(T))\big). \tag{6.14}$$

We proceed by following the steps from the proof of Theorem 1.5 in [KG12]. Defining $\delta_j := f(\mathrm{OPT}(k,T)) - f(A_j(T))$, we can rewrite (6.14) as $\delta_{j-1} \leq \beta k(\delta_{j-1} - \delta_j)$. By rearranging, we obtain

$$\delta_j \leq \left(1 - \frac{1}{\beta k}\right)\delta_{j-1}.$$

Applying this recursively, we obtain $\delta_l \leq \big(1 - \frac{1}{\beta k}\big)^l \delta_0$, where $\delta_0 = f(\mathrm{OPT}(k,T))$ since f is normalized (i.e., $f(\emptyset) = 0$). Finally, applying $1 - x \leq e^{-x}$ and rearranging, we obtain

$$f(\mathcal{A}_l(T)) \geq \left(1 - e^{-\frac{l}{\beta k}}\right) f(\mathrm{OPT}(k,T)).$$

6.A.4 Proof of Theorem 6.2.1

Technical Lemmas

We first provide several technical lemmas that will be used throughout the proof. We begin with a simple property of submodular functions.

Lemma 6.A.1. *For any submodular function f on a ground set V, and any sets $A, B, R \subseteq V$, we have*

$$f(A \cup B) - f(A \cup (B \setminus R)) \leq f(R \mid A).$$

Proof. Define $R_2 := A \cap R$, and $R_1 := R \setminus A = R \setminus R_2$. We have

$$\begin{aligned} f(A \cup B) - f(A \cup (B \setminus R)) &= f(A \cup B) - f((A \cup B) \setminus R_1) \\ &= f(R_1 \mid (A \cup B) \setminus R_1) \\ &\leq f(R_1 \mid (A \setminus R_1)) && (6.15) \\ &= f(R_1 \mid A) && (6.16) \\ &= f(R_1 \cup R_2 \mid A) && (6.17) \\ &= f(R \mid A), \end{aligned}$$

where (6.15) follows from the submodularity of f, (6.16) follows since A and R_1 are disjoint, and (6.17) follows since $R_2 \subseteq A$. □

The next lemma provides a simple lower bound on the maximum of two quantities; it is stated formally since it will be used on multiple occasions.

Lemma 6.A.2. *For any set function f, sets A, B, and constant $\alpha > 0$, we have*

$$\max\{f(A), f(B) - \alpha f(A)\} \geq \left(\frac{1}{1+\alpha}\right) f(B), \tag{6.18}$$

and

$$\max\{\alpha f(A), f(B) - f(A)\} \geq \left(\frac{\alpha}{1+\alpha}\right) f(B). \tag{6.19}$$

Proof. Starting with (6.18), we observe that one term is increasing in $f(A)$ and the other is decreasing in $f(A)$. Hence, the maximum over all possible $f(A)$ is achieved when the two terms are equal, i.e., $f(A) = \frac{1}{1+\alpha} f(B)$. We obtain (6.19) via the same argument. □

The following lemma relates the function values associated with two buckets formed by PRO, denoted by X and Y. It is stated with respect to an arbitrary set E_Y, but when we apply the lemma, this will correspond to the elements of Y that are removed by the adversary.

Lemma 6.A.3. *Under the setup of Theorem 6.2.1, let X and Y be buckets of* PRO *such that Y is constructed at a later time than X. For any set $E_Y \subseteq Y$, we have*

$$f(X \cup (Y \setminus E_Y)) \geq \frac{1}{1+\alpha} f(Y),$$

and

$$f(E_Y \mid X) \leq \alpha f(X), \tag{6.20}$$

where

$$\alpha = \beta^{\frac{|E_Y|}{|X|}}.$$

Proof. Inequality (6.20) follows from the β-iterative property of $\mathcal{A}$; specifically, we have from (6.5) that

$$f(e|X) \leq \beta \frac{f(X)}{|X|},$$

where e is any element of the ground set that is neither in X nor any bucket constructed before X. Hence, we can write

$$f(E_Y \mid X) \leq \sum_{e \in E_Y} f(e|X) \leq \beta \frac{|E_Y|}{|X|} f(X) = \alpha f(X),$$

where the first inequality is by submodularity. This proves (6.20).

Next, we write

$$\begin{aligned} f(Y) - f(X \cup (Y \setminus E_Y)) &\leq f(X \cup Y) - f(X \cup (Y \setminus E_Y)) && (6.21) \\ &\leq f(E_Y \mid X), && (6.22) \end{aligned}$$

(6.21) is by monotonicity, and (6.22) is by Lemma 6.A.1 with $A = X$, $B = Y$, and $R = E_Y$.

Combining (6.20) and (6.22), together with the fact that $f(X \cup (Y \setminus E_Y)) \geq f(X)$ (by monotonicity), we have

$$f(X \cup (Y \setminus E_Y)) \geq \max\{f(X), f(Y) - \alpha f(X)\} \qquad \geq \frac{1}{1+\alpha} f(Y), \tag{6.23}$$

where (6.23) follows from (6.18). □

Finally, we provide a lemma that will later be used to take two bounds that are known regarding the previously-constructed buckets, and use them to infer bounds regarding the next bucket.

Lemma 6.A.4. *Under the setup of Theorem 6.2.1, let Y and Z be buckets of* PRO *such that Z is constructed at a later time than Y, and let $E_Y \subseteq Y$ and $E_Z \subseteq Z$ be arbitrary sets. Moreover, let X be a set (not necessarily a bucket) such that*

$$\begin{aligned} f((Y \setminus E_Y) \cup X) &\geq \frac{1}{1+\alpha} f(Y), && (6.24) \\ f(E_Y \mid X) &\leq \alpha f(X). && (6.25) \end{aligned}$$

Then, we have

$$\begin{aligned} f(E_Z \mid (Y \setminus E_Y) \cup X) &\leq \alpha_{\text{next}} f((Y \setminus E_Y) \cup X), && (6.26) \\ f((Z \setminus E_Z) \cup (Y \setminus E_Y) \cup X) &\geq \frac{1}{1+\alpha_{\text{next}}} f(Z), && (6.27) \end{aligned}$$

where $\alpha_{\text{next}} = \beta \frac{|E_Z|}{|Y|}(1+\alpha) + \alpha$.

Proof. We first prove (6.26):

$$
\begin{aligned}
f(E_Z \mid (Y \setminus E_Y) \cup X) &= f((Y \setminus E_Y) \cup X \cup E_Z) - f((Y \setminus E_Y) \cup X) \\
&\leq f(X \cup Y \cup E_Z) - f((Y \setminus E_Y) \cup X) && (6.28)\\
&= f(E_Z \mid X \cup Y) + f(X \cup Y) - f((Y \setminus E_Y) \cup X) \\
&\leq f(E_Z \mid Y) + f(X \cup Y) - f((Y \setminus E_Y) \cup X) && (6.29)\\
&\leq \beta \frac{|E_Z|}{|Y|} f(Y) + f(X \cup Y) - f((Y \setminus E_Y) \cup X) && (6.30)\\
&\leq \beta \frac{|E_Z|}{|Y|}(1+\alpha) f((Y \setminus E_Y) \cup X) + f(X \cup Y) - f((Y \setminus E_Y) \cup X) && (6.31)\\
&\leq \beta \frac{|E_Z|}{|Y|}(1+\alpha) f((Y \setminus E_Y) \cup X) + f(E_Y \mid (Y \setminus E_Y) \cup X) && (6.32)\\
&\leq \beta \frac{|E_Z|}{|Y|}(1+\alpha) f((Y \setminus E_Y) \cup X) + f(E_Y \mid X) && (6.33)\\
&\leq \beta \frac{|E_Z|}{|Y|}(1+\alpha) f((Y \setminus E_Y) \cup X) + \alpha f(X) && (6.34)\\
&\leq \beta \frac{|E_Z|}{|Y|}(1+\alpha) f((Y \setminus E_Y) \cup X) + \alpha f((Y \setminus E_Y) \cup X) && (6.35)\\
&= \left(\beta \frac{|E_Z|}{|Y|}(1+\alpha) + \alpha\right) f((Y \setminus E_Y) \cup X)., && (6.36)
\end{aligned}
$$

where: (6.28) and (6.29) follow by monotonicity and submodularity, respectively; (6.30) follows from the second part of Lemma 6.A.3; (6.31) follows from (6.24); (6.32) is obtained by applying Lemma 6.A.1 for $A = X$, $B = Y$, and $R = E_Y$; (6.33) follows by submodularity; (6.34) follows from (6.25); (6.35) follows by monotonicity. Finally, by defining $\alpha_{\text{next}} := \beta \frac{|E_Z|}{|Y|}(1+\alpha) + \alpha$ in (6.36) we establish the bound in (6.26).

In the rest of the proof, we show that (6.27) holds as well. First, we have

$$
f((Z \setminus E_Z) \cup (Y \setminus E_Y) \cup X) \geq f(Z) - f(E_Z \mid (Y \setminus E_Y) \cup X) \tag{6.37}
$$

by Lemma 6.A.1 with $B = Z$, $R = E_Z$ and $A = (Y \setminus E_Y) \cup X$. Now we can use the derived bounds (6.36) and (6.37) to obtain

$$
\begin{aligned}
f((Z \setminus E_Z) \cup (Y \setminus E_Y) \cup X) &\geq f(Z) - f(E_Z \mid (Y \setminus E_Y) \cup X) \\
&\geq f(Z) - \left(\beta \frac{|E_Z|}{|Y|}(1+\alpha) + \alpha\right) f((Y \setminus E_Y) \cup X).
\end{aligned}
$$

Finally, we have

$$\begin{aligned} f((Z \setminus E_Z) \cup (Y \setminus E_Y) \cup X) &\geq \max \Bigg\{ f((Y \setminus E_Y) \cup X), \\ &\qquad f(Z) - \left(\beta \frac{|E_Z|}{|Y|}(1+\alpha) + \alpha \right) f((Y \setminus E_Y) \cup X) \Bigg\} \\ &\geq \frac{1}{1+\alpha_{\text{next}}} f(Z), \end{aligned}$$

where the last inequality follows from Lemma 6.A.1.

□

Observe that the results we obtain on $f(E_Z \mid (Y \setminus E_Y) \cup X)$ and on $f((Z \setminus E_Z) \cup (Y \setminus E_Y) \cup X)$ in Lemma 6.A.4 are of the same form as the pre-conditions of the lemma. This will allow us to apply the lemma recursively.

Characterizing the Adversary

Let E denote a set of elements removed by an adversary, where $|E| \leq \tau$. Within S_0, PRO constructs $\lceil \log \tau \rceil + 1$ partitions. Each partition $i \in \{0, \ldots, \lceil \log \tau \rceil\}$ consists of $\lceil \tau / 2^i \rceil$ buckets, each of size $2^i \eta$, where $\eta \in \mathbb{N}$ will be specified later. We let B denote a generic bucket, and define E_B to be all the elements removed from this bucket, i.e. $E_B = B \cap E$.

The next lemma identifies a bucket in each partition for which not too many elements are removed.

Lemma 6.A.5. *Under the setup of Theorem 6.2.1, suppose that an adversary removes a set E of size at most τ from the set S constructed by* PRO*. Then for each partition i, there exists a bucket B_i such that $|E_{B_i}| \leq 2^i$, i.e., at most 2^i elements are removed from this bucket.*

Proof. Towards contradiction, assume that this is not the case, i.e., assume $|E_{B_i}| > 2^i$ for every bucket of the i-th partition. As the number of buckets in partition i is $\lceil \tau / 2^i \rceil$, this implies that the adversary has to spend a budget of

$$|E| \geq 2^i |E_{B_i}| > 2^i \lceil \tau / 2^i \rceil = \tau,$$

which is in contradiction with $|E| \leq \tau$. □

We consider $B_0, \ldots, B_{\lceil \log \tau \rceil}$ as above, and show that even in the worst case, we have that $f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \right)$ is almost as large as $f\left(B_{\lceil \log \tau \rceil} \right)$ for appropriately set η. To achieve this, we apply Lemma 6.A.4 multiple times, as illustrated in the following lemma. We henceforth write $\eta_h := \eta / 2$ for brevity.

Lemma 6.A.6. *Under the setup of Theorem 6.2.1, suppose that an adversary removes a set E of size at most τ from the set S constructed by* PRO*, and let $B_0, \cdots, B_{\lceil \log \tau \rceil}$ be buckets such that $|E_{B_i}| \leq 2^i$ for each $i \in \{1, \cdots \lceil \log \tau \rceil\}$ (*cf.*, Lemma 6.A.5). Then,*

$$f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right) \geq \left(1 - \frac{1}{1+\frac{1}{\alpha}}\right) f\left(B_{\lceil \log \tau \rceil}\right) = \frac{1}{1+\alpha} f\left(B_{\lceil \log \tau \rceil}\right), \tag{6.38}$$

and

$$f\left(E_{B_{\lceil \log \tau \rceil}} \;\middle|\; \bigcup_{i=0}^{\lceil \log \tau \rceil - 1} (B_i \setminus E_{B_i})\right) \leq \alpha f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil - 1} (B_i \setminus E_{B_i})\right), \tag{6.39}$$

for some

$$\alpha \leq \beta^2 \frac{(1+\eta_h)^{\lceil \log \tau \rceil} - \eta_h^{\lceil \log \tau \rceil}}{\eta_h^{\lceil \log \tau \rceil}}. \tag{6.40}$$

Proof. In what follows, we focus on the case where there exists some bucket B_0 in partition $i = 0$ such that $B_0 \setminus E_{B_0} = B_0$. If this is not true, then E must be contained entirely within this partition, since it contains τ buckets. As a result, (i) we trivially obtain (6.38) even when α is replaced by zero, since the union on the left-hand side contains $B_{\lceil \log \tau \rceil}$; (ii) (6.39) becomes trivial since the left-hand side is zero is a result of $E_{B_{\lceil \log \tau \rceil}} = \emptyset$.

We proceed by induction. Namely, we show that

$$f\left(\bigcup_{i=0}^{j} (B_i \setminus E_{B_i})\right) \geq \left(1 - \frac{1}{1+\frac{1}{\alpha_j}}\right) f(B_j) = \frac{1}{1+\alpha_j} f(B_j), \text{ and} \tag{6.41}$$

$$f\left(E_{B_j} \;\middle|\; \bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i})\right) \leq \alpha_j f\left(\bigcup_{i=0}^{j-1} (B_i \setminus E_{B_i})\right), \tag{6.42}$$

for every $j \geq 1$, where

$$\alpha_j \leq \beta^2 \frac{(1+\eta_h)^j - \eta_h^j}{\eta_h^j}. \tag{6.43}$$

Upon showing this, the lemma is concluded by setting $j = \lceil \log \tau \rceil$.

Base case $j = 1$. In the case that $j = 1$, taking into account that $E_{B_0} = \emptyset$, we observe from (6.41) that our goal is to bound $f(B_0 \cup (B_1 \setminus E_{B_1}))$. Applying Lemma 6.A.3 with $X = B_0$, $Y = B_1$, and $E_Y = E_{B_1}$, we obtain

$$\begin{aligned} f(B_0 \cup (B_1 \setminus E_{B_1})) &\geq \frac{1}{1+\alpha_1} f(B_1), \\ f(E_{B_1} \mid B_0) &\leq \alpha_1 f(B_0), \\ \alpha_1 &= \beta \tfrac{|E_{B_1}|}{|B_0|}. \end{aligned}$$

We have $|B_0| = \eta$, while $|E_{B_1}| \leq 2$ by assumption. Hence, we can bound α_1 and rewrite as

$$\alpha_1 \leq \beta\frac{2}{\eta} = \beta\frac{1}{\eta_h} = \beta\frac{(1+\eta_h)-\eta_h}{\eta_h} \leq \beta^2\frac{(1+\eta_h)-\eta_h}{\eta_h},$$

where the last inequality follows since $\beta \geq 1$ by definition.

Inductive step. Fix $j \geq 2$. Assuming that the inductive hypothesis holds for $j-1$, we want to show that it holds for j as well.

We write

$$f\left(\bigcup_{i=0}^{j}(B_i \setminus E_{B_i})\right) = f\left(\left(\bigcup_{i=0}^{j-1}(B_i \setminus E_{B_i})\right) \cup (B_j \setminus E_{B_j})\right),$$

and apply Lemma 6.A.4 with $X = \bigcup_{i=0}^{j-2}(B_i \setminus E_{B_i})$, $Y = B_{j-1}$, $E_Y = E_{B_{j-1}}$, $Z = B_j$, and $E_Z = E_{B_j}$. Note that the conditions (6.24) and (6.25) of Lemma 6.A.4 are satisfied by the inductive hypothesis. Hence, we conclude that (6.41) and (6.42) hold with $\alpha_j = \beta\frac{|E_{B_j}|}{|B_{j-1}|}(1+\alpha_{j-1}) + \alpha_{j-1}$. It remains to show that (6.43) holds for α_j, assuming it holds for α_{j-1}. We have

$$\begin{aligned}
\alpha_j &= \beta\frac{|E_{B_j}|}{|B_{j-1}|}(1+\alpha_{j-1}) + \alpha_{j-1} \\
&\leq \beta\frac{1}{\eta_h}\left(1 + \beta\frac{(1+\eta_h)^{j-1} - \eta_h^{j-1}}{\eta_h^{j-1}}\right) + \beta\frac{(1+\eta_h)^{j-1} - \eta_h^{j-1}}{\eta_h^{j-1}} && (6.44)\\
&\leq \beta^2\left(\frac{1}{\eta_h}\left(1 + \frac{(1+\eta_h)^{j-1} - \eta_h^{j-1}}{\eta_h^{j-1}}\right) + \frac{(1+\eta_h)^{j-1} - \eta_h^{j-1}}{\eta_h^{j-1}}\right) && (6.45)\\
&= \beta^2\left(\frac{1}{\eta_h}\frac{(1+\eta_h)^{j-1}}{\eta_h^{j-1}} + \frac{(1+\eta_h)^{j-1} - \eta_h^{j-1}}{\eta_h^{j-1}}\right) \\
&= \beta^2\left(\frac{(1+\eta_h)^{j-1}}{\eta_h^{j}} + \frac{\eta_h(1+\eta_h)^{j-1} - \eta_h^{j}}{\eta_h^{j}}\right) \\
&= \beta^2\frac{(1+\eta_h)^{j} - \eta_h^{j}}{\eta_h^{j}},
\end{aligned}$$

where (6.44) follows from (6.43) and the fact that

$$\beta\frac{|E_{B_j}|}{|B_{j-1}|} \leq \beta\frac{2^j}{2^{j-1}\eta} = \beta\frac{2}{\eta} = \beta\frac{1}{\eta_h},$$

by $|E_{B_j}| \leq 2^j$ and $|B_{j-1}| = 2^{j-1}\eta$; and (6.45) follows since $\beta \geq 1$. □

Inequality (6.43) provides an upper bound on α_j, but it is not immediately clear how the bound varies with j. The following lemma provides a more compact form.

Lemma 6.A.7. *Under the setup of Lemma 6.A.6, we have for* $2\lceil \log \tau \rceil \leq \eta_h$ *that*

$$\alpha_j \leq 3\beta^2 \frac{j}{\eta} \tag{6.46}$$

Proof. We unfold the right-hand side of (6.43) in order to express it in a simpler way. First, consider $j = 1$. From (6.43) we obtain $\alpha_1 \leq 2\beta^2 \frac{1}{\eta}$, as required. For $j \geq 2$, we obtain:

$$\begin{aligned}
\beta^{-2}\alpha_j &\leq \frac{(1+\eta_h)^j - \eta_h^j}{\eta_h^j} \\
&= \sum_{i=0}^{j-1} \binom{j}{i} \frac{\eta_h^i}{\eta_h^j} && (6.47) \\
&= \frac{j}{\eta_h} + \sum_{i=0}^{j-2} \binom{j}{i} \frac{\eta_h^i}{\eta_h^j} && (6.48) \\
&= \frac{j}{\eta_h} + \sum_{i=0}^{j-2} \left(\frac{\prod_{t=1}^{j-i}(j-t+1)}{\prod_{t=1}^{j-i} t} \frac{\eta_h^i}{\eta_h^j} \right) \\
&\leq \frac{j}{\eta_h} + \frac{1}{2} \sum_{i=0}^{j-2} j^{j-i} \frac{\eta_h^i}{\eta_h^j} && (6.49) \\
&= \frac{j}{\eta_h} + \frac{1}{2} \sum_{i=0}^{j-2} \left(\frac{j}{\eta_h} \right)^{j-i} \\
&= \frac{j}{\eta_h} + \frac{1}{2} \left(-1 - \frac{j}{\eta_h} + \sum_{i=0}^{j} \left(\frac{j}{\eta_h} \right)^{j-i} \right),
\end{aligned}$$

where (6.47) is a standard summation identity, and (6.49) follows from $\prod_{t=1}^{j-i}(j-t+1) \leq j^{j-i}$ and $\prod_{t=1}^{j-i} t \geq 2$ for $j - i \geq 2$. Next, we explicitly evaluate the summation of the last equality:

$$\begin{aligned}
\beta^{-2}\alpha_j &\leq \frac{j}{\eta_h} + \frac{1}{2} \left(-1 - \frac{j}{\eta_h} + \frac{1 - \left(\frac{j}{\eta_h}\right)^{j+1}}{1 - \frac{j}{\eta_h}} \right) \\
&\leq \frac{j}{\eta_h} + \frac{1}{2} \left(-1 - \frac{j}{\eta_h} + \frac{1}{1 - \frac{j}{\eta_h}} \right) \\
&= \frac{j}{\eta_h} + \frac{1}{2} \left(\frac{\left(\frac{j}{\eta_h}\right)^2}{1 - \frac{j}{\eta_h}} \right) && (6.50) \\
&= \frac{j}{\eta_h} + \frac{j}{2\eta_h} \left(\frac{\frac{j}{\eta_h}}{1 - \frac{j}{\eta_h}} \right), && (6.51)
\end{aligned}$$

where (6.50) follows from $(-a-1)(-a+1) = a^2 - 1$ with $a = j/\eta_h$.

Next, observe that if $j/\eta_h \leq 1/2$, or equivalently

$$2j \leq \eta_h, \tag{6.52}$$

then we can weaken (6.51) to

$$\beta^{-2}\alpha_j \leq \frac{j}{\eta_h} + \frac{j}{2\eta_h} = \frac{3}{2}\frac{j}{\eta_h} = 3\frac{j}{\eta}, \tag{6.53}$$

which yields (6.46). □

Completing the Proof of Theorem 6.2.1

We now prove Theorem 6.2.1 in several steps. We define μ to be a constant such that $f(E_1 \mid (S \setminus E)) = \mu f(S_1)$ holds, and we write $E_0 := E_S^* \cap S_0$, $E_1 := E_S^* \cap S_1$, and $E_{B_i} := E_S^* \cap B_i$, where E_S^* is defined in (6.6). We also make use of the following lemma characterizing the optimal adversary. The proof is straightforward, and can be found in Lemma 2 of [OSU16].

Lemma 6.A.8. [OSU16] *Under the setup of Theorem 6.2.1, for all $X \subset V$ with $|X| \leq \tau$ holds*

$$f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}) \leq f(\mathrm{OPT}(k - \tau, V \setminus X)).$$

Initial lower bounds: We start by providing three lower bounds on $f(S \setminus E_S^*)$. First, we observe that $f(S \setminus E_S^*) \geq f(S_0 \setminus E_0)$ and $f(S \setminus E_S^*) \geq f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right)$. We also have

$$\begin{aligned}
f(S \setminus E) &= f(S) - f(S) + f(S \setminus E) \\
&= f(S_0 \cup S_1) + f(S \setminus E_0) - f(S \setminus E_0) - f(S) + f(S \setminus E) && (6.54)\\
&= f(S_1) + f(S_0 \mid S_1) + f(S \setminus E_0) - f(S) - f(S \setminus E_0) + f(S \setminus E) \\
&= f(S_1) + f(S_0 \mid (S \setminus S_0)) + f(S \setminus E_0) - f(E_0 \cup (S \setminus E_0)) \\
&\quad - f(S \setminus E_0) + f(S \setminus E) && (6.55)\\
&= f(S_1) + f(S_0 \mid (S \setminus S_0)) - f(E_0 \mid (S \setminus E_0)) - f(S \setminus E_0) + f(S \setminus E) \\
&= f(S_1) + f(S_0 \mid (S \setminus S_0)) - f(E_0 \mid (S \setminus E_0)) - f(E_1 \cup (S \setminus E)) + f(S \setminus E) \\
& && (6.56)\\
&= f(S_1) + f(S_0 \mid (S \setminus S_0)) - f(E_0 \mid (S \setminus E_0)) - f(E_1 \mid S \setminus E) \\
&= f(S_1) - f(E_1 \mid S \setminus E) + f(S_0 \mid (S \setminus S_0)) - f(E_0 \mid (S \setminus E_0)) \\
&\geq (1 - \mu) f(S_1), && (6.57)
\end{aligned}$$

where (6.54) and (6.55) follow from $S = S_0 \cup S_1$, (6.56) follows from $E_S^* = E_0 \cup E_1$, and (6.57) follows from $f(S_0 \mid (S \setminus S_0)) - f(E_0 \mid (S \setminus E_0)) \geq 0$ (due to $E_0 \subseteq S_0$ and $S \setminus S_0 \subseteq S \setminus E_0$), along with the definition of μ.

By combining the above three bounds on $f(S \setminus E_S^*)$, we obtain

$$f(S \setminus E_S^*) \geq \max \left\{ f(S_0 \setminus E_0), (1-\mu) f(S_1), f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \right) \right\}. \tag{6.58}$$

We proceed by further bounding these terms.

Bounding the first term in (6.58)**:** Defining $S_0' := \mathrm{OPT}(k-\tau, V \setminus E_0) \cap (S_0 \setminus E_0)$ and $X := \mathrm{OPT}(k-\tau, V \setminus E_0) \setminus S_0'$, we have

$$\begin{aligned} f(S_0 \setminus E_0) + f(\mathrm{OPT}(k-\tau, V \setminus S_0)) &\geq f(S_0') + f(X) && (6.59)\\ &\geq f(\mathrm{OPT}(k-\tau, V \setminus E_0)) && (6.60)\\ &\geq f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}), && (6.61) \end{aligned}$$

where (6.59) follows from monotonicity, i.e. $(S_0 \setminus E_0) \subseteq S_0'$ and $(V \setminus S_0) \subseteq (V \setminus E_0)$, (6.60) follows from the fact that $\mathrm{OPT}(k-\tau, V \setminus E_0) = S_0' \cup X$ and submodularity,[3] and (6.61) follows from Lemma 6.A.8 and $|E_0| \leq \tau$. We rewrite (6.61) as

$$f(S_0 \setminus E_0) \geq f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)). \tag{6.62}$$

Bounding the second term in (6.58)**:** Note that S_1 is obtained by using $\mathcal{A}$ that satisfies the β-iterative property on the set $V \setminus S_0$, and its size is $|S_1| = k - |S_0|$. Hence, from Lemma 6.2.1 with $k-\tau$ in place of k, we have

$$f(S_1) \geq \left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0)). \tag{6.63}$$

Bounding the third term in (6.58)**:** We can view S_1 as a large bucket created by our algorithm after creating the buckets in S_0. Therefore, we can apply Lemma 6.A.4 with $X = \bigcup_{i=0}^{\lceil \log \tau \rceil - 1} (B_i \setminus E_{B_i})$, $Y = B_{\lceil \log \tau \rceil}$, $Z = S_1$, $E_Y = E_S^* \cap Y$, and $E_Z = E_1$. Conditions (6.24) and (6.25) needed to apply Lemma 6.A.4 are provided by Lemma 6.A.6. From Lemma 6.A.4, we obtain the following with α as in (6.40):

$$\begin{aligned} f\left(E_1 \;\middle|\; \left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \right) \cup (S_1 \setminus E_1) \right) \leq \left(\beta \frac{|E_1|}{|B_{\lceil \log \tau \rceil}|} (1+\alpha) + \alpha \right) \\ \times f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i}) \right). \end{aligned} \tag{6.64}$$

[3]The submodularity property can equivalently be written as $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.

Furthermore, noting that the assumption $\eta \geq 4(\log k+1)$ implies $2\lceil \log \tau \rceil \leq \eta_h$, we can upper-bound α as in Lemma 6.A.7 by (6.46) for $j = \lceil \log \tau \rceil$. Also, we have $\beta \frac{|E_1|}{|B_{\lceil \log \tau \rceil}|} \leq \beta \frac{\tau}{2^{\lceil \log \tau \rceil}\eta} \leq \frac{\beta}{\eta}$. Putting these together, we upper bound (6.64) as follows:

$$f\left(E_1 \;\middle|\; \left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right) \cup (S_1 \setminus E_1)\right) \leq \left(\frac{\beta}{\eta}\left(1 + \frac{3\beta^2\lceil \log \tau \rceil}{\eta}\right) + \frac{3\beta^2\lceil \log \tau \rceil}{\eta}\right) \times f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right) \leq \frac{5\beta^3\lceil \log \tau \rceil}{\eta} f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right),$$

where we have used $\eta \geq 1$ and $\lceil \log \tau \rceil \geq 1$ (since $\tau \geq 2$ by assumption). We rewrite this as

$$\begin{aligned} f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right) &\geq \frac{\eta}{5\beta^3\lceil \log \tau \rceil} f\left(E_1 \;\middle|\; \left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right) \cup (S_1 \setminus E_1)\right) \\ &\geq \frac{\eta}{5\beta^3\lceil \log \tau \rceil} f(E_1 \mid (S \setminus E)) \qquad (6.65) \\ &= \frac{\eta}{5\beta^3\lceil \log \tau \rceil} \mu f(S_1), \qquad (6.66) \end{aligned}$$

where (6.65) follows from submodularity, and (6.66) follows from the definition of μ.

Combining the bounds: Returning to (6.58), we have

$$\begin{aligned} f(S \setminus E_S^*) &\geq \max\left\{f(S_0 \setminus E_0), (1-\mu)f(S_1), f\left(\bigcup_{i=0}^{\lceil \log \tau \rceil} (B_i \setminus E_{B_i})\right)\right\} \\ &\geq \max\left\{f(S_0 \setminus E_0), (1-\mu)f(S_1), \frac{\eta}{5\beta^3\lceil \log \tau \rceil}\mu f(S_1)\right\} \qquad (6.67) \\ &\geq \max\{f(\mathrm{OPT}(k,V,\tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \\ &\quad (1-\mu)\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \\ &\quad \frac{\eta}{5\beta^3\lceil \log \tau \rceil}\mu\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0))\} \qquad (6.68) \\ &\geq \max\{f(\mathrm{OPT}(k,V,\tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \\ &\quad \frac{\frac{\eta}{5\beta^3\lceil \log \tau \rceil}}{1 + \frac{\eta}{5\beta^3\lceil \log \tau \rceil}}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0))\} \qquad (6.69) \\ &= \max\{f(\mathrm{OPT}(k,V,\tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \\ &\quad \frac{\eta}{5\beta^3\lceil \log \tau \rceil + \eta}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0))\} \\ &\qquad (6.70) \end{aligned}$$

Finally, we obtain:

$$f(S \setminus E_S^*) \geq \frac{\frac{\eta}{5\beta^3\lceil \log \tau \rceil + \eta}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right)}{1 + \frac{\eta}{5\beta^3\lceil \log \tau \rceil + \eta}\left(1 - e^{-\frac{k-|S_0|}{\beta(k-\tau)}}\right)} f(\mathrm{OPT}(k, V, \tau) \setminus E^*_{\mathrm{OPT}(k,V,\tau)}), \tag{6.71}$$

where (6.67) follows from (6.66), (6.68) follows from (6.62) and (6.63), (6.69) follows since $\max\{1-\mu, c\mu\} \geq \frac{c}{1+c}$ analogously to (6.18), and (6.71) follows from (6.19). Hence, we have established (6.71).

Turning to the permitted values of τ, we have from Proposition 6.2.1 that

$$|S_0| \;\leq\; 3\eta\tau(\log k + 2).$$

For the choice of τ to yield valid set sizes, we only require $|S_0| \leq k$; hence, it suffices that

$$\tau \leq \frac{k}{3\eta(\log k + 2)}. \tag{6.72}$$

Finally, we consider the second claim of the lemma. For $\tau \in o\big(\frac{k}{\eta(\log k)}\big)$ we have $|S_0| \in o(k)$. Furthermore, by setting $\eta \geq \log^2 k$ (which satisfies the assumption $\eta \geq 4(\log k + 1)$ for large k), we get $\frac{k-|S_0|}{\beta(k-\tau)} \to \beta^{-1}$ and $\frac{\eta}{5\beta^3\lceil \log \tau \rceil + \eta} \to 1$ as $k \to \infty$. Hence, the constant factor converges to $\frac{1-e^{-1/\beta}}{2-e^{-1/\beta}}$, yielding (6.8). In the case that GREEDY is used as the subroutine, we have $\beta = 1$, and hence the constant factor converges t $\frac{1-e^{-1}}{2-e^{-1}} \geq 0.387$. If THRESHOLDING-GREEDY is used, we have $\beta = \frac{1}{1-\epsilon}$, and hence the constant factor converges to $\frac{1-e^{\epsilon-1}}{2-e^{\epsilon-1}} \geq (1-\epsilon)\frac{1-e^{-1}}{2-e^{-1}} \geq (1-\epsilon)0.387$.

7 Adversarially Robust Maximization of Non-Submodular Objectives

In this chapter, we consider the same adversarially robust optimization formulation from Chapter 6 in the case when the objective function is *non-submodular*. This setting is motivated by the various machine learning problems in which the objective function does not satisfy the natural notion of diminishing returns, i.e., submodularity. While there exist constant factor approximation guarantees when the objective is submodular (see Chapter 6 and Section 2.3), it is not known whether similar guarantees hold in the non-submodular case. We propose a simple and practical algorithm OBLIVIOUS-GREEDY and prove the first constant-factor approximation guarantees for a wider class of non-submodular objectives. We consider two applications in which the objective is non-submodular, namely, *sparse feature selection* and *batch Bayesian optimization*, and obtain the first robust theoretical guarantees for these problems.

This chapter is based on the joint work with Junyao Zhao* and Volkan Cevher [BZC18].

7.1 Introduction

An important problem in machine learning is the one of feature selection, where the goal is to extract a subset of features that are informative with respect to a given task (e.g., classification). For some tasks, it is of great importance to select features that exhibit robustness against deletions. This is particularly relevant in domains with non-stationary feature distributions or with input sensor failures [GR06]. Another important example is the optimization of an unknown function from point evaluations that require performing costly experiments (this problem is extensively studied in Chapters 3–5). When the expensive experiments can fail it is essential to protect against worst-case failures. In these and many other applications, the objective function is not submodular and hence the guarantees that are obtained in Chapter 6 are not applicable.

In this chapter, we consider the robust maximization of non-submodular functions in the case of adversarial removals, and we investigate whether the constant factor approximation guarantees can be obtained in this setting.

7.1.1 Problem Statement

Let V be a ground set with cardinality $|V| = n$, and let $f : 2^V \to \mathbb{R}_+$ be a normalized monotone set function defined on V. We consider the same robust problem formulation as in the previous chapter, that is

$$\max_{S \subseteq V, |S| \leq k} \min_{E \subseteq S, |E| \leq \tau} f(S \setminus E), \tag{7.1}$$

with the important difference that the objective function f no longer satisfies submodularity. As before, τ is the size of subset E that is removed from the solution set S. The solution set is of size k, and the budget of the adversary τ is at most k. When the objective function exhibits submodularity, a constant factor approximation guarantee can be obtained for this problem (see Chapter 6). However, in many important applications such as the mentioned feature selection problem, the objective function $f(\cdot)$ is not submodular.

7.1.2 Related Work

It is well-known that when the objective is submodular, the simple GREEDY algorithm [NWF78] achieves a $(1 - 1/e)$-multiplicative approximation guarantee for the problem of monotone submodular maximization subject to cardinality constraint (see Section 2.3). The constant factor can be further improved by exploiting the properties of the objective function, such as the *closeness* to being modular captured by the notion of *curvature* [CC84, Von10, IJB13]. In many cases, the GREEDY algorithm performs well empirically even when the objective function deviates from being submodular. An important class of such objectives is γ-*weakly* submodular functions. Simply put, *submodularity ratio* γ is a quantity that characterizes how *close* the function is to being submodular. It was first introduced in [DK11], where it was shown that for such functions the approximation ratio of GREEDY degrades slowly as the submodularity ratio decreases, i.e., as $(1 - e^{-\gamma})$. In [BBKT17], the authors obtain the approximation guarantee of the form $\alpha^{-1}(1 - e^{-\gamma\alpha})$, that further depends on the curvature α.

In the submodular setting, the first efficient algorithm and constant factor guarantees for Problem (7.1) were obtained in [OSU16] for $\tau = o(\sqrt{k})$. Our algorithm PRO-GREEDY (presented in the previous chapter) attains the same 0.387-guarantee but it allows for greater robustness, i.e. the allowed number of removed elements is $\tau = o(k)$. It is not clear how the obtained guarantees generalize for non-submodular functions. The curvature-dependent constant factor approximation guarantees that hold for any number of removals have been recently obtained in the submodular setting [TGJP17].

One important class of non-submodular functions that we consider in this chapter are those used for support/feature selection:

$$f(S) := \max_{\boldsymbol{x} \in \mathcal{X}, \text{supp}(\boldsymbol{x}) \subseteq S} l(\boldsymbol{x}), \tag{7.2}$$

where l is a continuous function, $\mathcal{X}$ is a convex set and $\text{supp}(\boldsymbol{x}) = \{i : x_i \neq 0\}$ is the support set.

A popular way to solve the problem of finding a k-sparse vector that maximizes $l(\cdot)$, i.e.,

$$\boldsymbol{x} \in \underset{\boldsymbol{x}\in\mathcal{X}, \|\boldsymbol{x}\|_0 \leq k}{\arg\max} \; l(\boldsymbol{x})$$

is to maximize the auxiliary set function in (7.2) subject to the cardinality constraint k. This setting and its variants have been used in various applications, e.g., sparse approximation [DK11, CK11], feature selection [KED+17], sparse recovery [CRT06], sparse M-estimation [JTK14] and column subset selection problems [ABF+16]. An important result from [EKDN16] states that if $l(\cdot)$ is (m, L)-(strongly concave, smooth) then $f(S)$ is weakly submodular with submodularity ratio $\gamma \geq \frac{m}{L}$. Consequently, this result enlarges the number of problems where GREEDY comes with guarantees. In Section 7.4.1, we consider the robust version of this problem with the goal of protecting against the worst-case deletions of features.

7.1.3 Contributions

The main contributions of this chapter are:

- We initiate the study of the robust optimization Problem (7.1) for a wider class of monotone non-submodular functions. In Section 7.3, we present a simple and practical algorithm OBLIVIOUS-GREEDY and prove the first constant factor approximation guarantees for Problem (7.1). When the function is submodular and *under specific conditions*, we recover the approximation guarantees obtained in the previous chapter.

- In the non-submodular setting, we obtain the *first* constant factor approximation guarantees for the *linear regime*, that is, when the number of removals $\tau = \lceil ck \rceil$ for some $c \in (0, 1)$.

- Our theoretical bounds are expressed in terms of parameters that further characterize a set function. We prove some interesting relations between these parameters in Section 7.2 and obtain theoretical bounds for them in two important applications: (i) sparse feature selection and (ii) variance reduction objective used in batch Bayesian optimization. This allows us to obtain the *first robust* guarantees for these two important objectives.

- In Section 7.4.1, we show the new connection between *strong convexity* and *weak supermodularity*. Our result (in Proposition 7.4.1) complements the one in [EKDN16] where the same connection between strong convexity and weak submodularity is established. This result can be of interest in other combinatorial optimization works, where it can potentially enlarge the number of applications where (weak) supermodularity can be exploited.

- In Section 7.5, we experimentally validate the robustness of OBLIVIOUS-GREEDY in several scenarios and show that it outperforms other robust and non-robust algorithms. In the feature selection task, we empirically demonstrate that the set of features selected by OBLIVIOUS-GREEDY achieves better robust generalization performance in comparison to other algorithms when it comes to the previously unseen test data.

7.2 Set Function Ratios

In this section, we consider a normalized monotone set function $f : 2^V \to \mathbb{R}_+$; we proceed by defining several quantities that characterize such a set function and show some useful *new relations* among them. Our main theoretical results from subsequent sections will depend on these quantities. Some of these parameters were introduced and used in different previous works, while a few of them are novel.

Definition 7.2.1 (Submodularity [DK11] and Supermodularity ratio)**.** *The submodularity ratio of $f(\cdot)$ is the largest scalar $\gamma \in [0, 1]$ s.t.*

$$\frac{\sum_{i\in\Omega} f(\{i\}|S)}{f(\Omega|S)} \geq \gamma, \quad \forall \textit{ disjoint } S, \Omega \subseteq V. \tag{7.3}$$

while the supermodularity ratio is the largest scalar $\check{\gamma} \in [0, 1]$ s.t.

$$\frac{f(\Omega|S)}{\sum_{i\in\Omega} f(\{i\}|S)} \geq \check{\gamma}, \quad \forall \textit{ disjoint } S, \Omega \subseteq V. \tag{7.4}$$

The function $f(\cdot)$ is submodular (supermodular) *iff* $\gamma = 1$ ($\check{\gamma} = 1$). Hence, the submodularity/supermodularity ratio measures to what extent the function has submodular/supermodular properties. While $f(\cdot)$ is modular *iff* $\gamma = \check{\gamma} = 1$, in general, γ can be different from $\check{\gamma}$. Hence, $f(\cdot)$ can be both γ-weakly submodular and $\check{\gamma}$-weakly supermodular.

Definition 7.2.2 (Generalized curvature [Von10, BBKT17] and inverse generalized curvature)**.** *The generalized curvature of $f(\cdot)$ is the smallest scalar $\alpha \in [0, 1]$ s.t.*

$$\frac{f(\{i\}|S \setminus \{i\} \cup \Omega)}{f(\{i\}|S \setminus \{i\})} \geq 1 - \alpha, \quad \forall S, \Omega \subseteq V, i \in S \setminus \Omega, \tag{7.5}$$

while the inverse generalized curvature is the smallest scalar $\check{\alpha} \in [0, 1]$ s.t.

$$\frac{f(\{i\}|S \setminus \{i\})}{f(\{i\}|S \setminus \{i\} \cup \Omega)} \geq 1 - \check{\alpha}, \quad \forall S, \Omega \subseteq V, i \in S \setminus \Omega. \tag{7.6}$$

The function $f(\cdot)$ is submodular (supermodular) *iff* $\check{\alpha} = 0$ ($\alpha = 0$). The function is modular *iff* $\alpha = \check{\alpha} = 0$. In general, α can be different from $\check{\alpha}$.

Definition 7.2.3 (Sub/Superadditivity ratio)**.** *The subadditivity ratio of $f(\cdot)$ is the largest scalar $\nu \in [0, 1]$ such that*

$$\frac{\sum_{i\in S} f(\{i\})}{f(S)} \geq \nu, \quad \forall S \subseteq V. \tag{7.7}$$

The superadditivity ratio is the largest scalar $\check{\nu} \in [0, 1]$ such that

$$\frac{f(S)}{\sum_{i\in S} f(\{i\})} \geq \check{\nu}, \quad \forall S \subseteq V. \tag{7.8}$$

If the function is submodular (supermodular) then $\nu = 1$ ($\check{\nu} = 1$).

The following proposition captures the relation between the above quantities.

Proposition 7.2.1. *For any $f(\cdot)$, the following relations hold:*

$$\nu \geq \gamma \geq 1 - \check{\alpha} \quad \textit{and} \quad \check{\nu} \geq \check{\gamma} \geq 1 - \alpha.$$

We also provide a more general definition of the bipartite subadditivity ratio used in [KED+17].

Definition 7.2.4 (Bipartite subadditivity ratio)**.** *The bipartite subadditivity ratio of $f(\cdot)$ is the largest scalar $\theta \in [0, 1]$ s.t.*

$$\frac{f(A) + f(B)}{f(S)} \geq \theta, \quad \forall S \subseteq V, A \cup B = S, A \cap B = \emptyset. \tag{7.9}$$

Remark 1. *For any $f(\cdot)$, it holds that $\theta \geq \check{\nu}\nu$.*

Greedy guarantee. Different works [DK11, BBKT17] have studied the performance of the GREEDY algorithm [NWF78] for the problem of monotone set function maximization subject to caridinality constraint (Section 2.3) when the objective is γ-weakly submodular. In our analysis, we are going to make use of the following important result from [DK11].

Lemma 7.2.1. *For a monotone normalized set function $f : 2^V \to \mathbb{R}_+$, with submodularity ratio $\gamma \in [0, 1]$ the* GREEDY *algorithm when run for $l \in \mathbb{N}_+$ steps, returns a set S_l of size l such that*

$$f(S_l) \geq \left(1 - e^{-\gamma \frac{l}{k}}\right) f(\text{OPT}(k, V)),$$

where $\text{OPT}(k, V)$ *is used to denote the optimal set of size k, i.e.,*

$$\text{OPT}(k, V) \in \underset{S \subseteq V, |S| \leq k}{\arg\max}\ f(S). \tag{7.10}$$

7.3 Oblivious Greedy Algorithm and its Guarantees

We present our OBLIVIOUS-GREEDY algorithm in Algorithm 11. The algorithm requires a non-negative monotone set function $f : 2^V \to \mathbb{R}_+$, and the ground set of items V. It constructs two sets S_0 and S_1. The first set S_0 is constructed via oblivious selection, i.e. $\lceil \beta\tau \rceil$ items with the individually highest objective values are selected. Here, $\beta \in \mathbb{R}_+$ is an input parameter, that together with τ, determines the size of S_0 ($|S_0| = \lceil \beta\tau \rceil \leq k$). We provide more information on this parameter in the next section. The second set S_1, of size $k - |S_0|$, is obtained by running the GREEDY algorithm on the remaining items $V \setminus S_0$. Finally, the algorithm outputs the set $S = S_0 \cup S_1$ of size k that is robust against the worst-case removal of τ elements.

Algorithm 11 OBLIVIOUS-GREEDY [BZC18]

Input: Set V, k, τ, $\beta \in \mathbb{R}_+$ and $\lceil \beta\tau \rceil \leq k$
Output: Set $S \subseteq V$ such that $|S| \leq k$

$S_0, S_1 \leftarrow \emptyset$
for $i \leftarrow 0$ **to** $\lceil \beta\tau \rceil$ **do**
    $v \leftarrow \arg\max_{v \in V \setminus S_0} f(\{v\})$
    $S_0 \leftarrow S_0 \cup \{v\}$
$S_1 \leftarrow \text{GREEDY}(k - |S_0|,\ (V \setminus S_0))$
$S \leftarrow S_0 \cup S_1$
return S

Intuitively, the role of S_0 is to ensure robustness, as its elements are selected independently of each other and have high marginal values. On the other hand, the set S_1 is obtained greedily and it is near-optimal on the set $V \setminus S_0$.

OBLIVIOUS-GREEDY is simpler than the submodular algorithms PRO-GREEDY (Sec. 6.2.1) and OSU (Sec. 2.3). Both of these algorithms construct multiple sets (buckets) whose number and size depend on the input parameters k and τ. In contrast, OBLIVIOUS-GREEDY always constructs two sets, where the first set is obtained by the fast OBLIVIOUS selection.

For problem $\max_{|S| \leq k} f(S)$ and the weakly submodular objective $f(\cdot)$, the GREEDY algorithm achieves a constant factor approximation (Lemma 7.2.1), while OBLIVIOUS selection achieves (γ/k)-approximation [KED+17]. For the harder Problem (7.1), GREEDY can fail arbitrarily badly (see Section 6.1.1). Interestingly enough, the combination of these two algorithms reflected in OBLIVIOUS-GREEDY leads to a constant factor approximation for Problem (7.1).

7.3.1 Approximation guarantee

The quantity of interest in this section is the remaining utility after the adversarial removal of elements $f(S \setminus E_S^*)$, where S is the set of size k returned by OBLIVIOUS-GREEDY, and E_S^* is the set of size τ chosen by the adversary, i.e., $E_S^* \in \arg\min_{E \subset S, |E| \leq \tau} f(S \setminus E)$. Let $\mathrm{OPT}(k - \tau, V \setminus E_S^*)$ denote the optimal solution (Eq. 7.10), of size $k - \tau$, when the ground set is $V \setminus E_S^*$. The goal in this section is to compare $f(S \setminus E_S^*)$ to $f(\mathrm{OPT}(k - \tau, V \setminus E_S^*))$.[1] All the omitted proofs from this section can be found in Appendix 7.A.2.

Intermediate results. Before stating our main result, we provide three lower bounds on $f(S \setminus E_S^*)$. For the returned set $S = S_0 \cup S_1$, we let E_0 denote elements removed from S_0, i.e., $E_0 := E_S^* \cap S_0$ and similarly $E_1 := E_S^* \cap S_1$. The result of the first lemma is borrowed from the previous chapter (the proof can be found in Appendix 6.A.4), and states that $f(S \setminus E_S^*)$ is at least some constant fraction of the utility of the elements obtained greedily in the second stage.

[1] We have $f(\mathrm{OPT}(k - \tau, V \setminus X)) \geq f(\mathrm{OPT} \setminus E_{\mathrm{OPT}}^*)$ (from Lemma 6.A.8), where OPT is the optimal solution to Problem (7.1) and $X \subset V$ is any set of size τ. The main result from the previous chapter also holds if we replace $f(\mathrm{OPT} \setminus E_{\mathrm{OPT}}^*)$ with its upper bound $f(\mathrm{OPT}(k - \tau, V \setminus E_S^*))$.

Lemma 7.3.1. *For any $f(\cdot)$ (not necessarily submodular), let $\mu \in [0,1]$ be a constant such that $f(E_1 \mid (S \setminus E_S^*)) = \mu f(S_1)$ holds. Then, $f(S \setminus E_S^*) \geq (1-\mu) f(S_1)$.*

The next lemma applies to any monotone set function with bipartite subadditivity ratio θ.

Lemma 7.3.2. *Let $\theta \in [0,1]$ be a bipartite subadditivity ratio defined in Eq.* (7.9)*. Then*

$$f(S \setminus E_S^*) \geq \theta f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) - (1 - e^{-\frac{k-|S_0|}{k-\tau}})^{-1} f(S_1).$$

In other words, if $f(S_1)$ is *small* compared to the utility of the optimal solution, then $f(S \setminus E_S^*)$ is at least a constant factor away from the optimal solution. Next, we present our key lemma that further relates $f(S \setminus E_S^*)$ to the utility of the set S_1 with no deletions.

Lemma 7.3.3. *Let β be a constant such that $|S_0| = \lceil \beta\tau \rceil$, $|S_0| \leq k$, and let $\check{\nu}, \check{\alpha} \in [0,1]$ be a superadditivity ratio and generalized inverse curvature (Eq.* (7.8) *and Eq.* (7.6)*, respectively). Finally, let μ be a constant defined as in Lemma 7.3.1. Then,*

$$f(S \setminus E_S^*) \geq (\beta - 1)\check{\nu}(1-\check{\alpha})\mu f(S_1).$$

Proof. We have:

$$\begin{aligned}
f(S \setminus E_S^*) &\geq f(S_0 \setminus E_0) \\
&\geq \check{\nu} \sum_{e_i \in S_0 \setminus E_0} f(\{e_i\}) && (7.11)\\
&\geq \frac{|S_0 \setminus E_0|}{|E_1|} \check{\nu} \sum_{e_i \in E_1} f(\{e_i\}) && (7.12)\\
&\geq \frac{(\beta-1)\tau}{\tau} \check{\nu} \sum_{e_i \in E_1} f(\{e_i\}) && (7.13)\\
&\geq (\beta-1)\check{\nu}(1-\check{\alpha}) \\
&\quad \times \sum_{i=1}^{|E_1|} f\left(\{e_i\} | (S \setminus E_S^*) \cup E_1^{(i-1)}\right) && (7.14)\\
&= (\beta-1)\check{\nu}(1-\check{\alpha}) f\left(E_1 | (S \setminus E_S^*)\right) && (7.15)\\
&= (\beta-1)\check{\nu}(1-\check{\alpha})\mu f(S_1). && (7.16)
\end{aligned}$$

Eq. (7.11) follows by the superadditivity. Eq. (7.12) follows from the way S_0 is constructed, i.e. via OBLIVIOUS selection that ensures $f(\{i\}) \geq f(\{j\})$ for every $i \in S_0 \setminus E_0$ and $j \in E_1$. Eq. (7.13) follows from $|S_0 \setminus E_0| = \lceil \beta\tau \rceil - |E_0| \geq \beta\tau - \tau = (\beta-1)\tau$, and $|E_1| \leq \tau$.

To prove Eq. (7.14), let $E_1 = \{e_1, \cdots e_{|E_1|}\}$, and let $E_1^{(i-1)} \subseteq E_1$ denote the set $\{e_1, \cdots, e_{i-1}\}$.

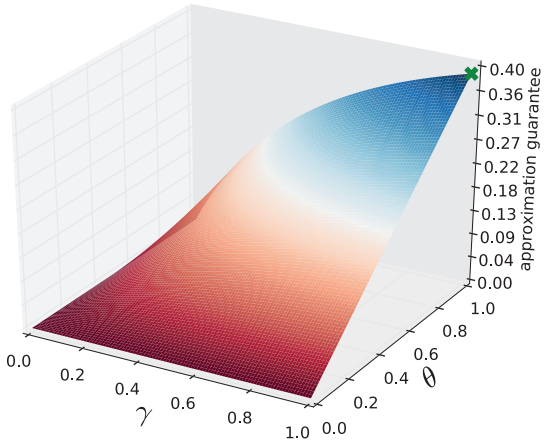


Figure 7.1: Approximation guarantee obtained in Remark 2. The green cross represents the approximation guarantee when f is submodular ($\gamma = \theta = 1$).

Also, let $E_1^{(0)} = \emptyset$. Eq. (7.14) then follows from

$$f(\{e_i\}) \geq (1 - \check{\alpha}) f\left(\{e_i\} | (S \setminus E_S^*) \cup E_1^{(i-1)}\right),$$

which in turns follows from (7.6) by setting $S = \{e_i\}$ and $\Omega = (S \setminus E_S^*) \cup E_1^{(i-1)}$.

Finally, Eq. (7.15) follows from $f\left(E_1 | (S \setminus E_S^*)\right) = \sum_{e_i \in E_1} f\left(\{e_i\} | (S \setminus E_S^*) \cup E_1^{(i-1)}\right)$ (telescoping sum) and Eq. (7.16) follows from the definition of μ. □

Main result. We obtain the main result by examining the maximum of the obtained lower bounds in Lemma 7.3.1, 7.3.2 and 7.3.3. Note, that all three obtained lower bounds depend on $f(S_1)$. In Lemma 7.3.2, we benefit from $f(S_1)$ being small while the opposite is true for Lemma 7.3.1 and 7.3.3 (both bounds are increasing in $f(S_1)$). By examining the latter two, we observe that in Lemma 7.3.1 we benefit from μ being small (i.e. the utility that we lose due to E_1 is small compared to the utility of the whole set S_1) while the opposite is true for Lemma 7.3.3. By carefully balancing between these cases (see Appendix 7.A.2) we arrive at our main result.

Theorem 7.3.1. *Let $f : 2^V \to \mathbb{R}_+$ be a normalized, monotone set function with submodularity ratio γ, bipartite subadditivity ratio θ, inverse curvature $\check{\alpha}$ and superadditivity ratio $\check{\nu}$, every parameter in $[0, 1]$. For a given budget k and $\tau = \lceil ck \rceil$ for some fixed $c \in (0, 1)$, the* OBLIVIOUS-GREEDY *algorithm with $\beta \in \left(1, \frac{1}{c}\right)$ returns a set S such that as $k \to \infty$ we have*

$$f(S \setminus E_S^*) \geq \frac{\theta P \left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)}{1 + P \left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)} f(\mathrm{OPT}(k - \tau, V \setminus E_S^*)),$$

where P is used to denote $\frac{(\beta-1)\check{\nu}(1-\check{\alpha})}{1+(\beta-1)\check{\nu}(1-\check{\alpha})}$.

Remark 2. *Consider $f(\cdot)$ from Theorem 7.3.1 with $\check{\nu} \in (0,1]$ and $\check{\alpha} \in [0,1)$. When $\tau = o\left(\frac{k}{\beta}\right)$ and $\beta \geq \log k$, as $k \to \infty$ we have:*

$$f(S \setminus E_S^*) \geq \left(\theta \frac{1-e^{-\gamma}}{2-e^{-\gamma}} + o(1)\right) f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)).$$

Interpretation. In Theorem 7.3.1 we obtain the first asymptotic constant factor approximation in the *linear regime*, i.e., when the number of removals is $\tau = \lceil ck \rceil$ for some fixed constant $c \in (0,1)$. Our result holds in the non-submodular setting and depends on the few different parameters that further characterize our objective function. In the subsequent sections, we provide bounds for these parameters in the case of specific classes of set functions.

Additionally, when f is submodular, all the parameters in the obtained bound are fixed ($\check{\alpha} = 0$ and $\gamma = \theta = 1$ due to submodularity) except the superadditivity ratio $\check{\nu}$ which can take any value in $[0,1]$. The approximation factor improves for greater $\check{\nu}$, i.e. the closer the function is to being superadditive. On the other hand, if f is supermodular then $\check{\nu} = 1$ while $\check{\alpha}, \theta, \gamma$ are in $[0,1]$, and the approximation factor improves for larger θ and γ, and smaller $\check{\alpha}$.

From Remark 2, when f is submodular (and assuming the conditions of Remark 2 hold), OBLIVIOUS-GREEDY achieves an asymptotic approximation factor of at least 0.387. In such case, the obtained approximation factor matches the one obtained in [BMSC17b, OSU16]. More importantly, the obtained result also holds for a wider range of non-submodular functions. In Figure 7.1 we show how the asymptotic approximation factor changes as a function of γ and θ.

We also obtain an alternative formulation of our main result, which we present here.

Corollary 7.3.1. *Consider the setting from Theorem 7.3.1 and let $P := \frac{(\beta-1)\check{\nu}\nu}{1+(\beta-1)\check{\nu}(1-\nu)}$. We have*

$$f(S \setminus E_S^*) \geq \frac{\theta^2 P\left(1 - e^{-\gamma\frac{1-\beta c}{1-c}}\right)}{1 + \theta P\left(1 - e^{-\gamma\frac{1-\beta c}{1-c}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)).$$

Additionally, consider $f(\cdot)$ with $\check{\nu}, \nu \in (0,1]$. When $\tau = o\left(\frac{k}{\beta}\right)$ and $\beta \geq \log k$, as $k \to \infty$, we have that $f(S \setminus E_S^*)$ is at least

$$\left(\frac{\theta^2(1-e^{-\gamma})}{1+\theta(1-e^{-\gamma})} + o(1)\right) f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)).$$

The key observation is that the approximation factor depends on ν instead of inverse curvature $\check{\alpha}$. The asymptotic approximation ratio is slightly worse here compared to the one obtained in Theorem 7.3.1. However, depending on the considered application, it might be significantly harder to provide bounds for the inverse curvature than bipartite subadditivty ratio, and hence in such cases, this formulation might be more suitable.

7.4 Applications

In this section, we consider two important real-world applications where deletion robust optimization is of interest. We show that the parameters used in the statement of our main theoretical result can be explicitly characterized, which implies that the obtained guarantees are applicable.

7.4.1 Robust Support Selection

We first consider the recent results that connect submodularity with concavity [EKDN16, KED$^+$17]. In order to obtain bounds for robust support selection, we make use of the theoretical bounds obtained for OBLIVIOUS-GREEDY in Corollary 7.3.1.

Given a differentiable concave function $l : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subseteq \mathbb{R}^p$ is a convex set, and $k \leq p$, the support selection problem is: $\max_{\|\boldsymbol{x}\|_0 \leq k} l(\boldsymbol{x})$. We let $\text{supp}(\boldsymbol{x}) = \{i : x_i \neq 0\}$ and consider the normalized monotone set function from [EKDN16]: $f(S) := \max_{\text{supp}(\boldsymbol{x}) \subseteq S, \boldsymbol{x} \in \mathcal{X}} l(\boldsymbol{x}) - l(\mathbf{0})$.

Let $T_l(\boldsymbol{x}, \boldsymbol{y}) := l(\boldsymbol{y}) - l(\boldsymbol{x}) - \langle \nabla l(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \rangle$. An important result obtained in [EKDN16] can be rephrased as follows: if $l(\cdot)$ is L-smooth and m-strongly concave then

$$-\frac{m}{2}\|\boldsymbol{y} - \boldsymbol{x}\|_2^2 \geq T_l(\boldsymbol{x}, \boldsymbol{y}) \geq -\frac{L}{2}\|\boldsymbol{y} - \boldsymbol{x}\|_2^2,$$

holds for all $\boldsymbol{x}, \boldsymbol{y} \in \text{dom}(l)$, and f's submodularity ratio γ is lower bounded[2] by $\frac{m}{L}$. Subsequently, in [KED$^+$17] it is shown that θ can also be lower bounded by the same ratio $\frac{m}{L}$.

In this chapter, we consider the robust support selection problem, that is, finding a set of features $S \subseteq [d]$ of size k that is robust against the deletion of limited number of features. More formally, the goal is to maximize the following objective over all $S \subseteq [d]$:

$$\min_{|E_S| \leq \tau, E_S \subseteq S} \max_{\text{supp}(\boldsymbol{x}) \subseteq S \setminus E_S, \boldsymbol{x} \in \mathcal{X}} l(\boldsymbol{x}) - l(\mathbf{0}).$$

By inspecting the bound obtained in Corollary 7.3.1, it remains to bound the (super/sub)additive ratio ν and $\check{\nu}$. The first bound follows by combining the result $\gamma \geq \frac{m}{L}$ with Proposition 7.2.1: $\nu \geq \gamma \geq \frac{m}{L}$. To prove the second bound, we make use of the following result that complements the one from [EKDN16] and shows the novel connection between the strong concavity and weak supermodularity.

Proposition 7.4.1. *The supermodularity ratio $\check{\gamma}$ of the considered objective $f(\cdot)$ can be lower bounded by $\frac{m}{L}$.*

The second bound follows by the result of this proposition and Proposition 7.2.1: $\check{\nu} \geq \check{\gamma} \geq \frac{m}{L}$.

[2] In fact, as shown in [EKDN16], a similar result holds in the case when $l(\cdot)$ is restricted smooth and restricted strongly concave on the domain of all pairs of k-sparse vectors that differ in at most k entries. All our result can be easily adapted to hold in such a setting as well.

7.4.2 Variance Reduction in Robust Batch Bayesian Optimization

In batch Bayesian optimization, the goal is to optimize an unknown non-convex function from *costly* concurrent function evaluations [DKB14, GDHL16, AJF12]. Most often, the concurrent evaluations correspond to running an expensive batch of experiments. In the case where experiments can fail, it is beneficial to select a set of experiments in a robust way.

Different acquisition (i.e. auxiliary) functions have been proposed to evaluate the utility of candidate points for the next evaluations of the unknown function (see Section 2.2). In Chapter 3, we used the *variance reduction* objective as the part of our acquisition function – the unknown function is evaluated at the points that maximally reduce variance of the posterior distribution over the given set of points that represent *potential maximizers*. We now recall the problem setup.

Setup. Let $f(\boldsymbol{x})$ be an unknown function defined over a finite domain $\mathcal{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$, where $\boldsymbol{x}_i \in \mathbb{R}^p$. Once we evaluate the function at some point $\boldsymbol{x}_i \in \mathcal{X}$, we receive a noisy observation $y_i = f(\boldsymbol{x}_i) + z$, where $z \sim \mathcal{N}(0, \sigma^2)$. In Bayesian optimization, f is modeled as a sample from a Gaussian process. We use a Gaussian process with zero mean and kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$, i.e. $f \sim \mathrm{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$. Let $S = \{e_1, \cdots, e_{|S|}\} \subseteq [n]$ denote the set of points, and $\boldsymbol{X}_S := [\boldsymbol{x}_{e_1}, \cdots, \boldsymbol{x}_{e_{|S|}}] \in \mathbb{R}^{|S| \times d}$ and $\boldsymbol{y}_S := [y_1, \cdots, y_{|S|}]$ denote the corresponding data matrix and observations, respectively. The posterior distribution of f given the points $\boldsymbol{X}_S$ and observations $\boldsymbol{y}_S$ is again a GP, with the posterior variance given by:

$$\sigma^2_{\boldsymbol{x}|S} = k(\boldsymbol{x}, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{X}_S) \left(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I}_{|S|}\right)^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}).$$

For a given set of potential maximizers $M \subseteq [n]$, the variance reduction objective is defined as:

$$F_M(S) := \sum_{\boldsymbol{x} \in X_M} \sigma^2_{\boldsymbol{x}} - \sigma^2_{\boldsymbol{x}|S}, \tag{7.17}$$

where $\sigma^2_{\boldsymbol{x}} = k(\boldsymbol{x}, \boldsymbol{x})$. We prove that this objective is *not* submodular in general (see Appendix 7.A.3).

Finally, our goal is to find a set of points S of size k that maximizes

$$\min_{|E_S| \leq \tau, E_S \subseteq S} \sum_{\boldsymbol{x} \in X_M} \sigma^2_{\boldsymbol{x}} - \sigma^2_{\boldsymbol{x}|S \setminus E_S}.$$

Proposition 7.4.2. *Assume the kernel function is such that $k(\boldsymbol{x}_i, \boldsymbol{x}_i) \leq k_{max}$, for every $i \in [n]$. The objective function in* (7.17) *is normalized and monotone, and both its curvature α and inverse curvature $\check{\alpha}$ can be upper bounded by $\frac{k_{max}}{\sigma^2 + k_{max}}$.*

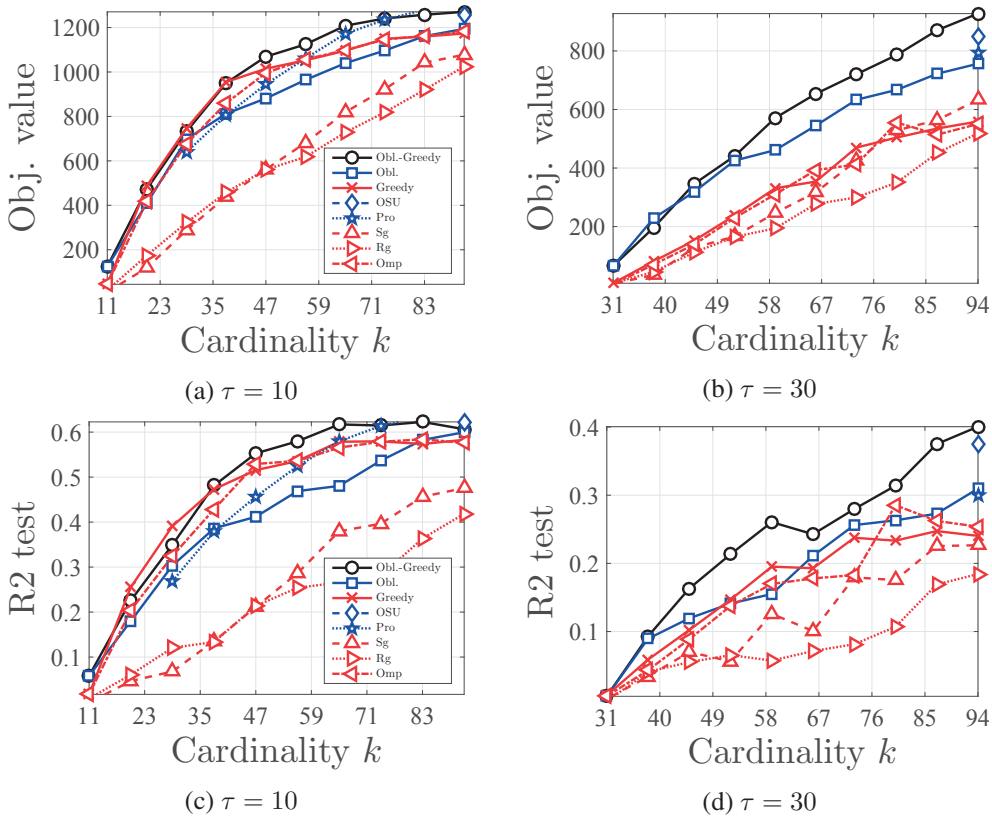
We can combine this result with Proposition 7.2.1, to obtain $\nu \geq \gamma \geq \frac{\sigma^2}{\sigma^2 + k_{\max}}$ and $\check{\nu} \geq \check{\gamma} \geq \frac{\sigma^2}{\sigma^2 + k_{\max}}$. Also, we have $\theta \geq \frac{\sigma^4}{(\sigma^2 + k_{\max})^2}$ (Remark 1). Consequently, all the parameters from Theorem 7.3.1 are explicitly characterized.

7.5 Experimental Evaluation

Optimization performance. For a returned set S, we measure the performance in terms of $\min_{E\subseteq S,|E|\leq\tau} f(S \setminus E)$. Note that $f(S \setminus E)$ is a submodular function in E. Finding the minimizer E s.t. $|E| \leq \tau$ is NP-hard even to approximate [SF11]. We rely on the following methods in order to find E of size τ that degrades the solution as much as possible:

– Greedy adversaries: (i) *Greedy Min* – iteratively removes elements to reduce the objective value $f(S \setminus E)$ as much as possible, and (ii) *Greedy Max* – iteratively adds elements from S to maximize the objective $f(E)$.

– Random Greedy adversaries:[3] In order to introduce randomness in the removal process we consider (iii) *Random Greedy Min* – iteratively selects a random element from the top τ elements whose marginal gains are the highest in terms of reducing the objective value $f(S \setminus E)$ and (iv) *Stochastic Greedy Min* – iteratively selects an element, from a random set $R \subseteq V$, with the highest marginal gain in terms of reducing $f(S \setminus E)$. At every step, R is obtained by subsampling $(|S|/\tau)\log(1/\epsilon)$ elements from S.

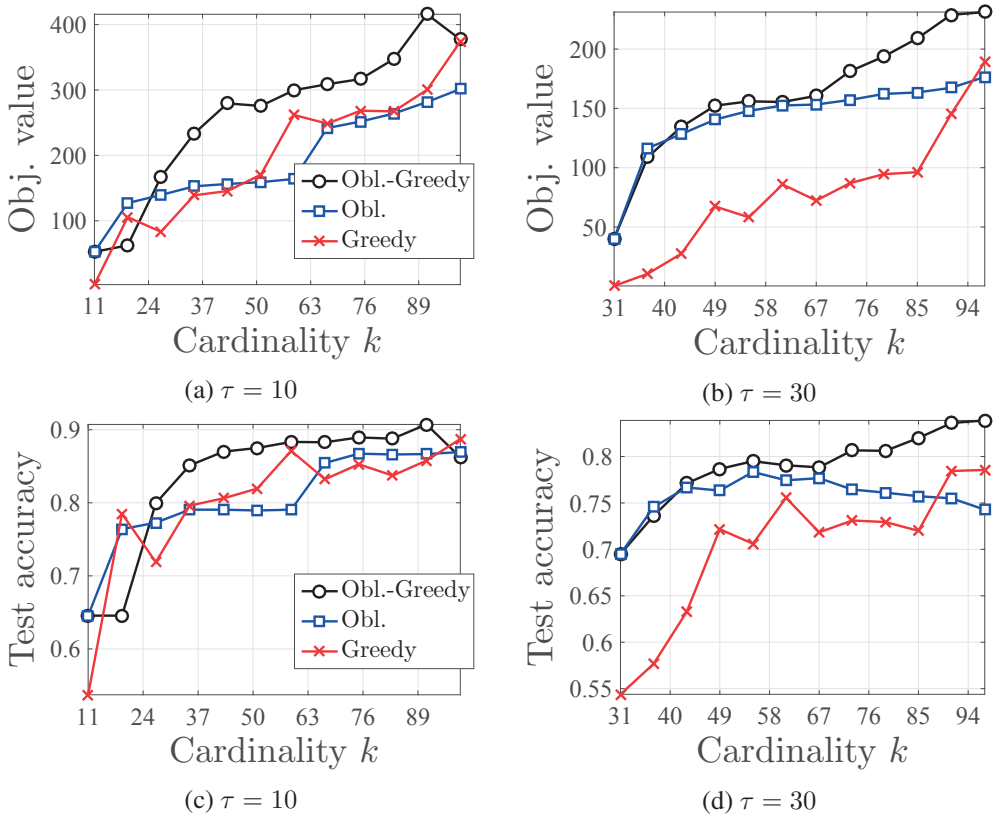


(a) $\tau = 10$ (b) $\tau = 30$

(c) $\tau = 10$ (d) $\tau = 30$

Figure 7.2: Comparison of the algorithms on the linear regression task.

[3]The random adversaries are inspired by the works of [BFNS14] and [MBK+15].



(a) $\tau = 10$ (b) $\tau = 30$

(c) $\tau = 10$ (d) $\tau = 30$

Figure 7.3: Logistic regression task with synthetic dataset.

The minimum objective value $f(S \setminus E)$ among all obtained sets E is reported. We observed that while *Greedy Max* is effective in degrading the solution S obtained by GREEDY, it has little effect on the solutions produced by the robust algorithms.

7.5.1 Robust Support Selection

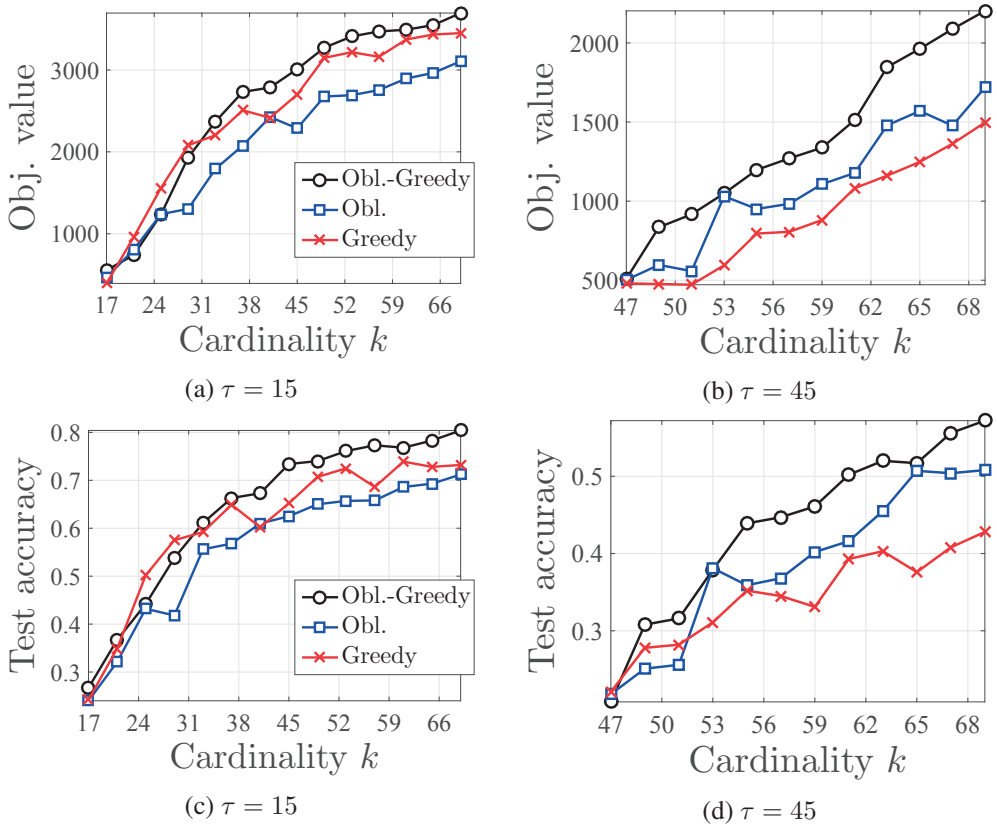
Linear Regression. Our setup is similar to the one in [KED+17]. Each row of the design matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ is generated by an autoregressive process ($t \in [p]$),

$$X_{i,t+1} = \sqrt{1 - \alpha^2} X_{i,t} + \alpha \epsilon_{i,t}, \tag{7.18}$$

where $\epsilon_{i,t}$ is i.i.d. standard Gaussian with variance $\alpha^2 = 0.5$. We use $n = 800$ training data points and $p = 1000$. An additional 2400 points are used for testing. We generate a 100-sparse regression vector by selecting random entries of $\boldsymbol{\omega}$ and set them

$$\boldsymbol{\omega}_s = (-1)^{\text{Bern}(1/2)} \left(5 \sqrt{\frac{\log d}{n}} + \delta_s \right),$$

where $\delta_s \sim \mathcal{N}(0, 1)$. The target is given by $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\omega} + \boldsymbol{z}$, where $\forall i \in [n],\ z_i \sim \mathcal{N}(0, 5)$.



(a) $\tau = 15$ (b) $\tau = 45$

(c) $\tau = 15$ (d) $\tau = 45$

Figure 7.4: Logistic regression with MNIST dataset.

We compare the performance of OBLIVIOUS-GREEDY against:

- robust algorithms (in blue) such as OBLIVIOUS, PRO-GREEDY and OSU [OSU16], and
- greedy-type algorithms (in red) such as GREEDY, STOCHASTIC-GREEDY [MBK$^+$15], RANDOM-GREEDY [BFNS14] and ORTHOGONAL-MATCHING-PURSUIT.

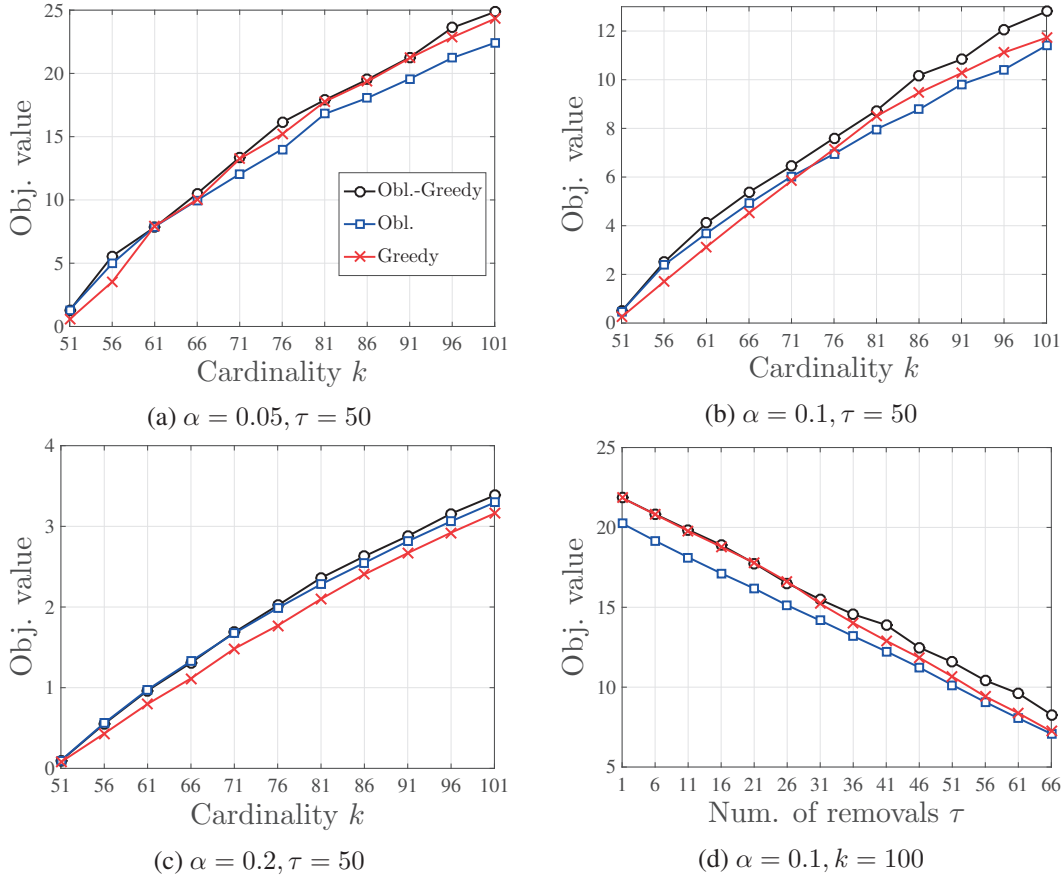
We require $\beta > 1$ for our asymptotic results to hold, but we found out that in practice (small k regime) $\beta \leq 1$ usually gives the best performance. We use OBLIVIOUS-GREEDY with $\beta = 1$ unless stated otherwise.

The results are shown in Fig. 7.2. Since PRO-GREEDY and OSU only make sense in the regime where τ is relatively small, the plots show their performance only for feasible values of k. It can be observed that OBLIVIOUS-GREEDY achieves the best performance among all the methods in terms of both training error and test score. Also, the greedy-type algorithms become less robust for larger values of τ.

Logistic Regression. We compare the performance of OBLIVIOUS-GREEDY against GREEDY and OBLIVIOUS selection on both synthetic and real-world data.

– *Synthetic data:* We generate a 100-sparse $\boldsymbol{\omega}$ by letting $\omega_s = (-1)^{\text{Bern}(1/2)} \times \delta_s$, with $\delta_s \sim \text{Unif}([-1, 1])$. The design matrix $\boldsymbol{X}$ is generated as in (7.18), with $\alpha^2 = 0.09$. We set $p = 200$, and use $n = 600$ points for training and additional 1800 points for testing. The label of the i-th data point $\boldsymbol{X}_{(i,\cdot)}$ is set to 1 if $1/(1 + \exp(\boldsymbol{X}_{(i,\cdot)}\boldsymbol{\beta})) > 0.5$ and 0 otherwise. The results are shown in Fig. 7.3. We can observe that OBLIVIOUS-GREEDY outperforms other methods both in terms of the achieved objective value and generalization error. We also note that the performance of GREEDY decays significantly when τ increases.

– *MNIST:* We consider the 10-class logistic regression task on the MNIST [LBBH98] dataset. In this experiment, we set $\beta = 0.5$ in OBLIVIOUS-GREEDY, and we sample 200 images for each digit for the training phase and 100 images of each for testing. The results are shown in Fig. 7.4. It can be observed that OBLIVIOUS-GREEDY has a distinctive advantage over GREEDY and OBLIVIOUS, while when τ increases the performance of GREEDY decays significantly and more robust OBLIVIOUS starts to outperform it.



(a) $\alpha = 0.05, \tau = 50$

(b) $\alpha = 0.1, \tau = 50$

(c) $\alpha = 0.2, \tau = 50$

(d) $\alpha = 0.1, k = 100$

Figure 7.5: Comparison of the algorithms on the variance reduction task.

7.5.2 Robust Batch Bayesian Optimization via Variance Reduction

Setup. We conducted the following synthetic experiment. A design matrix X of size 600×20 is obtained via the autoregressive process from (7.18). The function values at these points are generated from a GP with $3/2$-Mátern kernel with both lengthscale and output variance set to 1.0. The samples of this function are corrupted by Gaussian noise, $\sigma^2 = 1.0$. Objective function used is the variance reduction (Eq. (7.17)). Finally, half of the points randomly chosen are selected in the set M, while the other half is used in the selection process. We use $\beta = 0.5$ in our algorithm.

Results. In Figure 7.5 (a), (b), (c), the performance of all three algorithms is shown when τ is fixed to 50. Different figures correspond to different α values. We observe that when $\alpha = 0.1$, GREEDY outperforms OBLIVIOUS for most values of k, while OBLIVIOUS clearly outperforms GREEDY when $\alpha = 0.2$. For all presented values of α, OBLIVIOUS-GREEDY outperforms both GREEDY and OBLIVIOUS selection. For larger values of α, the correlation between the points becomes small and consequently so do the objective values. In such cases, all three algorithms perform similarly. In Figure 7.5 (d), we show how the performance of all three algorithms decreases as the number of removals increases. When the number of removals is small both GREEDY and our algorithm perform similarly, while as the number of removals increases the performance of GREEDY drops more rapidly.

7.A Proofs

7.A.1 Proofs from Section 7.2

Proof of Proposition 7.2.1

We prove the following relations:

- $\nu \geq \gamma, \check{\nu} \geq \check{\gamma}$:
  By setting $S = \emptyset$ in both Eq. (7.3) and Eq. (7.4), we obtain $\forall S \subseteq V$:

  $$\sum_{i \in S} f(\{i\}) \geq \gamma f(S), \tag{7.19}$$

  and

  $$f(S) \geq \check{\gamma} \sum_{i \in S} f(\{i\}). \tag{7.20}$$

  The result follows since, by definition of ν and $\check{\nu}$, they are the largest scalars such that Eq. (7.19) and Eq. (7.20) hold, respectively.

- $\gamma \geq 1 - \check{\alpha}, \check{\gamma} \geq 1 - \alpha$:
  Let $S, \Omega \subseteq V$ be two arbitrary disjoint sets. We arbitrarily order elements of $s\Omega = \{e_1, \cdots, e_{|\Omega|}\}$ and we let Ω_{j-1} denote the first $j-1$ elements of Ω. We also let Ω_0 be an empty set.

  By the definition of $\check{\alpha}$ (see Eq. (7.6)) we have:

  $$\begin{aligned}
  \sum_{j=1}^{|\Omega|} f\left(\{e_j\}|S\right) &= \sum_{j=1}^{|\Omega|} f\left(\{e_j\}|S \cup \{e_j\} \setminus \{e_j\}\right) \\
  &\geq \sum_{j=1}^{|\Omega|} (1 - \check{\alpha}) f\left(\{e_j\}|S \cup \{e_j\} \setminus \{e_j\} \cup \Omega_{j-1}\right) \\
  &= (1 - \check{\alpha}) f\left(\Omega|S\right),
  \end{aligned} \tag{7.21}$$

  where the last equality is obtained via telescoping sums.

  Similarly, by the definition of α (see Eq. (7.5)) we have:

  $$\begin{aligned}
  (1 - \alpha) \sum_{j=1}^{|\Omega|} f\left(\{e_j\}|S\right) &= \sum_{j=1}^{|\Omega|} (1 - \alpha) f\left(\{e_j\}|S \cup \{e_j\} \setminus \{e_j\}\right) \\
  &\leq \sum_{j=1}^{|\Omega|} f\left(\{e_j\}|S \cup \{e_j\} \setminus \{e_j\} \cup \Omega_{j-1}\right) \\
  &= f\left(\Omega|S\right).
  \end{aligned} \tag{7.22}$$

Because S and Ω are arbitrary disjoint sets, and both γ and $\check{\gamma}$ are the largest scalars such that for all disjoint sets $S, \Omega \subseteq V$ the following holds $\sum_{j=1}^{|\Omega|} f(\{e_j\}|S) \geq \gamma f(\Omega|S)$ and $\check{\gamma} \sum_{j=1}^{|\Omega|} f\left(\{e_j\}|S\right) \leq f\left(\Omega|S\right)$, it follows from Eq. (7.21) and Eq. (7.22), respectively, that $\gamma \geq 1 - \check{\alpha}$ and $\check{\gamma} \geq 1 - \alpha$.

Proof of Remark 1

Proof. Consider any set $S \subseteq V$, and A and B such that $A \cup B = S$, $A \cap B = \emptyset$. We have

$$\begin{aligned}\frac{f(A) + f(B)}{f(S)} &\geq \frac{\check{\nu} \sum_{i \in A} f(\{i\}) + \check{\nu} \sum_{i \in B} f(\{i\})}{f(S)} \\ &= \frac{\check{\nu} \sum_{i \in S} f(\{i\})}{f(S)} \geq \nu\check{\nu},\end{aligned}$$

where the first and second inequality follow by the definition of ν and $\check{\nu}$ (Eq. (7.7) and Eq. (7.8)), respectively. By the definition (see Eq. (7.9)), θ is the largest scalar such that $f(A) + f(B) \geq \theta f(S)$ holds, hence, it follows $\theta \geq \nu\check{\nu}$. □

7.A.2 Proofs of the Main Result (Section 7.3)

Proof of Lemma 7.3.2

Proof. We start by defining

$$S_0' := \mathrm{OPT}(k - \tau, V \setminus E_0) \cap (S_0 \setminus E_0),$$

and $X := \mathrm{OPT}(k - \tau, V \setminus E_0) \setminus S_0'$.

$$\begin{aligned}f(S_0 \setminus E_0) + f(\mathrm{OPT}(k - \tau, V \setminus S_0)) &\geq f(S_0') + f(X) && (7.23)\\ &\geq \theta f(\mathrm{OPT}(k - \tau, V \setminus E_0)) && (7.24)\\ &\geq \theta f(\mathrm{OPT}(k - \tau, V \setminus E_S^*)), && (7.25)\end{aligned}$$

where (7.23) follows from monotonicity as $S_0' \subseteq (S_0 \setminus E_0)$ and $(V \setminus S_0) \subseteq (V \setminus E_0)$. Eq. (7.24) follows from the fact that $\mathrm{OPT}(k - \tau, V \setminus E_0) = S_0' \cup X$ and the bipartite subadditive property (7.9). The final equation follows from the definition of the optimal solution and the fact that $E_S^* = E_0 \cup E_1$.

By rearranging and noting that $f(S \setminus E_S^*) \geq f(S_0 \setminus E_0)$ due to $(S_0 \setminus E_0) \subseteq (S \setminus E_S^*)$ and monotonicity, we obtain

$$f(S \setminus E_S^*) \geq \theta f(\mathrm{OPT}(k - \tau, V \setminus E_S^*)) - f(\mathrm{OPT}(k - \tau, V \setminus S_0)).$$

□

Proof of Theorem 7.3.1

Before proving the theorem we recall the following auxiliary result that can be found in Lemma 6.A.2:

Lemma 7.A.1. *For any set function f, sets A, B, and constant $\alpha > 0$, we have*

$$\max\{\alpha f(A), \beta f(B) - f(A)\} \geq \left(\frac{\alpha}{1+\alpha}\right) \beta f(B). \tag{7.26}$$

Next, we prove the main theorem.

Proof. We consider two cases, when $\mu = 0$ and $\mu \neq 0$. When $\mu = 0$, from Lemma 7.3.1 we have

$$f(S \setminus E_S^*) \geq f(S_1) \tag{7.27}$$

On the other hand, when $\mu \neq 0$, by Lemma 7.3.1 and 7.3.3 we have

$$\begin{aligned} f(S \setminus E_S^*) &\geq \max\{(1-\mu)f(S_1), (\beta-1)\check{\nu}(1-\check{\alpha})\mu f(S_1)\} \\ &\geq \frac{(\beta-1)\check{\nu}(1-\check{\alpha})}{1+(\beta-1)\check{\nu}(1-\check{\alpha})} f(S_1). \end{aligned} \tag{7.28}$$

By denoting

$$P := \frac{(\beta-1)\check{\nu}(1-\check{\alpha})}{1+(\beta-1)\check{\nu}(1-\check{\alpha})}$$

we observe that $P \in (0,1)$ once $\beta > 1$. Hence, by setting $\beta > 1$ and taking the minimum between the two bounds in Eq. (7.28) and Eq. (7.27) we conclude that Eq. (7.28) holds for any $\mu \in [0,1]$.

By combining Eq. (7.28) with Lemma 7.2.1 we obtain

$$f(S \setminus E_S^*) \geq P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0)). \tag{7.29}$$

By further combining this with Lemma 7.3.2 we have

$$\begin{aligned} f(S \setminus E_S^*) \geq \max\{&\theta f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \\ &P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0))\} \end{aligned} \tag{7.30}$$

$$\geq \theta \frac{P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right)}{1 + P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) \tag{7.31}$$

where the second inequality follows from Lemma 7.A.1.

By plugging in $\tau = \lceil ck \rceil$ we further obtain

$$\begin{aligned} f(S \setminus E_S^*) &\geq \theta \frac{P\left(1 - e^{-\gamma \frac{k-\beta\lceil ck\rceil - 1}{(1-c)k}}\right)}{1 + P\left(1 - e^{-\gamma \frac{k-\beta\lceil ck\rceil - 1}{(1-c)k}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) \\ &\geq \theta \frac{P\left(1 - e^{-\gamma \frac{1-\beta c - \frac{1}{k} - \frac{\beta}{k}}{1-c}}\right)}{1 + P\left(1 - e^{-\gamma \frac{1-\beta c - \frac{1}{k} - \frac{\beta}{k}}{1-c}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) \\ &\xrightarrow{k\to\infty} \frac{\theta P\left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)}{1 + P\left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)). \end{aligned}$$

It remains to set β. From the previous analysis we have $\beta > 1$. We note that β should be chosen such that the following condition holds $|S_0| = \lceil \beta\tau \rceil \leq k$. When $\tau = \lceil ck \rceil$ for $c \in (0,1)$ and $k \to \infty$ the condition $\beta < \frac{1}{c}$ suffices.

Finally, Remark 2 follows from Eq. (7.31) when $\tau \in o\left(\frac{k}{\beta}\right)$ and $\beta \geq \log k$ (note that the condition $|S_0| = \lceil \beta\tau \rceil \leq k$ is thus satisfied), as $k \to \infty$, we have both $\frac{k - \lceil \beta\tau \rceil}{k - \tau} \to 1$ and $P = \frac{(\beta-1)\check{\nu}(1-\check{\alpha})}{1+(\beta-1)\check{\nu}(1-\check{\alpha})} \to 1$, when $\check{\nu} \in (0,1]$ and $\check{\alpha} \in [0,1)$.

□

Proof of Corollary 7.3.1

To prove this result we need the following two lemmas that can be thought of as the alternative to Lemma 7.3.1 and 7.3.3.

Lemma 7.A.2. *Let $\mu' \in [0,1]$ be a constant such that $f(E_1) = \mu' f(S_1)$ holds. Consider $f(\cdot)$ with bipartite subadditivity ratio $\theta \in [0,1]$ defined in Eq.* (7.2.4)*. Then*

$$f(S \setminus E_S^*) \geq (\theta - \mu') f(S_1). \tag{7.32}$$

Proof. By the definition of θ, $f(S_1 \setminus E_1) + f(E_1) \geq \theta f(S_1)$. Hence,

$$\begin{aligned} f(S \setminus E_S^*) &\geq f(S_1 \setminus E_1) \\ &\geq \theta f(S_1) - f(E_1) \\ &= (\theta - \mu') f(S_1). \end{aligned}$$

□

Lemma 7.A.3. *Let β be a constant such that $|S_0| = \lceil \beta\tau \rceil$ and $|S_0| \leq k$, and let $\check{\nu}, \nu \in [0,1]$ be superadditivity and subadditivity ratio (Eq.* (7.8) *and Eq.* (7.7)*, respectively). Finally, let μ' be a constant defined as in Lemma 7.A.2. Then,*

$$f(S \setminus E_S^*) \geq (\beta - 1)\check{\nu}\nu\mu' f(S_1). \tag{7.33}$$

Proof. The proof follows that of Lemma 7.3.3, with two modifications. In Eq. (7.34) we used the subadditive property of $f(\cdot)$, and Eq. (7.35) follows by the definition of μ'.

$$\begin{aligned}
f(S \setminus E_S^*) &\geq f(S_0 \setminus E_0) \\
&\geq \check{\nu} \sum_{e_i \in S_0 \setminus E_0} f(\{e_i\}) \\
&\geq \frac{|S_0 \setminus E_0|}{|E_1|} \check{\nu} \sum_{e_i \in E_1} f(\{e_i\}) \\
&\geq \frac{(\beta-1)\tau}{\tau} \check{\nu} \sum_{e_i \in E_1} f(\{e_i\}) \\
&\geq (\beta - 1)\check{\nu}\nu f\left(E_1\right) && (7.34) \\
&= (\beta - 1)\check{\nu}\nu\mu' f(S_1). && (7.35)
\end{aligned}$$

□

Next we prove the main corollary. The proof follows the steps of the proof from Appendix 7.A.2, except that here we make use of Lemma 7.A.2 and 7.A.3.

Proof. We consider two cases, when $\mu' = 0$ and $\mu' \neq 0$. When $\mu' = 0$, from Lemma 7.A.2 we have

$$f(S \setminus E_S^*) \geq \theta f(S_1).$$

On the other hand, when $\mu' \neq 0$, by Lemma 7.A.2 and 7.A.3 we have

$$\begin{aligned}
f(S \setminus E_S^*) &\geq \max\{(\theta - \mu')f(S_1), (\beta - 1)\check{\nu}\nu\mu' f(S_1)\} \\
&\geq \theta \frac{(\beta-1)\check{\nu}\nu}{1 + (\beta-1)\check{\nu}\nu} f(S_1). && (7.36)
\end{aligned}$$

By denoting $P := \frac{(\beta-1)\check{\nu}\nu}{1+(\beta-1)\check{\nu}\nu}$ and observing that $P \in (0,1)$ once $\beta > 1$, we conclude that Eq. (7.36) holds for any $\mu' \in [0,1]$ once $\beta > 1$.

By combining Eq. (7.36) with Lemma 7.2.1 we obtain

$$f(S \setminus E_S^*) \geq \theta P \left(1 - e^{-\gamma \frac{k - \lceil \beta\tau \rceil}{k - \tau}}\right) f(\mathrm{OPT}(k - \tau, V \setminus S_0)). \tag{7.37}$$

By further combining this with Lemma 7.3.2 we have

$$f(S \setminus E_S^*) \geq \max\{\theta f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)) - f(\mathrm{OPT}(k-\tau, V \setminus S_0)), \tag{7.38}$$
$$\theta P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right) f(\mathrm{OPT}(k-\tau, V \setminus S_0))\}$$
$$\geq \frac{\theta^2 P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right)}{1 + \theta P\left(1 - e^{-\gamma \frac{k-\lceil \beta\tau \rceil}{k-\tau}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)), \tag{7.39}$$

where the second inequality follows from Lemma 7.A.1. By plugging in $\tau = \lceil ck \rceil$ in the last equation and by letting $k \to \infty$ we arrive at:

$$f(S \setminus E_S^*) \geq \frac{\theta^2 P\left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)}{1 + \theta P\left(1 - e^{-\gamma \frac{1-\beta c}{1-c}}\right)} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)).$$

Finally, from Eq. (7.39), when $\tau \in o\left(\frac{k}{\beta}\right)$ and $\beta \geq \log k$, as $k \to \infty$, we have both $\frac{k-\lceil \beta\tau \rceil}{k-\tau} \to 1$ and $P = \frac{(\beta-1)\check{\nu}\nu}{1+(\beta-1)\check{\nu}\nu} \to 1$ (when $\nu, \check{\nu} \in (0, 1]$). It follows

$$f(S \setminus E_S^*) \xrightarrow{k\to\infty} \frac{\theta^2(1 - e^{-\gamma})}{1 + \theta(1 - e^{-\gamma})} f(\mathrm{OPT}(k-\tau, V \setminus E_S^*)).$$

□

7.A.3 Proofs from Section 7.4

Proof of Proposition 7.4.1

Proof. The goal is to prove: $\check{\gamma} \geq \frac{m}{L}$. Let $S \subseteq [d]$ and $\Omega \subseteq [d]$ be any two disjoint sets, and for any set $A \subseteq [d]$ let $\boldsymbol{x}^{(A)} = \arg\max_{\mathrm{supp}(\boldsymbol{x}) \subseteq A, \boldsymbol{x} \in \mathcal{X}} l(\boldsymbol{x})$. Moreover, for $B \subseteq [d]$ let $\boldsymbol{x}_B^{(A)}$ denote those coordinates of vector $\boldsymbol{x}^{(A)}$ that correspond to the indices in B.

We proceed by upper bounding the denominator and lower bounding the numerator in (7.4). By definition of $\boldsymbol{x}^{(S)}$ and strong concavity of $l(\cdot)$,

$$l(\boldsymbol{x}^{(S\cup\{i\})}) - l(\boldsymbol{x}^{(S)}) \leq \langle \nabla l(\boldsymbol{x}^{(S)}), \boldsymbol{x}^{(S\cup\{i\})} - \boldsymbol{x}^{(S)} \rangle - \frac{m}{2} \left\| \boldsymbol{x}^{(S\cup\{i\})} - \boldsymbol{x}^{(S)} \right\|^2 \tag{7.40}$$
$$\leq \max_{\boldsymbol{v}: \boldsymbol{v}_{(S\cup\{i\})^c} = 0} \langle \nabla l(\boldsymbol{x}^{(S)}), \boldsymbol{v} - \boldsymbol{x}^{(S)} \rangle - \frac{m}{2} \left\| \boldsymbol{v} - \boldsymbol{x}^{(S)} \right\|^2 \tag{7.41}$$
$$= \frac{1}{2m} \left\| \nabla l(\boldsymbol{x}^{(S)})_i \right\|^2, \tag{7.42}$$

where the last equality follows by plugging in the maximizer $\boldsymbol{v} = \boldsymbol{x}^{(S)} + \frac{1}{m}\nabla l(\boldsymbol{x}^{(S)})_i$. Hence,

$$\sum_{i\in\Omega}\left(l(\boldsymbol{x}^{(S\cup\{i\})}) - l(\boldsymbol{x}^{(S)})\right) \leq \sum_{i\in\Omega}\frac{1}{2m}\left\|\nabla l(\boldsymbol{x}^{(S)})_i\right\|^2 = \frac{1}{2m}\left\|\nabla l(\boldsymbol{x}^{(S)})_\Omega\right\|^2.$$

On the other hand, from the definition of $\boldsymbol{x}^{(S\cup\Omega)}$ and due to smoothness of $l(\cdot)$ we have

$$\begin{aligned} l(\boldsymbol{x}^{(S\cup\Omega)}) - l(\boldsymbol{x}^{(S)}) &\geq l(\boldsymbol{x}^{(S)} + \frac{1}{L}\nabla l(\boldsymbol{x}^{(S)})_\Omega) - l(\boldsymbol{x}^{(S)}) \\ &\geq \langle \nabla l(\boldsymbol{x}^{(S)}), \frac{1}{L}\nabla l(\boldsymbol{x}^{(S)})_\Omega\rangle - \frac{L}{2}\left\|\frac{1}{L}\nabla l(\boldsymbol{x}^{(S)})_\Omega\right\|^2 \\ &= \frac{1}{2L}\left\|l(\boldsymbol{x}^{(S)})_\Omega\right\|^2. \end{aligned}$$

It follows that

$$\frac{l(\boldsymbol{x}^{(S\cup\Omega)}) - l(\boldsymbol{x}^{(S)})}{\sum_{i\in\Omega}\left(l(\boldsymbol{x}^{(S\cup\{i\})}) - l(\boldsymbol{x}^{(S)})\right)} \geq \frac{m}{L}, \quad \forall \text{ disjoint } S, \Omega \subseteq [d]$$

We finish the proof by noting that $\check{\gamma}$ is the largest constant for the above statement to hold.

□

Variance Reduction in GPs

Non-submodularity of Variance Reduction

The goal of this section is to show that the GP variance reduction objective is not submodular in general. Consider the following PSD kernel matrix:

$$\boldsymbol{K} = \begin{bmatrix} 1 & \sqrt{1-z^2} & 0 \\ \sqrt{1-z^2} & 1 & z^2 \\ 0 & z^2 & 1 \end{bmatrix}.$$

We consider a single $x = \{3\}$ (i.e. M is a singleton) that corresponds to the third data point. The objective is as follows:

$$F(i|S) = \sigma^2_{\{3\}|S} - \sigma^2_{\{3\}|S\cup i}.$$

The submodular property implies $F(\{1\}) \geq F(\{1\}|\{2\})$. We have:

$$\begin{aligned} F(\{1\}) &= \sigma^2_{\{3\}} - \sigma^2_{\{3\}|\{1\}} \\ &= 1 - K(\{3\},\{3\}) - K(\{3\},\{1\})(K(\{1\},\{1\}) + \sigma^2)^{-1}K(\{1\},\{3\}) \\ &= 1 - 1 + 0 = 0, \end{aligned}$$

$$\begin{aligned}F(\{2\}) &= \sigma^2_{\{3\}} - \sigma^2_{\{3\}|\{2\}} \\ &= 1 - K(\{3\},\{3\}) - K(\{3\},\{2\})(K(\{2\},\{2\}) + \sigma^2)^{-1}K(\{2\},\{3\}) \\ &= 1 - (1 - z^2(1+\sigma^2)^{-1}z^2) = \frac{z^4}{1+\sigma^2},\end{aligned}$$

and

$$\begin{aligned}F(\{1,2\}) &= \sigma^2_{\{3\}} - \sigma^2_{\{3\}|\{1,2\}} \\ &= 1 - K(\{3\},\{3\}) + [K(\{3\},\{1\}), K(\{3\},\{2\})] \begin{bmatrix} 1+\sigma^2, K(\{2\},\{1\}) \\ K(\{1\},\{2\}), 1+\sigma^2 \end{bmatrix}^{-1} \begin{bmatrix} K(\{1\},\{3\}) \\ K(\{2\},\{3\}) \end{bmatrix} \\ &= 1 - 1 + [0, z^2] \begin{bmatrix} 1+\sigma^2, \sqrt{1-z^2} \\ \sqrt{1-z^2}, 1+\sigma^2 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ z^2 \end{bmatrix} \\ &= \frac{z^4(1+\sigma^2)}{(1+\sigma^2)^2 - (1-z^2)}.\end{aligned}$$

We obtain,

$$\begin{aligned}F(\{1\}|\{2\}) &= F(\{1,2\}) - F(\{2\}) \\ &= \frac{z^4}{(1+\sigma^2) - (1-z^2)(1+\sigma^2)^{-1}} - \frac{z^4}{1+\sigma^2}.\end{aligned}$$

When $z \in (0,1)$, $F(\{1\}|\{2\})$ is strictly greater than 0, and hence greater than $F(\{1\})$. This is in contradiction with the submodular property which implies $F(\{1\}) \geq F(\{1\}|\{2\})$.

Proof of Proposition 7.4.2

Proof. We are interested in lower bounding the following ratios: $\frac{f(\{i\}|S\setminus\{i\}\cup\Omega)}{f(\{i\}|S\setminus\{i\})}$ and $\frac{f(\{i\}|S\setminus\{i\})}{f(\{i\}|S\setminus\{i\}\cup\Omega)}$.

Let $k_{\max} \in \mathbb{R}_+$ be the largest variance, i.e. $k(\boldsymbol{x}_i, \boldsymbol{x}_i) \leq k_{\max}$ for every i. Consider the case when M is a singleton set:

$$f(i|S) = \sigma^2_{\boldsymbol{x}|S} - \sigma^2_{\boldsymbol{x}|S\cup i}.$$

By using $\Omega = \{i\}$ in Eq. (7.43), we can rewrite $f(i|S)$ as

$$f(i|S) = a_i^2 B_i^{-1},$$

where $a_i, B_i \in \mathbb{R}_+$, and are given by:

$$a_i = k(\boldsymbol{x}, \boldsymbol{x}_i) - k(\boldsymbol{x}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}_i)$$

and

$$B_i = \sigma^2 + k(\boldsymbol{x}_i, \boldsymbol{x}_i) - k(\boldsymbol{x}_i, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}_i).$$

By using the fact that $k(\boldsymbol{x}_i, \boldsymbol{x}_i) \leq k_{\max}$, for every i and S, we can upper bound B_i by $\sigma^2 + k_{\max}$ (note that $k(\boldsymbol{x}_i, \boldsymbol{x}_i) - k(\boldsymbol{x}_i, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}_i) \geq 0$ as variance cannot be negative), and lower bound by σ^2. It follows that for every i and S we have:

$$\frac{a_i^2}{\sigma^2 + k_{\max}} \leq f(i|S) \leq \frac{a_i^2}{\sigma^2}.$$

Therefore,

$$\frac{f(\{i\}|S \setminus \{i\} \cup \Omega)}{f(\{i\}|S \setminus \{i\})} \geq \frac{a_i^2/(\sigma^2 + k_{\max})}{a_i^2/\sigma^2} = \frac{\sigma^2}{\sigma^2 + k_{\max}}, \quad \forall S, \Omega \subseteq V, i \in S \setminus \Omega,$$
$$\frac{f(\{i\}|S \setminus \{i\})}{f(\{i\}|S \setminus \{i\} \cup \Omega)} \geq \frac{a_i^2/(\sigma^2 + k_{\max})}{a_i^2/\sigma^2} = \frac{\sigma^2}{\sigma^2 + k_{\max}}, \quad \forall S, \Omega \subseteq V, i \in S \setminus \Omega.$$

It follows: $(1 - \alpha) \geq \frac{\sigma^2}{\sigma^2 + k_{\max}}$ and $(1 - \breve{\alpha}) \geq \frac{\sigma^2}{\sigma^2 + k_{\max}}$. The obtained result also holds for any set $M \subseteq [n]$. □

Alternative GP variance reduction form

Here, the goal is to show that the variance reduction can be written as

$$F(\Omega|S) = \sigma^2_{\boldsymbol{x}|S} - \sigma^2_{\boldsymbol{x}|S\cup\Omega} = \boldsymbol{a}\boldsymbol{B}^{-1}\boldsymbol{a}^T, \tag{7.43}$$

where $\boldsymbol{a} \in \mathbb{R}_+^{1\times|\Omega\setminus S|}$, $\boldsymbol{B} \in \mathbb{R}_+^{|\Omega\setminus S|\times|\Omega\setminus S|}$ and are given by:

$$\boldsymbol{a} := k(\boldsymbol{x}, \boldsymbol{X}_{\Omega\setminus S}) - k(\boldsymbol{x}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{X}_{\Omega\setminus S}),$$

and

$$\boldsymbol{B} := \sigma^2 \boldsymbol{I} + k(\boldsymbol{X}_{\Omega\setminus S}, \boldsymbol{X}_{\Omega\setminus S}) - k(\boldsymbol{X}_{\Omega\setminus S}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{X}_{\Omega\setminus S}).$$

This form is used in the proof in Appendix 7.A.3.

Proof. Recall the definition of the posterior variance:

$$\sigma^2_{\boldsymbol{x}|S} = k(\boldsymbol{x}, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{X}_S) \left(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I}_{|S|}\right)^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}).$$

We have

$$\begin{aligned} F(\Omega|S) &= \sigma^2_{\boldsymbol{x}|S} - \sigma^2_{\boldsymbol{x}|S\cup\Omega} \\ &= k(\boldsymbol{x}, \boldsymbol{X}_{S\cup\Omega}) \left(k(\boldsymbol{X}_{S\cup\Omega}, \boldsymbol{X}_{S\cup\Omega}) + \sigma^2 \boldsymbol{I}_{|\Omega\cup S|}\right)^{-1} k(\boldsymbol{X}_{S\cup\Omega}, \boldsymbol{x}) - k(\boldsymbol{x}, \boldsymbol{X}_S) \\ &\quad \times \left(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I}_{|S|}\right)^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}) \end{aligned}$$

$$= [\boldsymbol{m}_1, \boldsymbol{m}_2] \begin{bmatrix} \boldsymbol{A}_{11}, \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21}, \boldsymbol{A}_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{m}_1^T \\ \boldsymbol{m}_2^T \end{bmatrix} - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T,$$

where we use the following notation:

$$\begin{aligned} &\boldsymbol{m}_1 := k(\boldsymbol{x}, \boldsymbol{X}_S), \boldsymbol{m}_2 := k(\boldsymbol{x}, \boldsymbol{X}_{\Omega \setminus S}) \\ &\boldsymbol{A}_{11} := k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I}_{|S|}, \\ &\boldsymbol{A}_{12} := k(\boldsymbol{X}_S, \boldsymbol{X}_{\Omega \setminus S}), \\ &\boldsymbol{A}_{21} := k(\boldsymbol{X}_{\Omega \setminus S}, X_S), \\ &\boldsymbol{A}_{22} := k(\boldsymbol{X}_{\Omega \setminus S}, \boldsymbol{X}_{\Omega \setminus S}) + \sigma^2 \boldsymbol{I}_{|\Omega \setminus S|}. \end{aligned}$$

By using the inverse formula [PP+08, Section 9.1.3] we obtain:

$$F(\Omega|S) = [\boldsymbol{m}_1, \boldsymbol{m}_2] \begin{bmatrix} \boldsymbol{A}_{11}^{-1} + \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \boldsymbol{B}^{-1} \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1}, & -\boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \boldsymbol{B}^{-1} \\ -\boldsymbol{B}^{-1} \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1}, & \boldsymbol{B}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{m}_1^T \\ \boldsymbol{m}_2^T \end{bmatrix} - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T,$$

where $\boldsymbol{B} := \boldsymbol{A}_{22} - \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12}$.

Finally, we obtain:

$$\begin{aligned} F(\Omega|S) &= \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T + \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \boldsymbol{B}^{-1} \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T - \boldsymbol{m}_2 \boldsymbol{B}^{-1} \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T \\ &\quad - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \boldsymbol{B}^{-1} \boldsymbol{m}_2^T + \boldsymbol{m}_2 \boldsymbol{B}^{-1} \boldsymbol{m}_2^T - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T \\ &= \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \boldsymbol{B}^{-1} (\boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T - \boldsymbol{m}_2^T) - \boldsymbol{m}_2 \boldsymbol{B}^{-1} (\boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T - \boldsymbol{m}_2^T) \\ &= (\boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} - \boldsymbol{m}_2) \boldsymbol{B}^{-1} (\boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T - \boldsymbol{m}_2^T) \\ &= (\boldsymbol{m}_2 - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12}) \boldsymbol{B}^{-1} (\boldsymbol{m}_2^T - \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T). \end{aligned}$$

By setting

$$\begin{aligned} \boldsymbol{a} &:= \boldsymbol{m}_2 - \boldsymbol{m}_1 \boldsymbol{A}_{11}^{-1} \boldsymbol{A}_{12} \\ &= k(\boldsymbol{x}, \boldsymbol{X}_{\Omega \setminus S}) - k(\boldsymbol{x}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{X}_{\Omega \setminus S}) \end{aligned}$$

and

$$\begin{aligned} \boldsymbol{a}^T &:= \boldsymbol{m}_2^T - \boldsymbol{A}_{21} \boldsymbol{A}_{11}^{-1} \boldsymbol{m}_1^T \\ &= k(\boldsymbol{X}_{\Omega \setminus S}, \boldsymbol{x}) - k(\boldsymbol{X}_{\Omega \setminus S}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}_S, \boldsymbol{x}), \end{aligned}$$

we have $F(\Omega|S) = \boldsymbol{a} \boldsymbol{B}^{-1} \boldsymbol{a}^T$, where

$$\boldsymbol{B} = \sigma^2 \boldsymbol{I}_{|\Omega \setminus S|} + k(\boldsymbol{X}_{\Omega \setminus S}, \boldsymbol{X}_{\Omega \setminus S}) - k(\boldsymbol{X}_{\Omega \setminus S}, \boldsymbol{X}_S)(k(\boldsymbol{X}_S, \boldsymbol{X}_S) + \sigma^2 \boldsymbol{I}_{|S|})^{-1} k(\boldsymbol{X}_S, \boldsymbol{X}_{\Omega \setminus S}).$$

□

8 Conclusions and Future Work

As we discussed in Chapter 1, sequential optimization of an unknown black-box function is a central task in many interactive machine learning systems. In this dissertation, we have considered Bayesian optimization (BO), a powerful technique for adaptive experimentation with a broad range of applications, where the primary goal is the *model-based* optimization of the black-box function via sequentially chosen decisions. We investigated a variety of settings in Bayesian optimization that are of practical interest, with a particular focus on robust aspects such as robustness to adversarial input perturbations, unknown and uncertain parameters, time-variations in the objective, and more. In addition, we considered various other aspects of interest including pointwise costs and heteroscedastic noise, multi-fidelity setting, and a setting where the goal is to produce multiple sufficiently good solutions, i.e., the level-set estimation problem. The main goal of studying these diverse aspects is to enhance both robust and adaptive decision making in Bayesian optimization and related methods. Moreover, for many of the considered problems and settings, we demonstrated that the standard Bayesian optimization algorithms are inadequate or suffer from poor performance. We proposed novel algorithms for these problems that are based on the powerful idea of *confidence bounds* which, despite the lack of knowledge of the actual function, permit decision making based on the underlying model and the previous interactions. A common feature of all the proposed algorithms in this dissertation is that they are designed to exploit particular underlying structures that can be identified in many practical tasks and corresponding decision spaces: They take advantage of the fact that in many situations adjacent decisions result in similar outcomes and make use of the flexible Gaussian process models, and/or they make use of the (near) submodular structure that can be found in many applications and problems in machine learning. Furthermore, we obtained rigorous *regret guarantees* for our proposed algorithms, and for most of the settings studied in this dissertation, we complemented our results with algorithm-independent regret lower bounds.

Motivated by the batch setting in BO in which some of the selected experiments can fail, we have also considered a subset selection problem in which some of the elements from the selected set can be adversarially removed. We formalized this problem as a robust counterpart of the classical problem of submodular set function maximization subject to cardinality constraints.

We also investigated the case of non-submodular objectives, and we proposed efficient and practical algorithms that achieve constant factor approximation guarantees and allow for a greater number of removals in comparison to the previously known results. In numerous real-world applications, we demonstrated the robust performance of our algorithms by showing that they outperform other robust and non-robust selection strategies.

We continue by giving an overview of the main contributions of each chapter in this dissertation as well as discussing some ideas for future research.

In Chapter 3, we proposed a novel algorithm for Bayesian optimization and level-set estimation that is both versatile and efficient. We highlight the following aspects in which TRUVAR is versatile:

- Unified optimization and level-set estimation: These are typically treated separately, whereas TRUVAR and its theoretical guarantees are essentially identical in both cases.
- Actions with costs: TRUVAR naturally favors cost-effective points, as this is directly incorporated into the acquisition function.
- Heteroscedastic noise: TRUVAR chooses points that effectively shrink the variance of *other* points, thus directly taking advantage of situations in which some points are noisier than others. It is unclear how to incorporate this into most existing algorithms.
- Multi-fidelity setting: We provided strong theoretical guarantees for the case that the algorithm can choose both a point and a noise level, *cf.*, Corollary 3.5.1.

Hence, TRUVAR directly handles several important phenomena that are non-trivial to incorporate into most existing LSE and BO algorithms. Compared to other BO algorithms that perform a lookahead (e.g., ES and MRS), TRUVAR avoids the computationally expensive task of averaging over the posterior and/or measurements, and comes with rigorous theoretical guarantees.

Our obtained upper bounds on the simple regret for TRUVAR in the non-Bayesian GP optimization setting nearly match existing lower bounds [SBC17] for the squared exponential kernel, while in the case of the Matérn kernel the gaps are more significant. A direction for future research is to resolve whether the presence of γ_T^2 is necessary in our upper regret bound in comparison to having γ_T only (*cf.* Section 3.6). If such an improvement in the upper bound is possible, the new upper bound for the squared exponential kernel would match the lower bound up to the factor of 2 in the exponent to $\log T$. In the case of the Matérn kernel, this improvement in the bound would lead to the removal of the present condition (i.e., $2\nu - p(p+1)$ in Table 3.1), and hence, the bound would be valid for any dimension p and Matérn parameter ν.

In Chapter 4, we introduced and studied a variant of GP optimization in which one requires robustness to an adversarial input perturbation. We demonstrated the failures of existing algorithms and proposed a new algorithm STABLEOPT that is based on two principles, *optimism in the face of uncertainty* when it comes to deciding where to sample, and *pessimism in the face of uncertainty* when it comes to anticipating the perturbation of the considered decision.

In this setting, StableOpt is able to overcome the limitations of the standard algorithms and find the robust maximizer. We provided rigorous regret bounds for StableOpt and complemented them with algorithm-independent lower regret bounds. Moreover, we showed that our framework naturally applies to several max-min optimization formulations including robust Bayesian optimization, robust group identification, and settings in which robustness to unknown or uncertain parameters is required. In our experiments that involve real-world applications such as environmental monitoring, movie recommendation, and robotic manipulation tasks, we demonstrated significant improvements over several natural baselines.

In the robust setting that we considered in Chapter 4, the returned point is the only one that undergoes perturbation, while in some practical applications, even the points that we sample (i.e., experiments that we implement and run) might be adversarially altered. This setting can be formalized as follows: At time t, we decide to sample the function at $\boldsymbol{x}_t$, but the observation that we receive instead corresponds to some other adversarially perturbed input $\boldsymbol{x}_t + \boldsymbol{\delta}_t$. This setting is motivated by practical applications in which we cannot implement the selected design exactly, but we can only do so while encountering some implementation errors [BNT10b]. An interesting direction for future work is to investigate the appropriate regret metrics for this setting and discover the best achievable performance for different adversarial budgets. Another exciting direction for the future is to study the ϵ-stable optimization formulation in the context of hyperparameter tuning (e.g., for deep neural networks). One might expect that wide function maxima in hyperparameter space provide better generalization than narrow maxima, but establishing this requires investigation. Similar considerations are an ongoing topic of debate in understanding the *parameter space* rather than the hyperparmeter space, e.g., see [DPBB17].

In Chapter 5, we studied the GP optimization problem with time-varying rewards, taking a new approach based on a GP that evolves according to a simple Markov model. We introduced the R-GP-UCB and TV-GP-UCB algorithms, which, in contrast to previous algorithms, simultaneously trade off forgetting and remembering while also exploiting both spatial and temporal correlations. Our regret bounds for these algorithms provide, to our knowledge, the first explicit characterizations of the trade-off between the time horizon T and rate at which the function varies ϵ. We also provided an algorithm-independent bound revealing that a linear dependence on T for fixed ϵ is unavoidable. Despite the simplicity of our model, we saw that the algorithms performed well on real world data sets that need not be matched to this model.

An immediate direction for future research is to determine to what extent the dependence on ϵ can be improved in our upper and lower bounds. Moreover, one could move to the non-Bayesian setting and consider classes of time-varying functions whose smoothness is dictated by an RKHS norm. Furthermore, while our time-varying model is primarily suited to handling steady changes, it could potentially be made even more effective by explicitly handling *sudden* changes, e.g., by a combination of our techniques with those from previous works studying changepoint detection [AM07, GOR09, STR10]. In addition, different works [GSA14, GKX$^+$14] have addressed the standard Bayesian optimization in the case of additional constraints that are unknown a priori (i.e., constrained Bayesian optimization), and that are also modeled via Gaussian Processes.

In this setting, both the unknown objective and constraints can be (independently) evaluated. A natural extension would be to study the setup in which both constraints and objective vary with time. The goal would be to model these changes and to extend our algorithms, TV-GP-UCB and R-GP-UCB, to incorporate such information. Finally, we observed in our experiments (in Section 5.4) that R-GP-UCB is computationally more efficient while the gradual forgetting of TV-GP-UCB performs favorably compared to the sharp resetting of R-GP-UCB. An immediate direction is to consider new algorithmic ideas that would combine these two approaches and achieve best of both worlds, e.g., a sliding window (of fixed size) when it comes to which points to keep (instead of sharp resetting) and gradual forgetting inside this window.

Motivated by settings in which our decisions can fail, but we have no prior information on how failures occur, we have considered the problem of deciding on a robust set of decisions that maximize some objective of interest. In Chapter 6, we formalized this problem as a problem of submodular maximization in the case some number of the selected elements can be adversarially removed. We provided a new Partitioned Robust (PRO) submodular algorithm attaining a constant-factor approximation guarantee for general $\tau = o(k)$, thus resolving an open problem posed in [OSU16]. Our algorithm uses a novel partitioning structure with partitions consisting of buckets with exponentially decreasing size, thus providing a "robust part" of size $O(\tau \mathrm{poly} \log \tau)$. We considered applications such as robust influence maximization and data summarization and proved the first robust guarantees for the corresponding objectives. In a variety of numerical experiments, we demonstrated the robust performance of PRO by showing that it outperforms other robust and non-robust algorithms. In addition, in Chapter 7, we presented a practical algorithm OBLIVIOUS-GREEDY that achieves a constant-factor approximation guarantee for the robust maximization of monotone non-submodular objectives. The theoretical guarantees obtained hold in the linear regime, that is, for general $\tau = \lceil ck \rceil$ for some $c \in (0, 1)$. We also obtained the first robust guarantees for the support selection and GP variance reduction objectives. In various experiments, we demonstrated the robust performance of OBLIVIOUS-GREEDY by showing that it outperforms both OBLIVIOUS selection and GREEDY, and hence achieves the best of both worlds.

An immediate direction for future work is to understand whether the best-known approximation ratio of 0.387 can be obtained in the linear regime for general submodular objectives (in the case of fixed memory size k). More generally, the best possible polynomial-time approximation guarantee for this problem is not known for any non-trivial adversarial budget. Moreover, an interesting research direction is to go beyond the simple cardinality constraints and to investigate other types of constraints that are typically studied in the non-robust submodular maximization, e.g., the more general case of matroid constraints [CCPV07]. Furthermore, we have only considered monotone set functions in this dissertation; hence, an interesting direction is to study the same type of robust problem formulations in the case of non-monotone objectives [FMV11].

In Section 7.4.1, we considered the robust feature/support selection problem, where we assumed that the continuous loss function is (restricted) strongly convex and smooth. A potential direction for future work is to consider practical and theoretical approaches for robust feature selection

in the case of other more complex and non-convex loss functions. An interesting direction to investigate is whether the combination of such robust selection strategies and adversarial training [GSS15] can potentially result in neural network models that are robust to ℓ_0-norm attacks [CW17] (i.e., missing features in the test data). When it comes to other applications, it would be interesting to investigate if the methods developed in Chapter 6 and 7, can be used to provide more robust interpretations of machine learning models [RSG16, KL17]. Providing both robust and human interpretable set of features that are responsible for the model's prediction can be of great importance in real-world scenarios and decision supporting systems (e.g., in medical domains) where the expert might find some number of the returned features (i.e., explanations or symptoms) to be irrelevant.

Bibliography

[AB10] Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *Conference on Learning Theory (COLT)*, pages 13–p, 2010.

[ABF+16] Jason Altschuler, Aditya Bhaskara, Gang Fu, Vahab Mirrokni, Afshin Rostamizadeh, and Morteza Zadimoghaddam. Greedy column subset selection: New bounds and distributed algorithms. In *International Conference on Machine Learning (ICML)*, pages 2539–2548, 2016.

[ACBFS98] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. Technical report, http://www.dklevine.com/archive/refs4462.pdf, 1998.

[AFF10] Javad Azimi, Alan Fern, and Xiaoli Z. Fern. Batch bayesian optimization via simulation matching. In *Conference on Neural Information Processing Systems (NIPS)*, pages 109–117. 2010.

[AHP+17] Nima Anari, Nika Haghtalab, Sebastian Pokutta, Mohit Singh, Alfredo Torrico, et al. Robust submodular maximization: Offline and online algorithms. *arXiv preprint arXiv:1710.04740*, 2017.

[AJF12] Javad Azimi, Ali Jalali, and Xiaoli Zhang Fern. Hybrid batch Bayesian optimization. In *International Conference on Machine Learning (ICML)*, 2012.

[AM07] Ryan Prescott Adams and David J.C. MacKay. Bayesian online changepoint detection. http://arxiv.org/abs/0710.3742, 2007.

[BBKT17] Andrew An Bian, Joachim Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *International Conference on Machine Learning (ICML)*, 2017.

[BCB12] S. Bubeck and N. Cesa-Bianchi. *Regret Analysis of Stochastic and Nonstochastic Multi-Armed Bandit Problems*. Foundation and Trends in Machine Learning. Now Publishers, 2012.

[BCHS15] Ilija Bogunovic, Volkan Cevher, Jarvis Haupt, and Jonathan Scarlett. Active learning of self-concordant like multi-index functions. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2189–2193, 2015.

[BDKP15] Sébastien Bubeck, Ofer Dekel, Tomer Koren, and Yuval Peres. Bandit convex optimization: $\sqrt{T}$ regret in one dimension. In *Conference on Learning Theory (COLT)*, pages 266–278, 2015.

[BFNS14] Niv Buchbinder, Moran Feldman, Joseph Seffi Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Symposium on Discrete Algorithms (SODA)*, pages 1433–1452. SIAM, 2014.

[BGZ14] Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. In *Conference on Neural Information Processing Systems (NIPS)*, pages 199–207. 2014.

[Bha97] Rajendra Bhatia. *Matrix Analysis*. Springer, 1997.

[BLS+16] Luca Baldassarre, Yen-Huan Li, Jonathan Scarlett, Baran Gözcü, Ilija Bogunovic, and Volkan Cevher. Learning-based compressive subsampling. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):809–822, 2016.

[BMSC17a] Ilija Bogunovic, Slobodan Mitrović, Jonathan Scarlett, and Volkan Cevher. A distributed algorithm for partitioned robust submodular maximization. In *Comp. Adv. in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5. IEEE, 2017.

[BMSC17b] Ilija Bogunovic, Slobodan Mitrović, Jonathan Scarlett, and Volkan Cevher. Robust submodular maximization: A non-uniform partitioning approach. In *International Conference on Machine Learning (ICML)*, pages 508–516, 2017.

[BN17] Justin J. Beland and Prasanth B. Nair. Bayesian optimization under uncertainty. NIPS BayesOpt 2017 workshop, 2017.

[BNM00] Dimitris Bertsimas and José Niño-Mora. Restless bandits, linear programming relaxations, and a primal-dual index heuristic. *INFORMS Operations Research*, 48(1):80–90, 2000.

[BNT10a] Dimitris Bertsimas, Omid Nohadani, and Kwong Meng Teo. Nonconvex robust optimization for problems with constraints. *INFORMS Journal on Computing*, 22(1):44–58, 2010.

[BNT10b] Dimitris Bertsimas, Omid Nohadani, and Kwong Meng Teo. Robust optimization for unconstrained simulation-based problems. *INFORMS Operations Research*, 58(1):161–178, 2010.

[Bog12] Ilija Bogunovic. Robust protection of networks against cascading phenomena. Master's thesis, Department of Computer Science, ETH Zürich, 2012.

[BP18] Ricardo Baptista and Matthias Poloczek. Bayesian optimization of combinatorial structures. In *International Conference on Machine Learning (ICML)*, pages 471–480, 2018.

[BS08] Brent Bryan and Jeff G. Schneider. Actively learning level-sets of composite functions. In *International Conference on Machine Learning (ICML)*, 2008.

[BSC16] Ilija Bogunovic, Jonathan Scarlett, and Volkan Cevher. Time-varying Gaussian process bandit optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 314–323, 2016.

[BSJC18] Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust optimization with Gaussian processes. In *Conference on Neural Information Processing Systems (NIPS)*, 2018. "to appear".

[BSK16] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with Gaussian processes. In *International Conference on Robotics and Automation (ICRA)*, pages 491–496. IEEE, 2016.

[BSKC16] Ilija Bogunovic, Jonathan Scarlett, Andreas Krause, and Volkan Cevher. Truncated variance reduction: A unified approach to Bayesian optimization and level-set estimation. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1507–1515, 2016.

[BV14] Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *Symposium on Discrete Algorithms (SODA)*, pages 1497–1514, 2014.

[BZC18] Ilija Bogunovic, Junyao Zhao, and Volkan Cevher. Robust maximization of non-submodular objectives. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[CBRV13] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel Gaussian process optimization with upper confidence bound and pure exploration. In *Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 225–240. Springer, 2013.

[CC84] Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3):251–274, 1984.

[CCPV07] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 182–196. Springer, 2007.

[CG17] Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *International Conference on Machine Learning (ICML)*, pages 844–853, 2017.

[CK11] Volkan Cevher and Andreas Krause. Greedy dictionary selection for sparse representation. *Selected Topics in Signal Processing*, 5(5):979–988, 2011.

[CLB+17] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Conference on Neural Information Processing Systems (NIPS)*, pages 4299–4307, 2017.

[CLSS17] Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. Robust optimization for non-convex objectives. In *Conference on Neural Information Processing Systems (NIPS)*, pages 4708–4717, 2017.

[CLT+16] Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. Robust influence maximization. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 795–804. ACM, 2016.

[CRT06] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[CT01] Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 2001.

[CW17] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, pages 39–57, 2017.

[DK08] Abhimanyu Das and David Kempe. Algorithms for subset selection in linear regression. In *Symposium on Theory of Computing (STOC)*, pages 45–54, 2008.

[DK11] Abhimanyu Das and David Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *International Conference on Machine Learning (ICML)*, pages 1057–1064, 2011.

[DKB14] Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *Journal of Machine Learning Research*, 15(1):3873–3923, 2014.

[DKC13] Josip Djolonga, Andreas Krause, and Volkan Cevher. High-dimensional Gaussian process bandits. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1025–1033, 2013.

[DPBB17] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning (ICML)*, 2017.

[Duv14] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[EG16] Richard Evans and Jim Gao. Deepmind ai reduces google data centre cooling bill by 40%. https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/, 2016. Online; accessed 17 September 2018.

[EKDN16] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. Restricted strong convexity implies weak submodularity. *arXiv preprint arXiv:1612.00804*, 2016.

[FMV11] Uriel Feige, Vahab S Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[FPD08] Peter I Frazier, Warren B Powell, and Savas Dayanik. A knowledge-gradient policy for sequential information collection. *Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[GBWD+18] 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.

[GCHK13] Alkis Gotovos, Nathalie Casati, Gregory Hitz, and Andreas Krause. Active learning for level set estimation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1344–1350, 2013.

[GDDL] Javier González, Zhenwen Dai, Andreas Damianou, and Neil D. Lawrence. Preferential Bayesian optimization. In *International Conference on Machine Learning (ICML)*, page 1282.

[GDHL16] Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch Bayesian optimization via local penalization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 648–657, 2016.

[GKX+14] Jacob R. Gardner, Matt J. Kusner, Zhixiang Eddie Xu, Kilian Q. Weinberger, and John P. Cunningham. Bayesian optimization with inequality constraints. In *International Conference on Machine Learning (ICML)*, 2014.

[GOR09] Roman Garnett, Michael A Osborne, and Stephen J Roberts. Sequential Bayesian prediction in the presence of changepoints. In *International Conference on Machine Learning (ICML)*, 2009.

[GR06] Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *International Conference on Machine Learning (ICML)*, 2006.

[GSA14] Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In *Uncertainty in Artificial Intelligence (UAI)*, 2014.

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

[GWB97] Paul W Goldberg, Christopher KI Williams, and Christopher M Bishop. Regression with input-dependent noise: A Gaussian process treatment. *Conference on Neural Information Processing Systems (NIPS)*, 10:493–499, 1997.

[HJ12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 2 edition, 2012.

[HK16] Xinran He and David Kempe. Robust influence maximization. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2016.

[HLHG14] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Conference on Neural Information Processing Systems (NIPS)*, 2014.

[HPG+12] Gregory Hitz, Francois Pomerleau, Marie-Eve Garneau, Edric Pradalier, Thomas Posch, Jakob Pernthaler, and Roland Y. Siegwart. Autonomous inland water monitoring: Design and application of a surface vessel. *IEEE Robotics and Automation Magazine*, 19(1):62–72, 2012.

[HS12] Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13:1809–1837, 2012.

[HS17] Avinatan Hassidim and Yaron Singer. Submodular optimization under noise. In *Conference on Learning Theory (COLT)*, pages 1069–1122, 2017.

[HSL+17] Nicolas Heess, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, Ali Eslami, Martin Riedmiller, et al. Emergence of locomotion behaviours in rich environments. *preprint arXiv:1707.02286*, 2017.

[IJB13] Rishabh K Iyer, Stefanie Jegelka, and Jeff A Bilmes. Curvature and optimal algorithms for learning and minimizing submodular functions. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2742–2750, 2013.

[JN14] Kevin Jamieson and Robert Nowak. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *Information Sciences and Systems (CISS), 2014 48th Annual Conference on*, pages 1–6. IEEE, 2014.

[JTK14] Prateek Jain, Ambuj Tewari, and Purushottam Kar. On iterative hard thresholding methods for high-dimensional m-estimation. In *Conference on Neural Information Processing Systems (NIPS)*, pages 685–693, 2014.

[KCG16] Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best-arm identification in multi-armed bandit models. *Journal of Machine Learning Research*, 17(1):1–42, 2016.

[KED+17] Rajiv Khanna, Ethan Elenberg, Alexandros G. Dimakis, Sahand Negahban, and Joydeep Ghosh. Scalable greedy feature selection via weak submodularity. In

International Conference on Artificial Intelligence and Statistics (AISTATS), pages 1560–1568, 2017.

[KG05a] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 324–331, 2005.

[KG05b] Andreas Krause and Carlos Guestrin. A note on the budgeted maximization of submodular functions. Technical Report, 2005.

[KG07] Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *Conference on Artificial Intelligence (AAAI)*, 2007.

[KG12] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3(19):8, 2012.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.

[KL17] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, pages 1885–1894, 2017.

[KMGG08] Andreas Krause, H Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(Dec):2761–2801, 2008.

[KO11] Andreas Krause and Cheng S Ong. Contextual Gaussian process bandit optimization. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2447–2455, 2011.

[KSG08] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

[KSP15] Kirthevasan Kandasamy, Jeff Schneider, and Barnabás Póczos. High dimensional Bayesian optimisation and bandits via additive models. In *International Conference on Machine Learning (ICML)*, pages 295–304, 2015.

[KSU08] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *ACM Symposium on Theory of Computing (STOC)*, 2008.

[Kus64] Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.

[KZK17] Ehsan Kazemi, Morteza Zadimoghaddam, and Amin Karbasi. Deletion-robust submodular maximization at scale. *arXiv preprint arXiv:1711.07112*, 2017.

[LB11] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Association for Computational Linguistics: Human Language Technologies - Volume 1*, 2011.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBZK16] Mario Lucic, Olivier Bachem, Morteza Zadimoghaddam, and Andreas Krause. Horizontally scalable submodular maximization. In *International Conference on Machine Learning (ICML)*, 2016.

[LLZ13] Haoyang Liu, Keqin Liu, and Qing Zhao. Learning in a changing world: Restless multiarmed bandit with unknown dynamics. *IEEE Transactions on Information Theory*, 59(3):1902–1916, March 2013.

[LR85] T.L Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4 – 22, 1985.

[LWBS07] Daniel J Lizotte, Tao Wang, Michael H Bowling, and Dale Schuurmans. Automatic gait optimization with Gaussian process regression. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 944–949, 2007.

[MBK+15] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Conference on Artificial Intelligence (AAAI)*, 2015.

[MBNF+17] Slobodan Mitrovic, Ilija Bogunovic, Ashkan Norouzi-Fard, Jakub M Tarnawski, and Volkan Cevher. Streaming robust submodular maximization: A partitioned thresholding approach. In *Conference on Neural Information Processing Systems (NIPS)*, pages 4560–4569, 2017.

[MBS09] Rémi Munos, Sébastien Bubeck, and Gilles Stoltz. Pure exploration for multi-armed bandit problems. *Lecture Notes in Computer Science*, 2009.

[MCTM18] Ruben Martinez-Cantin, Kevin Tee, and Michael McCourt. Practical Bayesian optimization in the presence of outliers. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[Met16] Jan Hendrik Metzen. Minimum regret search for single-and multi-task optimization. In *International Conference on Machine Learning (ICML)*, 2016.

[Min78] Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pages 234–243. Springer, 1978.

[Mir17] Baharan Mirzasoleiman. *Big Data Summarization Using Submodular Functions*. PhD thesis, ETH Zurich, 2017.

[MKK17] Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Deletion-robust submodular maximization: Data summarization with "the right to be forgotten". In *International Conference on Machine Learning (ICML)*, pages 2449–2458, 2017.

[ML14] Julian Mcauley and Jure Leskovec. Discovering social circles in ego networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2014.

[MLG04] Omid Madani, Daniel J Lizotte, and Russell Greiner. The budgeted multi-armed bandit problem. In *Conference on Learning Theory (COLT)*, pages 643–645. Springer, 2004.

[NFBB+16] Ashkan Norouzi-Fard, Abbas Bazzi, Ilija Bogunovic, Marwa El Halabi, Ya-Ping Hsieh, and Volkan Cevher. An efficient streaming algorithm for the submodular cover problem. In *Conf. on Neural Information Processing Systems (NIPS)*, 2016.

[NMCBJ16] J. Nogueira, R. Martinez-Cantin, A. Bernardino, and L. Jamone. Unscented Bayesian optimization for safe robot grasping. In *International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[NW78] George L Nemhauser and Leonard A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

[NWF78] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[OR12] Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1772–1780. 2012.

[ORR+08] Michael A Osborne, SJ Roberts, A Rogers, SD Ramchurn, and Nicholas R Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *International Conference on Information Processing in Sensor Networks*, pages 109–120, 2008.

[OSU16] James B Orlin, Andreas S Schulz, and Rajan Udwani. Robust monotone submodular function maximization. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*. Springer, 2016.

[PBW+16] Thomas Powers, Jeff Bilmes, Scott Wisdom, David W Krout, and Les Atlas. Constrained robust submodular optimization. NIPS OPT2016 workshop, 2016.

[PP+08] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[Rec18] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *arXiv preprint arXiv:1806.09460*, 2018.

[RMGO18] Bin Xin Ru, Mark McLeod, Diego Granziol, and Michael A. Osborne. Fast information-theoretic bayesian optimisation. In *International Conference on Machine Learning (ICML)*, pages 4381–4389, 2018.

[RSBC18] Paul Rolland, Jonathan Scarlett, Ilija Bogunovic, and Volkan Cevher. High-dimensional Bayesian optimization via additive models with overlapping groups. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 298–307, 2018.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 1135–1144, 2016.

[Rup15] Matthias Rupp. Machine learning for quantum mechanics in a nutshell. *International Journal of Quantum Chemistry*, 115(16):1058–1073, 2015.

[RW06] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[SAMR18] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Control, Robotics, and Autonomous Systems*, 1:187–210, 2018.

[SBC17] Jonathan Scarlett, Ilijia Bogunovic, and Volkan Cevher. Lower bounds on regret for noisy Gaussian process bandit optimization. In *Conference on Learning Theory (COLT)*, 2017.

[SF11] Zoya Svitkina and Lisa Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. *SIAM Jour. on Computing*, 40(6):1715–1737, 2011.

[SGBK15] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with Gaussian processes. In *International Conference on Machine Learning (ICML)*, pages 997–1005, 2015.

[SJ17a] Shubhanshu Shekhar and Tara Javidi. Gaussian process bandits with adaptive discretization. *arXiv preprint arXiv:1712.01447*, 2017.

[SJ17b] Matthew Staib and Stefanie Jegelka. Robust budget allocation via continuous submodular functions. In *International Conference on Machine Learning (ICML)*, pages 3230–3240, 2017.

[SKKS10] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Inter. Conference on Machine Learning (ICML)*, pages 1015–1022, 2010.

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2951–2959, 2012.

[SND18] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations (ICLR)*, 2018.

[SS98] Alex J Smola and Bernhard Schölkopf. *Learning with kernels*. Citeseer, 1998.

[SSA13] Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task Bayesian optimization. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2004–2012, 2013.

[SSA14] Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw Bayesian optimization. http://arxiv.org/abs/1406.3896, 2014.

[SSS+17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550, 2017.

[SSW+16] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[SSZA14] Jasper Snoek, Kevin Swersky, Richard Zemel, and Ryan P. Adams. Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning (ICML)*, 2014.

[STR10] Yunus Saat, Ryan Turner, and Carl Edward Rasmussen. Gaussian process change point models. In *International Conference on Machine Learning (ICML)*, 2010.

[SU08] Aleksandrs Slivkins and Eli Upfal. Adapting to a changing environment: the Brownian restless bandits. In *Conference on Learning Theory (COLT)*, 2008.

[SWJ18] Matthew Staib, Bryan Wilder, and Stefanie Jegelka. Distributionally robust submodular maximization. *arXiv preprint arXiv:1802.05249*, 2018.

[TFF08] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[TGJP17] Vasileios Tzoumas, Konstantinos Gatsis, Ali Jadbabaie, and George J Pappas. Resilient monotone submodular function maximization. In *Conference on Decision and Control (CDC)*, pages 1362–1367. IEEE, 2017.

[TJP18] Vasileios Tzoumas, Ali Jadbabaie, and George J Pappas. Resilient monotone sequential maximization. *arXiv preprint arXiv:1803.07954*, 2018.

[TL12] C. Tekin and Mingyan Liu. Online learning of rested and restless bandits. *IEEE Transactions on Information Theory*, 58(8):5588–5611, Aug. 2012.

[Udw17] Rajan Udwani. Multi-objective maximization of monotone submodular functions with cardinality constraint. *arXiv preprint arXiv:1711.06428*, 2017.

[VNDBK14] Hastagiri P Vanchinathan, Isidor Nikolic, Fabio De Bona, and Andreas Krause. Explore-exploit in top-n recommender systems via Gaussian processes. In *Conference on Recommender Systems (RecSys)*, pages 225–232. ACM, 2014.

[Von10] Jan Vondrák. Submodularity and curvature: The optimal algorithm. *Kokyuroku Bessatsu*, page 23:253–266, 2010.

[VVLGS12] Steven Van Vaerenbergh, Miguel Lázaro-Gredilla, and Ignacio Santamaría. Kernel recursive least-squares tracker for time-varying regression. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1313–1326, 2012.

[VVSLG12] Steven Van Vaerenbergh, Ignacio Santamaría, and Miguel Lázaro-Gredilla. Estimation of the forgetting factor in kernel recursive least squares. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2012.

[WdF14] Ziyu Wang and Nando de Freitas. Theoretical analysis of Bayesian optimisation with unknown Gaussian process hyper-parameters. *arXiv preprint arXiv:1406.7758*, 2014.

[Whi88] P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25:287–298, 1988.

[WIB15] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning (ICML)*, pages 1954–1963, 2015.

[Wil17] Bryan Wilder. Equilibrium computation for zero sum games with submodular structure. In *Conference on Artificial Intelligence (AAAI)*, 2017.

[WJ17] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *International Conference on Machine Learning (ICML)*, pages 3627–3635, 2017.

[WLJK17] Zi Wang, Chengtao Li, Stefanie Jegelka, and Pushmeet Kohli. Batched high-dimensional Bayesian optimization via structural kernel learning. In *International Conference on Machine Learning (ICML)*, pages 3656–3664, 2017.

[Wol82] Laurence A Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

[WSJF14] Ziyu Wang, Babak Shakibi, Lin Jin, and Nando Freitas. Bayesian multi-scale optimistic optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1005–1014, 2014.

[WZH+13] Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando de Freitas. Bayesian optimization in high dimensions via random embeddings. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

**Ce document a été imprimé au Centre d'impression EPFL,
imprimerie climatiquement neutre, certifiée myClimate.**