NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

AN ONLINE LEARNING FRAMEWORK FOR INTELLIGENCE COLLECTION IN UNCERTAIN ENVIRONMENTS

by

Amit Carmeli

September 2021

Thesis Advisor: Roberto Szechtman
Second Reader: Moshe Kress

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2021	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE AN ONLINE LEARNING FRAMEWORK FOR INTELLIGENCE COLLECTION IN UNCERTAIN ENVIRONMENTS	**5. FUNDING NUMBERS**
6. AUTHOR(S) Amit Carmeli	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The process of creating an intelligence status report requires a continuous collection of intelligence from various sources of varying inaccuracies and reliability. Consequently, managing many intelligence sources is not only a costly operation to establish but also to maintain continuously. Our objective is to use the Multi-armed Bandit (MAB) framework to model intelligence collection. The proposed framework generalizes the classical MAB model by accounting for censoring in sampled observations in a resource-constrained environment. We devise an online optimization framework, accompanied by rigorous analysis and comprehensive numerical experiments, that sheds light on this real-world problem.

14. SUBJECT TERMS online learning, Multi-armed Bandit, MAB, intelligence collection, censoring	**15. NUMBER OF PAGES** 81
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

AN ONLINE LEARNING FRAMEWORK FOR INTELLIGENCE COLLECTION IN UNCERTAIN ENVIRONMENTS

Amit Carmeli
Seren, Israel Defence Forces
B. Sc in Software Engineering, Technion, Israel Institute of Technology, 2013

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 2021

Approved by: Roberto Szechtman
Advisor

Moshe Kress
Second Reader

W. Matthew Carlyle
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The process of creating an intelligence status report requires a continuous collection of intelligence from various sources of varying inaccuracies and reliability. Consequently, managing many intelligence sources is not only a costly operation to establish but also to maintain continuously. Our objective is to use the Multi-armed Bandit (MAB) framework to model intelligence collection. The proposed framework generalizes the classical MAB model by accounting for censoring in sampled observations in a resource-constrained environment. We devise an online optimization framework, accompanied by rigorous analysis and comprehensive numerical experiments, that sheds light on this real-world problem.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

MAB	Multi-armed Bandit
UCB	Upper Confidence Bound
OGA	Online Gradient Ascent
OCO	Online Convex Optimization
FKM	Flaxman, Kalai, and McMahan
DKW	Dvoretzky-Kiefer-Wolfowitz
DKWUCB	Dvoretzky-Kiefer-Wolfowitz Inequality based Upper Confidence Bound
KMUCB	Kaplan-Meier based Upper Confidence Bound
UCBMB	Upper Confidence Bound for Multi-Play with Budget

THIS PAGE INTENTIONALLY LEFT BLANK

Executive Summary

This work focuses on the problem of intelligence collection in uncertain environment under budgetary constraints. The problem tackled by the research is difficult not only due to the uncertain nature of information gained, but also since the value of the information is often tied to the level of effort required to collect that information.

Our research provides an optimization framework, based on Multi-armed Bandits and online optimization, to address the problem. The framework is novel, as it incorporates together several modeling elements that have been previously explored only separately.

We model information gained from intelligence sources by censoring. That is, efforts to extract information from an intelligence source must exceed an unknown threshold to be successful. Censoring of information ensures that the decision maker must spend sufficient resources to collect valuable information. The goal of the framework is to maximize the value of information gained and observed (not censored) in a limited time period under budgetary constraints.

We provide rigorous analysis of the proposed optimization problem, and propose an elegant algorithm, called $(K+1)$-UCB, to solve it. The algorithm aims to explore each intelligence source as much as possible. By fusing the information gained from all intelligence sources, the algorithm uncovers an exploitation point - an optimal allocation of the budget that can maximize the value of information observed.

Our analysis shows that the algorithm performance is competitive with other algorithms designed to solve similar problems. Through various numerical experiments, we support our theoretical analysis and show that in practice our algorithm performance is often better than existing algorithms.

Finally, we highlight potential future work within the proposed framework that may be able to further improve our results.

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I wish to thank my advisor for guiding me through this research work. It was a great opportunity to explore a unique and innovative problem in a domain that was new and exciting to me. Your guidance and support of me throughout the research is greatly appreciated, as I was made to feel not only as an active participant and a contributor to our work, but also a research colleague.

I would like to thank my second reader for their comments and ensuring this work is of the highest quality as possible.

I also thank my parents and my sister, Dana, for their remote support while conducting this research, as well as my commanding officers, Ohad, Yoav and Aviv, for their belief in me and my work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

1.1 Motivation

Modern intelligence collection faces many challenges, and collectors of intelligence must be able to practice many disciplines to create a reliable intelligence picture [1]. The importance of maintaining a plethora of informative intelligence sources cannot be overstated, as also evident by the billion dollar U.S. Intelligence Community Budget [2].

The large budget and the multitude of disciplines aim to mitigate the highly-uncertain nature of intelligence collection. Typically, one can model uncertainty of an intelligence source via a random variable, as a method to account for both variability of information and enemy mitigation [3], [4]. We wish to expand on this modeling approach of intelligence as a random variable by also tying the information gained from an intelligence source to the resources the intelligence community is willing to commit to process that source.

An online learning framework can be very attractive to explore in the context of this problem as the framework can model the time-critical decision making process. Our goal is to use this framework to appropriately model the uncertainty tied to both Blue and Red actions to the information gain and design methods to optimize that information gain.

1.2 Model

A decision maker is given the task of allocating resources (such as manpower, budget or otherwise) to the intelligence community, and specifically, to different intelligence sources or operations. Each source or operation yields useful information, referred to as a "reward," which is valuable to the decision maker.

Importantly, the reward each source can yield is tied to both Red and Blue actions in the following ways. Firstly, we assume for simplicity that each non-negative reward is independently sampled from some distribution. These distributions are unknown to the decision maker. Secondly, if the resources allocated to a certain source are insufficient,

a reward may be censored from the decision maker. A censored reward implies that no information (that is, information of zero value) was gained from a source. The problem faced by the decision maker is to allocate resources in order to maximize the expected value of information collected.

1.3 Novelty

We focus our attention to the Multi-armed Bandit (MAB) [5] framework in order to solve the problem. The MAB framework can model each intelligence source as a different arm with an unknown underlying distribution of rewards (information). In addition, we incorporate both a budget allocated to play each of the arms as well as a censoring mechanism that can eliminate rewards (setting them to zero) in a manner described in [6]. We note that while many of these elements have been explored in the context of MAB in the past, our literature review (Chapter 2) shows that no singular work attempted to incorporate them all together, let alone in the context of intelligence collection.

1.4 Research Questions of Interest

Our research is mainly concerned with the following questions:

- What is the best approach to collection information from intelligence source in order to evaluate it, given a budget and the censored-nature of the information?
- What assumptions are required in order to address the problem algorithmically?
- Given the answers to the previous question, can we devise an algorithm to address the problem, despite the censored-nature of information? What is the performance of such an algorithm?

CHAPTER 2:
Background

In this chapter, we shall introduce the online optimization framework and the MAB problem. We will note different algorithms designed to address the MAB problem and its variants. We will also discuss how different MAB variants can prove useful for the modeling of our problem.

2.1 Online Optimization Framework

We begin by describing a general online optimization framework. An online optimization framework includes a *learner* (e.g., a decision-maker) that interacts with a time-varied environment in an attempt to maximize reward. Formally, the learner is given a *time-horizon* $T \in \mathbb{N}$ and each time-step $t = 1, 2, ..., T$, must decide on a certain K-dimensional action $\mathbf{x}_t \in \mathcal{S} \subset \mathbb{R}^K$, where $\mathcal{S}$ denotes the set of allowable policies. Once an action has been set, the learner receives feedback in the form of a time-dependent reward function $f_t(\mathbf{x}_t)$.

A policy π dictates the actions taken by the learner over the time horizon, $\pi = \{\mathbf{x}_t\}_{t=1}^T$, and typically depends on the information collected so far. The goal of the learner is to minimize the *regret* from deviating from the optimal policy. Formally, the regret of a policy π is defined via

$$r_T(\pi) = \max_{\{\mathbf{x}_t \in \mathcal{S}\}_{t=1}^T} \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} \mathbb{E}(f_t(\mathbf{x}_t)). \tag{2.1}$$

If the functions f_t for $t = 1, ..., T$ are convex, and the set $\mathcal{S}$ is also convex, this framework is also known as Online Convex Optimization (OCO) [7]. The convexity property of the problem allows us to address the problem using Online Gradient Ascent (OGA) [7]. The OGA algorithm continuously improves the chosen action $\mathbf{x}_t$ at each time-step by observing the *gradient* of the function f_t at the point $\mathbf{x}_t$ and taking a proximal step in the ascent direction projected by the gradient.

2.2 From Online Optimization to MAB

The classical MAB problem is inspired by the notion of a gambler aiming to choose the best sequence of one-armed bandit machines to play each time [5]. The gambler's decisions are based solely on past observations, and the gambler can only play one machine each time. This problem can be cast as a special case of the online optimization framework.

In the MAB problem, we have a set of allowable actions $\mathcal{S}_{MAB} = \{e_i \in \mathbb{R}^K\}$, where e_i is a standard basis vector, whose i^{th} coordinate is 1 and all other coordinates are zero. Each action plays one of K arms, whose underlying reward distribution has a mean $\mu_i, i \in [K]$. We let $\mu \in \mathbb{R}^K$ be the vector whose coordinates are μ_i, so that $\mathbb{E}(f_t(\mathbf{x}_t)) = \mu^T \mathbf{x}_t$. Since $\mathbf{x}_t = e_i$ for some i, then $\mathbb{E}(f_t(\mathbf{x}_t)) = \mu_i$. The motivation behind exploring the MAB within the online optimization framework is that it allows to easily expand the problem to support *mixed* policies, that is $\mathbf{x} \notin \mathcal{S}_{MAB}$.

We note that this formulation of MAB is *not* an OCO. While the functions f_t can be expressed as the linear functions $f_t(\mathbf{x}_t) = \mu^T \mathbf{x}_t$, the set $\mathcal{S}_{MAB} = \{e_i \in \mathbb{R}^K\}$ is not convex. If we wish to address the MAB problem using the OCO framework, one potential relaxation [7] is to replace $\mathcal{S}$ with the $K-$dimensional simplex, denoted Δ_K.

2.2.1 Solving the MAB Problem

In this section, we discuss two important algorithms used to solve the MAB problem: Upper Confidence Bound (UCB) and the Flaxman, Kalai, and McMahan (FKM) algorithm [7].

UCB

MAB algorithms aim to balance between *exploration* of different arms and *exploitation* of arms whose mean reward is the best sampled so far. The UCB algorithm [5] balances exploration and exploitation by adopting the "Optimism Under Uncertainty" principle, that is, by assuming that arm means are as high as possible, based on what observed. The UCB algorithm makes use of *indices* for each arm that serve as a ranking mechanism that guides the algorithm which arm to play at each time-step t. It has been shown [5] that the regret is logarithmic in the time-horizon, $r_T(\pi_{UCB}) = O(\log T)$, where π_{UCB} is the policy dictated by the UCB algorithm. A pseudo-code of the UCB method is shown in Algorithm 1.

Algorithm 1 defines an *index* for each arm to balance between *exploration* and *exploitation*. With exploration, the algorithm spends a time-step to sample an arm to increase its confidence in its evaluation of the mean. With exploitation, the algorithm will sample the arm deemed most valuable. UCB implements this balance elegantly by accounting for a confidence interval for each sampled mean of each arm. The algorithm takes an optimistic approach in the sense that we only use the upper-bound of the confidence interval - hence, the name Upper Confidence Bound.

Importantly, we note that the form of the upper-bound of the confidence interval is dependent on both the current time-step t as well as the number of samples of each arm $N_{k,t}$. As t increases, the confidence interval becomes much looser for under-explored arms, allowing the algorithm to perform exploration of those arms. If $N_{k,t}$ increases, the confidence interval becomes tighter, offsetting the natural expansion of the confidence interval. Observe that the confidence interval dependency in those factors that allow this behavior is of the form $o(t)/N_{k,t}$. Further details can be found in [5].

Algorithm 1: UCB for Classical MAB

Input: $K \in \mathbb{N}_+$ (number of arms),
$T \in \mathbb{N}_+$ (time-horizon)

Output: $\pi_{UCB} = \{\mathbf{x}_t\}_{t=1}^{T}$ (UCB policy)

$\forall k = 1, ..., K : UCB_{k,0} \leftarrow \infty, N_{k,0} \leftarrow 0, \hat{\mu}_{k,0} \leftarrow 0$

for *t = 1, 2, ..., T* **do**

- $k_t \leftarrow \arg\max_{k=1,..,K} UCB_{k,t-1}$ (choose best arm)
- $\mathbf{x}_t \leftarrow e_{k_t}$
- $y_t \leftarrow f_t(\mathbf{x}_t)$ (play arm i_t and collect information of value y_t)
- $N_{k_t,t} \leftarrow N_{k_t,t-1} + 1$
- $\hat{\mu}_{k_t,t} \leftarrow (\hat{\mu}_{k_t,t-1} + y_t)/N_{k_t,t-1}$
- $UCB_{k_t,t} \leftarrow \hat{\mu}_{k_t,t} + \sqrt{2 \log t / N_{k_t,t}}$ (update UCB index)

end

FKM Algorithm

While the OGA algorithm is a compact method to solve problems formulated as OCO, we cannot directly use it for the MAB problem. First, as mentioned above, we must relax the

requirement on the set of allowable policies from $\mathcal{S} = \{e_i \in \mathbb{R}^K\}$ to Δ_K. Second, the MAB problem does not have access to the gradients of the functions f_t.

In order to address these issues, a variant known as FKM [7] can be used to estimate the gradients of f_t without the additional assumption that the learner has access of such gradients. For the FKM algorithm, it was shown that the regret is $r_T(\pi_{FKM}) = O(T^{3/4})$.

Algorithm 2: FKM algorithm for Convex Online Optimization

Input: $K \in \mathbb{N}_+$ (number of arms)
$T \in \mathbb{N}_+$ (time-horizon)
$\delta > 0$
$\rho > 0$

Output: $\pi_{FKM} = \{\mathbf{x}_t\}_{t=1}^T$ (FKM policy)

$\mathbf{x}_1 \leftarrow \mathbf{0}$

for *t = 1, 2, ..., T* **do**
  Randomly generate $\mathbf{u}_t \in \mathbb{S}_1 = \{\mathbf{u} \in \mathbb{R}^K | ||\mathbf{u}||_2 = 1\}$
  $\mathbf{z}_t \leftarrow \mathbf{x}_t + \delta \mathbf{u}_t$
  $y_t \leftarrow f_t(\mathbf{z}_t)$ (play action $\mathbf{z}_t$)
  $\mathbf{g}_t \leftarrow \frac{K}{\delta} f_t(\mathbf{z}_t)\mathbf{u}_t$
  $\mathbf{x}_{t+1} = P_{\Delta_K(\delta)}(\mathbf{x}_t + \rho \mathbf{g}_t)$ (take a projected-gradient step in ascent direction)
end

We note that the algorithm requires the use of $P_{\Delta_K(\delta)}$, an orthogonal projection operator onto the set $\Delta_K(\delta) = \{\mathbf{x} \in \mathbb{R}^K | \frac{1}{1-\delta}\mathbf{x} \in \Delta_K\}$. There are several algorithms existing for this purpose such as [8] and [9]. Importantly, since FKM projects onto $\Delta_K(\delta)$ and not Δ_K, the choice of δ affects the *actual* set of proposed policies by FKM.

2.3 Variants on the MAB Problem

We now focus our attention on certain variants on the original MAB problem. These variants introduce additional components to the classical MAB problem that make the MAB problem more challenging but also more relevant to our modeling efforts.

2.3.1 Continuum MAB

Kleinberg [10] has explored a variant known as "Continuum MAB", where $\mathbf{x}_t$ is not bounded to the classical definition of $\mathcal{S}$ discussed above. Specifically, the work in [10] introduces an optimal algorithm for the specific case where $K = 2$.

Algorithm 3: Kleinberg's Continuum MAB for $K = 2$ Arms

Input: $T \in \mathbb{N}_+$ (time-horizon)

Output: $\pi_{Kleinberg} = \{\mathbf{x}_t\}_{t=1}^{T}$ (Kleinberg's policy)

$t \leftarrow 1$

while $t \leq T$ **do**

  $K_t \leftarrow (t/\log t)^{1/3}$

  Initialize MAB algorithm over K_t arms

  for $t_k = t, t+1, ..., \min(2t-1, T)$ **do**

    $i_t \leftarrow$ Best arm from MAB algorithm

    $\mathbf{x}_t \leftarrow (\frac{i_t}{K_t}, 1 - \frac{i_t}{K_t})^T$ (transform the discrete arm i_t into a 2-dimensional point)

    $y_t \leftarrow f_t(\mathbf{x}_t)$ (play 2-dimensional point corresponding to i_t)

    Update MAB algorithm with reward y_t from playing i_t

  end

  $t \leftarrow 2t$

end

Algorithm 3 details Kleinberg's approach for the $K = 2$ case. The algorithm divides the time horizon $1, ..., T$ into phases, each with twice as many time steps as the previous phase (but such that the total number of time steps is still T). In each phase, an MAB with K_t arms is used, with each arm representing a point in 2-dimensional space. For $i \in [K_t]$, the i^{th} arm is the vector $(\frac{i}{K_t}, 1 - \frac{i}{K_t})^T \in \Delta^2$. If the internal MAB used in each phase chooses a specific arm index i to be played, Kleinberg's method then samples the corresponding vector.

The method proposed by Kleinberg runs an MAB algorithm with each discrete arm index mapped to a point in continuous space. By iteratively increasing the number of arms, the internal MAB algorithm can sample more points in the $\mathbb{R}^2$ space and further approach to the optimal point. Consequently, Kleinberg's work ties together the discrete nature of MAB and the continuous nature of the problem. Kleinberg's algorithm achieves a regret of $r_T(\pi_{Kleinberg}) = O(T^{2/3})$ [10]. The importance of Kleinberg's algorithm is the fact it can work in a continuous space without needing to tune a step-size or approximate the space

with a parameter as FKM requires.

We note that Kleinberg et al. developed a generalization of their technique to arbitrary metric spaces in [11]. In their work, an algorithm called the *zooming algorithm* is capable of zooming into a ball in the metric space in which the optimal policy lies. The regret of the zooming algorithm is of order $O(\log T \cdot T^{\frac{K}{K+1}})$ for the K-dimensional Euclidean space, provided the rewards are Lipschitz [11].

2.3.2 Censored MAB

Abernethy et al. [6] addressed a variant of MAB where feedback from each arm may be censored according to an unknown threshold and therefore not observed by the learner.

In this setting, each arm has an underlying unknown but constant threshold. When the arm is sampled, the reward is either 1 or 0, depending on whether the sample surpasses the threshold or not. We refer to this type of censoring as "self-censored" since the censoring threshold are fixed and are independent of the specific policy of the learner.

Abbernethy et al. discuss two variants for this setting —one with feedback on the sample and one with feedback only on samples that surpass the threshold. The authors propose an algorithm to address each variant: the Dvoretzky-Kiefer-Wolfowitz Inequality based Upper Confidence Bound (DKWUCB) algorithm for the variant with feedback and the Kaplan-Meier based Upper Confidence Bound (KMUCB) algorithm for the variant without feedback. Both algorithms are shown to have a regret that is of order $O(\log T)$ [6].

2.3.3 Budget-Constraint MAB

Zhou and Tomlin tackled a budget-constraint variant of MAB, where multiple plays are allowed at each time-step [12]. Their approach focuses on the combinatorial aspects of choosing which arms to play, resulting in a time horizon-independent regret.

The setting proposed by Zhou and Tomlin suggests a fixed budget $0 < B \in \mathbb{R}_+$ to be allocated across the entire time horizon. Moreover, the time horizon T depends on the specific allocation of the budget and the algorithm terminates when the budget runs out. Consequently, while we can draw inspiration from the algorithm suggest in [12], we believe

its applicability to an intelligence-collection setting is limited, as time is a critical factor in our setting.

We note that the algorithm, called Upper Confidence Bound for Multi-Play with Budget (UCBMB) algorithm, is a budget-aware version of UCB that plays a fixed-number L of arms with the highest UCB indices at each iteration. The regret performance of UCBMB is shown to be of order $O(KL^4 \log B)$.

2.4 Intelligence-collection Using MAB

We end this chapter by briefly discussing how the different MAB variants (summarized in table 2.1) can contribute to our model. The MAB variants we explored introduce new elements introduced to classical MAB: continuous decision-space, censoring of information and budget-constraints. We wish to incorporate all of these elements in some way into our model.

As noted in Chapter 1, we aim to assist decision makers in resource allocation when the intelligence collection plan is created. As such, incorporating a budget-constraint is desirable. However, we shall note that our model should still be limited by a time horizon, as intelligence collection is a time-critical procedure.

The continuous decision-space (Continuum MAB), as opposed to the discrete decision-space used in classical MAB, better reflects trade-offs a decision maker faces where multiple option are present. In other words, there is a continuum of intelligence collection opportunities, rather than a discrete set of opportunities.

Importantly, we can use censoring of rewards from the decision maker to represent an interaction between Red and Blue actions. That is, Blue needs to invest resources in order to stand a chance to receive a reward. The randomness of each intelligence source (arm) in the classical MAB captures the variety of information Red keeps in each individual source. The censoring allows us to model potential obscurity of information Blue can receive from a given source, if they do not allocate sufficient resources for exploring that source.

By combining these three elements into our proposed model, we will be able to better express the complexities of resource-allocation for intelligence collection. From an algorithmic

Table 2.1. Summary of MAB Algorithms

Problem Property	Algorithm	Assumptions	Performance
Classical MAB	UCB [5]	None	$O(\log T)$
Continuum MAB	Kleinberg's Algorithm [10]	$K = 2$	$O(T^{2/3})$
Continuum MAB	Zooming Algorithm [11]	Lipschitz rewards	$O(\log T \cdot T^{\frac{K}{K+1}})$
Continuum MAB	FKM Algorithm [7]	None	$O(T^{3/4})$
Censored with Feedback	DKWUCB Algorithm [6]	Self-censored	$O(\log T)$
Censored without Feedback	KMUCB Algorithm [6]	Self-censored	$O(\log T)$
Budget-Dependent Horizon	UCBMB [12]	Time-independent	$O(KL^4 \log B)$

perspective, to the best of our knowledge, addressing such a model with all of the above elements has not been attempted before.

2.4.1 Additional Related Work

While the elements introduced in our model are novel, we acknowledge that an MAB or Online Optimization-inspired setting for intelligence collection has been previously devised.

We specifically pay attention to the works in [3] and [4]. The work in [3] is specifically tailored for intelligence collection in the cyber domain and incorporate domain-specific elements into the model, such as the network structure from which information is extracted. In [4], the authors specifically address the process of intelligence collection including extraction of information, processing and analysis. As described in Chapter 3, we abstract this chain into a single step in our model that captures the collection, processing and analysis (evaluation) of information. We further note that the models presented in these work make some specific assumptions regarding the distributions involved. We will refrain from making any specific assumptions regarding any distributions.

CHAPTER 3:
Proposed Model and Algorithm

In this chapter, we present a mathematical formulation of our model and discuss the implications of introducing multiple mechanics into our model.

3.1 Model Formulation

A decision maker is provided with a set of $K \in \mathbb{N}$ intelligence sources and a time horizon $T \in \mathbb{N}$. During each time step $t = 1, 2, ..., T$, each intelligence source $k \in \{1, ..., K\}$ can provide a piece of information of some value. We model the value of that information using a random variable X_k. For all $k \in [K]$, X_k are independent random variables with support $\mathcal{I} \subset \mathbb{R}$. We also refer to $\mathcal{I}$ as the *information space*.

To collect information from a given source, the decision maker must pay a cost. For each $k \in [K]$, we assume the existence of an *information-cost function*, $C_k : \mathcal{I} \to \mathbb{R}_+$. Paying a cost $C_k(x)$ guarantees that all information with value at most $x \in \mathcal{I}$ will be accessible to the decision maker from source k. Since the value of information is random, if the value generated from source k is greater than x, the decision maker will not be able to observe it due to insufficient resources. In other words, the value observed by the decision maker becomes 0.

The decision maker is limited by a *budget*, $B \in \mathbb{R}_+$ for each time step $t = 1, ..., T$. The budget can be allocated freely between any number of intelligence sources, and any portion of the budget not spent in a specific time step is lost.

In this model, we use the *expected observable value* of source. It is the expected value of information up to a certain specified value within $\mathcal{I}$, referred to as the information threshold. The expected observable value expresses the expected value accessible to the decision maker. Formally, the expected observable value from a source $k \in [K]$ at time t, given an information value threshold $x \in \mathcal{I}$, is expressed through

$$\mathbb{E}(X_k; X_k \leq x) = \int_{z \in \mathcal{I}} z f_k(z) I(z \leq x) dz, \tag{3.1}$$

where f_k is the probability density function of intelligence source k at all times $t = 1, ..., T$, and I is an indicator function. The goal of the decision maker is to maximize the total expected observable value of all sources given the budget B in each time step $t = 1, ..., T$,

$$\max_{\mathbf{x} \in \mathcal{I}^K} \sum_{k=1}^{K} \mathbb{E}(X_k; X_k \leq x_k) \quad \text{subject to} \sum_{k=1}^{K} C_k(x_k) \leq B, \tag{3.2}$$

where x_k denotes the k^{th} coordinate of a $K-$dimensional vector $\mathbf{x} \in \mathcal{I}^k$.

3.1.1 Connection to the Online Optimization and MAB Frameworks

Equation (3.1) describes the goal of the decision maker at each time step $t \in \{1, 2, ..., T\}$. In practice, while the functions C_k are known to the decision maker, the distribution of each X_k (and consequently, the structure of the function $\mathbb{E}(X_k; X_k \leq x)$) is not available.

We can recast the formulation as an online optimization problem. At each time step t, the decision maker decides on an allocation action $\mathbf{x}_t \in \mathcal{I}^K$, where $\mathbf{x}_t$ is a K-dimensional vector and $\mathcal{I}^K$ denotes the K-dimensional space whose coordinates all lie in $\mathcal{I}$. The learner then receives feedback of the form $f_t(\mathbf{x}_t) = \sum_{k=1}^{K} X_{k,t} I(X_{k,t} < x_{k,t})$, where $x_{k,t}$ is a sample of the value of information of source k at time step t. Observe that, by definition, we have $\mathbb{E}(f_t(\mathbf{x}_t)) = \sum_{k=1}^{K} \mathbb{E}(X_k; X_k \leq x_k)$, which is exactly our objective function from equation (3.1). Maximizing the expected value of f_t is there equivalent to minimizing the regret of the online learning problem induced by f_t, as defined by equation (2.1). For completeness, note that the set of feasible policies $\mathcal{S}$ is given by

$$\mathcal{S} = \{\mathbf{x} \in \mathcal{I}^K | \sum_{k=1}^{K} C_k(x_k) \leq B\}. \tag{3.3}$$

The shape and properties of the set $\mathcal{S}$ depends on the specific information-cost function, as discussed below. We note that in most cases, however, it is a set defined within the

continuous domain. Consequently, when casting the formulation in equation (3.1) to the MAB framework, we must use a continuous decision space, such as the one described as Continuum MAB in Chapter 2.

The MAB interpretation of the formulation is straightforward, otherwise: in each time step, a decision maker allocates a budget B to a set of K arms based on past observations from each arm. If the cost paid for a certain arm is below its produced value, that is, $I(X_{k,t} \leq x_{k,t}) = 0$, the value is never observed by the decision maker.

3.2 The Information-cost Function

The information-cost function is assumed to be non-decreasing, that is, the decision maker must pay more in order to observe higher values of information. If the information-cost function is invertible within $\mathcal{I}$ we can transform the optimization problem in equation (3.2) from an information-space problem to a cost-space problem:

$$\max_{\mathbf{x}\in\mathbb{R}^K} \sum_{k=1}^{K} \mathbb{E}(X_k; X_k \leq C_k^{-1}(x_k)) \quad \text{subject to} \sum_{k=1}^{K} x_k \leq B \tag{3.4}$$

One potential advantage of this cost-space formulation is that the constraint is linear in the decision variables. In fact, by denoting $\mathbf{y} = \frac{1}{B}\mathbf{x}$, we can rewrite the formulation as an optimization problem over the K-dimensional simplex Δ^K:

$$\max_{\mathbf{y}\in\Delta^K} \sum_{k=1}^{K} \mathbb{E}(X_k; X_k \leq C_k^{-1}(By_k)), \tag{3.5}$$

where y_k is the k^{th} coordinate of the vector $\mathbf{y}$. Moreover, we are specifically interested in information-cost affine functions. If we let $C_k(z) = a_k z + b_k$, then we can recast the inequality $X_k \leq C_k^{-1}(By_k)$ as $X_k \leq (By_k - b_k)/a_k$, and we can once more rewrite the formulation as

$$\max_{\mathbf{y}\in\Delta^K} \sum_{k=1}^{K} \Big[\frac{B}{a_k}\mathbb{E}(Y_k; Y_k \leq y_k) - \frac{b_k}{B}\Big] \tag{3.6}$$

where $Y_k = (a_k X_k + b_k)/B$. The above formulation benefits from a simple constraint set (the decision space is the $K-$dimensional simplex) and a relatively simple form for the objective function that depends directly on the censored mean function of the random variables $\{Y_k\}_{k=1}^K$.

3.2.1 Assumptions In This Work

Unless noted otherwise, we shall hereafter assume that $\mathcal{I} = [0, 1]$, $B = 1$ and $C_k(x) = x$ for all $k \in [K]$. These assumptions will allow us to better analyze the properties and algorithms of the model, and it is standard that they generalize the problem for any bounded interval. The problem, given these assumption, is

$$\max_{\mathbf{x}\in\Delta^K} \sum_{k=1}^{K} \mathbb{E}(X_k; X_k \leq x_k). \tag{3.7}$$

We note that since the information-cost function is the identity function, we can say that the decision maker operates within the cost space or the information value space. We will therefore refer hereafter to x_k as the cost paid to gather intelligence from source k.

3.3 Estimating the Objective Function

A unique challenge for this model is that the decision maker can make a decision resulting in no new information at all. As a result, estimating the censored mean of each intelligence source does not only require the decision maker to allocate resources to spend on a certain source, but also requires the decision maker to allocate *sufficient* resources. This section discusses how we can estimate the censored mean of a source based on the policies employed by the decision maker.

To simplify our notation, we will consider how to estimate the expected observable value of a single source whose value is a random variable X with support on $[0, 1]$. We consider a series of n collection attempts, designed in two fashions: one uses the entirety of the budget towards a singular source, the other splits the budget between two sources. These two types of policies shed light about the limits of different sampling strategies, and can be generalized to address an arbitrary number of sources to focus upon.

3.3.1 Full-cost Policy Estimator

Consider this case: we use the entire budget on a singular intelligence source throughout all n attempts. Since the budget $B = 1 \geq \max_{v \in I} v$, all n attempts are uncensored and completely observable by the decision maker. To simplify our notation, for $x \in [0, 1]$ denote

$$\mu(x) = \mathbb{E}(X; X \leq x). \tag{3.8}$$

$\mu(x)$ is the *true* expected observable value for some threshold $x \in [0, 1]$. Using the set of samples $\{x_i\}_{i=1}^n$, we can construct an estimator for $\mu(x)$ via

$$\hat{\mu}(x; n) = \frac{1}{n} \sum_{i=1}^{n} x_i I(x_i \leq x). \tag{3.9}$$

We refer to $\hat{\mu}(x)$ as an empirical censored mean function at point $x \in [0, 1]$. Note the indicator expression $I(x_i \leq x)$ nullifies the i^{th} sample if its value is more than the parameter x. If $x = 0$, $\hat{\mu}(x; n) = 0$, whereas if $x = 1$, $\hat{\mu}(x; n)$ is simply the sample mean of all uncensored samples.

Since we do not have access to the censored mean function directly in our setting, the empirical censored mean function is critical to our ability to estimate the objective function. The challenge in evaluating $\hat{\mu}(x)$ is that samples of X might be censored if the decision maker does not allocate enough resources to observe them. Consequently, even if we attempt to sample a certain source n times, we may end up with fewer usable samples.

How good is this proposed approximation of the censored mean function? The following lemma addresses this exact question.

Lemma 3.3.1 *For any $\delta > 0$, given n IID* uncensored *samples of a source:*

$$P(\sup_{x \in [0,1]} |\hat{\mu}(x; n) - \mu(x)| > \delta) \leq 4 \cdot e^{-2n\delta^2}. \tag{3.10}$$

The proof of this lemma is given in Appendix A.1. The above results implies that the error for a given threshold decays exponentially with the number of samples. Also, the error

threshold is controlled by an observer and can be independent of the distribution of the intelligence sources. This strength of the proposed estimator also has an inherit weakness - lemma 3.3.1 requires n uncensored samples from a singular source. If there are K different sources, we need to collect Kn uncensored samples in order to achieve the same level of error for all of them. In other words, the cost is linear in the number of intelligence sources.

3.3.2 Twin-cost Policy Estimator

In the previous section, the simplicity of the full-cost policy estimator was apparent due to the bypassing of the censoring altogether. In this section, we wish to expand the full-cost policy estimator.

A natural expansion to the full-cost policy estimator is dubbed the *twin-cost policy estimator*. We derive this estimator for the case where $K = 2$. Here, we pay a fixed cost $0 < \beta < 1$ for one source, and $1 - \beta$ for the other source. The twin-cost policy estimator allows the learner to sample the simplex space Δ^2 at the 2-dimensional points $(\beta, 1-\beta)^T$ and $(1-\beta, \beta)^T$.

For a specific source, assume the learner pays β exactly n_β times and pays $1 - \beta$ exactly $n_{1-\beta}$ times. Without loss of generality, assume that $\beta < 1/2 < 1 - \beta$. In order to estimate the expected observable value of the random variable X at some point $x \in [0, 1]$, we have to consider the exact point of evaluation. If $x > 1 - \beta$, then x is beyond any observable sample and we cannot tell anything about the expected observable value. Conversely, if $x \leq \beta$, then all *uncensored* samples from amongst the $n_\beta + n_{1-\beta}$ collection attempts can be used to estimate the expected observable value in a similar manner to the one presented in subsection 3.3.1.

The case where $x \in (\beta, 1 - \beta]$ is the most complex. We must carefully consider how to use uncensored samples collected. To do so, observe that for some $x \in (\beta, 1 - \beta]$ we have

$$\mu(x) = \int_0^x z f_k(z) dz = \int_0^\beta z f_k(z) dz + \int_\beta^x z f_k(z) dz = \mu(\beta) + \underbrace{\int_\beta^x z f_k(z) dz}_{\text{residual}}. \quad (3.11)$$

Since all samples equal to or less than β than uncensored, we can use all of the $n_\beta + n_{1-\beta}$ samples to estimate $\mu(\beta)$, then use the remaining uncensored samples to estimate the residual.

To formalize this observation, let $\mathcal{U}_\beta$ and $\mathcal{U}_{1-\beta}$ denote the subset of uncensored samples for each cost. We denote the proposed estimator via $\hat{\mu}^{(\beta)}(x; \beta, n_\beta, n_{1-\beta})$,

$$\hat{\mu}^{(\beta)}(x; \beta, n_\beta, n_{1-\beta}) = \begin{cases} \hat{\mu}(x; n_\beta + n_{1-\beta}) & x \leq \beta \\ \hat{\mu}(\beta; n_\beta + n_{1-\beta}) + \frac{1}{n_{1-\beta}} \sum_{z \in \mathcal{U}_{1-\beta}} z I(z \in (\beta, x]) & \beta < x \leq 1 - \beta \\ 0 & x > 1 - \beta. \end{cases} \tag{3.12}$$

The accuracy of this estimator highly depends on the choice of β and the cumulative distribution function of X. We shall not provide an analytical analysis of the estimator performance, but we will explore the estimator numerically in Chapter 4.

3.3.3 Beyond Full-cost and Twin-cost Policy Estimators

The strength of both the full-cost policy estimator and the twin-cost policy estimator stems from their simplicity. However, we note that more complex and involved methods exist, such as control variates estimators [13] or the Kaplan-Meir survival analysis method [14]. We leave these estimators for future work on the subject.

3.4 From Model to Algorithm

Equation (3.7) defines the optimization problem we wish to solve using online optimization and MAB techniques. In this section, we will explore simple proposed algorithms to address this optimization problem. Some algorithms in this section use an estimator function, as discussed in section 3.3. We denote the estimator at time step $t \in \{1, ..., T\}$ by $\mathcal{F}_t(\mathbf{x})$. The performance of the algorithm greatly depends on the performance of $\mathcal{F}_t(\mathbf{x})$, as we shall discuss in this section.

3.4.1 The $(K+1)$-UCB Algorithm

We begin by introducing the $(K+1)$-UCB algorithm. Inspired by the original UCB (see Algorithm 1), this algorithm exploits the fact that $B = \max_{v \in \mathcal{I}} v = 1$, by sampling the K corners of the K-dimensional simplex, denoted by e_k for $k \in [K]$. In other words, the exploration rounds spend entire budget is allocated to collect information from a single source at each time step. Importantly, since $B = 1$, no sample in this approach is ever censored, thereby nullifying the censoring effect.

The exploitation rounds make use of an "extra" sampling point (hence, the +1 in the algorithm's name) that uses all samples collected so far and an estimator $\mathcal{F}_t(x)$ to estimate the maximal value of the objective function on the K-dimensional simplex. Since no samples are censored, the estimator $\mathcal{F}_t(\mathbf{x})$ is empirical censored mean function introduced in subsection 3.3.1,

$$\mathcal{F}_t(x) = \sum_{k=1}^{K} \hat{\mu}_{k,t}(x; N_{k,t}), \tag{3.13}$$

where $N_{k,t}$ is the number of samples collected from source $k \in [K]$ up until time t, and $\hat{\mu}_{k,t}(x; N_{k,t})$ is the empirical censored mean of all samples collected from source $k \in [K]$ up until time t, as defined via equation (3.9). The algorithm is shown below as Algorithm 4. The total value of information collected at time step t is given by $\sum_{k=1}^{K} X_{k,t} I(X_{k,t} \leq x_k)$. Note that $X_{k,t} I(X_{k,t} \leq x_k)$ can be zero if the value of information is higher than x_k. However, during exploration, since $B = 1$ and $x_k \neq 0$ for exactly one intelligence source k, no censoring can occur.

The $(K+1)$-UCB is a simple and effective approach as it eliminates complex elements introduced in our model. However, this approach is not very effective from a practical standpoint, since allocating the entire budget on a single source at time step is a potentially wasteful policy. Nevertheless, this algorithm is very attractive to introduce because it serves

as a baseline, and its performance is analytically traceable.

Algorithm 4: $(K+1)$-UCB

Input: $K \in \mathbb{N}_+$ (number of sources)
$T \in \mathbb{N}_+$ (time-horizon)
$B = 1$ (budget)
$\xi \in \mathbb{R}_+$

Output: $\pi_{(K+1)-UCB} = \{\mathbf{x}_t\}_{t=1}^{T}$ ($(K+1)$-UCB policy)

$\forall k = 1, ..., K : UCB_{k,0} \leftarrow \infty, N_{k,0} \leftarrow 0, \hat{\mu}_{k,0} \leftarrow 0$

$UCB_{(K+1),0} \leftarrow 0$

for *t = 1, 2, ..., T* **do**

$k_t \leftarrow \arg\max_{k=1,..,K,K+1} UCB_{k,t-1}$ (choose source to explore or mixture to exploit)

$$\mathbf{x}_t \leftarrow \begin{cases} e_{k_t} & k_t \neq K+1 \\ \arg\max_{x \in \Delta^K} \mathcal{F}_t(x) & k_t = K+1 \end{cases}$$

$y_t \leftarrow \sum_{k=1}^{K} X_{k,t} I(X_{k,t} \leq x_k)$ (play arm i_t)

if $k_t \neq K+1$ **then**

$N_{k_t,t} \leftarrow N_{k_t,t-1} + 1$

$\hat{\mu}_{k_t,t} \leftarrow (\hat{\mu}_{k_t,t-1} + y_t)/N_{k_t,t-1}$

$UCB_{k,t} \leftarrow \hat{\mu}_{k_t,t} + \sqrt{\xi t^{2/3}/N_{k_t,t}}$ (update UCB index according to the index type)

end

$UCB_{(K+1)} \leftarrow \max_{\mathbf{x} \in \Delta^K} \mathcal{F}_t(\mathbf{x})$ (optimize the estimator)

end

Algorithm 4 balances between exploration and exploitation differently than the original UCB algorithm (Algorithm 1). The algorithm performs exploration steps by sampling the K corner points of the K-dimensional simplex Δ^K, each corresponding to a full exploration of a different intelligence source.

Using the information collected through exploration, the algorithm uses the estimator (3.13) to find a potential exploitation point. The exploitation point divides the budget B between the intelligence sources, in a manner that is deemed optimal given the current information. Consequently, the point of exploitation is in fact a mixture of collection attempts from different sources, and is not limited to just a singular source. The next section details how to uncover this exploitation point.

3.4.2 Optimizing the Estimator

As noted in Algorithm 4, a critical step in the implementation requires an optimization of the estimator $\mathcal{F}_t(\mathbf{x})$. In this subsection, we discuss several techniques that can be used to implement this step. We draw inspiration from techniques discussed in Chapter 2.

First, note that if $K = 2$, we can use an exhaustive search to find the optimal value of $x \in [0, 1]$ such that $\mathcal{F}_t(\mathbf{x})$ is maximized. We note that, while accurate, this approach is not the most efficient. In fact, for $K > 2$, this approach becomes infeasible in terms of computational power required.

Our go-to approach is to therefore employ the FKM algorithm (Algorithm 2). We use our estimator of the objective function $\mathcal{F}_t(\mathbf{x})$ to perform a few offline gradient steps, typically 5. We start our search from the center of the unit simplex, whose coordinates are all $1/K$. Following each gradient ascent step, we project the resultant vector onto the unit simplex using the method described in [8].

3.4.3 $(K + 1)$-UCB Performance

In this section, we discuss the performance of the $(K+1)$-UCB algorithm. For our analysis, we use a similar technique to the one presented in [6]. The main result is stated below in corollary 3.4.1:

Lemma 3.4.1 *The regret $r_T(\pi_{(K+1)-UCB})$ resulting from Algorithm 4 using a perfect oracle to solve $\max_{\mathbf{x}\in\Delta^K} \mathcal{F}_t(\mathbf{x})$ is of order $O(T^{2/3}\sqrt{\log T})$.*

In what follows, we argue why the above result holds.

Notations

We begin by introducing several notations used throughout this subsection. First, the notation $\hat{\mu}_{k,t}$ from Algorithm 4 satisfies,

$$\hat{\mu}_{k,t} = \hat{\mu}_{k,t}(1; N_{k,t}), \tag{3.14}$$

and we shall use them interchangeably in this section. We also omit the indication of $N_{k,t}$ throughout.

Denote by $\mathbf{x}^* \in \Delta^K$ the optimal solution of the formulation in equation (3.7), and let $\hat{\mathbf{x}}_t$ be the optimal solution of the estimator at time t. We also denote the j^{th} coordinate of $\hat{\mathbf{x}}_t$ by $\hat{x}_{j,t}$.

For $k \in [K]$, we let ϵ be the quantity

$$\epsilon_k = \sum_{j=1}^{K} \mu_j(\mathbf{x}_j^*) - \mu_k(1), \tag{3.15}$$

that is, ϵ_k is the difference between the value of the estimator at the optimal solution and the k^{th} corner point of Δ^K (called e_k).

We also let $\delta_{k,t}$ be a time-dependent per-arm bound on the estimation error. We use $\delta_{k,t}$ to simplify our notation when applying lemma 3.3.1.

Preliminary Analysis - Bounding $N_{k,t}$

The regret is accounted for by the number of times an arm $k \in [K]$ is played is given, $N_{k,t}$. We therefore focus our attention on deriving an upper bound for $N_{k,T}$ by a function of T. In that follows we will derive a bound on $N_{k,t}$ under certain probabilistic conditions.

The algorithm plays some arm $k \in [K]$ at time step t if and only if:

$$\begin{aligned} &(1) \quad \forall j \neq k : UCB_{k,t} > UCB_{j,t} \\ &(2) \quad UCB_{k,t} > \max_{\mathbf{x} \in \Delta^K} \sum_{k=1}^{K} \hat{\mu}_{k,t}(x_k). \end{aligned} \tag{3.16}$$

Consider the case where the respective arm empirical means are close to the true arm means, that is,

$$\forall k \in [K], \forall x \in [0,1] : |\hat{\mu}_{k,t}(x) - \mu_k(x)| < \delta_k(t). \tag{3.17}$$

Using the inequalities (3.16), we have in this case

$$\hat{\mu}_{k,t}(1) + \sqrt{\xi t^{2/3}/N_{k,t}} \geq \sum_{j=1}^{K} \hat{\mu}_{j,t}(\hat{\mathbf{x}}_{j,t}) \geq \sum_{j=1}^{K} \hat{\mu}_{j,t}(\mathbf{x}_j^*), \tag{3.18}$$

where we used the fact that the maximizer of the right-hand side in equality (2) from 3.16 is at least as good a candidate to optimize the estimator of the objective function as the *true* maximizer of the actual objective function. We can further reduce this inequality to

$$\mu_{k,t}(1) + \delta_{k,t} + \sqrt{\xi t^{2/3}/N_{k,t}} \geq \sum_{j=1}^{K} [\mu_j(\mathbf{x}_j^*) - \delta_{j,t}], \tag{3.19}$$

or, simply,

$$\delta_{k,t} + \sum_{j=1}^{K} \delta_{j,t} + \sqrt{\xi t^{2/3}/N_{k,t}} \geq \sum_{j=1}^{K} \mu_j(\mathbf{x}_j^*) - \mu_{k,t}(1) = \epsilon_k \tag{3.20}$$

Observe that the right-hand side of inequality (3.20) is non-negative, since it is the difference between the maximizer of the estimator at $\mathbf{x}^*$ and the empirical mean of the k^{th} corner point.

Set $\delta_{k,t} = \sqrt{\xi t^{2/3}/N_{k,t}}$ for all $k \in [K]$. The right-hand side of the above inequality is non-negative, whereas the left-hand side decays as $t \to \infty$. This results in the following inequality:

$$3\sqrt{\xi t^{2/3}/N_{k,t}} + \sum_{j=1, j\neq k}^{K} \sqrt{\xi t^{2/3}/N_{j,t}} \geq \epsilon_k. \tag{3.21}$$

Inequality (3.21) holds with the stated probability not only for a singular source k, but all for all sources $k \in [K]$. We thus have a system of linear inequalities.

As $t \to \infty$, these inequalities become equations since the left-hand side decays while the right-hand side is constant non-negative. Let $\mathbf{g}(t) \in \mathbb{R}^K$ be a vector whose k^{th} coordinate

is $\sqrt{\xi t^{2/3}/N_{k,t}}$, $\mathbf{e} \in \mathbb{R}^K$ be a vector whose coordinates are ϵ_k, and consider the matrix $A_{K,t} \in \mathbb{R}^{K\times K}$,

$$A_{K,t} = \begin{pmatrix} 3 & 1 & \dots & 1 \\ 1 & 3 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \dots & 3 \end{pmatrix}. \tag{3.22}$$

The system derived from (3.21) can be recast as $A_{K,t}\mathbf{g}(t) = \mathbf{e}$. The matrix $A_{K,t}$ is invertible, and its inverse structure is given by

$$(A_{K,t}^{-1})_{i,j} = \begin{cases} \frac{-1}{2(K+2)} & i \neq j \\ \frac{K+1}{2(K+2)} & i = j \end{cases}, \tag{3.23}$$

implying its structure is only dependent on K. We can solve directly to find that

$$\sqrt{\xi t^{2/3}/N_{k,t}} = \frac{1}{2}\epsilon_k - \frac{1}{2(K+2)}\sum_{j=1}^{K}\epsilon_k. \tag{3.24}$$

We can express $N_{k,t}$ as a function of t at the time of equality, allowing us to have a bound on the number of times a specific source is sampled individually:

$$N_{k,t} \leq \underbrace{\frac{2\xi}{(\epsilon_k - \frac{1}{K+2}\sum_{j=1}^{K}\epsilon_k)^2}}_{\rho_k} t^{2/3}, \tag{3.25}$$

where ρ_k is a problem dependent constant, implying $N_{k,t} = O(t^{2/3})$ for all $k \in [K]$. In other words, $N_{k,T} = O(T^{2/3})$.

Regret Analysis for a Corner Point

We now proceed to bound the regret resulting from playing the corner points of Δ^K. Using inequality (3.25), we can bound the regret in the case where $|\hat{\mu}_{k,t}(x) - \mu_k(x)| \leq \delta_{k,t}$ for all $k \in [K]$.

To use this bound, we write the total regret from sampling a corner point k as decomposition of three cases:

- The case where $|\hat{\mu}_{j,t}(1) - \mu_j(1)| \leq \delta_{j,t}$ for all $j \in [K]$.
- The case where $|\hat{\mu}_{k,t}(1) - \mu_k(1)| > \delta_{k,t}$ and $N_{k,t} > \rho_k t^{2/3}$
- The case where $|\hat{\mu}_{k,t}(1) - \mu_k(1)| > \delta_{k,t}$ and $N_{k,t} \leq \rho_k t^{2/3}$

Formally, the total regret induced by playing *any* of the corner points is bounded by

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{k=1}^{K} I(\text{arm } k \text{ is selected at time step } t, \bigcap_{j\in[K]} |\hat{\mu}_{j,t}(1) - \mu_j(1)| \leq \delta_{k,t}) + \\
&\sum_{t=1}^{T}\sum_{k=1}^{K} \mathbb{E}(\text{arm } k \text{ is selected at time step } t, |\hat{\mu}_{k,t}(1) - \mu_k(1)| > \delta_{k,t}, N_{k,t} > \rho_k t^{2/3}) + \\
&\sum_{t=1}^{T}\sum_{k=1}^{K} \mathbb{E}(\text{arm } k \text{ is selected at time step } t, |\hat{\mu}_{k,t}(1) - \mu_k(1)| > \delta_{k,t}, N_{k,t} \leq \rho_k t^{2/3}).
\end{aligned} \tag{3.26}$$

From the preliminary analysis above, we know that the first term in equation (3.26) corresponds to the case where for all $k \in [K]$, $N_{k,t} \leq \rho_k t^{2/3}$. Therefore, the regret in this case

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{k=1}^{K} \mathbb{E}(\text{arm } k \text{ is selected at time step } t, \bigcap_{j\in[K]} |\hat{\mu}_{j,t}(1) - \mu_j(1)| \leq \delta_{k,t}) \leq \\
&\sum_{t=1}^{T}\sum_{k=1}^{K} I(N_{k,t} \leq \rho_k t^{2/3}) \leq \sum_{k=1}^{T} \rho_k T^{2/3}.
\end{aligned} \tag{3.27}$$

This reasoning also stands for the third term in equation (3.26), since $N_{k,t} \leq \rho_k t^{2/3}$. For the

second term, where $N_{k,t} > \rho_k t^{2/3}$, the regret is bounded by

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathbb{E}(\text{arm } k \text{ is selected at time step } t, |\hat{\mu}_{k,t}(1) - \mu_k(1)| > \delta_{k,t}, N_{k,t} > \rho_k t^{2/3}) \leq$$
$$\sum_{t=1}^{T}\sum_{k=1}^{K} P\Big(sup_{x\in[0,1]}|\hat{\mu}_{k,t}(x) - \mu_k(x)| \geq \delta_{k,t}, N_{k,t} > \rho_k t^{2/3}\Big) \underbrace{\leq}_{\text{lemma 3.3.1}}$$
$$\sum_{t=1}^{T}\sum_{k=1}^{K} 4\cdot\exp(-2N_{k,t}\delta_{k,t}^2) \underbrace{=}_{\text{replace } \delta_{k,t}} \sum_{t=1}^{T}\sum_{k=1}^{K} 4\cdot\exp(-2\xi t^{2/3}) \leq$$
$$4K\exp(-2\xi)\int_0^{\infty} e^{-t^{2/3}} \leq 4K\exp(-2\xi)\cdot\frac{3\sqrt{\pi}}{4} = 3K\sqrt{\pi}\exp(-2\xi), \tag{3.28}$$

where we used numerical integration to evaluate the last step. We can see that this error is constant and is not dependent on T. Combining this bound with the other two terms, results in equation 3.26 implying regret of order $O(T^{2/3})$.

Regret Analysis for the Oracle Estimator

Now we consider the error resulting from sampling the point yielded by the estimator as the optimal point. We assume that the estimator is optimized using a perfect Oracle. At time step t, the error from playing the extra arm is given by

$$|\sum_{k=1}^{K}\hat{\mu}_{k,t}(\hat{x}_{k,t}) - \sum_{k=1}^{K}\mu_k(\mathbf{x}_k^*)| \tag{3.29}$$

Assume that, at time step t, for all $k \in [K]$, we have $\sup_{x\in[0,1]}|\hat{\mu}_{k,t}(x) - \mu_k(x)| < \eta_{k,t}$. Both $\hat{\mu}_k(x)$ and $\mu_k(x)$ are increasing functions of x, so we can bound the error by

$$
\begin{aligned}
&|\sum_{k=1}^{K} \hat{\mu}_{k,t}(\hat{x}_{k,t}) - \sum_{k=1}^{K} \mu_k(\mathbf{x}_k^*)| \leq \sum_{k=1}^{K} |\hat{\mu}_{k,t}(\hat{x}_{k,t}) - \mu_k(\mathbf{x}_k^*)| \leq \\
&\sum_{k=1}^{K} |\hat{\mu}_{k,t}(1) - \mu_k(\mathbf{x}_k^*)| \leq \sum_{k=1}^{K} |\hat{\mu}_{k,t}(1) - \mu_k(1)| \leq \\
&\sum_{k=1}^{K} \sup_{x\in[0,1]} |\hat{\mu}_{k,t}(x) - \mu_k(x)| \leq \sum_{k=1}^{K} \eta_{k,t}.
\end{aligned}
\tag{3.30}
$$

Once again, since this bound is only valid on specific time steps, we decompose the regret into three cases:

- The case where $\sup_{x\in[0,1]} |\hat{\mu}_{k,t}(x) - \mu_k(x)| < \eta_{k,t}$ for all $k \in [K]$.
- The case where $\sup_{x\in[0,1]} |\hat{\mu}_{k,t}(x) - \mu_k(x)| < \eta_{k,t}$ for all $k \in [K]$ and $N_{k,t} > \rho_k t^{2/3}$
- The case where $\sup_{x\in[0,1]} |\hat{\mu}_{k,t}(x) - \mu_k(x)| < \eta_{k,t}$ for all $k \in [K]$ and $N_{k,t} \leq \rho_k t^{2/3}$

In the first case, the bound at time step t is given by (3.30). If we set $\eta_{k,t} = \lambda_k t^{-1/3}\sqrt{\log t}$, then by summing over all time steps, $t = 1, ..., T$, we have

$$
\begin{aligned}
&\sum_{t=1}^{T}\sum_{k=1}^{K} \eta_{k,t} = \sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k t^{-1/3}\sqrt{\log t} \leq \\
&\sum_{k=1}^{K} \lambda_k \sqrt{\log T} \int_1^T t^{-1/3} dt = O(T^{2/3}\sqrt{\log T}).
\end{aligned}
\tag{3.31}
$$

For the third case, using the same reasoning as when analyzing the regret for a corner point, the bound is given by

$$
\sum_{k=1}^{K} \rho_k T^{2/3} = O(T^{2/3})..
\tag{3.32}
$$

In the second case, suppose we choose $\eta_{k,t} = \lambda_k t^{-1/3}\sqrt{\log t}$. Then the bound is

$$\sum_{t=1}^{T}\sum_{k=1}^{K} P\Big(\sup_{x\in[0,1]} |\hat{\mu}_{k,t}(x) - \mu_k(x)| > \eta_{k,t}, N_{k,t} > \rho_k t^{2/3}\Big) \leq$$
$$4\sum_{t=1}^{T}\sum_{k=1}^{K} e^{-2\lambda_k^2\rho_k \frac{t^{2/3}\log t}{t^{2/3}}} = 4\sum_{t=1}^{T}\sum_{k=1}^{K} t^{-2\rho_k\lambda_k^2}. \tag{3.33}$$

Setting λ_k such that $2\rho_k\lambda_k^2 = \frac{1}{3}$, results in a bound of the form $\sum_{t=1}^{T}\sum_{k=1}^{K} t^{-1/3} = O(T^{2/3})$.

We conclude that the oracle error is of order $O(T^{2/3}\sqrt{\log T}) + O(T^{2/3}) + O(T^{2/3}) = O(T^{2/3}\sqrt{\log T})$. Combining this result with our analysis of regret for a corner point, we end up with the results stated in 3.4.1.

Regret Analysis - Beyond the Oracle Case

As mentioned above, our argument is based on the notion that the maximizer of the estimator has 0 error due to the existence of an oracle estimator. In practice, if $K = 2$, then an exhaustive search approach of Δ^K (which is reduced to the segment $[0, 1]$ in this case) is plausible, leading to the following result:

Lemma 3.4.2 *For $K = 2$, regret $r_T(\pi_{(K+1)-UCB})$ resulting from Algorithm 4 using exhaustive search in the estimator optimization step is of order $O(T^{2/3} \cdot \sqrt{\log T})$.*

For $K > 2$, however, the exhaustive search approach is not feasible, and we instead employ the use of FKM algorithm to maximize our estimator. Note that for $K = 2$, an *optimal* method - proposed by Kleinberg in [10] - yields better performance than the $(K + 1)$-UCB algorithm using FKM.

Hazan [7] showed that the regret performance of the FKM algorithm is of order $O(T^{3/4})$, which immediately results in the following adaptation of our result to the case where we use FKM:

Lemma 3.4.3 *The regret $r_T(\pi_{(K+1)-UCB})$ resulting from Algorithm 4 using FKM in the estimator optimization step is of order $O(T^{3/4})$.*

3.4.4 Tuning $(K+1)$-UCB Parameter ξ

The parameter ξ, appearing in the expression for $g_k(t)$, controls the rate-of-decay of the UCB indices $UCB_{k,t}$ in Algorithm 4. Our analysis above for the oracle estimator shows that the error is of $O(T^{2/3})$. As discussed above, we have two error components: a "misplay error", resulting from sampling a corner point of Δ^K instead of sampling the point determined by the oracle estimator; and a "probabilistic error," occurring when the conditions for lemma 3.3.1 do not hold.

This expected error, denoted $E(T,K)$, is a function of K and is bounded by:

$$E(T,K) \leq \underbrace{3K\sqrt{\pi}\exp(-2\xi)}_{\text{probabilistic error, equation (3.28)}} + \underbrace{\sum_{k=1}^{K}\sum_{t=1}^{T}\frac{2\xi}{(\epsilon_k - \frac{1}{K+2}\sum_{j=1}^{K}\epsilon_k)^2}t^{2/3}}_{\text{misplay error}}. \tag{3.34}$$

Note that the misplay error can be also written as $\sum_{k=1}^{K}\frac{2H_T^{-2/3}}{(\epsilon_k - \frac{1}{K+1}\sum_{j=1}^{K}\epsilon_k)^2}\xi$, where $H_T^{-2/3}$ is a generalized harmonic number [15], and is of order $O(\log T)$. We can minimize by the bound by selecting ξ to be

$$\xi_{opt} = \frac{1}{2}\log\left(\frac{3K\sqrt{\pi}}{\sum_{k=1}^{K}\frac{2H_T^{-2/3}}{(\epsilon_k - \frac{1}{K+1}\sum_{j=1}^{K}\epsilon_k)^2}}\right) \tag{3.35}$$

Since ϵ_k's are constant with respect to K, then $\frac{1}{K+2}\sum_{j=1}^{K}\epsilon_k = O(1)$, and therefore we have $\sum_{k=1}^{K}(\epsilon_k - \frac{1}{K+2}\sum_{j=1}^{K}\epsilon_k)^2 = O(K)$. Consequently, we claim $\xi_{opt} = O(\log K)$.

3.4.5 Discussing $(K+1)$-UCB Versus Kleinberg's Algorithms

Both of Kleinberg's algorithms (Table 2.1) are based on nearly-optimal sampling of the decision space in order to identify potential regions where the optimal solution lies. These algorithm do not exploit the gradient (or a gradient estimator) and base their decision on confidence regions, similarly to UCB.

In our problem, this sampling approach becomes sub-optimal due to the censoring mecha-

nism incorporated into our model. Consider, for instance, an arbitrary decision point $\mathbf{x} \in \Delta^K$, whose k^{th} coordinate is given by x_k. The probability that any of the sampled sources are censored is $1 - \prod_{k=1}^{K} \left[P(X_k \leq x_k)^{I(x_k>0)}\right]$. Note that if a specific coordinate is 0, we expect no value from the source corresponding to that coordinate.

Kleinberg's strategy for $K = 2$ is to create a uniform-grid over the $[0, 1]$ segment, $\{1/m, 2/m, ..., (m-1)/m, 1\}$, where m is some factor dependent on T [10]. The algorithm focuses its search to the best candidate (using MAB) by continuously increasing m and making the grid finer. If we sample the grid at j/m for some j, the probability the total sampled value is censored becomes is $1 - P(X_1 \leq j/m) \cdot P(X_2 \leq (1 - j/m))$. The expression suggests that if the algorithm samples points near but not at the edges of Δ^2 (that is, j is either very small or very large), the probability to lose information increases. As such, Kleinberg's algorithm has a significant disadvantage in settings where the optimal solution resides near the edges of Δ^2.

In the $(K + 1)$-UCB case, however, the probability to lose information due to censoring becomes 0. The trade-off, however, lies in the fact that for larger values of K, $(K + 1)$-UCB must spend many iterations sampling each individual source, whereas Kleinberg's methods sample in the decision space in less-stable but potentially more "interesting" points.

Zooming Algorithm Performance

The zooming algorithm operates in phases $i = 1, 2, ...$, each requiring 2^i time steps. If there are $p \in \mathbb{N}$ phases, the total number of time steps required is $2^{p+1} - 2$. In this section, we assume that $T = 2^{p+1} - 2$ for some $p \in \mathbb{N}$, and without loss of generality, assume the first center used by the zooming algorithm is $e_1 \in \Delta^K$.

Let $n_j(t)$ be the number of plays of the arm corresponding to the ball centered at e_j at time step t. Appendix A.2 shows that if $n_j(t) \leq 4i - 1$, then the ball covers the entirety of Δ^K, and that the zooming algorithm uncovers K balls, whose centers are the vectors $\{e_j\}_{j=1}^{K}$. Without loss of generality, assume the balls are added to the collection in the order $e_1, e_2, ..., e_K$. The key observation is that while at least of the one balls in the collection covers the entirety of Δ^K, we will only ever sample the objective function at one of the points $\{e_j\}_{j=1}^{K}$.

3.5 Addressing the $K = 2$ Case

In this section, we focus on the case where $K = 2$. While the $(K+1)$-UCB algorithm is still applicable in this case, we focus our attention on the twin-cost policy estimator introduced in subsection 3.3.2.

Recall that we define the twin-cost policy estimator using a parameter $\beta \in [0, 1]$. This parameter defines 2-dimensional points $(\beta, 1-\beta)^T$ and $(1-\beta, \beta)^T$ that we can use in exploration steps. Each such 2-dimensional point defines an "arm", in a similar manner to how the corner points of Δ^K defined "arms" for the $(K+1)$-UCB algorithm. Furthermore, we use "+1-arm" that is based on the twin-cost policy estimator (3.12). We dub this algorithm Mixed-$(2+1)$-UCB, which is detailed in Figure 5. Note that the "+1-arm" is indexed as $(2+1)$ to highlight its difference from arms 1 and 2.

The Mixed-$(2+1)$-UCB algorithm can be viewed as an extension of Algorithm 4. Setting the parameter β to either 0 or 1 results in the $(K+1)$-UCB algorithm for the $K = 2$ case exactly. We note that this algorithm may not be as appealing as the $(K+1)$-UCB algorithm due to its sampling strategy. While the algorithm uses a more complex estimator (3.12) that is potentially more accurate, the algorithm is more prone to censoring effects, especially as β approaches the midpoint of $1/2$. Also, the exact location of the optimal solution with the 2-dimensional simplex may also affect the performance of this algorithm.

Algorithm 5: Mixed-$(2+1)$-UCB

Input: $T \in \mathbb{N}_+$ (time-horizon)

$B = 1$ (budget)

$\xi \in \mathbb{R}_+$

$\beta \in [0, 1]$ (parameter)

Output: $\pi_{(K+1)-UCB} = \{\mathbf{x}_t\}_{t=1}^{T}$ ($(K+1)$-UCB policy)

$\forall k = 1, 2 : UCB_{k,0} \leftarrow \infty, N_{k,0} \leftarrow 0, \hat{\mu}_{k,0} \leftarrow 0$

$UCB_{(2+1),0} \leftarrow 0$

for *t = 1, 2, ..., T* **do**

$k_t \leftarrow \arg\max\{UCB_{1,t-1}, UCB_{2,t-1}, UCB_{(2+1),t-1}\}$

$$\mathbf{x}_t \leftarrow \begin{cases} (\beta, 1-\beta)^T & k_t = 1 \\ (1-\beta, \beta)^T & k_t = 2 \\ \arg\max_{x \in \Delta^K} \mathcal{F}_t(x) & k_t = (2+1) \end{cases}$$

$y_t \leftarrow \sum_{k=1}^{K} f_{k,t}(x_k)$ (play arm i_t)

if $k_t \neq (2+1)$ **then**

$N_{k_t,t} \leftarrow N_{k_t,t-1} + 1$

$\hat{\mu}_{k_t,t} \leftarrow (\hat{\mu}_{k_t,t-1} + y_t)/N_{k_t,t-1}$

$UCB_{k,t} \leftarrow \hat{\mu}_{k_t,t} + \sqrt{\xi t^{2/3}/N_{k_t,t}}$ (update UCB index according to the index type)

end

$UCB_{(K+1)} \leftarrow \max_{\mathbf{x} \in \Delta^K} \mathcal{F}_t(\mathbf{x})$ (optimize the estimator)

end

3.6 Limitations and Caveats

In this section, we discuss several limitations and caveats of our proposed model and algorithm. We first address the assumptions presented in the model and propose some potential extensions to explore, and then proceed to discuss the limitations of our proposed algorithm $(K+1)$-UCB from a practical perspective.

3.6.1 Model Limitations

In section 3.1, we gradually designed a mathematical model, including its associated assumptions, for the purpose of intelligence collection. A key assumption in our analysis is that fact that each intelligence source contains information that can be evaluated into

a real number in the range $[0, 1]$. While this assumption addresses an organizational issue, requiring a consistent evaluation process of every piece of information collected, it is nonetheless crucial for an organization to address should they choose to employ our model and algorithm.

Another important assumption, albeit not explicitly stated within the model, is the fact that intelligence sources cannot change significantly between collection attempts. Each time step $t \in [T]$ should represent an isolated period of time in which the decision maker can direct collection attempts *and* also observe the value of information collected. If extraction of information takes too long, the source of information might change or become irrelevant.

3.6.2 $(K + 1)$-UCB Algorithm Limitations

In subsection 3.4.3, we showed that Algorithm 4, dubbed the $(K + 1)$-UCB, provides satisfactory results in terms of regret. Since the algorithm samples the corner points of Δ^K, it is able to collect uncensored information and construct a better estimator for the objective function. This property of the algorithm also presents a limitation when the number of intelligence sources K increases – since the algorithm has to spend multiple time steps to sample each source individually, if $K \approx T$, then $(K + 1)$-UCB will be only exploring – not exploiting.

Another potential caveat in our design of the $K + 1$-UCB algorithm is the fact that the algorithm treats the set of intelligence source as a singular objective function. In practice, since we sample each intelligence source individually, the decision maker has a set of K values to evaluated, some of which are uncensored. As a result, a decision maker noting a source $k \in [K]$ is censored when paying a cost x_k, can avoid paying the same or lower cost on that source, seeing that it was censored. This proposed decomposition of the objective function into its K component may be key to producing better results, while potentially sacrificing the ability to rigorously analyze any proposed algorithms. In this work, we made it our priority to address models and algorithms that can be accompanied by a sound mathematical analysis of their performance.

CHAPTER 4:
Numerical Experiments and Results

In this chapter, we study our results from Chapter 3 in various scenarios. Our experiments include both the estimation of the objective function in a censored environment (as detailed in section 3.3), as well empirical evaluation of algorithm implementations.

Since we assume the distribution of the value of information of all sources is on $[0, 1]$, we will typically address the following three types of intelligence sources' distributions:

- A triangular distribution, denoted by $Tri(a, b, c)$ for $a, b, c \in [0, 1]$, and $a \leq c \leq b$, whose mean is $\frac{a+b+c}{3}$.
- A beta distribution, denoted by $Beta(\alpha, \beta)$, with $\alpha, \beta \in [0, 1]$, whose mean is $\frac{\alpha}{\alpha+\beta}$,
- A combination of the above distributions, with some sources having a triangular distribution and others having a beta distribution.

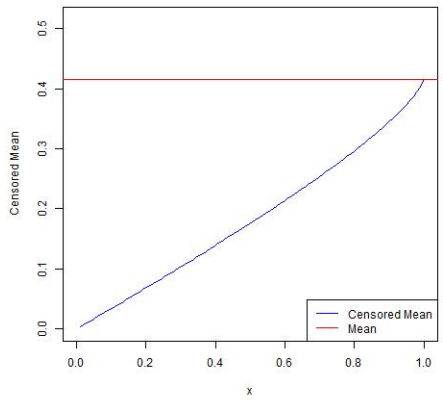
4.1 Evaluation of Censored Mean Estimators

We first explore the context of censored environment and how to estimate the expected observable value, as discussed in section 3.3. The expected observable value, $\mu(x)$, depends on the underlying distribution of the random variables representing the intelligence sources. Figure 4.1 shows, for example, that this function can be either convex or concave, depending on the specific distribution. This point is crucial, because the OCO framework from Chapter 2 could be employed under the assumption of concavity - an assumption we do not make.

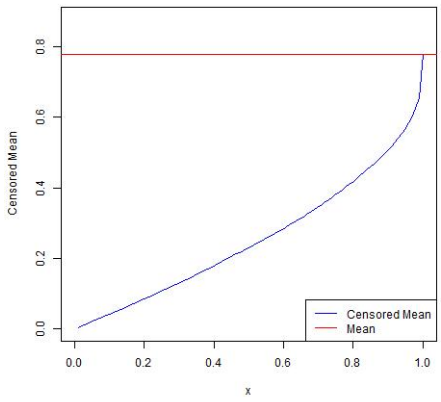
4.1.1 Visualizing the Full-cost Policy Estimator

In this subsection, we create various estimates of the censored mean in the $[0, 1]$ with varying amounts of samples. For each sample-size, we simulate the probability that the largest error between the estimates the true censored mean surpasses some threshold δ. The simulation results are based on 1000 replications for each sample size. In turn, this allows us to approximate the following expression from lemma 3.3.1,

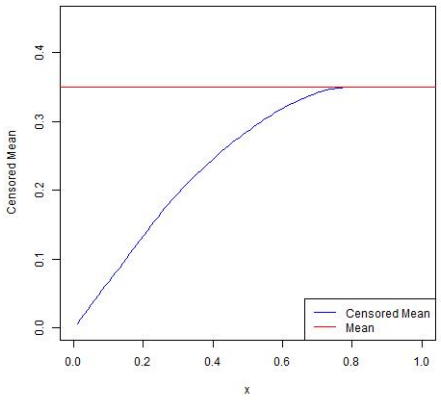
$$P(\sup_{x\in[0,1]} |\hat{\mu}(x;n) - \mu(x)| > \delta). \tag{4.1}$$



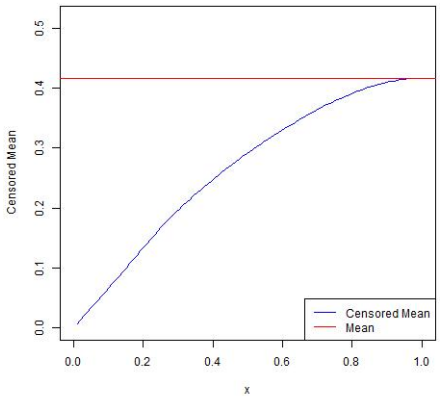
(a) Censored Mean of $Beta(0.5, 0.5)$



(b) Censored Mean of $Beta(0.7, 0.2)$

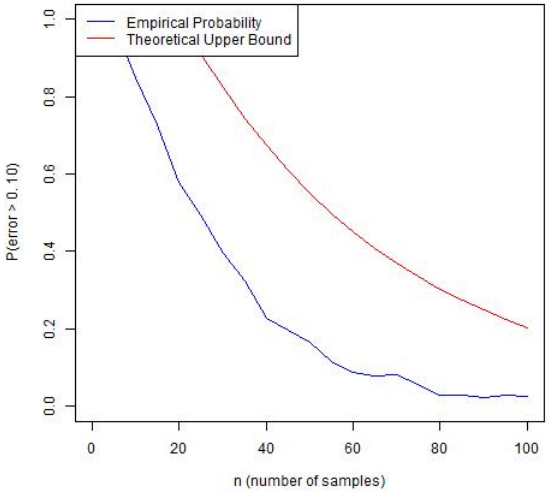


(c) Censored Mean of $Tri(0, 0.8, 0.25)$

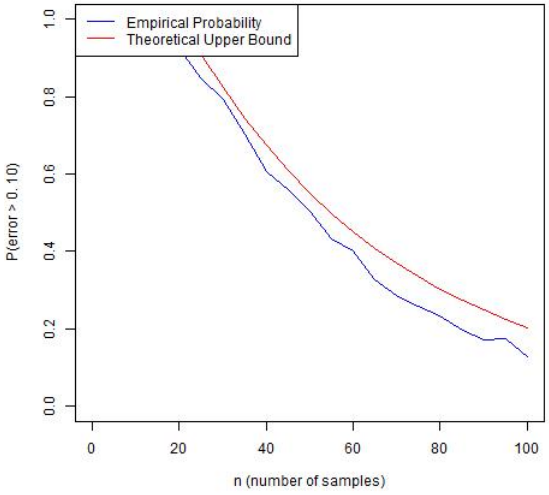


(d) Censored Mean of $Tri(0, 1, 0.25)$

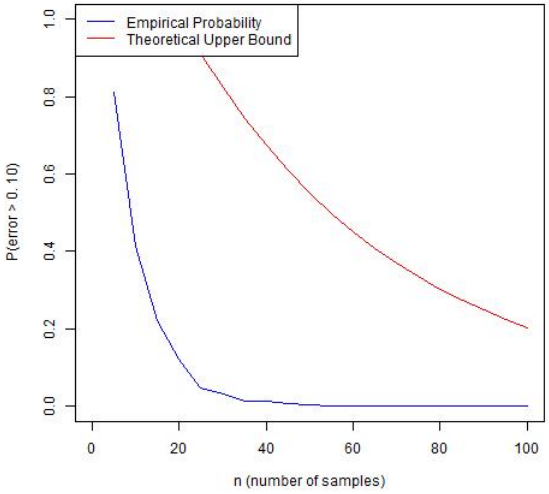
Figure 4.1. Censored means examples of different distributions and parameters. Observe that the censored mean function can be convex or concave, and its rate of increase depends on the specific distribution and its parameters.



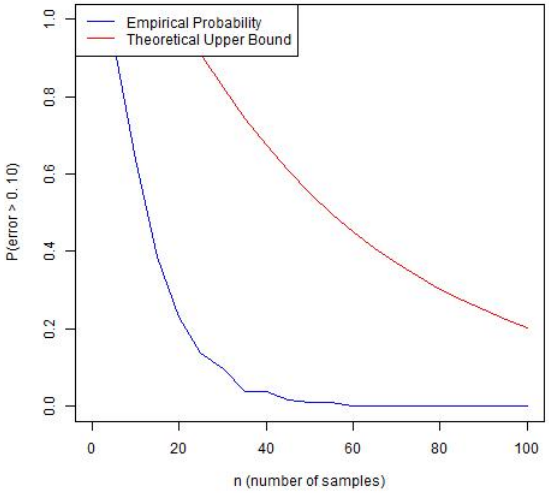
(a) $Beta(0.5, 0.5)$



(b) $Beta(0.7, 0.2)$



(c) $Tri(0, 0.8, 0.25)$



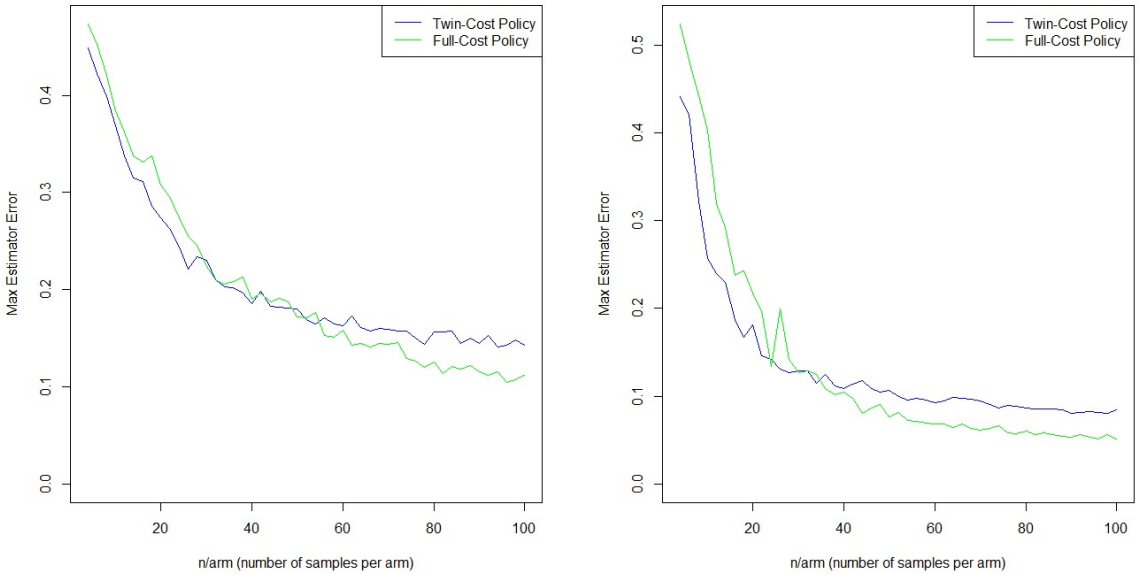
(d) $Tri(0, 1, 0.25)$

Figure 4.2. Empirical probability for a full-cost policy estimator error to exceed threshold $\delta = 0.1$ and its associated theoretical bound. Observe the theoretical bound is indeed an upper bound for our simulation, and that the simulated probability has the expected shape of decaying exponential. All results are based on 10^3 replications of simulated estimation.

Figure 4.2 shows the results of this experiment, as well as the associated theoretical bound from lemma 3.3.1. While the bound is not tight, we see it is effective for our analysis. Also, note that the simulated probability is decaying exponentially with the number of samples, regardless of the actual distribution.

4.1.2 Visualizing the Twin-cost Policy Estimator

Here we demonstrate the strengths and weaknesses of the twin-cost policy estimator discussed in subsection 3.3.2. To do so, we observe the distribution of $K = 2$ intelligence sources and evaluate the objective function from equation (3.7), instead of noting each source's individual censored mean.

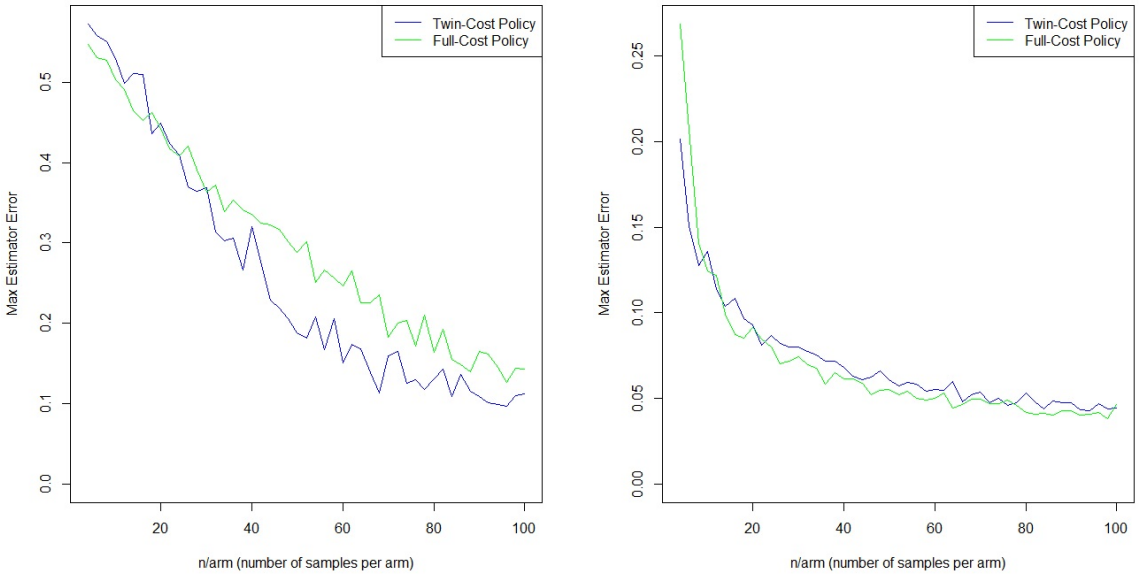


(a) $Beta(0.5, 0.5), Beta(0.7, 0.2)$ (b) $Tri(0, 0.8, 0.25), Tri(0, 1, 0.25)$

Figure 4.3. For the above distribution configurations, the plots show the max error per number of samples per arm, using the full-cost and twin-cost estimators. The maximal error is taken in the range $[0.2, 0.8]$ ($\beta = 0.2$). Observe that for smaller sample sizes per arm, the twin-cost policy estimator is more accurate.

The twin-cost policy estimator is dependent on a parameter $\beta \in [0, 1]$. Our intention was to design an estimator that is capable of operating with less samples, focusing on the range $[\beta, 1-\beta]$. While evaluating the objective function at point $x \in [0, 1]$, if $x < \beta$ or $x > 1-\beta$, we expect the results to not be as good as in the full-cost policy case.

Figure 4.3 shows two distribution configurations for the Beta and triangular distributions, and their respective maximal errors for each estimator. The maximal error is taken in the range $[0.2, 0.8]$ (that is, $\beta = 0.2$). For each estimator and each sample size, we simulate both full-cost policy estimator and twin-cost policy estimator and compare the maximal error in the range $[\beta, 1-\beta]$. The figure shows that for smaller sample sizes, the twin-cost policy estimator is more accurate.



(a) $\beta = 0.1$ (b) $\beta = 0.4$

Figure 4.4. For $K = 2$ sources, with distributions $Tri(0, 0.8, 0.25)$ and $Tri(0, 1, 0.25)$, respectively, we show the maximal error of the full-cost policy estimator and the twin-cost policy estimator per number of samples per arm. Each sub figure shows a different β value used for the twin-cost estimator. The max error is taken in the range $[\beta, 1-\beta]$ only.

As mentioned above, the performance of the twin-cost estimator depends on the specific distributions and the exact location of the optimal solution in Δ^K. If the optimal solution is either of the corner points $(1,0)$ or $(0,1)$, the twin-cost policy will not be able to collect samples that capture the optimal solution.

We demonstrate this observation in Figure 4.5. Here, we use the distributions $Beta(0.2, 0.4)$ and $Beta(0.3, 0.6)$ and $\beta = 0.2$. Unlike the previous figures, here we focus on the maximal error in the region where $x < \beta$ or $x > 1 - \beta$, that is, beyond the region where the twin-cost policy estimator is expected to prove useful. We can see that the twin-cost policy estimator error shows an almost constant error as the number of samples per arm increases, whereas the full-cost policy estimator shows a decline in the maximal error. As a consequence, the regret grows linearly under this configuration.

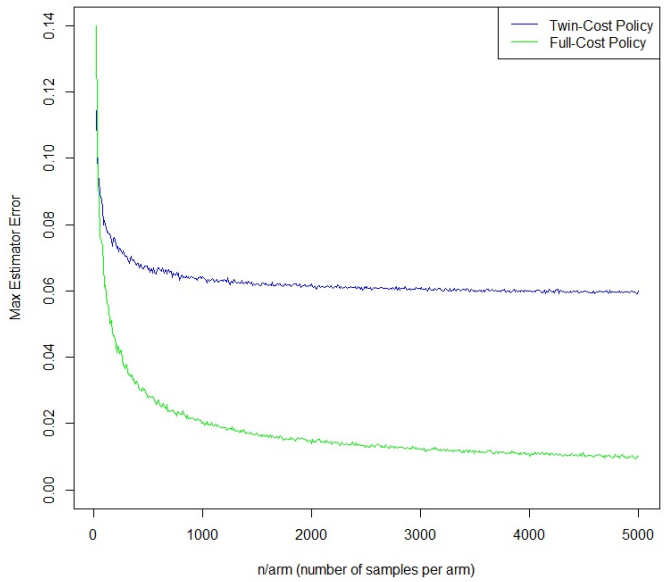


Figure 4.5. For $K = 2$ sources, with distributions $Beta(0.2, 0.4)$ and $Beta(0.3, 0.6)$, respectively, we show the maximal error of the full-cost policy estimator and the twin-cost policy estimator per number of samples per arm, with $\beta = 0.2$. The max error is taken in the range where $x < \beta$ or $x > 1 - \beta$.

4.2 Algorithm Evaluation and Comparison

In this section, we evaluate the performance of $(K+1)$-UCB algorithm (Algorithm 4), and compare it to the performance of other continuum MAB algorithms, both by Kleinberg [10], [11]. The original algorithms by Kleinberg et al. did not account for censoring of the samples. As a result, we add additional measures in our implementation of these algorithms in the form of censoring, following the same procedure as described in Chapter 3. Note that the algorithm detailed in [10] is suitable only in the case where $K = 2$.

Throughout our experiments, we use simulation over a discrete grid in Δ^K to uncover the optimal solution of equation (3.7). This step is required in order to evaluate and plot the regret of an algorithm. The plots shown in this section are based on the averaged simulated regret resulting from running each algorithm 100 times, and comparing the accumulated rewards (value of information collected) to the simulated optimal value.

In this section, we often discuss the performance of implemented algorithms. Unless noted otherwise, by performance we refer to the mean regret of the algorithm in a particular setting, as detailed in Chapters 2 and 3.

4.2.1 The $K = 2$ Case

We first address the $K = 2$ case where exactly 2 intelligence sources are present. Here, we explore both the $(K+1)$-UCB algorithm (Algorithm 4) and the Mixed-$(2+1)$-UCB algorithm (Algorithm 5).

We use different configurations that determine the distribution of both sources, as detailed in Table 4.1. One configuration, referred to as All-beta, uses information value distributions of $Beta(0.2, 0.4)$ and $Beta(0.3, 0.6)$, and one configuration, referred to as All-triangular, uses information value distributions of $Tri(0, 0.7, 0.2)$ and $Tri(0, 0.8, 0.3)$ (referred to as All-triangular). We also explore a mixture of distribution, referred to as Mixed-1-And-1, with one source having a Beta distribution, and one source having a triangular distribution.

Table 4.1. Experiment Configuration for $K = 2$

Distribution	Source 1	Source 2	Optimal Solution	Optimal Value
All-beta	$Beta(0.2, 0.4)$	$Beta(0.7, 0.6)$	$(0, 1)$	0.538
All-triangular	$Tri(0, 0.7, 0.2)$	$Tri(0, 0.8, 0.3)$	$(0.4, 0.6)$	0.5
Mixed-1-And-1	$Tri(0, 0.6, 0.4)$	$Beta(0.4, 0.6)$	$(0.525, 0.475)$	0.434

Mixed-$(2 + 1)$-UCB Parameter β Tuning

Figures 4.6 and 4.7 depict the Mixed-$(2 + 1)$-UCB algorithm (Algorithm 5) with parameter values of $\beta \in \{0.2, 0.3, 0.4\}$ for the two different distribution configurations, respectively.

We can see that the performance of the algorithm per choice of β depends on the particular distribution. For the All-beta configuration, whose optimal solution uncovered through simulation is either of the corner points of Δ^2, the performance of the algorithm for all values of β is similar, with $\beta = 0.2$ yielding the best results. Note that the corner points $(0, 1)$ and $(1, 0)$ cannot be captured by the twin-cost policy estimator, which explains the relatively similar performance for all three parameter values.

In the All-triangular configuration, however, significant differences in performance are observed, with $\beta = 0.4$ producing the best results. Through simulation, we know that the optimal solution in this case is $(0.4, 0.6) \in \Delta^2$. As such, we can except that the choice $\beta = 0.4$ will produce the best results.

We also note that both figures 4.6 and 4.7 show regret that linearly growing. As we use the twin-cost policy estimator, this behavior is expected.

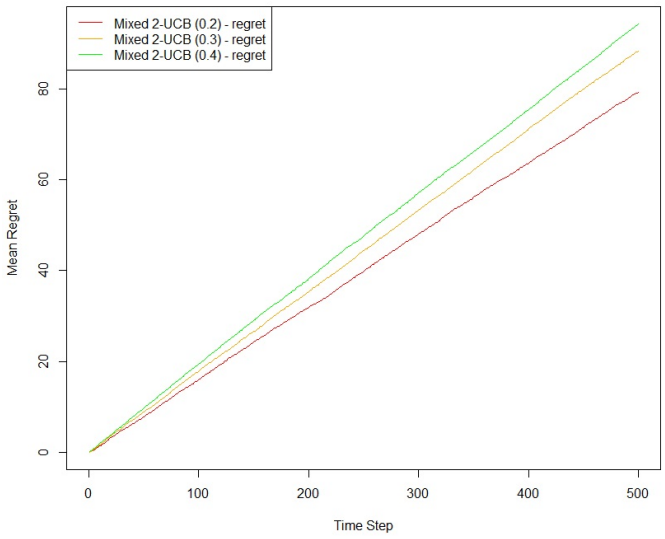


Figure 4.6. Algorithm performance depends on both the specific choice of β. For the All-beta configuration, $\beta = 0.2$ yields the best performance. Note the linear growth of the regret.

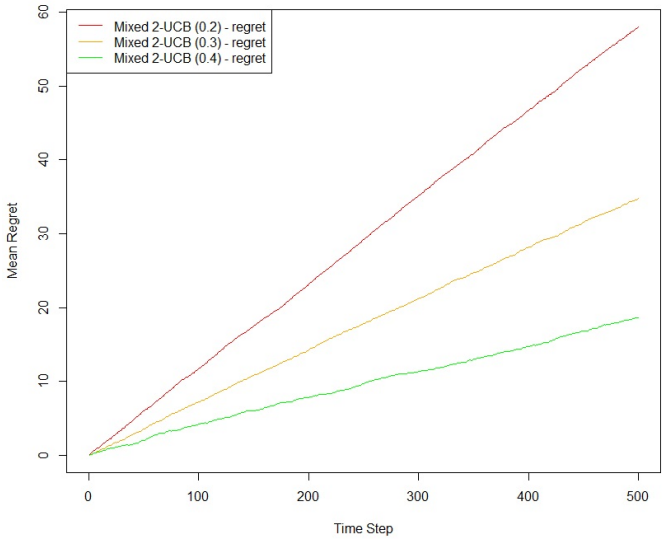


Figure 4.7. Algorithm performance depends on both the specific choice of β. For the All-triangular configuration, $\beta = 0.4$ yields the best performance. Note the linear growth of the regret.

$(K+1)$-UCB Parameter ξ Tuning

In Chapter 3, when presenting the $(K+1)$-UCB algorithm, we discussed the upper bound on the expected error. We also derived an expression for an optimal value of the parameter ξ, for which the upper bound is minimized. Figure 4.8 shows a demonstration of this analysis for the All-beta configuration, with the curve to the derived constant yielding the best performance in terms of mean regret. We remind the reader that derivation of this constant is not applicable in practice, as it requires knowledge of the distribution of each intelligence source. As expected, the figure shows sub-linear growth of regret that is of order of approximately $O(T^{2/3}\sqrt{\log T})$.

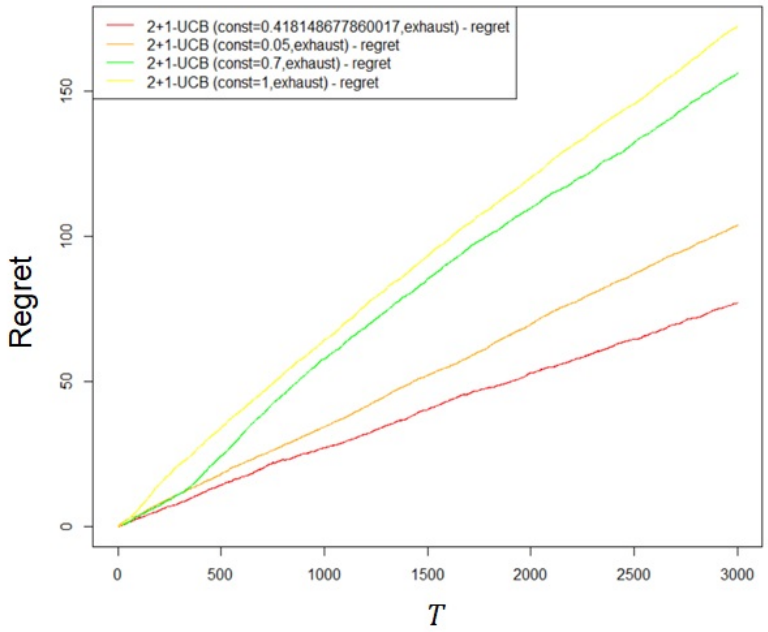


Figure 4.8. Demonstration of the performance of $(K+1)$-UCB for different values of ξ. The best performing curve corresponds to the optimal value of ξ, derived in Chapter 3. Note the regret is $\approx O(T^{2/3}\sqrt{\log T})$.

Algorithm Performance

In this subsection, we show the performance of different algorithms in the above settings. Our results are based on simulation of both algorithms across 100 replications for each setting. We separately simulate each scenario on a discretized grid approximating Δ^2 to

locate the optimal solution against which the regret is computed. Note that for the $(K+1)$-UCB algorithm, we explore two variants - one that uses FKM [7] to optimize the estimator, and another that employs an Oracle to optimize the estimator.

Figure 4.9 shows the performance of $(K+1)$-UCB with 100 steps of FKM gradient-like ascent [7] versus Kleinberg's optimal sampling strategy for $K = 2$ [10] in the All-beta setting. It is apparent that in the proposed algorithm is superior in terms of mean regret, with Kleinberg's algorithm sampling often-censored points in Δ^K, resulting in a linear regret. Moreover, while the Mixed-$(2+1)$-UCB algorithm shows linear regret, whereas Kleinbergs' algorithm and both variants of $(K+1)$-UCB show sub-linear regret.

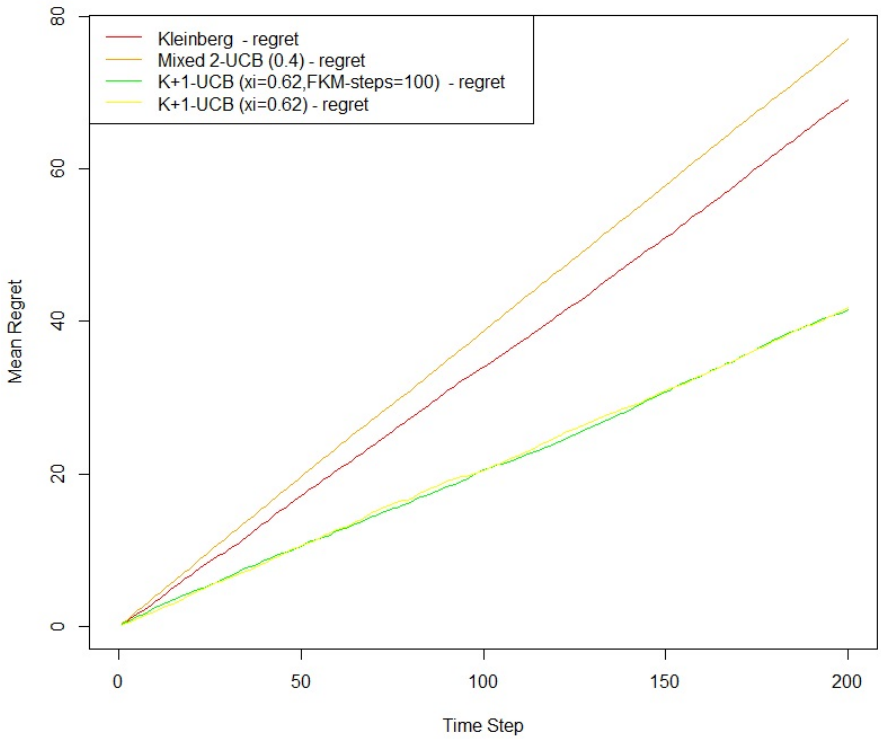


Figure 4.9. Mean regret performance of $(2+1)$-UCB algorithm, Kleinberg's algorithm, and the Mixed-(2+1)-UCB algorithm in the All-beta setting (Table 4.1). Note how both variants of $(K+1)$-UCB algorithms (with FKM and with an Oracle) outperform Kleinberg's algorithm and the Mixed-(2+1)-UCB algorithm. Note the linear vs. sub-linear regrets for each algorithm.

Figure 4.10 depicts a similar comparison of the algorithms, this time in the All-triangular setting. Here, the optimal solution lies within Δ^2 at $(0.4, 0.6)$. We can that in this example, Kleinberg's algorithm for $K = 2$ and the Mixed-$(2+1)$-UCB algorithm outperform the two variants of $(K+1)$-UCB.

In contrast, Figure 4.11 shows another comparison where the optimal solution is not a corner point (see Table 4.1). Here, we observe the opposite effect, where $(K+1)$-UCB's performance across the two variants perform better than both Kleinberg's algorithm and the Mixed-$(2+1)$-UCB algorithm. Note how in both figures 4.10 and 4.11 depict regret sub-linear growth for the $(K+1)$-UCB algorithm for both variants.

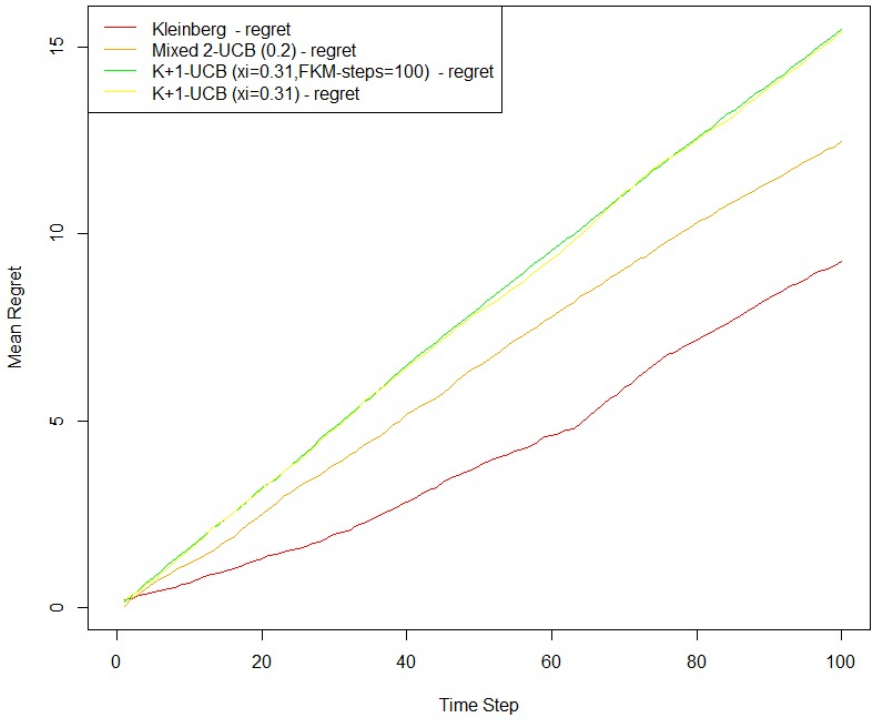


Figure 4.10. Mean regret performance the two variants of $(2+1)$-UCB algorithm in the All-triangular setting (Table 4.1). The yellow curve corresponds to the variants using an Oracle. Note the sub-linear regret growth for $(K+1)$-UCB in both variants.

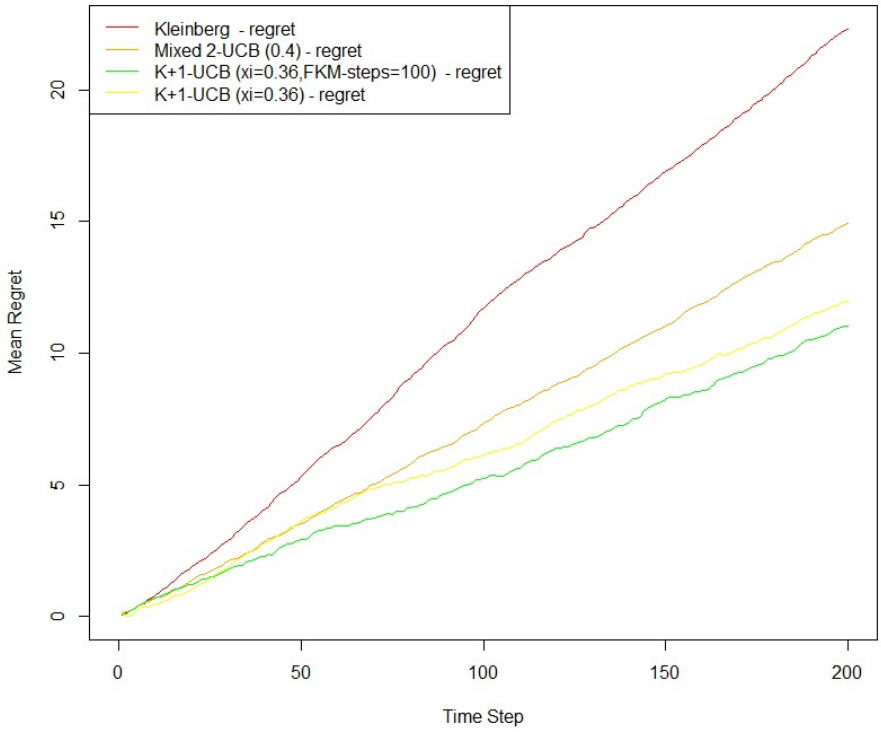


Figure 4.11. Mean regret performance the two variants of $(2+1)$-UCB algorithm in the Mixed-1-And-1 setting (Table 4.1). The yellow curve corresponds to the variants using an Oracle. Note the sub-linear regret growth for $(K+1)$-UCB in both variants.

.

4.2.2 The $K > 2$ Case

In this subsection, we focus our attention to comparison of the $(K+1)$-UCB algorithm with the zooming algorithm [11]. With $K > 2$, the use of an Oracle as part of running $(K+1)$-UCB algorithm becomes impractical, and we instead use the FKM variants discussed in Chapter 3. We observed that the implementation of the FKM variant is quite fast, even if we perform tens of iterations each time we invoke the optimizer.

In Chapter 3, we already discussed how the zooming algorithm operates, and why we believe its performance is not expected to be as good as the performance of the $(K+1)$-

UCB algorithm. We also note that parts of our analysis are detailed in Appendix A.2. Our simulations initialize the zooming algorithm from a random corner of Δ^K, as this approach was observed to be the best performing amongst all those tested.

Our results are based on simulation of both algorithms across 100 replications for each setting. We separately simulate each scenario on a discretized grid approximating Δ^K to locate the optimal solution against which the regret is computed. Consequently, we are limited by the maximal dimension K for which we can readily compute the optimal solution, and present here results only for $K = 4$. The specific distribution configurations are depicted in Table 4.2. For each configuration, we use equation (3.35) to optimally tune the constant ξ for the $(K + 1)$-UCB algorithm.

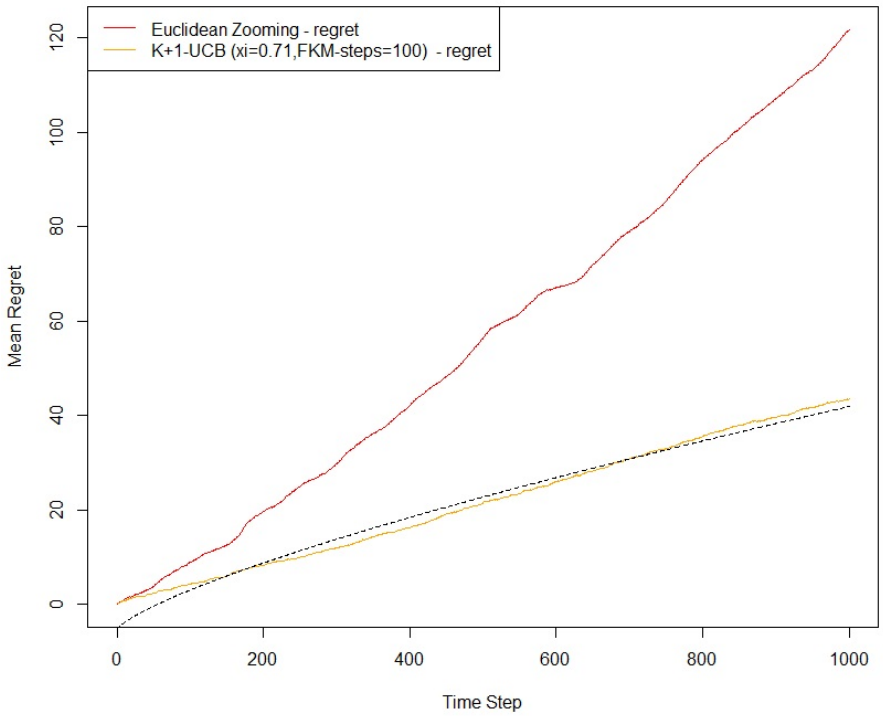


Figure 4.12. Mean regret performance of $(K + 1)$-UCB algorithm versus the zooming algorithm in the All-beta setting (Table 4.2). The black dotted curve denotes the a polynomial regression fit of the form $O(T^{3/4})$.

Figures 4.12, 4.13 and 4.14 depict the mean regret across simulations. The figures show the regret resultant from running both the zooming algorithm and $(K+1)$-UCB, when equipped with the FKM algorithm [7] as the optimizer of the estimator (see Chapter 3 for further details). We have also highlighted a fitted regression curve of order $T^{3/4}$, showing the regret of $(K+1)$-UCB is in accordance to result 3.4.3.

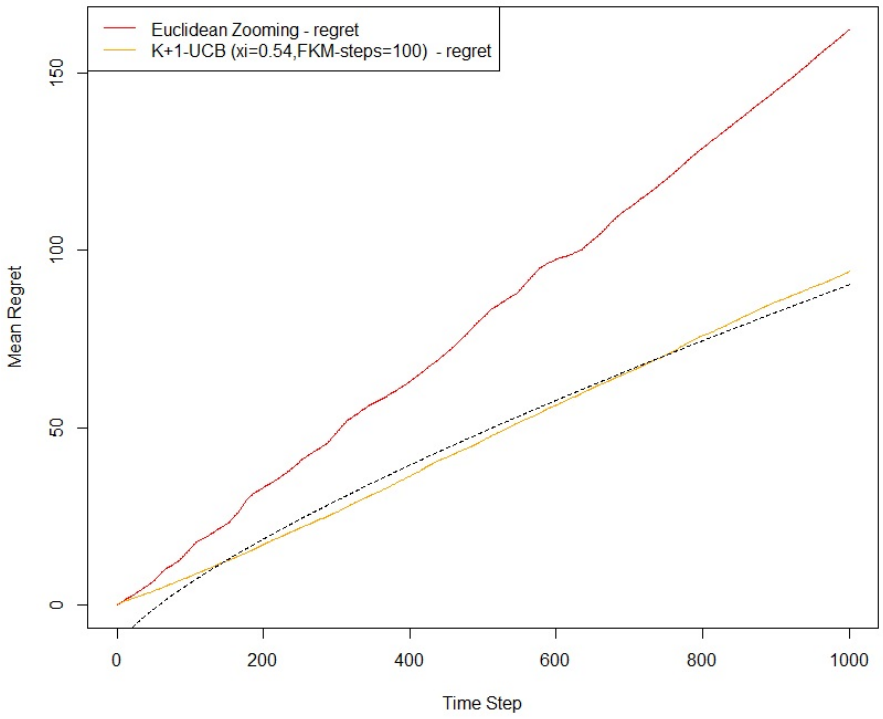


Figure 4.13. Mean regret performance of $(K+1)$-UCB algorithm versus the zooming algorithm in the All-triangular setting (Table 4.2). The black dotted curve denotes the a polynomial regression fit of the form $O(T^{3/4})$.

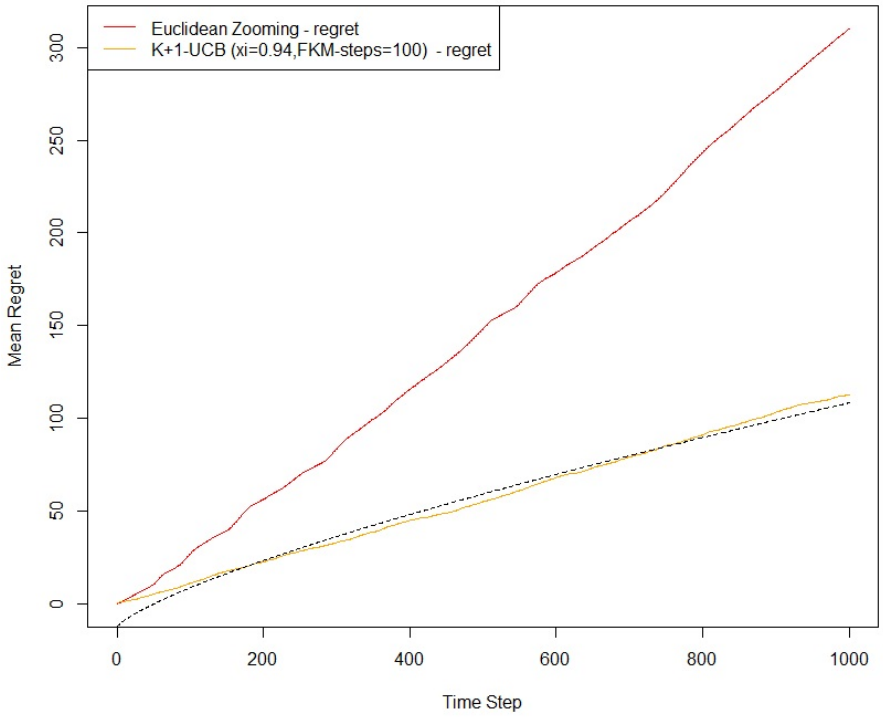


Figure 4.14. Mean regret performance of $(K+1)$-UCB algorithm versus the zooming algorithm in the Mixed setting (Table 4.2). The black dotted curve denotes the a polynomial regression fit of the form $O(T^{3/4})$.

Effect of K on the Regret

In order to explore the effect of K on the regret, we construct a simulation of a scenario where adding more intelligence sources does not change the optimal solution. To do so, we have one source whose expected value is very high, and all other $K-1$ sources have a very low expected value. Figure 4.15 shows the results of this experiment. Note how all curves depict regret of approximate order of $O(T^{3/4})$, but becomes increasingly larger for larger values of K due to more time steps being spent on exploration.

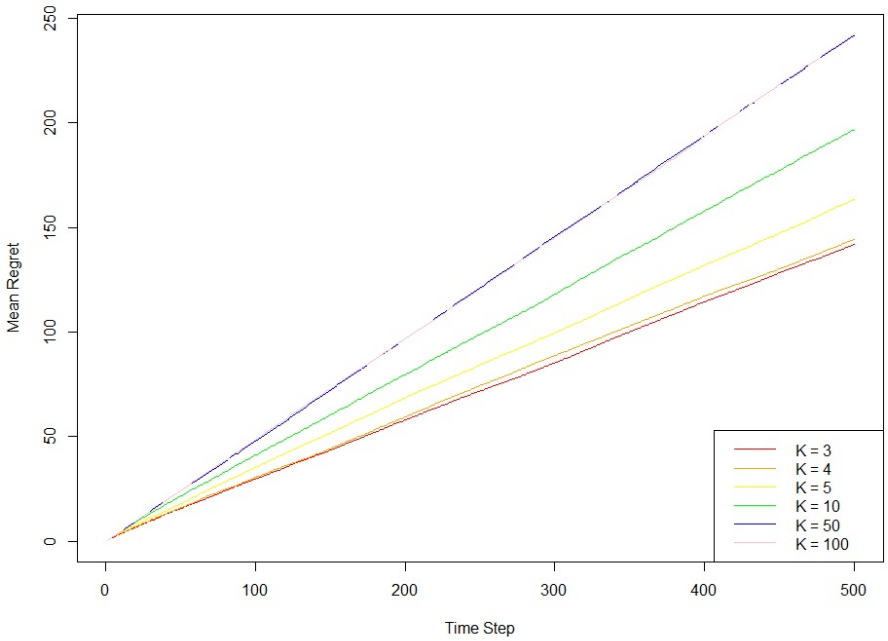


Figure 4.15. Mean regret performance of $(K+1)$-UCB algorithm for different values of K. The simulation is constructed such that the optimal value is identical in all cases. Note that for K, the regret is of order $O(T^{3/4})$.

4.3 Summary

In this chapter, we presented numerical simulation results that trend the various theoretical results and claims presented in Chapter 3. Our demonstrations used simulation of the decision space to uncover the true optimal solution, allowing us to compute the regret where appropriate, and observe the performance of both existing and proposed algorithms.

Importantly, our results are consistent with the ones presented in Chapter 3. Moreover, in our experiments, we observed that the performance of the $(K+1)$-UCB algorithm was better than that of both Kleinberg's sampling algorithm for $K = 2$ [10] and the zooming algorithm [11]. For the $(K+1)$-UCB algorithm, the growth of regret shown in this chapter is either $O(T^{2/3})$ for the Oracle variant, or $O(T^{3/4})$ for the FKM variant. Note that the Oracle variant is not practical for higher dimensions ($K > 2$), while the FKM variant is relatively easy and fast to compute, even for tens of iterations. We also observed linear growth of

regret for the Mixed-$(2+1)$-UCB algorithm.

Table 4.2. Experiment Configuration for $K > 2$

Distribution	**Per-Arm Mean**	**Optimal Value**	**Is Solution a Corner?**
All-beta	$(0.333, 0.5, 0.416, 0.466)$	0.5	Yes
All-triangular	$(0.233, 0.433, 0.4, 0.5)$	0.522	No
Mixed	$(0.233, 0.433, 0.4, 0.75)$	0.75	Yes

CHAPTER 5:
Conclusion

This chapter summarizes our main results. Chapter 3 presented our proposed model and discussed its limitations, the methodology used to tackle the problem that motivates the model, and provided analysis of our proposed algorithm, $(K+1)$-UCB. In Chapter 4, we have verified our results numerically through simulation and comparison to existing approaches.

We have shown that by choosing to sample uncensored points we are able to slowly explore the continuous decision space, while simultaneously improving upon an exploitation point. The exploitation point can be uncovered using an oracle or by using censor-free online optimization technique, such as the FKM algorithm [7]. The proposed was shown to be useful, yielding regret of order $O(T^{3/4})$ using FKM algorithm. Consequently, $(K+1)$-UCB is expected to outperform high-dimension sampling strategies such as the zooming algorithm [11].

We also showed that the optimal confidence interval constant used by $(K+1)$-UCB is of order $O(\log K)$. Although this constant cannot be uncovered under normal circumstances as it depends on the specific distribution, the analysis shows that $(K+1)$-UCB exploits the internal relationship between the K intelligence sources imposed by the continuous decision space, resulting in sub-linearity in K.

An important emphasis of our work is the ability to provide rigorous analysis of the algorithms. We acknowledge that this self-induced limitation has led us to leave aside certain appealing aspects of the model, such as combining our +1 approach with other algorithms, or exploiting the distributed nature of the problem (using K inputs, instead of treating them as a singular function). We leave those endeavors to our future work.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX: Supplementary Proofs and Notes

A.1 Proof of Lemma 3.3.1

For any $\delta > 0$, given n IID uncensored samples of a source:

$$P(\sup_{x\in[0,1]} |\hat{\mu}(x;n) - \mu(x)| > \delta) \leq 4e^{-2n\delta^2} \tag{A.1}$$

A.1.1 Proof

Let there be n uncensored samples of the random variable X, denoted by $\mathcal{X} = \{x_i\}_{i=1}^n$.

Let $\hat{F}(x;n)$ be the sample cumulative distribution function of X at $x \in [0,1]$,

$$\hat{F}(x;n) = \frac{1}{n}\sum_{x_i\in\mathcal{X}} I(x_i \leq x). \tag{A.2}$$

Consider some $x \in [0,1]$. Without loss of generality, assume the samples in $\mathcal{X}$ are ordered such that $x_1 \leq x_2 \leq \ldots \leq x_k \leq x < x_{k+1} \leq \ldots \leq x_n$. Consequently, we have $\hat{F}(x;n) = k/n$. In addition, observe that

$$\int_0^x \hat{F}(z;n)dz = \frac{1}{n}\sum_{i=1}^{k}(x_i - x_{i-1})\cdot(i-1) = -\frac{1}{n}\sum_{i=1}^{k} x_i + x\frac{k}{n}, \tag{A.3}$$

where we set $x_0 = 0$. Note that $\hat{\mu}(x;n) = \frac{1}{n}\sum_{i=1}^{k} x_i$, and so we can conclude that:

$$\hat{\mu}(x;n) = \int_0^x (\hat{F}(x;n) - \hat{F}(z;n))dz. \tag{A.4}$$

Similarly, using the definition in equation 3.1, we have $\mu(x) = \int_0^x zf(z)dz$. Using integration

by parts, it is apparent that

$$\mu(x) = \int_0^x (F(x) - F(z))dz, \tag{A.5}$$

and by combining these two results, we have

$$\begin{aligned} |\mu(x) - \hat{\mu}(x;n)| = \\ |\int_0^x (F(x) - \hat{F}(x;n)dz) - \int_0^x (F(z) - \hat{F}(z;n))dz| \leq \\ 2x|F(x) - \hat{F}(x;n)|. \end{aligned} \tag{A.6}$$

The above expression holds for some arbitrary $x \in [0, 1]$. We can bound the probability the deviance between $\hat{\mu}(x;n)$ and $\mu(x)$ is greater than some $\delta > 0$, by using the Dvoretzky-Kiefer-Wolfowitz (DKW) inequality [16],

$$P(\sup_{x\in[0,1]} |\hat{\mu}(x;n) - \mu(x)| > \delta) \leq 2 \cdot 2 \cdot e^{-2n\delta^2}. \tag{A.7}$$

A.2 Zooming Algorithm on the Simplex Space

The work in [11] describes a general method to address the continuous space MAB through an algorithm referred to as the *zooming algorithm*. In this section, we discuss the implementation of the algorithm in the context of model and highlight some of its properties that are specific to our implementation.

In this section, we refer to a generalized version of the simplex space, defined via

$$\Delta^K = \{\mathbf{x} \in \mathbb{R}^K | \mathbf{x} \geq 0, \mathbf{1}^T\mathbf{x} = B\}, \tag{A.8}$$

where B is the available budget (see Chapter 3).

The zooming algorithm starts with a single ball centered at $\mathbf{c}_1 \in \Delta^K$ that covers the entirety of Δ^K. As time progresses, the radius of the ball $r_1(t)$ shrinks, while increasing the confidence

of the estimation of the objective value at the point $\mathbf{c}_1$. At any time step t, $r_1(t)$ is not sufficient to cover Δ^K, the algorithm must find a new vector $\mathbf{c}_2 \in \Delta^K$ such that both balls cover Δ^K, or simply, $\Delta^K \subset B(c_1, r_1(t)) \cup B(c_2, r_2(t))$.

The radius $r_j(t)$ is only dependent on the algorithm actions and is given by $r_j(t) = \sqrt{\frac{8 \cdot i}{1+n_j(t)}}$, where i is the *phase* of the algorithm and $n_j(t)$ is the number of times the ball centered at $\mathbf{c}_j$ is sampled. For further details, we refer the reader to [11].

A.3 Covering Oracle Implementation

As part of the zooming algorithm implementation, we must implement the *covering oracle*. Given a set of balls $\mathcal{B}(t) = \{B(\mathbf{c}_j, r_j(t))\}_{j=1}^J$, we must be able to find a point $\mathbf{v} \in \Delta^K$ that is not covered by the $\bigcup_{j=1}^J B(\mathbf{c}_j, r_j(t))$, or alert that no such point exists.

To do so, we propose the formulation

$$\begin{aligned} &\max_{\mathbf{v} \in \Delta^K} \sum_{j=1}^{J} ||\mathbf{v} - \mathbf{c}_j||_2^2 \\ &\text{subject to} \\ &\forall j = 1, 2, ...J : ||\mathbf{v} - \mathbf{c}_j||_2^2 \geq r_j^2(t). \end{aligned} \tag{A.9}$$

This formulation is highly non-convex, even when $J = 1$. However, recall we are addressing this formulation incrementally, starting with a single ball ($J = 1$).

A.3.1 The $J = 1$ Case

If $J = 1$, then we have a single ball to cover Δ^K. Note that we have $||\mathbf{v} - \mathbf{c}_1||_2^2 \leq 2B^2$, where we used the fact that within the simplex Δ^K, the ℓ_1-norm is bounded by 1. Consequentially, if $r_j^2(t) > 2B^2$ (or, equivalently, $n_j(t) \leq 4i/B^2 - 1$), the above formulation has no solution. This observation allows us to skip addressing the optimization when these conditions hold.

The choice of $\mathbf{c}_1$ is also critical, because the bound on $||\mathbf{v} - \mathbf{c}_1||_2^2$ can be tighter than $2B^2$ for specific choices of $\mathbf{c}_1$. Our implementation randomly chooses $\mathbf{c}_1$ to be drawn from among $\{B \cdot e_k\}_{k=1}^K$. As such, the zooming algorithm is evaluated on equal grounds as the

$(K+1)$-UCB algorithm, since $B \cdot e_k$ guarantees an uncensored sample.

Since $\mathbf{c}_1 = B \cdot e_k$, the formulation is reduced to

$$\begin{aligned} &\max_{\mathbf{v} \in \mathbb{R}^K} ||\mathbf{v} - B \cdot e_k||_2^2 \\ &\text{subject to} \\ &||\mathbf{v} - e_k||_2^2 \geq r_1^2(t). \\ &\mathbf{1}^T \mathbf{v} = B \\ &\mathbf{v} \geq 0. \end{aligned} \tag{A.10}$$

While this formulation is still not convex, observe that for any $i \neq k$, $||B \cdot e_i - B \cdot e_k||_2^2 = 2B^2$, also $B \cdot e_i \in \Delta^K$. Therefore, $B \cdot e_i$ $(i \neq k)$ is an optimal solution that holds while $r_1^2(t) \leq 2B^2$.

A.3.2 Beyond $J = 1$ Case

Using the above analysis, we can make the following observation: the first K centers will be the B-scaled standard basis of the K-dimensional Euclidean space.

We can show this by induction. Assume the first $J < K$ are all B-scaled standard basis vectors $\{B \cdot e_j\}_{j=1}^J$. Let t_J be the time step where the union of balls $\bigcup_{j=1}^J B(B \cdot e_j, r_j(t_J))$ does not cover Δ^K, and we must therefore find a new center $\mathbf{c}_{J+1}$ using the formulation in equation (A.9).

Let $B \cdot e_k$ be a B-scaled basis vector of the K-dimensional Euclidean space, such that $e_k \neq e_j$ for $j = 1, ..., J$. Note that $\sum_{j=1}^J ||\mathbf{v} - B \cdot e_j||_2^2 \leq 2JB^2$ for any $\mathbf{v} \in \Delta^K$, and for choice $\mathbf{v} = B \cdot e_k$, we have exactly $\sum_{j=1}^J ||e_k - e_j||_2^2 = 2JB^2$. As such, $B \cdot e_k$ is the solution of equation (A.9), if it is a feasible solution.

Immediately we have $B \cdot e_k \in \Delta^K$, so we must only confirm that $||B \cdot e_k - B \cdot e_j||_2^2 \geq r_j^2(t_J)$. Recall that if $r_j^2(t) > 2B^2$ the formulation has no solution, i.e., the simplex is covered by the union of ball. Since we assumed the simplex is not completely covered, then $r_j^2(t) \leq 2B^2 = ||B \cdot e_k - B \cdot e_j||$, and thus, $B \cdot e_k$ is indeed a feasible solution.

This analysis is especially interesting in the context of our comparison to the Algorithm

4 where $B = 1$. While Algorithm 4 is specifically *designed* to sample the simplex space Δ^K at standard basis points of the K-dimensional Euclidean space, the zooming algorithm *uncovers* these sampling points, instead. We will note that the zooming algorithm will eventually sample additional points from Δ^K as time steps progress, whereas Algorithm 4 will not.

A.3.3 Numerical Approach

Our numerical approach includes the following steps, assuming $B = 1$:

Step 1: Check Condition

First, we verify the $J = 1$ condition $r_1^2(t) \leq 2B$. If that condition does not hold, the simplex is covered by the first ball $B(e_k, r_1(t))$ and we do not to explore further.

Step 2: Find Another Ball Center

Otherwise, we use the Augmented Lagrangian method [17] to find a solution to the above optimization problem. We limit the number of iterations for the Augmented Lagrangian by some value I_{AL}, and if a feasible solution is not found – we conclude the simplex is covered by the collection of balls in our possession. If a solution is found, and it is different than all other ball centers currently in the collection, we add it to our collection.

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] M. M. Lowenthal and R. M. Clark, *The Five Disciplines of Intelligence Collection*. SAGE, Jan. 2015, [Online]. Available: Google-Books-ID: rdI5DQAAQBAJ.

[2] "IC Budget," [Online]. Available: https://www.dni.gov/index.php/what-we-do/ic-budget.

[3] D. Kronzilber, "Multi-armed bandit models of network intrusion in the cyber domain," Thesis, Monterey, California, USA: Naval Postgraduate School, 2017, [Online]. Available: https://calhoun.nps.edu/handle/10945/56715.

[4] M. Tekin, "New perspectives on intelligence collection and processing," Thesis, Monterey, California, USA: Naval Postgraduate School, 2016, [Online]. Available: https://calhoun.nps.edu/handle/10945/49398.

[5] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge University Press, July 2020, [Online]. Available: Google-Books-ID: bbjpDwAAQBAJ.

[6] J. D. Abernethy, K. Amin, and R. Zhu, "Threshold bandit, with and without censored feedback," *Advances In Neural Information Processing Systems*, vol. 29, pp. 4889–4897, 2016.

[7] E. Hazan, "Introduction to online convex optimization," *arXiv preprint arXiv:1909.05207*, 2019, [Online]. Available: https://arxiv.org/abs/1909.05207.

[8] W. Wang and M. A. Carreira-Perpinán, "Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application," *arXiv preprint arXiv:1309.1541*, Sep. 2013, [Online]. Available: https://arxiv.org/abs/1309.1541v1.

[9] Y. Chen and X. Ye, "Projection onto a simplex," *arXiv:1101.6081 [math]*, Feb. 2011, [Online]. Available: http://arxiv.org/abs/1101.6081.

[10] R. Kleinberg, "Nearly tight bounds for the continuum-armed bandit problem," *Advances in Neural Information Processing Systems*, vol. 17, p. 8, 2004.

[11] R. Kleinberg, A. Slivkins, and E. Upfal, "Bandits and experts in metric spaces," *Journal of the ACM (JACM)*, vol. 66, no. 4, pp. 1–77, 2019, [Online]. Available: https://arxiv.org/abs/1312.1277v4.

[12] D. Zhou and C. Tomlin, "Budget-constrained multi-armed bandits with multiple plays," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, number: 1.

[13] S. M. Ross, *Probability Models for Computer Science*. Harcourt Academic Press San Diego, USA, 2002.

[14] E. L. Kaplan and P. Meier, "Nonparametric estimation from incomplete observations," *Journal of the American statistical association*, vol. 53, no. 282, pp. 457–481, 1958.

[15] N. Wiener, "Generalized harmonic analysis," *Acta mathematica*, vol. 55, no. 1, pp. 117–258, 1930.

[16] A. Dvoretzky, J. Kiefer, and J. Wolfowitz, "Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator," *The Annals of Mathematical Statistics*, vol. 27, no. 3, pp. 642–669, 1956, institute of Mathematical Statistics.

[17] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Mar. 2004, [Online]. Available: Google-Books-ID: mYm0bLd3fcoC.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California