ADAPTIVE BAYES-OPTIMAL METHODS FOR STOCHASTIC SEARCH WITH APPLICATIONS TO PREFERENCE LEARNING

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Stephen N. Pallone

August 2017

ADAPTIVE BAYES-OPTIMAL METHODS FOR STOCHASTIC SEARCH WITH APPLICATIONS TO PREFERENCE LEARNING

Stephen N. Pallone, Ph.D.

Cornell University 2017

Many fundamental problems in mathematics can be considered search problems, where one can make sequential queries to find a point with desired properties. This includes convex optimization, in which queries are points within the feasible region, and the corresponding subgradient implies a separating hyperplane that allows non-optimal points to be discarded. Search problems can be solved from a Bayes-optimal viewpoint, where a prior distribution is maintained over the search space quantifying the likelihood of the desired point's location. In this manner, queries and their responses allow this prior to be updated with new information, and the objective is to ask the best query to learn the location of the desired point as efficiently as possible.

This framework can be adapted to solving variants of one dimensional deterministic search, which includes work such as developing query strategies associated with parallel function evaluations, or analyzing a bisection-like method that sorts the roots of functions that must be evaluated together.

Most significantly, this idea is used in multidimensional search in the application of learning the preferences of a single user. In this case, the queries are comparative questions, where multiple alternatives are presented to the user, and they choose the most preferred option. The preferences of the user are encoded in a linear classifier, where the user's utility for an alternative roughly corresponds to the dot product between this classifier and a feature vector. The

response model enables answers to be contaminated by probabilistic noise, allowing the analysis of responses that break mathematical principles such as transitivity. The objective is to minimize the expected posterior loss of this classifier by adaptively and sequentially selecting the best subset of alternatives to present to the user. We consider multiple forms of loss, most notably including posterior entropy, and provide results characterizing these best subsets. We develop theory yielding tight bounds for measures of loss under the optimal policy, show how computationally tractable algorithms can be implemented, and analyze these policies in two unique computational scenarios.

BIOGRAPHICAL SKETCH

Stephen Pallone was born on March 21, 1991 in Summit, New Jersey, and grew up in the suburban town of Mendham.

In the summer of 2009, he began his undergraduate studies at Johns Hopkins University, working for a double major in Applied Mathematics and Economics. He spent a wonderful three years in Baltimore and met the love of his life while working as a statistics tutor. Seeing how well studying math had worked out so far, he decided to invest another five years to pursue a doctoral degree. After graduating in 2012, he started his graduate studies in the school of Operations Research and Information Engineering at Cornell University. In his spare time, he could be found playing improvisational piano, singing with the Cornell University Glee Club, playing bar trivia, and exploring new places with Christina.

In August 2017, he will join Uber in San Francisco, California to work as a data scientist. He will fondly miss the days on the hill but is eager to start a new chapter.

Dedicated to Grandpa Pat, who knew there are more important things to do in life than write a thesis.

ACKNOWLEDGEMENTS

First, I would like to thank my committee members Peter Frazier, Shane Henderson, Adrian Lewis, and Thorsten Joachims for their useful feedback on the thesis and the overall candidacy process. I especially appreciate Shane and Peter for being patient, insightful, and all-around amazing advisors.

The entire field of ORIE has been wonderful over the past five years. In particular, there are many ORIE Ph.D. students (and one post-doc) I would thank for being enjoyable to work and hang out with, mostly the latter.

James Dong	Cory Girard	Julian Sun
James Davis	Jake Feldman	Zach Rayfield
Alice Paul	Weici Hu	Andrew Daw
Ravi Kumar	Kenneth Chong	Emily Fischer
Venus Lo	Calvin Wylie	Mika Sumida
Chamsi Hssaine	Amy Zhang	Pamela Badian-Pessot
Angela Zhou	Ben Grimmer	Matthias Poloczek

All of you made this journey into the incredible ride that it was, and I hope you continue to do this for future ORIE Ph.D. students.

When not in Rhodes Hall, I was often in Sage Chapel singing with the Cornell Glee Club and Chorus. There are too many of you to list here, but thank you all for the memorable experiences here in Ithaca and elsewhere on tours, and for helping to keep music in my life for the past five years. In particular, I have gratitude for John Rowehl, Robert Issacs, and Steve Spinelli for allowing me to make music and memories with such talented and passionate groups. I will always look forward to coming back and singing on the hill.

I would like to thank my family for supporting me throughout my life, and especially these past eight years while I was away at school. You made me into

the person I am today, and I am thankful for everything you have done and will do to allow me to become the best version of myself. Lastly, thank you Christina for always being there. Even though I will fondly remember these years at Cornell, I am looking forward to spending the rest of my life together with you.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

The problem of deterministic search is fundamental to the field of optimization, as solving a constrained continuous optimization problem is equivalent to finding a feasible point in a particular level set. For example, given some constraint set $D \subset \mathbb{R}^d$, minimizing a function $f : \mathbb{R}^d \to \mathbb{R}$ to tolerance ϵ is equivalent to finding some $\bar{x} \in \mathcal{L}(D, \epsilon)$, where

$$\mathcal{L}_f(D, \epsilon) = \left\{ x \in D : f(x) - \inf_{z \in D} f(z) \leq \epsilon \right\} \tag{1.1}$$

corresponds to an ϵ level set of f. This optimization approach requires an oracle that separates non-optimal and non-feasible points from the rest. In the paradigm of convex optimization, $\mathcal{L}_f(D, \cdot)$ is a convex set, and therefore, a subgradient vector provides a separating hyperplane that allows one to exclude a halfspace from the search. This directly translates into cutting plane methods that are fundamental to complexity theory such as the center of gravity method and the ellipsoidal method (see Nemirovskii, Yudin, and Dawson 1983, for a comprehensive review).

One of the simplest examples of these kinds of algorithms is deterministic bisection search, used to find the zero of a one-dimensional monotonic function. At iteration n, an interval $[a_n, b_n)$ containing the desired point x_* is maintained, and the function is evaluated at midpoint of this interval. Depending on the sign of the result, one can find the subinterval containing the desired point. The routine is repeated with this subinterval, while the other is eliminated from the

search.

Bisection can be interpreted as a sequential, Bayesian optimal learning algorithm, where the location of the optimal solution X_* is treated as a random variable that is distributed uniformly on the interval $[a_n, b_n)$. From this interval, a query x_{n+1} is selected, one finds a new interval $[A_{n+1}, B_{n+1})$ that depends on the outcome of this query. Here, A_{n+1} and B_{n+1} are respectively the left and right endpoints of the next subinterval. These are X_*-measurable random variables that also depend on the query x_{n+1}. Specifically,

$$[A_{n+1}, B_{n+1}) = \begin{cases} [a_n, x_{n+1}) & x_{n+1} > X_* \\ [x_{n+1}, b_n) & x_{n+1} \leq X_*. \end{cases} \tag{1.2}$$

For a given loss function ℓ that measures dispersion of this interval, the objective is to minimize expected loss by adaptively choosing optimal queries at every iteration, or in mathematical terms,

$$\min_{x_{n+1} \in [a_n, b_n)} \mathbb{E}_n \left[\ell \left[A_{n+1}, B_{n+1} \right) \right], \tag{1.3}$$

where the expectation is taken with respect to a uniform distribution on the current interval $[a_n, b_n)$. In cases where the loss function ℓ is convex with respect to interval length, the expected loss becomes a symmetric, convex function in x_{n+1} over the interval $[a_n, b_n)$, and therefore, an optimal query point lies at the midpoint of that interval, justifying bisection method from a Bayes-optimal point of view.

The interval at each time epoch represents the current knowledge of the location of the desired point. This notion of knowledge can be generalized even further. One can envision a prior distribution p_n over the interval $[a_n, b_n)$ that encodes where the point is more or less likely to be located. For the ordinary

bisection method, this prior is uniform over the entire length of the subinterval, but in many applications, there can be circumstances for believing the value of X_* is not uniformly distributed *a priori*. In the case of deterministic search, the likelihood function g_{n+1} can be one of two different indicator functions,

$$g_{n+1}(t) = \begin{cases} \mathbb{I}(t < x_{n+1}) & x_{n+1} > X_* \\ \mathbb{I}(t \geq x_{n+1}) & x_{n+1} \leq X_*, \end{cases} \tag{1.4}$$

depending on the result of query x_{n+1}. The posterior is then updated, defining $p_{n+1} \propto p_n \cdot g_{n+1}$. Since g_{n+1} is X_*-measurable, so is the posterior p_{n+1}. The loss function ℓ now becomes an operator on probability distributions that returns a measure of dispersion, and the goal is to minimize the expected dispersion of the posterior distribution p_{n+1}, choosing the query to satisfy

$$\inf_{x_{n+1} \in \text{supp}(p_n)} \mathbb{E}_n \left[\ell \left(p_{n+1} \right) \right]. \tag{1.5}$$

where the expectation is taken with respect to prior p_n, and query $x_{n+1} \in \text{supp}(p_n)$ is chosen to be a value in the support of distribution p_n.

The performance of such policies depends on maximizing the amount of information content of each query, which strongly depends on problem variants such as the set of feasible queries and the observation model. In full generality, there are many variants of such search problems that greatly affect the structure of such optimal policies. One can envision a scenario where query outcomes are contaminated by noise and non-deterministic. The approach would be nearly identical to (1.5), except the structure of likelihood g_{n+1} would be more complex. For instance, Waeber (2013) tackles a stochastic root-finding problem where noise does not vary within subintervals, implying that g_n is piecewise constant. Search problems are also common in multidimensional spaces, arising in areas such as pattern recognition (Huo, Chan, and Lee 1995; Bruzzone

and Prieto 2002; Jedynak, Frazier, Sznitman, et al. 2012), experimental design (Thall, Simon, and Estey 1995; Kontsevich and Tyler 1999; Chevret 2012), decision theory (Weitzman 1979; Cooper, Subramanian, and Torczon 2001), and preference elicitation (Houlsby et al. 2011; Yu, Goos, and Vandebroek 2012).

This thesis focuses on several search problems, showing structural results and deriving Bayes-optimal policies under various loss functions for these different scenarios. In particular, the work in this thesis falls under two variations of such search problems. Chapters 2 & 3 focus on variants of one-dimensional root-finding problems, whereas Chapters 4 & 5 study a multidimensional stochastic search problem applied to learning the preferences of a single user. Although these problems stem from similar ideas, they are different enough that the notation used for each problem is distinct.

1.2 Variants of One-Dimensional Search

The first half of the thesis focuses on two problems tackling one-dimensional bisection search with more complex query mechanisms. In Chapter 2, we consider a variant of bisection where multiple queries can be made at once and queries require time to evaluate. Evaluating multiple queries at once splits a given interval into many subintervals, hence the name *multisection*. The objective in this scenario is to find a policy that jointly and adaptively assigns queries to learn the root of a monotonic function as efficiently as possible.

To study this, we formulate the query allocation problem as a dynamic program, where the objective is to minimize the expected loss of the containing interval at a random stopping time, which models the importance of computa-

tional budget. We consider convex loss functions that model various degrees of risk aversion.

When query computation times are deterministic and equal for all possible points, we show the optimal policy is to split the interval into equal subintervals, always reducing length by a constant factor, regardless of computational budget and risk aversion.

However, this policy is not necessarily optimal when the computation times are i.i.d. random quantities. Because the outcome of certain queries become known before that of others, the value of information of unfinished queries can instantly change. This interjects enough uncertainty in the flow of information that risk aversion and computational budget become a significant factor. If computational budget is low or risk aversion is high, optimal policies tend to place new queries closer to the midpoint of the current interval, regardless of the locations of other queries. On the other hand, if the computational budget is large or if risk aversion is low, then new queries tend to be placed more strategically so that the information value of any query is not affected by any other queries that complete beforehand.

The most pronounced case of this are policies that make query allocations to use a minimal number of computational states, implying that information flow is relatively constant. We call these *self-similar policies*. Among the most notable of these is the *Golden Section policy*, named for an algorithm that makes use of the same type of query allocation. Given the ability to make two queries at once, it uses the golden ratio to place queries such that the same query allocation can always be made, regardless of whichever query finishes first. We show that self-similar policies are optimal under risk-neutrality, and conjecture that they

are optimal under risk-aversion when the computational budget is sufficiently large.

Chapter 3 considers a different kind of one-dimensional search problem. In this scenario, there are multiple monotonically increasing functions defined on the interval $[0, 1)$ with corresponding zeros. These functions are "coupled" in that they must be evaluated simultaneously for a given argument. Most notably, this problem setting manifests with the computation optimal Gittens and Whittle index policies when solving bandit problems. The zeros correspond to optimal Lagrange multipliers for the value function evaluated at n different states, and the value function must be evaluated for all n states at once for a single multiplier. Ordinarily, if the goal were to find n roots to a tolerance of ϵ, this would take $n \log_2 1/\epsilon$ function evaluations. However, the objective is find the correct *ordering* of the corresponding roots, and this can be done in substantially less time.

The routine can be described as follows: given an interval with n unsorted roots, query a point within the interval to split the interval into two smaller subintervals. Each of these subintervals will contain a proportion of the n roots. If a subinterval contains at most one root, it is fully sorted and no further action is required. Otherwise, recursively run the routine on all subintervals that contain unsorted roots. Since each subinterval is sorted with respect to the other subintervals, it is only necessary for each root to be contained within its own subinterval to derive a total ordering.

The objective is to find a querying policy that minimizes the expected number of evaluations needed to find a complete ordering of the roots. We derive structural results to show that the querying policy only depends on the number

of unsorted roots within a given interval, and it is only necessary to consider the original case where all n roots are contained within a single interval.

To model this, the roots are modeled to be i.i.d. uniformly on the unit interval. Under these assumptions, the problem of choosing a querying strategy can be represented as a one-dimensional dynamic program, which can be solved numerically. The most notable result is that the bisection policy is not optimal, and in fact, there are instances where querying away from the midpoint leads to a slightly faster expected computational time. Nonetheless, we show that the bisection policy achieves a linear time complexity with respect to the number of zeros, which is optimal, and the rate is almost indistinguishably close to optimal. This bisection routine sorts these n roots with an expected $1.44n$ function evaluations, which is much more efficient than splitting the interval into an ϵ-grid.

1.3 Multidimensional Search for Preference Learning

The latter half of the thesis considers a different kind of search problem. Chapters 4 & 5 focus on learning a user's preferences. The type of queries we consider are comparative questions in which the user is presented with a set of alternatives and is asked to choose the most preferred option. The objective is to adaptively select the subset of alternatives at every step to maximize the amount of information learned about a user's preferences.

We model a user's choice model using a linear classifier $\theta \in \mathbb{R}^d$. Each alternative has a corresponding feature vector in $\mathbb{R}^d$ that encapsulates all the distinguishing properties. Roughly speaking, the model predicts a user to prefer al-

ternative A over alternative B if $\theta^T x_A \geq \theta^T x_B$. Therefore, when we pose queries to the user, their response implies a set of linear inequalities satisfied by their linear classifier. Through these sequential queries, we seek to learn a user's linear classifier θ as efficiently as possible.

Unlike the previous search problems studied in this thesis, the search of θ takes place on a higher dimensional space, which presents its own set of challenges. Halfspaces in one dimension can only take two directions: facing towards positive infinity or facing towards negative infinity. However, in $\mathbb{R}^d$ space, halfspaces can take any number of directions on a $(d-1)$-dimensional sphere. Now when considering separation oracles, the direction of the separating hyperplane has a critical effect on the amount of partitioned mass on each side. This is often a challenge in cutting plane methods for higher dimensional convex optimization, where the direction of the gradient is unknown *a priori*. In this setting, however, the hyperplanes always pass through the origin, and we are tasked with choosing the alternatives that imply a specific direction, presenting us with a geometrically interesting yet manageable problem.

Another complication is human behavior: there are times when one's responses to preferential questions contradict, or otherwise violate mathematical principles such as transitivity. Any reasonable choice model needs to be able to handle these occurrences. One way to do this is to introduce probabilistic uncertainty into the model. Our approach involves sending the model-consistent response through a *noise channel*, and only this perturbed response is observed. In other words, conditioned on the alternative with the highest dot product with respect to the user's linear classifier, the observed response is chosen according to a fixed probability distribution. This allows us to handle responses that con-

tradict themselves according to the model.

To accommodate this, we take an approach similar to (1.5) by maintaining a prior distribution for linear classifier θ and update it after every user response. The loss function we largely focus on throughout the Chapters 4 & 5 is differential entropy, which generalizes the concepts behind cutting plane methods. Differential entropy of the prior density p_n is defined as

$$H\left(p_n\right) = \mathbb{E}_n\left[-\log_2 p_n(\theta)\right]. \tag{1.6}$$

When the distribution is uniform on a compact set, the corresponding differential entropy is equal to the logarithm of the set's volume. Thus, learning the linear classifier θ amounts to asking comparative questions that cuts the most probabilistic mass at every step. Another benefit of using entropy as a measure of loss is the massive foundation supporting it called *information theory* (see Cover 1991, for a comprehensive review) that can be used to derive structure for optimal querying policies and other useful results.

Taking this approach, we apply fundamental results in information theory to find how to choose the sets of alternatives. The optimal querying policy strongly depends on the noise channel. One common instance is when a noise channel does not take the order of alternatives into account, which is referred to as a *symmetric noise channel*. In the case of a symmetric noise channel, an optimal policy involves selecting alternatives that the user is equally likely to select *a priori*. This result is reminiscent of classic bisection method, but our result handles more complicated noise channels as well. In all such cases, there is a specific predictive distribution for user responses that maximizes differential entropy reduction. Finding this predictive distribution only involves solving one concave maximization problem, with dimensionality equal to the desired cardinal-

ity of the offered subset. Further, we show that the optimal entropy reduction policy is a greedy one. This policy is referred to as *entropy pursuit*. We provide a tight linear lower bound for entropy with respect to the number of questions asked.

The remaining difficulty lies in selecting the correct subset of alternatives that induces the optimal predictive distribution. In reality, the set of alternatives is finite but massive, and finding such a subset is very much a combinatorial and arduous problem. We show that presenting a subset with a predictive distribution approximately equal to optimal results in an approximately optimal decrease in entropy. Thus, it is possible to use subsampling to find a query that is sufficiently close to optimal. In the continuum case, where the set of alternatives has a non-empty interior, we show it is possible to construct a query with the desired predictive distribution, so long as the predictive distribution is reasonably diffuse relative to the prior density. A corollary is that the entropy pursuit policy achieves the optimal rate for a symmetric noise channel in the continuum case, and otherwise reduces entropy in a linear fashion, albeit with a non-optimal rate.

For all the benefits of using differential entropy as a measure of loss, the metric itself is difficult to interpret. We present another metric to compare against called *misclassification error*. After asking the user a number of comparative queries, we then are presented with a question ourselves, and the goal is to minimize the probability of answering incorrectly. Even though this metric is easier to interpret, finding the optimal querying policy to minimize the expected loss is intractable. Instead, we compare the optimal entropy reduction policy with a greedy one-step *knowledge gradient policy* that selects queries as if an assessment

question was posed immediately after. This one-step policy is still much more expensive to compute than entropy pursuit because it requires comparing questions against other questions in a combinatorial fashion. Nevertheless, we show that a linear function of differential entropy bounds the misclassification error from below. Further, we prove minimizing misclassification error is equivalent to minimizing a penalized version of differential entropy.

While Chapter 4 explores the theory behind different query policies, Chapter 5 studies the implementation of necessary components for finding approximate policies and evaluating their effectiveness. The first section delves into sampling from the posterior distributions generated by the previously defined noise model. We use a geometrically-inspired sampler called hit-and-run, and show exactly how it can be used to sample from mixed Normal priors updated with weighted piecewise uniform likelihoods. We then explore ways in which a sampler can be warm-started by leveraging the statistical properties of a similar sampler. This process is called "hot starting," and we apply it to when the posterior is updated and when the noise parameters of the posterior are adjusted. We then focus on two computational examples. The first example uses academic papers as real alternatives, but otherwise simulates user responses. In this scenario, we show entropy pursuit performs similarly to its misclassification error counterpart, even before considering the computational cost of both policies. The second scenario uses historical user interaction with lists of academic papers. Although we are not free to adaptively query the user, we attempt to passively learn the linear classifiers of tens of thousands of users and the noise channel parameters simultaneously using a Gibbs sampler. We show that the predictive power in this scenario is moderate at best, but pose that further investigation is needed to find if using fully adaptive query strategies under this

model can work well in real applications.

REFERENCES

[1] Lorenzo Bruzzone and D Fernandez Prieto. "An adaptive semiparametric and context-based approach to unsupervised change detection in multitemporal remote-sensing images". In: *IEEE Transactions on image processing* 11.4 (2002), pp. 452–466.

[2] Sylvie Chevret. "Bayesian adaptive clinical trials: a dream for statisticians only?" In: *Statistics in medicine* 31.11-12 (2012), pp. 1002–1013.

[3] Keith D Cooper, Devika Subramanian, and Linda Torczon. "Adaptive optimizing compilers for the 21st century". In: *The Journal of Supercomputing* 23.1 (2001), pp. 7–22.

[4] Thomas M Cover. *Elements of information theory*. John Wiley, 1991.

[5] Neil Houlsby et al. "Bayesian active learning for classification and preference learning". In: *arXiv preprint arXiv:1112.5745* (2011).

[6] Qiang Huo, Chorkin Chan, and Chin Hui Lee. "Bayesian adaptive learning of the parameters of hidden Markov model for speech recognition". In: *IEEE Transactions on Speech and Audio Processing* 3.5 (1995), pp. 334–345.

[7] Bruno Jedynak, Peter I Frazier, Raphael Sznitman, et al. "Twenty questions with noise: Bayes optimal policies for entropy loss". In: *Journal of Applied Probability* 49.1 (2012), pp. 114–136.

[8] Leonid L Kontsevich and Christopher W Tyler. "Bayesian adaptive estimation of psychometric slope and threshold". In: *Vision research* 39.16 (1999), pp. 2729–2737.

[9] Arkadii Nemirovskii, David Borisovich Yudin, and Edgar Ronald Dawson. “Problem complexity and method efficiency in optimization”. In: (1983).

[10] Peter F Thall, Richard M Simon, and Elihu H Estey. “Bayesian sequential monitoring designs for single-arm clinical trials with multiple outcomes”. In: *Statistics in medicine* 14.4 (1995), pp. 357–379.

[11] Rolf Waeber. *Probabilistic bisection search for stochastic root-finding*. Cornell University, 2013.

[12] Martin L Weitzman. “Optimal search for the best alternative”. In: *Econometrica: Journal of the Econometric Society* (1979), pp. 641–654.

[13] Jie Yu, Peter Goos, and Martina Vandebroek. “A comparison of different Bayesian design criteria for setting up stated preference studies”. In: *Transportation Research Part B: Methodological* 46.7 (2012), pp. 789–807.

CHAPTER 2

MULTISECTION: PARALLELIZED BISECTION

Abstract

We consider a one-dimensional bisection method for finding the zero of a function, where function evaluations can be performed asynchronously in a parallel computing environment. Using dynamic programming, we characterize the Bayes-optimal policy for sequentially choosing points at which to query the function. In choosing these points, we face a trade-off between aggressively reducing the search space in the short term, and maintaining a desirable spread of queries in the long-term. Our results provide insight on how this trade-off is affected by function evaluation times, risk preferences, and computational budget.

2.1 Introduction

Parallel computing platforms are now ubiquitous, no longer being restricted to the high-performance computing environments used for massive-scale computation. Platforms that are broadly accessible include cloud computing, local computer clusters, and personal computers containing multicore processors. These platforms have the potential to allow us to more rapidly solve existing formulations of optimization problems, or to solve much larger, in the sense of computational requirements, formulations, provided that we can devise algorithms that appropriately exploit parallelism.

Perhaps the simplest optimization problem is to find the zero (assumed to exist and be unique) of a continuous function f that is positive to the right of the root and negative to the left of the root on the interval $(0, 1)$, where f is known only through a computational procedure that returns $f(x)$ when queried at $x \in (0, 1)$. This problem can be viewed as an optimization problem since f can be interpreted as the derivative of a smooth unimodal function that attains its minimum in $(0, 1)$. We minimize that unimodal function by seeking a zero of its derivative. Even for this problem it is not clear how to exploit parallel computing, and thus this problem is our starting point and the subject of this chapter.

A natural sequential approach for solving root-finding problems of such functions is bisection, in which an interval containing the root is successively reduced through evaluations of the function at the midpoint of the interval. When multiple cores are available to compute f, we might generalize bisection in such a way that we evaluate the function at multiple points simultaneously. At which points should we evaluate the function in order to rapidly reduce the interval in which we know the root lies? Throughout this chapter we assume an idealized model of communication, whereby all information is available to all processors.

We use dynamic programming, in which the state of the dynamic program encapsulates the interval in which the root is known to lie and the points that are still being evaluated by cores, to attempt to identify optimal policies. The dynamic program uses a reward function that is a function of the width of the smallest interval we are certain contains the root. The dynamic program only uses the *sign* of the function values, rather than the values themselves, which

keeps the calculations tractable. While using the function values themselves makes sense from an intuitive standpoint, we do not know how to formulate and solve a tractable dynamic program in that setting.

First suppose that the time required to perform a function evaluation $f(x)$ is deterministic and does not depend on x. This abstraction seems reasonable if the time to complete a function evaluation is not too variable, and leads one to consider algorithms that are, in some sense, synchronous, in that the cores operate iteratively, where the next iteration is not initiated until all cores complete processing from the current iteration. In this setting, with n cores, it is perhaps natural to evaluate the function at n equally spaced x values in the interior of the interval. Since all cores complete processing simultaneously, the interval is reduced at each stage by a factor of $(n+1)^{-1}$. This indeed turns out to be optimal when the root is initially assumed to be equally likely to be anywhere in the interval, as we assume throughout this chapter.

In general, function evaluation times may depend on x and may be random. We do not address the former issue in this chapter, which admittedly limits the scope of our analysis, but with good reason; identifying the optimal locations at which to evaluate a function would likely be very difficult if the time to compute $f(x)$ varies in some complicated manner with x. However, we do address the latter issue.

When processing times are random, cores do not necessarily complete processing at the same time, and one is naturally drawn to algorithms that are asynchronous, in that they do not proceed in lockstep as with synchronous algorithms. When one core reports a function value, it enables us to reduce the interval in which the root is contained, and this will render function evalua-

tions proceeding at some of the other cores redundant. This makes the question of where to evaluate the function so as to rapidly reduce the interval a highly nontrivial one. We make the additional assumption that processing times are independent and identically exponentially distributed, which makes the analysis tractable.

The "memoryless" property of the exponential distribution ensures that the residual processing-time distribution is unchanged if we re-assign cores to different x values before they complete processing, so that nothing is lost in doing so. Furthermore, the minimum of independent exponential random variables is also exponential with a reduced mean, so one can obtain solutions more rapidly by "stacking" cores on a single x value. These properties are very special, and so while we do briefly consider them as being approximations of reality when processing times are extremely variable, as might arise in a cloud-computing environment where cores are shared by other users that are not visible to us, the main part of our analysis does not exploit them.

Our main findings in the random-processing-time context are two-fold.

- If stacking is not allowed, and one has only two cores available, then a policy that is similar to golden-section search is optimal under a specific risk-neutral reward function, and is close to optimal for others.
- When stacking *is* allowed, then for all risk preferences and computational budgets, it is always optimal to stack queries at the midpoint of the interval.

When we parallelize bisection, as in the present chapter, multiple points are simultaneously evaluated. Accordingly, the term "multisection" seems appro-

priate as a descriptor of algorithms like those considered here, hence the title of the chapter. We henceforth refer to parallelized bisection algorithms as "multisection algorithms" and the associated problem in deciding where to query the function as the "multisection problem."

The remainder of this chapter is organized as follows. In Section 2.2 we define a stochastic process that tracks the progress of multisection. In Section 2.3 we introduce a dynamic programming recursion that allows us to characterize some important properties of an optimal policy. Section 2.4 computes the optimal policy when processing times are deterministic and equal on all cores. Section 2.5 develops our primary results for random processing times in both the "no-stacking" case and the case where stacking is permitted.

2.2 Problem Specification

We model the allocation problem using dynamic programming and Bayesian statistics. A priori, we take the root $X^* \sim \text{Unif}(0,1)$. Observing the value of the function f at points within $(0,1)$ allows us to localize X^* more precisely. We keep track of the smallest interval in which X^* is known to reside, and those $m \leq n$ points within this interval currently being evaluated, through an ordered tuple of real numbers $S = (S^{(0)}, S^{(1)}, \ldots, S^{(m)}, S^{(m+1)})$. (See Section 2.5.2 for an example.) In this tuple, the first and last values, $S^{(0)}$ and $S^{(m+1)}$, describe the smallest interval $[S^{(0)}, S^{(m+1)}]$ in which X^* is currently known to reside, and $S^{(1)}, \ldots, S^{(m)}$ are points at which we are currently evaluating f. Without loss of generality, we assume $S^{(k)} \leq S^{(k+1)}$. We construct a dynamic program, and a tuple of this form will be our state variable. When m, the number of points

currently being evaluated, is strictly less than n, the number of available cores, this tuple or state indicates that we may assign idle cores to evaluate f at new points. We call such states "incomplete states." The set of incomplete states is contained in the set

$$\mathbb{S} = \{(s_0, s_1, \ldots, s_m, s_{m+1}) : 0 \leq m \leq n,\ s_k \leq s_{k+1} \text{ for } 0 \leq k \leq m\}.$$

We track the state variable over time. While our problem is a continuous-time problem, the state variable only changes at moments in time when a core finishes a function evaluation. These are also the moments in time when we make decisions about how to allocate newly idle cores to new points. We index this discrete set of decision epochs by j, and indicate the (pre-decision) state variable at the beginning of the jth epoch, before any idle cores are re-allocated, by $S_j = (S_j^{(0)}, \ldots, S_j^{(m_j+1)})$, where m_j is the number of active cores at the beginning of the jth decision epoch. We respectively denote the left and right endpoints of the search interval S_j as $A_j = \min(S_j)$ and $B_j = \max(S_j)$.

Let the time at which the jth decision epoch occurs be $\sigma(j) \geq 0$. The first decision epoch is indexed by $j = 0$, and starts at time $\sigma(0) = 0$. The state variable at this moment in time is S_0. Typically we consider the case $S_0 = (0, 1)$, but we also consider other values for S_0 in a dynamic program constructed below. At each decision epoch j, we make a decision about how to allocate our idle cores according to some decision rule or policy.

Mathematically, we define a policy π to be a function $\pi : \mathbb{S} \to \mathbb{X}(\mathbb{S})$ that maps states to actions, where the action is an ordered tuple $X = (x_0, x_1, \cdots, x_n, x_{n+1}) \in \mathbb{S}$ of the same form as the state variable, which assigns all n cores to evaluate $f(x_1), \ldots, f(x_n)$. We require that this action assign all n cores, leave unchanged those cores that are still actively performing function evalua-

tions, and not alter the interval in which X^* is known to reside. To achieve this, we define the space of allowed actions for a given state variable (also called the "action space") as

$$\mathbb{X}(S) = \left\{ (x_0, x_1, \ldots, x_n, x_{n+1}) : \begin{array}{l} X \supseteq S,\ x_k \leq x_{k+1} \text{ for } 0 \leq k \leq n \\ x_0 = \min(S),\ x_{n+1} = \max(S) \end{array} \right\}. \quad (2.1)$$

The set of allowed actions at the first decision epoch is $\mathbb{X}(S_0)$. The constraint $X \supseteq S$ ensures that we do not disturb active function evaluations. The constraints $x_0 = \min(S)$ and $x_{n+1} = \max(S)$ ensure that we leave unchanged the smallest known interval containing X^*. Changing this interval requires observing the results from function evaluations.

At this first decision epoch, we also draw, for each newly assigned core $i = 1, \ldots, n$, an independent random variable $\delta_0^{(i)} \sim F$, where F is some known probability distribution with support on $(0, \infty)$. The random variable $\delta_0^{(i)}$ is the wall-clock time at which the job just started on core i will complete. We let $\delta_0 = (\delta_0^{(1)}, \ldots, \delta_0^{(n)})$. Our model allows early termination when the point being evaluated is eliminated from the interval containing the root by another newly completed function evaluation, and so function evaluations do not always run for the full time period $\delta_0^{(i)}$. In this chapter, we consider two possibilities: either F takes a single value with probability one, or F is an exponential distribution.

Given these times, the first time at which a function evaluation will complete is $\sigma(1) = \min_i \delta_0^{(i)}$. Over the time period $(0, \sigma(1))$, the state of the cores is described by $\pi(S_0)$. Let $Q_1 = \arg\min_{i=1,\ldots,n} \delta_1^{(i)}$ be the jobs that complete at time $\sigma(1)$. More than one job may finish at a time. We observe function values for all points $(S_0^{(i)} : i \in Q_1)$. These cores become momentarily idle, and the new observations of f reduce the interval in which X^* is known to reside, and may

also eliminate some points not in Q_1 that are actively being evaluated. This then produces a new incomplete state variable, which we call S_1. Below we describe the conditional distribution of S_1 given $\pi(S_0)$ and Q_1. We then choose the next action using our policy, evaluating the points implied by $\pi(S_1)$.

We proceed iteratively in this fashion, constructing a sequence of random variables $(S_j, \delta_j, \sigma(j) : j = 0, 1, \ldots)$. At each decision epoch $j = 1, 2, \ldots$, let $Q_j = \arg\min_{i=1,\ldots,n} \delta_{j-1}^{(i)}$ be the jobs that complete at the beginning of this decision epoch, and let $\sigma(j) = \min_{i=1,\ldots,n} \delta_{j-1}^{(i)}$ be the wall-clock time at which this decision epoch occurs. At the beginning of this decision epoch, we draw S_j from its conditional distribution given S_{j-1} and Q_j, as described below. We then choose an action $\pi(S_j)$ that allocates all newly idle cores. For those cores that are still performing a previous function evaluation, we set $\delta_j^{(i)}$ to the previous value, which is $\delta_{j-1}^{(i')}$ for some i'. However, if a core is newly allocated, then we generate a new iid processing time for the new function evaluation, drawing $\delta_j^{(i)}$ such that $\delta_j^{(i)} - \sigma(j) \sim F$.

In this way, each policy implies a probability distribution P^π over the sequence of random variables $(S_j, \delta_j, \sigma(j) : j = 0, 1, 2, \ldots)$. Let $\mathbb{E}^\pi$ be the expectation operator corresponding to this probability distribution. Define the policy space Π to contain all policies $\pi : \mathbb{S} \mapsto \mathbb{X}(\mathbb{S})$ satisfying $\pi(S) \in \mathbb{X}(S)$ for all $S \in \mathbb{S}$. We show later that it suffices to define a policy on a much smaller domain.

The state variable does not change between decision epochs, taking the value $X_j = \pi(S_j)$ in the time interval $(\sigma(j), \sigma(j+1))$.

We now describe the conditional distribution of S_j under $\mathbb{P}^\pi$ given S_{j-1} and Q_j. For each core $i \in Q_j$ that finishes at the beginning of the jth epoch, we

observe $f\left(X_{j-1}^{(i)}\right)$. From the sign of this function value, we infer whether $X^* \leq X_{j-1}^{(i)}$ or $X^* > X_{j-1}^{(i)}$. We use this information to update the search interval and make it as small as possible. Accordingly, let

$$L_j = \left\{ i \in Q_j : X^* \geq X_{j-1}^{(i)} \right\} \quad \text{and} \quad U_j = \left\{ i \in Q_j : X^* \leq X_{j-1}^{(i)} \right\}. \tag{2.2}$$

For each core $i \in L_j$, the results from the most recent function evaluations tell us that $X^* \geq X_{j-1}^{(i)}$. Let ℓ_j be largest such index, i.e., let $\ell_j = \max L_j$ if L_j is nonempty, and otherwise set $\ell_j = 0$. Likewise, set $u_j = \min U_j$ if U_j is nonempty, and otherwise set $u_j = n + 1$.

We terminate a function evaluation at a point $X_{j-1}^{(i)}$ that has not finished before decision epoch j if the point is cut off by some point in Q_j, i.e., if $i < \ell_j$ or $i > u_j$. Any other jobs that have not yet completed will be allowed to run until at least the next epoch. The state S_j is then a tuple describing the new interval in which X^* is known to reside, and the jobs that are still running at time $\sigma(j)$. Formally, $S_j = (X_{j-1}^{(i)} : i = \ell_j, \ldots, u_j)$. The updated search interval is $[A_j, B_j] = [\min S_j, \max S_j]$.

Conditioning on which jobs complete at decision epoch j, we can find the probability of certain outcomes for (A_j, B_j) using the uniform prior on X^*. Given Q_j and X_{j-1}, (A_j, B_j) is conditionally independent of S_k and δ_k for all $k < j$, and

$$\mathbb{P}\left((A_j, B_j) = \left(Q_j^{(i-1)}, Q_j^{(i)} \right) \,\middle|\, Q_j,\, X_{j-1} \right) = \frac{Q_j^{(i)} - Q_j^{(i-1)}}{B_{j-1} - A_{j-1}}.$$

We now define a reward function $R : \mathbb{S} \to \mathbb{R}$ that measures progress in reducing the interval. For any state S_j, let $\|S_j\| = B_j - A_j$ be the length of the search interval. We choose the reward function $R(S) = \|S\|^{-r}$, where $r \in (0, 1]$. The parameter r controls our risk preference. As $r \to 0$ we become more risk

averse, because there is less gain for making the interval smaller. Choosing $r = 1$ minimizes risk aversion.

We choose a time horizon T, and seek to make decisions so that the reward function applied to the state at time T is large. Conceptually, one could use a fixed value for T, or allow T to be a random variable, modeling the situation in which one is uncertain when starting a computation about how long we will run it before asking it to produce an answer. For tractability, we assume that T is exponentially distributed with rate parameter α, independent of all else.

Let $\tau = \sup\{j : T \geq \sigma(j)\}$ so that the random time T falls in the interval $[\sigma(\tau), \sigma(\tau+1))$. Then the interval to which X^* has been localized at time T is $[A_\tau, B_\tau]$, which has associated reward $R(S_\tau)$.

The value, i.e., expected reward, attained by policy $\pi \in \Pi$ is $V^\pi = \mathbb{E}^\pi R(S_\tau)$. We want to find a policy that maximizes the expected reward. Let

$$V = \sup_{\pi \in \Pi} V^\pi. \tag{2.3}$$

An *optimal policy* is any policy π that attains the supremum in (2.3). An *ϵ-optimal policy* is any policy π such that $V^\pi \geq V - \epsilon$ for some given $\epsilon > 0$.

2.2.1 Deterministic and Equal Waiting Times for Queries

We first assume that it takes a deterministic and constant time to evaluate the function at any point in $[0, 1]$. That is, F represents a point mass at some $C > 0$. This assumption is reasonable when the variability of computation times is small. The wait time is independent of where the function is being evaluated, which may not reflect reality, but makes the model more tractable. Because

every task takes the same amount of time to complete, $Q_j = \{1, \ldots, n\}$ for all j, and hence all n cores are reallocated at each decision point and $\sigma(j) - \sigma(j-1) = C$ for all j. This predictability makes the state space smaller and the optimal allocation easier to identify. The deterministic case will serve as a benchmark for the exponential case.

2.2.2 Exponential Waiting Times for Queries

What is the effect of variability of function evalution times on the optimal allocation of points? We model this variability by assuming the evalution-time distribution F is exponential with rate λ. The choice of exponential evaluation times is a computational convenience that admits evaluation uncertainty while retaining tractability. Since there is probability zero of any two independent exponential random variables taking the same value, Q_j is a singleton with probability one. The memoryless property of the exponential distribution ensures that for every j and for every $i = 1, \ldots, n$, the residual processing time $\delta_j^{(i)} - \sigma(j) \sim$ exponential(λ), independent of all past history. Therefore, it is equally likely for any job to finish first, so that $\mathbb{P}(Q_j = \{i\}) = n^{-1}$ for every i. In addition, $\sigma(j+1) - \sigma(j)$ is exponentially distributed with rate $n\lambda$.

2.3 Desirable Properties of the Value Function

2.3.1 Value Function Recursion

The problem (2.3) is a stochastic control problem, and we study it using dynamic programming. We first show that under the assumptions of Section 2.2.1 or 2.2.2, the problem is an infinite-horizon discrete-time Markov decision process. Under either assumption the stopping index τ has a geometric distribution with $P(\tau = j) = (1-\gamma)\gamma^j$ for $j \geq 0$, where γ can be easily computed. Indeed, when processing times are deterministically equal to $C > 0$, $\gamma = e^{-\alpha C}$, and when they are exponentially distributed, $\gamma = n\lambda/(\alpha + n\lambda)$. Using the fact that τ is independent of S_j, S_0, it can be shown that for any policy π,

$$V^\pi(S) = (1-\gamma)\,\mathbb{E}^\pi\left[\sum_{j=0}^{\infty} \gamma^j R(S_j) \mid S_0 = S\right].$$

Thus, except for a constant factor of $1-\gamma$, solving (2.3) is equivalent to solving the infinite-horizon discrete-time discounted-cost Markov decision process $\sup_\pi \mathbb{E}^\pi\left[\sum_{j=0}^{\infty} \gamma^j R(S_j) \mid S_0 = S\right]$. This Markov decision process is over the time-homogeneous controlled Markov process $(S_j : j = 0, 1, 2, \ldots)$, which evolves independently of $(\delta_j : j = 0, 1, 2, \ldots)$ under the assumptions of either Section 2.2.1 or Section 2.2.2.

Define the value function $V(S) = \sup_\pi V^\pi(S)$. Under the conditions of Theorem 2.3.1 below, standard results in dynamic programming (Hernández-Lerma and Lasserre 1996, p. 46) show that

$$V(S) = (1-\gamma)\,R(S) + \gamma\,\left(\max_{X \in \mathbb{X}(S)} \mathbb{E}\left[V(S_1) \mid\ X_0 = X, S_0 = S\right]\right). \tag{2.4}$$

2.3.2 Scaling and Shifting Made Easy

Intuition tells us that there should be a relationship between states that are either scaled or shifted. This would greatly reduce the size of the state space. For notational convenience, for any $h, c \in \mathbb{R}$, let $(hS + c)^{(i)} = hS^{(i)} + c$ for every i. Recall that $R(S) = \|S\|^{-r}$ for $r \in (0, 1]$. Then one can verify that $R(aS + b) = a^{-r}R(S)$. This property extends to the value function, as follows.

Theorem 2.3.1. *Suppose that $R(S) = \|S\|^{-r}$ for $r \in (0, 1]$. For $c, h \in \mathbb{R}$ with $h > 0$, the value function V has the following properties:*

1. ***Shift Invariance:*** $V(S + c) = V(S)$
2. ***Inverse Scaling:*** $V(hS) = h^{-r}\, V(S)$

Proof. We show the result by induction. For a function $f : \mathbb{S} \to \mathbb{R}$, the Bellman operator T is given by

$$T\, f(\cdot) = (1 - \gamma)\, R(\cdot) + \gamma \left(\max_{X \in \mathbb{X}(\cdot)} \mathbb{E}\left[\, f(S_{j+1}) \mid S_j = \cdot\,,\, X_j = X \right] \right).$$

We claim that the result holds for $T^{(k)}\, R$ for all $k \in \mathbb{N}$. As a base-case, consider $k = 0$. We let $T^{(0)}\, R(\cdot) = R$, so the result holds trivially. Now assume $T^{(k)}\, R$ has the above properties, and let $\bar{S} = hS + c$, $\bar{X} = hX + c$.

$$T(T^{(k)}\, R(\bar{S})) = (1 - \gamma)\, R(\bar{S}) + \gamma \left(\max_{\bar{X} \in \mathbb{X}(\bar{S})} \mathbb{E}\left[T^{(k)} R(S_{j+1}) \mid S_j = \bar{S}\,,\, X_j = \bar{X} \right] \right).$$

The transition kernel itself is shift and scale invariant. That is, $\mathbb{P}(S_{j+1} = S' \,|\, S_j = S,\, X) = \mathbb{P}(S_{j+1} = hS' + c \,|\, S_j = hS + c,\, hX + c)$. Therefore, for any measurable function g, we have $\mathbb{E}[g(S_{j+1}) \,|\, S_j = S,\, X_j = X] = \mathbb{E}[g(hS_{j+1} + c) \,|\, S_j = \bar{S},\, X_j = \bar{X}]$. We can rewrite the expectation using the process $(S_j : j \in \mathbb{N})$. Using the induction hypothesis, we see

$$T(T^{(k)}\, R(\bar{S})) = (1-\gamma)\, R(hS+c) + \gamma \left(\max_{X \in \mathbb{X}(S)} \mathbb{E}\left[T^{(k)} R(hS_{j+1} + c) \,|\, S_j = S\,,\, X_j = X \right] \right)$$

$$T^{(k+1)}\, R(\bar{S}) = h^{-r}\, (1-\gamma)\, R(S) + h^{-r}\gamma \left(\max_{X \in \mathbb{X}(S)} \mathbb{E}\left[T^{(k)} R(S_{j+1}) \,|\, S_j = S\,,\, X_j = X \right] \right),$$

i.e., that $T^{(k+1)} R(\bar{S}) = h^{-r}\, T^{(k+1)} R(S)$. Therefore, we have proved that the result holds for $T^{(k)}R$, for all k. We now wish to assert $\lim_{k\to\infty} T^{(k)} R(S) = V(S)$ for all $S \in \mathbb{S}$. According to Hernández-Lerma and Lasserre (1996, p. 46), this holds under the following three conditions.

- The action space is compact: As we defined it, the action space $\mathbb{X}(S)$ for every S is a bounded subset of $\mathbb{R}^{n+2}$ that is the intersection of a finite number of closed halfspaces; see (2.1).
- The reward R is lower semi-continuous and non-negative: We have $R(S_j) = [B_j - A_j]^{-r}$, and if we define $R(S_j) = \infty$ for states where $A_j = B_j$, then the inverse image of every open set of rewards is an open set of states, which is exactly the definition of lower semi-continuity. Hence, R is lower semi-continuous on a compact set.
- The transition kernel is continuous: The construction in Section 2.2 ensures this, although a full verification is cumbersome and thus omitted.

Therefore, $T^{(k)}R \to V$ pointwise as $k \to \infty$. Moreover, for any state S and parameters h, c, there exists K_1 and K_2 such that for all $k \geq K_1$, we have $|V(S) -$

$T^k R(S)| \leq \epsilon$, and for all $k \geq K_2$, we have $|V(hS+c) - T^k R(hS+c)| \leq \epsilon$. So for all $k \geq \max\{K_1, K_2\}$,

$$\begin{aligned}
\epsilon &\geq |V(hS+c) - T^{(k)} R(hS+c)| \\
&= |V(hS+c) - h^{-r} V(S) + h^{-r} V(S) - h^{-r} T^{(k)} R(S)| \\
&\geq |V(hS+c) - h^{-r} V(S)| - h^{-r}\epsilon,
\end{aligned}$$

which shows that the value function also has the desired properties. □

2.3.3 Normalized Scaling and State Space Reduction

Corollary 2.3.2. *For some state $S \in \mathbb{S}$, let $\bar{S} = \frac{S - S^{(0)}}{S^{(n+1)} - S^{(0)}}$ so that $\bar{S}^{(0)} = 0$ and $\bar{S}^{(n+1)} = 1$. Then from the previous theorem, we have*

$$V(S) = V\left((S^{(n+1)} - S^{(0)})\bar{S} + S^{(0)}\right) = \left(S^{(n+1)} - S^{(0)}\right)^{-r} V(\bar{S}) = R(S)\, V(\bar{S})$$

The corollary demonstrates the structure embedded in the bisection algorithm. From a theoretical standpoint, we can reduce our attention to all states within the interval $[0, 1]$. This will help us identify the structure of optimal policies.

2.4 Optimal Policy Structure for Deterministic and Equal Processing Times

Suppose that the evaluations take a deterministic amount of time to compute. If we start the first batch of function evaluations simultaneously, then the jobs

finish all at once, and all n points can be allocated. Using the recursion, we can prove that an optimal allocation at every step is for the points to be equally spaced in the interval.

Theorem 2.4.1. *Suppose the function evaluations require a deterministic and equal amount of time to compute. Let $R(\cdot)$ be defined as before, with $r \in (0,1)$. If $\tilde{X}_j = (\tilde{X}_j^{(0)}, \tilde{X}_j^{(1)}, \ldots, \tilde{X}_j^{(n)}, \tilde{X}_j^{(n+1)})$ is defined by*

$$\tilde{X}_j^{(i)} = A_j + \frac{i}{n+1}\left(B_j - A_j\right),$$

then $\tilde{X}$ is an optimal allocation for $S = (A_j, B_j)$, and this defines an optimal policy.

Proof. At decision epoch j we must allocate all n points because $Q_j = \{1, \ldots, n\}$ for all $j \geq 0$ with probability 1. Thus, it is only necessary for the state to keep track of the endpoints of the interval because there are no jobs that are left to continue. For an allocation X to be optimal for incomplete state S, it must attain the maximum in

$$\max_{X \in \mathbb{X}(S)} \mathbb{E}\, V(S) = \max_{X \in \mathbb{X}(S)} \sum_{i=0}^{n} \left(\frac{X^{(i+1)} - X^{(i)}}{X^{(n+1)} - X^{(0)}} \right) V\left((X^{(i)}, X^{(i+1)})\right).$$

Using the inverse scaling property,

$$\begin{aligned}\max_{X \in \mathbb{X}(S)} \mathbb{E}\, V(S) &= \max_{X \in \mathbb{X}(S)} \sum_{i=0}^{n} \frac{X^{(i+1)} - X^{(i)}}{\|S\|} \left(X^{(i+1)} - X^{(i)}\right)^{-r} V((0,1)) \\ &= \|S\|^{-1} V((0,1)) \max_{X \in \mathbb{X}(S)} \sum_{i=0}^{n} (X^{(i+1)} - X^{(i)})^{1-r}.\end{aligned}$$

In this maximization, $X \in \mathbb{X}(S)$ implies we can choose $S^{(0)} = X^{(0)} \leq X^{(1)} \leq \cdots X^{(n)} \leq X^{(n+1)} = S^{(n+1)}$ arbitrarily, since no jobs are ongoing. Changing variables to $u_i = X^{(i+1)} - X^{(i)}$ for $i = 0, 1, \ldots, n$, the problem becomes that of maximizing $\sum_{i=0}^{n} u_i^{1-r}$, subject to the constraints that $\sum_{i=0}^{n} u_i = \|S\|$ and $u_i \geq 0$

for every i. This is a concave maximization problem with local, and therefore global, minimum when $u_0 = u_1 = \cdots = u_n = \|S\|(n+1)^{-1}$, which corresponds to $\tilde{X}$. The maximum is unique when $r \in (0, 1)$ since the objective function is strictly concave. When $r = 1$, $\tilde{X}$ is still optimal, but it is not the uniquely optimal solution. Lastly, iteratively choosing the optimal allocation $\tilde{X}$ for every S yields an optimal policy Hernández-Lerma and Lasserre (1996, p. 46). □

Hence equally-spaced allocation provides the optimal solution to the dynamic program for deterministic function evaluations. It is encouraging to see a result that is consistent with the classical bisection method.

2.5 Optimal Policy Structure for I.I.D. Exponential Processing Times

We now characterize optimal policies in the presence of exponentially distributed processing times. Function evaluations no longer finish simultaneously, so we must reallocate points without the information from jobs that are still running.

2.5.1 A Simplified Recursion

Recall that each core $i = 1, 2, \ldots, n$ is equally likely to return a function value at every decision point. If Core i returns with the information that $X^* > X^{(i)}$, then the resulting incomplete state is $X_+^{(i)} = (X^{(i)}, X^{(i+1)}, \ldots, X^{(n)}, X^{(n+1)})$. Otherwise, if the process returns with $X^* < X^{(i)}$, then the resulting incomplete state

is $X_-^{(i)} = (X^{(0)}, X^{(1)}, \ldots, X^{(i-1)}, X^{(i)})$. Therefore

$$\mathbb{P}\left(S_{j+1} = X_+^{(i)} \;\middle|\; S_j,\, X_j\right) = \left(\frac{1}{n}\right)\left(\frac{B_j - X_j^{(i)}}{B_j - A_j}\right)$$

and

$$\mathbb{P}\left(S_{j+1} = X_-^{(i)} \;\middle|\; S_j,\, X_j\right) = \left(\frac{1}{n}\right)\left(\frac{X_j^{(i)} - A_j}{B_j - A_j}\right).$$

We can now analyze the dynamic programming recursion. Conditioning on the action X_j,

$$\begin{aligned} V(S_j) &= (1-\gamma)R(S_j) + \gamma \max_{X \in \mathbb{X}(S_j)} \mathbb{E}[V(S_{j+1}) \,|\, S_j, X_j = X] \\ &= (1-\gamma)R(S_j) + \gamma \max_{X \in \mathbb{X}(S_j)} \frac{1}{n}\sum_{i=1}^{n}\left[\left(\frac{X^{(i)} - A_j}{B_j - A_j}\right)V(X_-^{(i)}) + \left(\frac{B_j - X^{(i)}}{B_j - A_j}\right)V(X_+^{(i)})\right]. \end{aligned}$$

Now assume S_j is normalized, i.e., $\|S_j\| = 1$. We want to express $V(S_j)$ in terms of the value of other normalized states. Scaling and shifting, we find that

$$\begin{aligned} V(S_j) &= (1-\gamma)(1) + \gamma \max_{X \in \mathbb{X}(S_j)} \frac{1}{n}\sum_{i=1}^{n}\left[X^{(i)}\, V(X_-^{(i)}) + (1 - X^{(i)})\, V(X_+^{(i)})\right] \\ &= (1-\gamma) + \gamma \max_{X \in \mathbb{X}(S_t)} \frac{1}{n}\sum_{i=1}^{n}\left[(X^{(i)})^{1-r}\, V(\bar{X}_-^{(i)}) + (1 - X^{(i)})^{1-r}\, V(\bar{X}_+^{(i)})\right], \end{aligned}$$

where $\bar{X}_+^{(i)}$ and $\bar{X}_-^{(i)}$ are the respective normalized states for $X_+^{(i)} = (X^{(i)}, \ldots, X^{(n+1)})$ and $X_-^{(i)} = (X^{(0)}, \ldots, X^{(i)})$. Now we can restrict the state space to $\{S \in \mathbb{S} : \min(S) = 0,\, \max(S) = 1\}$. In fact, for the rest of the chapter, we consider only normalized states and allocations.

In addition to expressing the value function in terms of the current state, we can also express it in terms of the action X. One can view X as the post-decision state, and $W(X)$, the expected downstream reward resulting from action X, as a so-called Q-factor or post-decision value function (Powell 2011, see). Formally, we define $W(X)$ as

$$W(X) = \frac{1}{n}\sum_{i=1}^{n}(X^{(i)})^{1-r}\, V(\bar{X}_-^{(i)}) + (1 - X^{(i)})^{1-r}\, V(\bar{X}_+^{(i)}). \tag{2.5}$$

For $z \in [0,1]$, let $\beta(z) = z^{1-r} + (1-z)^{1-r}$, with $r \in (0,1]$ defined as in the reward function R.In an abuse of notation, for an allocation X, we define $\beta(X) = n^{-1}\sum_{i=1}^{n}\beta(X^{(i)})$, and $\beta((0,1)) = 1$. Using the original definition V, we can write W as a recursion, yielding

$$W(X) = (1-\gamma)\beta(X) + \frac{\gamma}{n}\sum_{i=1}^{n}(X^{(i)})^{1-r}\left[\max_{Y\in\mathbb{X}(\bar{X}_{-}^{(i)})} W(Y)\right] + (1-X^{(i)})^{1-r}\left[\max_{Z\in\mathbb{X}(\bar{X}_{+}^{(i)})} W(Z)\right]. \tag{2.6}$$

2.5.2 Golden Section Policy

The above recursions and shifting/scaling properties reveal the structure inherent in the Multisection Algorithm. We now want to leverage this structure in deriving an optimal policy. When we have two cores, i.e., $n = 2$, we can describe a policy that reduces the interval size by a constant factor in every step, and guarentees that the (normalized) allocation at every step is identical.

Let $X = X_0 = \pi((0,1))$. When $n = 2$, we have $X = (0, a, b, 1)$, with $X^{(1)} = a$ and $X^{(2)} = b$. By normalizing, we have $\bar{X}_{+}^{(1)} = \left(0, \frac{b-a}{1-a}, 1\right)$ and $\bar{X}_{-}^{(2)} = (0, \frac{a}{b}, 1)$. We only need to consider these states because $\bar{X}_{-}^{(1)} = \bar{X}_{+}^{(2)} = (0,1)$, meaning that both cores need to be reallocated. If we want $\pi(\bar{X}_{+}^{(i)}) = \pi(\bar{X}_{-}^{(i)}) = X$ for $i = 1, 2$ and some allocation X, i.e., that the allocation is the same for both states, then we require that $(0, \frac{b-a}{1-a}, 1) \subset (0, a, b, 1)$ and $(0, \frac{a}{b}, 1) \subset (0, a, b, 1)$. One way to satisfy this is for $a = (b-a)/(1-a)$ and $b = a/b$. This system of equations has the unique solution

$$X^{(1)} = a = \frac{1}{2}\left(3 - \sqrt{5}\right) \qquad X^{(2)} = b = \frac{1}{2}\left(\sqrt{5} - 1\right). \tag{2.7}$$

Consider a policy π such that $\pi((0,1)) = \pi((0,a,1)) = \pi((0,b,1)) = (0,a,b,1)$ and $\pi(hS + c) = h\pi(S) + c$. We call this policy the Golden Section Policy because

it uses the same queries as the Golden Section Search Algorithm (Kiefer 1953) for finding the extremum (either maximum or minimum) of a unimodular function. In both situations, Golden Section uses symmetry to maintain the same spread of queries. In the context of multisection, the construction of the Golden Section policy implies that if $X_j = X$, then $\pi(\bar{X}_+^{(i)}) = \pi(\bar{X}_-^{(i)}) = X$ for $i = 1, 2$, implying that $X_{j+1} = X_j$ for all $j \geq 0$. This property can be used to show that Golden Section is optimal under a risk-neutral reward function.

On the Fringe: Optimal When Indifferent

We now show that under risk indifference, i.e., $r = 1$, Golden Section is optimal.

Theorem 2.5.1. *Let π be the Golden Section policy as defined above. If $r = 1$, then π is optimal.*

Proof. From the recursion in $W(\cdot)$, we see for any allocation $X \in \arg\max_{Z \in \mathbb{X}((0,1))} W(Z)$,

$$W(X) = (1-\gamma)\beta(X) + \frac{\gamma}{n}\sum_{i=1}^{n}(X^{(i)})^{1-r}\left[\max_{Z_1 \in \mathbb{X}(\bar{X}_-^{(i)})} W(Z_1)\right] + (1-X^{(i)})^{1-r}\left[\max_{Z_2 \in \mathbb{X}(\bar{X}_+^{(i)})} W(Z_2)\right].$$

If we relax the constraints in the two maxima, we see that

$$\begin{aligned} W(X) &\leq (1-\gamma)\beta(X) \\ &\quad + \frac{\gamma}{n}\sum_{i=1}^{n}(X^{(i)})^{1-r}\left[\max_{Z_1 \in \mathbb{X}((0,1))} W(Z_1)\right] + (1-X^{(i)})^{1-r}\left[\max_{Z_2 \in \mathbb{X}((0,1))} W(Z_2)\right] \\ &= (1-\gamma)\beta(X) + \gamma\, W(X)\,\frac{1}{n}\sum_{i=1}^{n}[(X^{(i)})^{1-r} + (1-X^{(i)})^{1-r}] \\ &= (1-\gamma)\beta(X) + \gamma\, W(X)\beta(X). \end{aligned}$$

By iteratively substituting the above inequality for $W(X)$, we can express the upper bound as the infinite geometric series

$$W(X) \leq (1-\gamma)\beta(X) \sum_{p=0}^{\infty} \gamma^p \, \beta(X)^p, \tag{2.8}$$

which converges when $\gamma\beta(X) < 1$ and diverges to infinity otherwise. But if we consider the value of the Golden Section policy, with Y being the Golden Section allocation,

$$\begin{aligned} W^\pi(Y) &= (1-\gamma)\beta(Y) + \frac{\gamma}{n} \sum_{i=1}^{n} (Y^{(i)})^{1-r} \, W^\pi(\pi(\bar{Y}_-^{(i)}))) + (1-Y^{(i)})^{1-r} \, W^\pi(\pi(\bar{Y}_+^{(i)})) \\ &= (1-\gamma)\beta(Y) + \frac{\gamma}{n} \sum_{i=1}^{n} (Y^{(i)})^{1-r} \, W^\pi(Y) + (1-Y^{(i)})^{1-r} \, W^\pi(Y), \end{aligned}$$

and we can also write the post-decision value of choosing the Golden Section as

$$W^\pi(Y) = (1-\gamma)\beta(Y) \sum_{p=0}^{\infty} \gamma^p \, \beta(Y)^p. \tag{2.9}$$

If $r = 1$, we have $\beta(X) = 2$ for any allocation X, which makes (2.8) and (2.9) both equal. If $\gamma < 1/2$, then the upper bound equals $2(1-\gamma)/(1-2\gamma)$. If $\gamma \geq 1/2$, then the series diverges to infinity. In either case, choosing the Golden Section policy π achieves the upper bound, and is therefore optimal. □

When $r < 1$, the Golden Section Policy still performs very well. In the $W(\cdot)$ recursion, the immediate reward obtained from assigning queries with allocation X is given by $\beta(X)$. Since the Golden Section Policy requires only one allocation, it is sufficent to use this as a metric for the performance of the entire policy. Now suppose X is the Golden Section Allocation. If we compare $\beta(X)$ versus $\max_z \beta(z) = 2^r$, which is achieved by the Bisection Method, there is an extremely small gap. At its smallest, the ratio $\beta(X)/2^r \geq 0.993$, achieved when $r = 0.502$. And from numerically solving discretized versions of the dynamic

program, there is strong computational evidence that the Golden Section Policy is optimal for any $0 < r < 1$ with γ sufficiently large.

We can also compare the Golden Section Policy to the traditional Bisection Method in terms of $W(\cdot)$. Let w_1 and w_2 be the respective post-decision state values for Bisection and Golden Section policies. Using (2.9) and the definition of γ in Section 2.3.1, we can derive closed form expressions for w_1 and w_2 in terms of α, λ, and r. In general, Golden Section achieves a higher expected reward because it effectively uses the additional information, and the loss from cutting off-center is relatively small. There are values of parameters (eg. $\alpha = 0.5$, $\lambda = 1.2$, $r = 0.5$) for which Golden Section performs infinitely better under this metric. However, in cases where λ is small, it is more advantageous to use Bisection.

Larger Numbers of Cores

For a general number of cores n, it is difficult to explicitly find policies similar to Golden Section. Even when $n = 3$, one cannot rely on a single allocation to define a policy. (For the case of three dimensions, finding such a policy is equivalent to solving a system with three variables and six nonlinear equations, none of which are redundant.) A fundamental idea behind Golden Section is that reducing the number of states visited over time is beneficial for the long-run performance of the algorithm. We might then relax the condition that we only visit a single state, i.e., $X_{j+1} = X_j$ for every j, and instead design a policy that visits a *finite* number of states. This might be a more achievable goal in higher dimensions, even though it is not guaranteed to yield an optimal policy.

For example, consider a policy for the case $n = 3$ defined by two allocations $Y = (0, 1/3, 1/2, 2/3, 1)$ and $Z = (0, 1/4, 1/2, 3/4, 1)$. The potential incomplete states arising from Y or Z, (exploiting the fact that both Y and Z are symmetric about $1/2$ so as to reduce the number of incomplete states), are

$$\begin{aligned} \bar{Y}_+^{(1)} &= (0, 1/4, 1/2, 1) \quad \bar{Y}_-^{(1)} = (0, 1) \quad \bar{Y}_+^{(2)} = (0, 1/3, 1) \\ \bar{Z}_+^{(1)} &= (0, 1/3, 2/3, 1) \quad \bar{Z}_-^{(1)} = (0, 1) \quad \bar{Z}_+^{(2)} = (0, 1/2, 1), \end{aligned} \tag{2.10}$$

which means that no matter what job finishes first, we will always be able to choose action Y or action Z to reallocate the idle processors. In other words, starting from Y or Z, for every $j \geq 0$ there always exists some $X_j \in \{Y, Z\} \cap \mathbb{X}(S_j)$. Such a choice of policy yields a Markov Chain on the allocation space $\mathbb{X}((0, 1))$ with only two states. This kind of policy is desirable because it easy to describe and easy to analyze. Even though such a policy is reminiscient of Golden Section, both actions are not equally preferable. The queries of Y are positioned closer to the midpoint than those of Z, which means Y is likely to reduce the interval length by a greater amount. Because of this, in cases where we have a choice between Y and Z, selecting Y is preferable, and one can verify that $W(Y) > W(Z)$. Even so, the existence of a policy with a small action space is promising. We suspect that similar policies exist for larger n, although finding them has proven to be difficult.

2.5.3 To Stack or Not To Stack

The memoryless property of the exponential distribution means that the residual processing time for a core is always exponentially distributed with the same mean. This observation suggests that each time a core completes, we might re-assign all cores that are still running as well as those that have been made re-

dundant. Our formulation in Section 2.2 did not allow this, because we required that $X \subset S$, i.e., that the action leaves cores running that are still in process. We continue to require that non-redundant cores continue to run, but there is a subtlety in our setup that has a substantial impact on the form of the optimal policy. With exponential processing times, with probability 1 we see a single core, i say, return a value at each decision point, so that either $(\ell_j, u_j) = (0, i)$ or $(\ell_j, u_j) = (i, n+1)$ in the discussion immediately following (2.2). If, for example, Core i tells us that $X^* > X^{(i)}$, then just after (2.2) our formulation allows us to reassign all cores with indices less than i. In practice, we have the option of doing more. If Core $i+1$ is also evaluating at $X^{(i)}$, then we can also reassign Core $i+1$, along with any other cores $j > i$ for which $X^{(j)} = X^{(i)}$. In our present formulation this is not allowed.

We made this modeling choice because it seemed, to us, to be realistic to not "stack cores," i.e., to not assign them to evaluate the same point. However, in situations such as in cloud-computing environments where cores can be quite unreliable and the speed of processing can be highly variable from one core to another, we may have incentive to evaluate very similar points on multiple cores, i.e., to adopt stacking policies. We now adapt our formulation to allow such policies.

Suppose now that when Core i concludes that $X^* > X^{(i)}$, we reassign all cores j for which $X^{(j)} \leq X^{(i)}$, *including those with indices greater than* i. Similarly, when $X^* \leq X^{(i)}$, we reassign all cores j for which $X^{(j)} \geq X^{(i)}$. Now the transition kernel is no longer continuous because, for example, there is a large difference between stacking two adjacent points, and instead evaluating two points that are minutely but positively separated. Hence, the argument in

Theorem 2.3.1 no longer applies, and we are no longer assured that an optimal value function exists and satisfies the Bellman recursion. *We conjecture that this is so, and proceed in this section under this conjecture.*

Optimal Stacking Policy Structure

It turns out that now the optimal policy for any set of parameters is one where all cores stack at the midpoint of the interval.

Theorem 2.5.2. *Stacking queries $X_j^{(i)} = \frac{1}{2}(A_j + B_j)$ for every core i is the optimal allocation for independent exponentially distributed waiting times with common rate.*

Proof. Since $V(S_0)$ is the optimal allocation when *all* n cores are assigned, $V(\bar{X}_-^{(i)}) \leq V(S_0)$ because the value function is more constrained. Therefore,

$$\begin{aligned} V(S_0) &= (1-\gamma) + \gamma \max_{X \in \mathbb{X}(S_0)} \frac{1}{n} \sum_{i=1}^{n} \left[(X^{(i)})^{1-r} V(\bar{X}_-^{(i)}) + (1 - X^{(i)})^{1-r} V(\bar{X}_+^{(i)}) \right] \\ &\leq (1-\gamma) + \gamma \max_{X \in \mathbb{X}(S_0)} \frac{1}{n} \sum_{i=1}^{n} \left[(X^{(i)})^{1-r} V(S_0) + (1 - X^{(i)})^{1-r} V(S_0) \right] \\ &= (1-\gamma) + V(S_0)\gamma \max_{X \in \mathbb{X}(S_0)} \frac{1}{n} \sum_{i=1}^{n} \left[(X^{(i)})^{1-r} + (1 - X^{(i)})^{1-r} \right]. \end{aligned}$$

The function $\beta(\cdot)$ is maximized at $x = 1/2$ with $\beta(x) \leq 2^r$. Each term in the sum is bounded above by this value, with equality if every queried point $X^{(i)} = 1/2$. Thus,

$$V(S_0) \leq (1-\gamma) + V(S_0)\gamma\, 2^r,$$

with equality if all points $X^{(i)}$ are stacked at $1/2$. Hence, stacking at $1/2$ is optimal regardless of the value of γ. Moreover, if $\gamma < 2^{-r}$ then the optimal value is finite. □

REFERENCES

[1] Onésimo Hernández-Lerma and Jean Bernard Lasserre. *Discrete-time Markov control processes*. Springer, 1996.

[2] Jack Kiefer. "Sequential minimax search for a maximum". In: *Proceedings of the American Mathematical Society* 4.3 (1953), pp. 502–506.

[3] Warren B Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Vol. 842. John Wiley & Sons, 2011.

CHAPTER 3

COUPLED BISECTION FOR ROOT ORDERING

3.1 Introduction

Consider a function $f : \mathcal{S} \times [0, 1) \to \mathbb{R}$, where $\mathcal{S}$ is finite but large, and for every s, $f(s, \cdot)$ is monotonic with unique root. If we are interested in finding the zero $x^*(s)$ of $f(s, \cdot)$ for all elements $s \in \mathcal{S}$, then for each $f(s, \cdot)$ we could employ the classical bisection method. However, if one evaluation of f for some x yields values of $f(s, x)$ for all $s \in \mathcal{S}$, then we could potentially solve multiple bisection problems at once. Furthermore, if we are only interested in the *ordering* of elements s with respect to their zeros $x^*(s)$, calculating the zeros to precision is computationally unnecessary.

The coupled root-ordering setting has applications in computing Gittins policies (Gittins and Jones 1974; Whittle 1988), respectively used in multi-arm bandit and restless bandit problems. We are then interested in ordering states in the state space according to their Gittins or Whittle indices, which correspond to the zero of a particular function. The ordering of the states is all that is required to implement the index policy. Methods for evaluating these indices to precision are prevalent in the literature (Niño-Mora 2011, see) for a discussion on computational methods). In practical applications where Gittins indices are computed, the problems typically have additional structure, such as sparse transition kernels or the ability to compute indices in an online fashion. The most competitive algorithms for computing Gittins index policies exploit these kinds of structure. Coupled bisection does not take advantage of any additional structure, and therefore is not competitive with these algorithms. However,

its generality allows it to compute index policies for a wide range of problems that be formulated as instances of coupled root-ordering (Hu, Frazier, Xie, et al. 2014; Zhao and Frazier 2014; Dayanik, Powell, and Yamazaki 2008; Glazebrook, Kirkbride, and Ouenniche 2009, see).

Coupled bisection can also be used in solving coupled root-finding problems, because it can be more computationally efficient to sort the roots before further refining their respective locations. One such example is the estimation of phase diagrams during the evaluation of piezoelectric materials (Shrout and Zhang 2007). Given a pair of chemical compounds A and B, one must determine for each temperature s in some set a critical threshold $x^*(s)$ such that mixtures of A and B with a fraction $x > x^*(s)$ of A form one crystal phase, while mixtures with $x < x^*(s)$ form another phase. Here, $f(s, x)$ is the crystal phase at temperature s and mixing coefficient x, and can be observed through a physical experiment. Synthesizing a particular linear combination x is time consuming, but allows easy observation of $f(s, x)$ for all temperatures s. This is a coupled root-finding problem.

Coupled root-finding also arises in remote sensing, when finding the boundary of a forest or other geographical feature from an airborne laser scanner (Castro and Nowak 2008). Here, an aircraft chooses a latitude x at which to fly and observes the presence or absence of the feature, $f(s, x) \in \{0, 1\}$, for all longitudes s in the flight path. The boundary at longitude s is given by the root $x^*(s)$.

Naively, we could discretize the interval $[0, 1)$ and calculate f with respect to all discretized values of x. Although this is easy to program and understand, the computational investment can be massive and unnecessary. We develop a coupled bisection method that can sort these elements in a more efficient fash-

ion.

We solve this problem by sequentially evaluating f at different values of x. When we evaluate f at some value x, we find $f(s, x)$ for every element s, and we can deduce for every element whether $x^*(s) \geq x$ or $x^*(s) < x$. At every iteration, we know of a subinterval that contains $x^*(s)$. These subintervals form a disjoint partition of $[0, 1)$. By evaluating f for a different value of x at each iteration, we refine the previous partition, choosing one subinterval to split into two. This continues until for every element each subinterval contains at most one root.

In this process, we must find a way to sequentially select the next value of x at which we evaluate f. One might conjecture by analogy that the optimal decision is to choose some subinterval and select the midpoint of that interval to be the next value of x. This policy is *not* optimal, but we show it can sort the elements by their associated zeros in $O(|\mathcal{S}|)$ iterations in expectation, and calculate the asymptotic constant to be bounded above by 1.44. We also provide a lower bound of $|\mathcal{S}| - 1$ for the minimum number of iterations for any policy, implying an approximation guarantee of 1.44. Moreover, we give computational evidence suggesting our proposed policy is even closer to optimal than the 1.44 guarantee suggests.

3.2 Problem Specification

We first model the underlying decision process. Suppose $X = \left\{[x^{(i)}, x^{(i+1)}) : i = 0, \ldots, m\right\}$ denotes a partition of the interval $[x^{(0)}, x^{(m+1)})$. We assume $x^{(i)} < x^{(i+1)}$ for all i. Let $N = (n^{(0)}, n^{(1)}, \ldots, n^{(m)})$ represent the num-

bers of roots that lie in the corresponding subintervals in X. Together, the pair (X, N) determine the *computational state*. Suppose at decision epoch j, our current computational state is (X_j, N_j). We choose a refined partition

$$X_{j+1} = \left\{ \left[x_j^{(0)}, x_j^{(1)}\right), \ldots, \left[x_j^{(\ell)}, \bar{x}_{j+1}\right), \left[\bar{x}_{j+1}, x_j^{(\ell+1)}\right), \ldots, \left[x_j^{(j)}, x_j^{(j+1)}\right) \right\},$$

where $\ell \in \{0, \ldots, j\}$ and $\bar{x}_{j+1} \in (x_j^{(\ell)}, x_j^{(\ell+1)})$. Accordingly, let $\mathbb{X}(X_j)$ be the set containing all refined partitions of X_j containing $j+1$ subintervals. We then evaluate f at $\bar{x}_{j+1}$. For all elements $s \in \mathcal{S}$ we observe $f(s, \bar{x}_{j+1})$, and therefore we can determine whether $x^*(s)$ is less than or greater than $\bar{x}_{j+1}$. At this point, the $n_j^{(\ell)}$ roots in the original subinterval $\left[x_j^{(\ell)}, x_j^{(\ell+1)}\right)$ are split among the two newly-created subintervals. Hence, we have

$$N_{j+1} = \left(n_j^{(0)}, \ldots, n_j^{(\ell-1)}, \bar{N}_{j+1}, n_j^{(\ell)} - \bar{N}_{j+1}, n_j^{(\ell+1)}, \ldots, n_j^{(j)}\right),$$

where $n_{j+1}^{(\ell)} = \bar{N}_{j+1}$ and $n_{j+1}^{(\ell+1)} = n_j^{(\ell)} - \bar{N}_{j+1}$. All other components in N_j remain the same because we learn nothing new about roots in the other subintervals.

A priori, we assign a prior distribution to the location of $x^*(s)$ for every element $s \in \mathcal{S}$. For simplicity, we assume that for every s, independent of all else, $x^*(s) \sim \text{Unif}[0, 1)$. Otherwise, as long as under the prior distribution the root locations are i.i.d. and absolutely continuous with respect to Lebesgue measure, we can use an inverse mapping to appropriately stretch the real line to yield the above case. Therefore, *a priori*, $\bar{N}_{j+1} \sim \text{Binomial}\left(n_j^{(\ell)}, (\bar{x}_{j+1} - x_j^{(\ell)})/(x_j^{(\ell+1)} - x_j^{(\ell)})\right)$. Since we would like to find an ordering for $x^*(\cdot)$, we stop evaluating f when every subinterval in the partition X contains at most one root, i.e., we stop when $n^{(i)} \leq 1$ for all $i \in \{0, \ldots, |N| - 1\}$. Define the stopping time $\tau = \inf\{j \in \mathbb{N} : N_j^{(i)} \leq 1 \quad \forall i = 0, 1, \ldots, j\}$.

We would like to model this multiple root-finding problem as a dynamic program that finds the Bayes-optimal policy minimizing the expected number

of evaluations of $f(\cdot, x)$ needed to find an ordering of all elements $s \in \mathcal{S}$ with respect to x^*. We define the value function for *computational effort* under policy π

$$W^{\pi}(X, N) = \mathbb{E}^{\pi}\left[\tau \,|\, X_0 = X,\ N_0 = N\right], \tag{3.1}$$

where π is a policy that maps computational states (X, N) to partitions $\bar{X} \in \mathbb{X}(X)$. The value function W^{π} counts the expected number of iterations needed to sort the elements $s \in \mathcal{S}$ with respect to x^* under policy π. We define the value function $W(X, N) = \inf_{\pi \in \Pi} W^{\pi}(X, N)$, where Π denotes the set of all policies π.

3.3 Recursion

Using the original definition of the value function in (3.1), we can derive a recursion for the computational dynamic program. For a computational state pair (X, N) that do not satisfy the stopping conditions, we have that

$$W^{\pi}(X, N) = 1 + \mathbb{E}^{\pi}\left[W^{\pi}(X_1, N_1) \,|\, (X_0, N_0) = (X, N)\right],$$

where $X_1 = \pi(X_0, N_0)$ indicates the next refinement of the partition, and N_1 is the subsequent spread of elements among subintervals.

By the principle of optimality, we can iteratively take the best partition X_1 over each step, which gives us a recursion for W, the value function under the optimal policy. Because the process $((X_j, N_j) :\ j \geq 0)$ is time-homogeneous, we can drop the subscripts and denote $\bar{X}$ as the refinement of partition X, and $\bar{N}$ as the resulting distribution of elements among subintervals. Thus, we have

$$W\left(X, N\right) = 1 + \min_{\bar{X} \in \mathbb{X}(X)} \mathbb{E}\left[W(\bar{X}, \bar{N}) \,\middle|\, X_0 = X,\ N_0 = N\right]. \tag{3.2}$$

3.3.1 Decomposition and Interval Invariance

We can greatly simplify this recursion by decomposing it by the different subintervals.

Theorem 3.3.1. *Under the optimal policy, the value function W has the following two properties.*

- *Decomposition:* $W(X,N) = \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$
- *Interval Invariance:* $W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right) = W\left(\{[0,1)\}, n^{(i)}\right)$.

Proof. We will first prove the decomposition result. For any initial computational state (X,N), consider the following policy. For each subinterval $[x^{(i)}, x^{(i+1)})$, take an optimal policy for the computational state $\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$, and once we find a partition of the subinterval satisfying the stopping conditions, we do the same for another subinterval. Therefore, it must be $W(X,N) \leq \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$.

Now we will prove the opposite inequality. First, note that the order we choose to further partition the subintervals is irrelevant, since we only seek to minimize the number of evaluations required, and each evaluation provides refinement only within its subinterval. Without loss of generality, consider only policies that evaluate the function with value within the leftmost subinterval that still does not satisfy the stopping conditions. Suppose this interval is $[x^{(i)}, x^{(i+1)})$ and contains the zeros of $n^{(i)}$ elements. Before we are allowed to move to the next subinterval, we must find a partition of $[x^{(i)}, x^{(i+1)})$ that satisfies the stopping conditions. By definition, this takes a minimum of $W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$ steps. Since we only evaluate the function at one value

at a time, we perform one evaluation on exactly one subinterval at each step. Therefore, repeating the same logic for every subinterval tells us $W(X, N) \geq \sum_{i=0}^{|N|-1} W\left(\left\{[x^{(i)}, x^{(i+1)})\right\}, n^{(i)}\right)$.

We will now prove the second claim of the theorem using a pathwise argument. Suppose we have initial computational state $(X_0, N_0) = \left([a, b), n^{(i)}\right)$. Define the operator $T((X, N)) = ((b-a)X + a, N)$. If we define $\left(\tilde{X}_j, \tilde{N}_j\right) = T^{-1}(X_j, N_j)$ for all time epochs j, there exists a one-to-one mapping between computational states. For any sample path of the process $((X_j, N_j) : j \geq 0)$ which reaches the stopping conditions at time epoch t, it must be that $(\tilde{X}_t, \tilde{N}_t)$ also satisfies the stopping conditions. Therefore, it must be that $W(\tilde{X}_0, \tilde{N}_0) \leq W(X_0, N_0)$. Symmetry gives the opposite inequality, and hence the result. $\square$

It may seem strange that the recursion relation does not depend on the size of the interval. In fact, it only depends on the number of elements in each subinterval, because we are only concerned with finding the correct *ordering*.

The decomposition is helpful both when solving the dynamic program and describing the optimal policy. Since the value function is additive among subintervals, the order in which we refine the partition does not affect the optimal number of evaluations of f. Thus, we can focus our attention on solving a one-dimensional dynamic program and without loss of generality solely consider the subinterval $[0, 1)$.

3.3.2 Simplified Recursion

Since we can write the value function W in terms of its smaller subintervals, we can just consider the special case where the partition $X = \{[0,1)\}$. In a slight abuse of notation, we define $W(n) = W(\{[0,1)\}, n)$ and have

$$W(n) = 1 + \min_{x \in (0,1)} \mathbb{E}\left[W(N_x) + W(n - N_x) \right], \tag{3.3}$$

where $N_x \sim$ Binomial(n, x), independent of all else. Intuitively, we choose a point $x \in (0,1)$ to evaluate the original dynamic program, and the n elements get split among the two newly-created sub-intervals. As before, we have the stopping conditions $W(0) = W(1) = 0$. Computationally, we cannot use this recursion to solve for $W(\cdot)$ explicitly, since N_x can equal 0 or n with positive probability, causing $W(n)$ to appear on both sides of (3.3). Proposition 3.3.2 accounts for this.

Proposition 3.3.2.

$$W(n) = \min_{x \in (0,1)} \left\{ \frac{1}{1 - x^n - (1-x)^n} + \mathbb{E}\left[W(N_x) + W(n - N_x) \,\middle|\, 1 \le N_x \le n-1 \right] \right\}. \tag{3.4}$$

Proof. From (3.3), for any $x \in [0,1]$,

$$\begin{aligned} W(n) \le 1 &+ \mathbb{E}\left[W(N_x) + W(n - N_x) \mid N_x \in [1, n-1] \right] \cdot \mathbb{P}\left(N_x \in [1, n-1] \right) \\ &+ \left(W(n) + W(0) \right) \mathbb{P}\left(N_x = n \right) + \left(W(0) + W(n) \right) \mathbb{P}\left(N_x = 0 \right), \end{aligned}$$

with equality for some $x \in [0,1]$ (since the interval is compact and the right side is continuous in x). Since $W(0) = 0$, we get

$$\begin{aligned} W(n) \le 1 &+ \mathbb{E}\left[W(N_x) + W(n - N_x) \mid N_x \in [1, n-1] \right] \cdot \mathbb{P}\left(N_x \in [1, n-1] \right) \\ &+ W(n) \left(1 - \mathbb{P}\left(N_x \in [1, n-1] \right) \right), \end{aligned}$$

$$\text{i.e.,} \qquad W(n) \le \frac{1}{\mathbb{P}\left(N_x \in [1, n-1] \right)} + \mathbb{E}\left[W(N_x) + W(n - N_x) \,\middle|\, N_x \in [1, n-1] \right].$$

This inequality is tight for the same x that made the previous inequality tight. Using $N_x \sim \text{Binomial}(n, x)$ gives the result. □

This recursion reveals the structure behind the coupled bisection algorithm. Suppose we have an interval that contains n elements. There are two possibilities when evaluating f at the next value of x: (i) splitting the interval into two subintervals, one of which contains all n elements, and (ii) splitting into two subintervals, both of which contain at least one element each. In case (i), we would have to perform the same procedure again on that smaller subinterval. The first term in (3.4) is exactly equal to the expected number of iterations it takes to break free of this loop. Case (ii) corresponds with the conditional expectation term in the same equation. Thus, the choice of x is a balancing act between limiting the iterations spent on the initial split and obtaining a desirable spread of elements in subintervals after that split occurs.

3.4 Bisection Policy

Consider the bisection policy $\beta \in \Pi$, where we choose to evaluate the function at the midpoint of the interval, i.e., choosing $x = 1/2$ for all n in the recursion (3.3). This yields

$$W^{\beta}(n) = 1 + 2\ \mathbb{E}[\, W^{\beta}(N_{1/2})\,], \tag{3.5}$$

where $N_{1/2} \sim \text{Binomial}(n, 1/2)$, with stopping conditions $W(0) = W(1) = 0$.

3.4.1 Greedy Shannon Entropy Reduction

The bisection policy β is a good candidate policy because it is intuitive and easy to implement. But there is also a metric where bisection is optimal. If our goal is to sort elements with respect to their zeros, we can view the problem as trying to find the correct permutation of elements among the $|\mathcal{S}|!$ possibilities. Since we assume *a priori* that the roots of all elements are i.i.d. uniformly distributed throughout $[0, 1)$, every permutation of elements in $\mathcal{S}$ is equally likely.

We define $H(X, N)$ to be the Shannon entropy of the distribution of possible permutations of $\mathcal{S}$ at computational state (X, N). In this case, since we are considering a uniformly discrete distribution, this is equivalent to

$$H(X, N) = \log_2 \left(\prod_{i=0}^{|N|-1} n^{(i)}! \right), \tag{3.6}$$

where the term inside the logarithm denotes the number of permutations of roots consistent with computational state (X, N). The first observation is that the entropy of a given computational state (X, N) does not depend on the partition X, but only on the number of zeros in each partition. Also, because of the convenient properties of the logarithm, the decomposition property comes for free, giving us

$$H(X, N) = \sum_{i=0}^{|N|-1} H(\{[x^{(i)}, x^{(i+1)})\}, n^{(i)}).$$

For the same reasons given for decomposing W, we only need to consider one subinterval at a time. Therefore, we can assume without loss of generality that we start with computational state $([0, 1), n)$. Similarly to W, we define $H(n) = H(\{[0, 1)\}, n)$. We would like to maximize the expected entropy reduction, meaning

$$\max_{x \in (0,1)} H(n) - \mathbb{E}\left[H(N_x) + H(n - N_x)\right], \tag{3.7}$$

where $N_x \sim \text{Binomial}(n, x)$.

Theorem 3.4.1. *An optimal solution for* (3.7) *is* $x^* = 1/2$ *for all* $n \geq 2$*, implying that the bisection policy maximally reduces entropy, given a single function evaluation, among possible permutations of elements in* $\mathcal{S}$*.*

Proof. The objective function in (3.7) is symmetric about $x = 1/2$, because $n - N_x \stackrel{d}{=} N_{1-x}$. If we can show that it is concave in x, then we are done. We know that $H(n) = \log_2 n!$, and therefore, we can compactly write the optimization problem in (3.7) as

$$\max_{x \in (0,1)} \mathbb{E}\left[\log_2 \binom{n}{N_x}\right]. \tag{3.8}$$

First, we use a property of binomial coefficients, namely that $\left(\binom{n}{k} : k = 0, \ldots, n\right)$ is a log-concave sequence (Stanley 1989), and hence, $\left(\log_2 \binom{n}{k} : k = 0, \ldots, n\right)$ is a concave sequence. Now we invoke a useful property of distributions in exponential families. A random variable Z_θ parameterized by θ is *convexly parameterized* if for any convex function f, $\mathbb{E}[f(Z_\theta)]$ is convex in θ. Mean-parameterized exponential family distributions are convexly parameterized (Shaked 1980; Schweder 1982), and since the binomial distribution is a member of that family, it follows that (3.8) is concave in x. □

We showed bisection is an optimal policy with respect to the one-step reduction of the Shannon entropy of possible orderings of roots. Now we explore how well the bisection policy can reduce computational effort in W.

3.4.2 Minimizing Computational Effort

It is reasonable to conjecture that bisection is also an optimal policy for minimizing computational effort. However, for $n = 6$, solving the dynamic program W computationally with (3.4) reveals that the optimal choice of x is not the midpoint of the interval, but rather is located at $x_6^* = 0.5 \pm 0.037$, with an optimality gap of 2.58×10^{-5}. This is the first of many disparities between the bisection policy and the optimal policy in this setting.

To bound the optimality gap, we derive upper bounds on $W^\beta(\cdot)$ and lower bounds on $W(\cdot)$. We can show a rather crude lower bound for $W(\cdot)$. If we want to sort n elements, we must perform at least $n - 1$ evaluations of f to separate them, and induction on n with (3.4) confirms that $W(n) \geq n - 1$.

In comparison, what upper bounds can we derive for $W^\beta(\cdot)$? We have computational evidence suggesting that W^β grows linearly for large n, so we focus on bounding the linear growth rate. For a function g defined on the integers, let $\Delta g(n) = g(n+1) - g(n)$. We will prove for large n that $\Delta W^\beta(n) \leq \gamma$ for some constant γ.

3.4.3 What's the difference?

To derive upper bounds, we use a structural result regarding the growth rate of W^β, which is proved using a coupling argument.

Lemma 3.4.2.

$$\Delta W^\beta(n) = \mathbb{E}\left[\, \Delta W^\beta(N_{1/2}) \,\right], \tag{3.9}$$

where $N_{1/2} \sim$ *Binomial*$(n, 1/2)$*, and with stopping conditions* $\Delta W^\beta(0) = 0$ *and* $\Delta W^\beta(1) = 2$*.*

Proof. We start with the recursion from (3.5) and take a difference, giving us

$$\Delta W^\beta(n) = 2\,\mathbb{E}\left[W^\beta(\hat{N}_{1/2}) - W^\beta(N_{1/2})\right],$$

where $N_{1/2} \sim$ Binomial$(n, 1/2)$ and $\hat{N}_{1/2} \sim$ Binomial$(n+1\,, 1/2)$. We can couple these two random variables since the expression involves only their expectation. Let $B_{1/2} \sim$ Bernoulli$(1/2)$, independent of all else, and since $\hat{N}_{1/2} \stackrel{\mathrm{d}}{=} N_{1/2} + B_{1/2}$, we have that

$$\begin{aligned}
\Delta W^\beta(n) &= 2\,\mathbb{E}\left[W^\beta(N_{1/2} + B_{1/2}) - W^\beta(N_{1/2})\right], \\
&= 2\,\mathbb{E}\left[\frac{1}{2}\left(W^\beta(N_{1/2}+1) - W^\beta(N_{1/2})\right) + \frac{1}{2}\left(W^\beta(N_{1/2}) - W^\beta(N_{1/2})\right)\right] \\
&= \mathbb{E}\left[\,\Delta W^\beta(N_{1/2})\,\right].
\end{aligned}$$

For the stopping conditions, we use the original stopping conditions for (3.3), which gives us $W^\beta(0) = W^\beta(1) = 0$. By direct calculation using (3.4), we find $W^\beta(2) = 2$ (which happens to correspond with W at $n = 2$). □

Here the growth rate $\Delta W^\beta(n)$ is a weighted average of all previous growth rates, suggesting that the sequence should converge. We use this idea to derive a method for calculating upper bounds on $W^\beta(n)$ for large but finite values of n, and conjecture these bounds hold for all n.

3.4.4 Upper Bounds

Before we show a computational method for deriving upper bounds on $\Delta W^\beta(n)$, we first prove a property of the Binomial distribution.

Lemma 3.4.3. *For non-negative integers ℓ and m such that $\ell \in [0, m-2]$, and for $n \geq m$, let $p_{\ell,m}$ be defined as*

$$p_{\ell,m}(n) = \mathbb{P}\left(N_{1/2}(n) \geq \ell + 1 \,\middle|\, N_{1/2}(n) \leq m - 1\right),$$

where $N_{1/2}(n) \sim$ Binomial$(n, 1/2)$. Then $p_{\ell,m}(n)$ is non-decreasing in n.

Proof. For any non-negative integer n, $p_{\ell,m}(n) \leq p_{\ell,m}(n+1)$ is equivalent to $N_{1/2}(n) \leq_{rh} N_{1/2}(n+1)$, where $\leq_{rh}$ refers to the *reverse hazard rate ordering* (Shaked and Shanthikumar 2007, p. 37). By definition, for two discrete random variables U and V, $U \leq_{rh} V$ if

$$\frac{\mathbb{P}(U = n)}{\mathbb{P}(U \leq n)} \leq \frac{\mathbb{P}(V = n)}{\mathbb{P}(V \leq n)}, \tag{3.10}$$

for all natural numbers n. It is clear from (3.10) that $N_{1/2}(n) \leq_{rh} N_{1/2}(n)$. We also have $0 \leq_{rh} B_{1/2}$, where $B_{1/2} \sim$ Bernoulli$(1/2)$, independent of all else.

Now we use the fact that reverse hazard rate ordering is closed under convolutions (Shaked and Shanthikumar 2007, p. 38), i.e., if we have random variables U_1, U_2 and V_1, V_2 such that $U_1 \leq_{rh} V_1$ and $U_2 \leq_{rh} V_2$, then $U_1 + U_2 \leq_{rh} V_1 + V_2$. Because $N_{1/2}(n) \leq_{rh} N_{1/2}(n)$ and $0 \leq_{rh} B_{1/2}$, we deduce $N_{1/2}(n) + 0 \leq_{rh} N_{1/2}(n) + B_{1/2}$, and since $N_{1/2}(n) + B_{1/2} \stackrel{d}{=} N_{1/2}(n+1)$, we get $N_{1/2}(n) \leq_{rh} N_{1/2}(n+1)$. □

Using (3.9), we now present computationally tractable upper bounds on the value of the bisection policy.

Theorem 3.4.4. *For some non-negative integer m, we define $g_m(n) = \max_{\ell \in [n, m-1]} \Delta W^{\beta}(\ell)$, and define h as*

$$h_m(n) = \mathbb{E}\left[g_m\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m - 1\right],$$

where $N_{1/2} \sim$ Binomial$(n, 1/2)$. Suppose we have $\gamma_m > 0$ so that the following condition holds:

$$\max\left\{\Delta W^{\beta}(m), h_m(m)\right\} \leq \gamma_m. \tag{3.11}$$

Then for all $n \geq m$, it must be that $\Delta W^{\beta}(n) \leq \gamma_m$.

Proof. We use induction on $n \geq m$. The condition gives us the base case $n = m$. Now suppose that for all $k \in [m, n-1]$ that $\Delta W^{\beta}(k) \leq \gamma_m$. From (3.9),

$$\begin{aligned}
\Delta W^{\beta}(n) &= \mathbb{E}\left[\Delta W^{\beta}\left(N_{1/2}\right)\right] \\
&= \Delta W^{\beta}(n)\ \mathbb{P}\left(N_{1/2} = n\right) + \mathbb{E}\left[\Delta W^{\beta}\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right]\ \mathbb{P}\left(N_{1/2} \leq m-1\right) \\
&\quad + \mathbb{E}\left[\Delta W^{\beta}\left(N_{1/2}\right) \,\middle|\, m \leq N_{1/2} \leq n-1\right]\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right) \\
&\leq \Delta W^{\beta}(n)\ \mathbb{P}\left(N_{1/2} = n\right) + \mathbb{E}\left[g\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right]\ \mathbb{P}\left(N_{1/2} \leq m-1\right) \\
&\quad + \gamma_m\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right) \\
&= \Delta W^{\beta}(n)\ \mathbb{P}\left(N_{1/2} = n\right) + h_m(n)\ \mathbb{P}\left(N_{1/2} \leq m-1\right) + \gamma_m\ \mathbb{P}\left(m \leq N_{1/2} \leq n-1\right),
\end{aligned}$$

where the inequality is due to the definition of g in the first term and the inductive hypothesis in the second term. Because we can solve for $\Delta W^{\beta}(n)$, all that remains is to show $h_m(n) \leq \gamma_m$. Consider

$$\begin{aligned}
h_m(n) &= \mathbb{E}\left[g\left(N_{1/2}\right) \,\middle|\, N_{1/2} \leq m-1\right] \\
&= g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot \mathbb{P}\left(N_{1/2} \geq \ell + 1 \,\middle|\, N_{1/2} \leq m-1\right) \\
&= g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot p_{\ell,m}(n).
\end{aligned}$$

Since g is non-increasing, Δg is non-positive. Also, we proved in Lemma 3.4.3 that $p_{\ell,m}(n)$ is non-decreasing in n. Since $n \geq m$,

$$h_m(n) \leq g(0) + \sum_{\ell=0}^{m-2} \Delta g\left(\ell\right) \cdot p_m(m) = h_m(m) \leq \gamma_m,$$

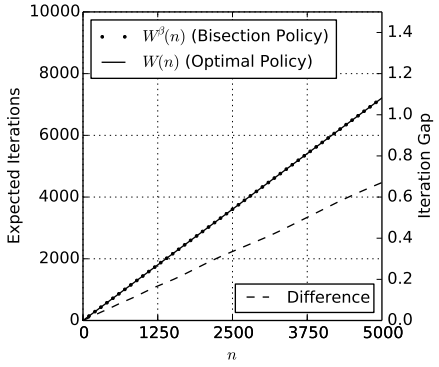
where the last inequality is true by the assumption. □

Theorem 3.4.4 is useful because we can compute ΔW^{β} for the first m terms using (3.9), then use the above result to find an arbitrarily tight bound on the policy for all $n \geq m$. The condition in (3.11) can be verified directly with the computed values of $W^{\beta}(n)$ for $n \in [0, m-1]$.

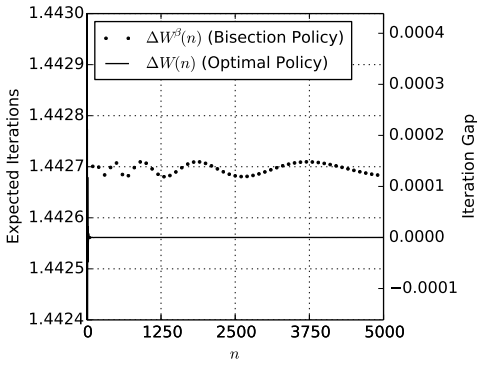
3.5 Computational Results and Concluding Remarks

Using Theorem 3.4.4, and choosing appropriate values for m, we computationally derive upper bounds on the bisection policy. In general, the bounds improve as m increases. Choosing $m = 15$, we find that the linear growth

Figure 3.1: Performance of the Bisection Policy



(a) The values $W^{\beta}(n)$, $W(n)$ and their difference for $n \leq 5000$. The bisection policy is close to optimal in the number of expected iterations required, enough for W^{β} and W to coincide above, as further evidenced by the scale in panel (b).



(b) The rates $\Delta W(n)$ and $\Delta W^{\beta}(n)$ for $n \leq 5000$. Although we show theoretically that $\Delta W(n) \geq 1$, it appears that $\Delta W(n)$ is bounded below by a larger constant for large n, and the true gap in rates is closer to 0.0001.

rate of the expected number of iterations required under the bisection policy is bounded above by $\gamma_{15} = 1.4440$, compared to the lower bound of 1 shown earlier on the growth rate of the optimal policy. This implies that, as n approaches infinity, the ratio of the expected number of iterations required under the two

policies is bounded above by 1.4440.

In fact, the gap in performance appears to even tighter. For $100 \leq n \leq 5000$, by using (3.4) to compute W directly, we empirically observe that $\Delta W(n) \geq 1.4425$, and the sequence seems to converge rather quickly, as shown in Figure 3.1b. Further, from Figure 3.1a, we see that the expected number of iterations required to sort n elements under the bisection policy is indistinguishably close to that under the optimal policy. This suggests that bisection is near-optimal, although proving this seems difficult. In any case, we have a linear-time algorithm for ordering elements by their associated zeros.

REFERENCES

[1] Rui Castro and Robert Nowak. "Active learning and sampling". In: *Foundations and Applications of Sensor Management*. Springer, 2008, pp. 177–200.

[2] Savas Dayanik, Warren Powell, and Kazutoshi Yamazaki. "Index policies for discounted bandit problems with availability constraints". In: *Advances in Applied Probability* (2008), pp. 377–400.

[3] John Gittins and D Jones. "A dynamic allocation index for the design of experiments". In: *Progress in Statistics* 2.9 (1974).

[4] Kevin D Glazebrook, Christopher Kirkbride, and Jamal Ouenniche. "Index policies for the admission control and routing of impatient customers to heterogeneous service stations". In: *Operations Research* 57.4 (2009), pp. 975–989.

[5] Weici Hu, Peter Frazier, Jing Xie, et al. "Parallel bayesian policies for finite-horizon multiple comparisons with a known standard". In: *Simulation Conference (WSC), 2014 Winter*. IEEE. 2014, pp. 3904–3915.

[6] José Niño-Mora. "Computing a Classic Index for Finite-Horizon Bandits". In: *INFORMS Journal on Computing* 23.2 (2011), pp. 254–267.

[7] Tore Schweder. "On the dispersion of mixtures". In: *Scandinavian Journal of Statistics* (1982), pp. 165–169.

[8] Moshe Shaked. "On mixtures from exponential families". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1980), pp. 192–198.

[9] Moshe Shaked and J George Shanthikumar. *Stochastic orders*. Springer Science & Business Media, 2007.

[10] Thomas R Shrout and Shujun J Zhang. “Lead-free piezoelectric ceramics: Alternatives for PZT?” In: *Journal of Electroceramics* 19.1 (2007), pp. 113–126.

[11] Richard P Stanley. “Log-Concave and Unimodal Sequences in Algebra, Combinatorics, and Geometrya”. In: *Annals of the New York Academy of Sciences* 576.1 (1989), pp. 500–535.

[12] Peter Whittle. “Restless bandits: Activity allocation in a changing world”. In: *Journal of Applied Probability* (1988), pp. 287–298.

[13] Xiaoting Zhao and Peter I Frazier. “A Markov Decision Process Analysis of the Cold Start Problem in Bayesian Information Filtering”. In: *arXiv preprint arXiv:1410.7852* (2014).

CHAPTER 4

BAYES-OPTIMAL ENTROPY PURSUIT FOR ACTIVE CHOICE-BASED PREFERENCE LEARNING

4.1 Introduction

The problem of preference learning is a well-studied and widely applicable area of study in the machine learning literature. Preference elicitation is by no means a new problem (Schapire and Singer 1998), and is now ubiquitous in many different forms in nearly all subfields of machine learning. One such scenario is the active learning setting, where one sequentially and adaptively queries the user to most efficiently learn his or her preferences. In general, learning in an online setting can be more efficient than doing so in an offline supervised learning setting, which is consequential when queries are expensive. This is often the case for preference elicitation, where a user may not be inclined to answer too many questions. The ability to adaptively query the user with particular exemplars that facilitate learning to the labels of the rest is invaluable in the context of preference elicitation.

In particular, there is great interest in using choice-based queries to learn the preferences of an individual user. In this setting, a user is offered two or more alternatives and is asked to select the alternative he or she likes most. There are other types of responses that can assess one's preferences among a set of alternatives, such as rating each of the items on a scale, or giving a full preference order for all alternatives in the set. However, choosing the most-preferred item in a given set is a natural task, and is a more robust measurement of preference than rating or fully-ranking items. For this reason, choice-based methods

have been shown to work well in practice (see Louviere, Hensher, and Swait 2000), and these are the types of queries we study. In this chapter, we formulate the problem of sequential choice-based preference elicitation as a finite horizon adaptive learning problem.

The marketing community has long been focused on preference elicitation and isolating features that matter the most to consumers. In this field, *conjoint analysis* is a class of methods that attempts to learn these important features by offering users a subset of alternatives (Green and Srinivasan 1978). Lately, there has been a push in the marketing community to design sequential methods that adaptively select the best subset of alternatives to offer the user. In the marketing research literature, this is referred to as adaptive choice-based conjoint analysis. In the past, geometrically-motivated heuristics have been used to adaptively choose questions (Toubia, Hauser, and Simester 2004). These heuristics have since evolved to include probabilistic modeling that captures the uncertainty in user responses (Toubia, Hauser, and Garcia 2007).

These problems are also tackled by the active learning community. For instance, Maldonado, Montoya, and Weber (2015) use existing support vector machine (SVM) technology to identify features users find important. In the context of preference elicitation in the active learning literature, there are two main approaches. The first is to take a non-parametric approach and infer a full preference ranking, labeling every pairwise combination of alternatives (Fürnkranz and Hüllermeier 2003). The benefit to this approach is the generality offered by a non-parametric model and its ability to capture realistic noise. Viewing preference learning as a generalized binary search problem, Nowak (2011) proves exponential convergence in probability to the correct preferential ordering for

all alternatives in a given set, and shows his algorithm is optimal to a constant factor. Unfortunately, this probabilistic upper bound is weakened by a coefficient that is quadratic in the total number of alternatives, and the running time of this optimal policy is proportional to the number of valid preferential orderings of all the alternatives. These issues are common for non-parametric ranking models. Using a statistical learning theoretic framework, Ailon (2012) develops an adaptive and computationally efficient algorithm to learn a ranking, but the performance guarantees are only asymptotic. In practice, one can only expect to ask a user a limited number of questions, and in this scenario, Yu, Goos, and Vandebroek (2012) show that taking a Bayesian approach to optimally and adaptively selecting questions is indispensable to the task of learning preferences for a given user. In the search for finite-time results and provable bounds, we opt to learn a parametric model using a Bayesian approach. In particular, this chapter largely focuses on a greedy policy that maximally reduces posterior entropy of a linear classifier, leveraging information theory to derive results pertaining to this policy.

Maximizing posterior entropy reduction has long been a suggested objective for learning algorithms (Lindley 1956; Bernardo 1979), especially within the context of active learning (MacKay 1992). But even within this paradigm of preference elicitation, there is a variety of work that depends on the user response model. For example, Dzyabura and Hauser (2011) study maximizing entropy reduction under different response heuristics, and Saure and Vielma (2016) uses ellipsoidal credibility regions to capture the current state of knowledge of a user's preferences. Using an entropy-based objective function allows one to leverage existing results in information theory to derive theoretical finite-time guarantees (Jedynak, Frazier, Sznitman, et al. 2012). Most simi-

lar to our methodology, Brochu, Brochu, and Freitas (2010) and Houlsby et al. (2011) model a user's utility function using a Gaussian process, updating the corresponding prior after each user response, and adaptively choose questions by minimizing an estimate of posterior entropy. However, while the response model is widely applicable and the method shows promise in practical situations, the lack of theoretical guarantees leaves much to be desired. Ideally, one would want concrete performance bounds for an entropy-based algorithm under a parameterized response model. In contrast, this proves information theoretic results in the context of adaptive choice-based preference elicitation for arbitrary feature-space dimension, leverages these results to derive bounds for performance, and shows that a greedy entropy reduction policy (hereafter referred to as *entropy pursuit*) optimally reduces posterior entropy of a linear classifier over the course of multiple choice-based questions. In particular, the main contributions are summarized as follows:

- In Section 4.2, we formally describe the response model for the user. For this response model, we prove a linear lower bound on the sequential entropy reduction over a finite number of questions in Section 4.3, and provide necessary and sufficient conditions for asking an optimal comparative question.
- Section 4.3.3 presents results showing that the linear lower bound can be attained by a greedy algorithm up to a multiplicative constant when we are allowed to fabricate alternatives (i.e., when the set of alternatives has a non-empty interior). Further, the bound is attained exactly with moderate conditions on the noise channel.
- Section 4.4 focuses on misclassification error, a more intuitive metric of measuring knowledge of a user's preferences, and explores a one-step

knowledge gradient policy that minimizes this metric in a greedy fashion. In the context of this metric, we show a Fano-type lower bound on the optimal policy in terms of an increasing linear function of posterior differential entropy.

- Lastly, in Section 4.4.5, we derive a new entropy policy that encapsulates the benefits of the knowledge gradient policy, and motivates the exploration of similarly structured policies.

4.2 Problem Specification

The alternatives $x^{(i)} \in \mathbb{R}^d$ are represented by d-dimensional feature vectors that encode all of their distinguishing aspects. Let $\mathbb{X}$ be the set of all such alternatives. Assuming a linear utility model, each user has her own linear classifier $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^d$ that encodes her preferences [1]. At time epoch k, given m alternatives $X_k = \{x_k^{(1)}, x_k^{(2)}, \ldots, x_k^{(m)}\} \in \mathbb{X}^m$, the user prefers to choose the alternative i that maximizes $\boldsymbol{\theta}^T x_k^{(i)}$. However, we do not observe this preference directly. Rather, we observe a signal influenced by a noise channel. In this case, the signal is the response we observe from the user.

Let $\mathbb{Z} = \{1, 2, \ldots, m\}$ denote the m possible alternatives. We define $\boldsymbol{Z}_k(X_k)$ to be the alternative that is consistent with our linear model after asking question X_k, that is, $\boldsymbol{Z}_k(X_k) = \min \left\{ \arg\max_{i \in \mathbb{Z}} \boldsymbol{\theta}^T x_k^{(i)} \right\}$. The minimum is just used as a tie-breaking rule; the specific rule is not important so long as it is deterministic. We do not observe $\boldsymbol{Z}_k(X_k)$, but rather observe a signal $\boldsymbol{Y}_k(X_k) \in \mathbb{Y}$, which depends on $\boldsymbol{Z}_k(X_k)$. We allow $\mathbb{Y}$ to characterize any type of signal that can be

[1] Throughout this chapter, we use boldface to denote a random variable.

received from posing questions in $\mathbb{X}$. In general, the density of the conditional distribution of $\boldsymbol{Y}_k(X_k)$ given $\boldsymbol{Z}_k(X_k) = z$ is denoted $f^{(z)}(\cdot)$. Here, we primarily consider the scenario in which $\mathbb{Y} = \mathbb{Z} = \{1, 2, \ldots, m\}$, where nature randomly perturbs $\boldsymbol{Z}_k(X_k)$ to some (possibly the same) element in $\mathbb{Z}$. In this scenario, the user's response to the preferred alternative is the signal $\boldsymbol{Y}_k(X_k)$, which is observed in lieu of the model-consistent "true response" $\boldsymbol{Z}_k(X_k)$. In this case, we define a noise channel stochastic matrix P by setting $P^{(zy)} = f^{(z)}(y)$ to describe what is called a *discrete noise channel*.

One sequentially asks the user questions and learns from each of their responses. Accordingly, let $\mathbb{P}_k$ be the probability measure conditioned on the σ-field generated by $\boldsymbol{\mathcal{Y}}_k = (\boldsymbol{Y}_\ell(X_\ell) : 1 \leq \ell \leq k-1)$. Similarly, let $\mathcal{Y}_k = \{Y_\ell(X_\ell) : 1 \leq \ell \leq k-1\}$ denote the history of user responses. As we update, we condition on the previous outcomes, and subsequently choose a question X_k that depends on all previous responses $\mathcal{Y}_k$ from the user. Accordingly, let policy π return a comparative question $X_k \in \mathbb{X}^m$ that depends on time epoch k and past response history $\mathcal{Y}_k$. The selected question X_k may also depend on i.i.d. random uniform variables, allowing for stochastic policies. We denote the space of all such policies π as Π. In this light, let $\mathbb{E}^\pi$ be the expectation operator induced by policy π.

We consider a specific noise model, which is highlighted in the following assumptions.

Noise Channel Assumptions. *For every time epoch k, signal $\boldsymbol{Y}_k(X_k)$ and true response $\boldsymbol{Z}_k(X_k)$ corresponding to comparative question X_k, we assume*

- *model-consistent response $\boldsymbol{Z}_k(X_k)$ is a deterministic function of question X and linear classifier $\boldsymbol{\theta}$, and*

- *given true response $\boldsymbol{Z}_k(X_k)$, signal $\boldsymbol{Y}_k(X_k)$ is conditionally independent of linear classifier $\boldsymbol{\theta}$ and previous history $\mathcal{Y}_k$, and*
- *the conditional densities $f = \{f^{(z)} : z \in \mathbb{Z}\}$ differ from each other on a set of Lebesgue measure greater than zero.*

The first two assumptions ensure that all the information regarding $\boldsymbol{\theta}$ is contained in some true response $\boldsymbol{Z}_k(X_k)$. In other words, the model assumes that no information about the linear classifier is lost if we focus on inferring the true response instead. The last assumption is focused on identifiability of the model: since we infer by observing a signal, it is critical that we can tell the conditional distributions of these signals apart, and the latter condition guarantees this.

One of the benefits this noise model provides is allowing us to easily update our beliefs of $\boldsymbol{\theta}$. For a given question $X \in \mathbb{X}^m$ and true response $z \in \mathbb{Z}$, let

$$A^{(z)}(X) = \left\{ \theta \in \Theta : \begin{array}{ll} \theta^T x^{(z)} \geq \theta^T x^{(i)} & \forall i > z \\ \theta^T x^{(z)} > \theta^T x^{(i)} & \forall i < z \end{array} \right\}. \tag{4.1}$$

These m sets form a partition of Θ that depend on the question X we ask at each time epoch, where each set $A^{(z)}$ corresponds to all linear classifiers $\boldsymbol{\theta}$ that are consistent with the true response $\boldsymbol{Z} = z$.

Let μ_k denote the prior measure of $\boldsymbol{\theta}$ at time epoch k. Throughout the thesis, we assume that μ_k is absolutely continuous with respect to d-dimensional Lebesgue measure, admitting a corresponding Lebesgue density p_k. At every epoch, we ask the user a comparative question that asks for the most preferred option in $\boldsymbol{X}_k = \{x_1, x_2, \ldots, x_m\}$. We observe signal $\boldsymbol{Y}_k(X_k)$, and accordingly update the prior.

Lemma 4.2.1. *Suppose that the Noise Channel Assumptions hold. Then we can write*

the posterior p_{k+1} as

$$p_{k+1}\left(\theta \mid \boldsymbol{Y}_k(X_k) = y\right) = \left(\frac{\sum_{z\in\mathbb{Z}} \mathbb{I}(\theta \in A^{(z)}(X_k))\ f^{(z)}(y)}{\sum_{z\in\mathbb{Z}} \mu_k\left(A^{(z)}(X_k)\right)\ f^{(z)}(y)}\right) p_k(\theta), \tag{4.2}$$

where $\mathbb{I}$ denotes the indicator function.

Proof. Using Bayes' rule, we see

$$\begin{aligned} p_{k+1}(\theta \mid \boldsymbol{Y}_k(X_k) = y) &\propto \mathbb{P}_k(\boldsymbol{Y}_k(X_k) = y \mid \boldsymbol{\theta} = \theta) \cdot p_k(\theta) \\ &= \sum_{z\in\mathbb{Z}} \mathbb{P}_k(\boldsymbol{Y}_k(X_k) = y \mid \boldsymbol{Z}_k(X_k) = z,\, \boldsymbol{\theta} = \theta) \cdot \mathbb{P}_k(\boldsymbol{Z}_k(X_k) = z \mid \boldsymbol{\theta} = \theta) \cdot p_k(\theta). \end{aligned}$$

Now we use a property of $\boldsymbol{Y}_k(X_k)$ and $\boldsymbol{Z}_k(X_k)$ from the Noise Channel Assumptions, namely that $\boldsymbol{Y}_k(X_k)$ and $\boldsymbol{\theta}$ are conditionally independent given $\boldsymbol{Z}_k(X_k)$. This implies

$$\begin{aligned} p_{k+1}(\theta \mid \boldsymbol{Y}_k(X_k) = y) &\propto \sum_{z\in\mathbb{Z}} \mathbb{P}_k(\boldsymbol{Y}_k(X_k) = y \mid \boldsymbol{Z}_k(X_k) = z) \cdot \mathbb{P}_k(\boldsymbol{Z}_k(X_k) = z \mid \boldsymbol{\theta} = \theta) \cdot p_k(\theta) \\ &= \sum_{z\in\mathbb{Z}} f^{(z)}(y) \cdot \mathbb{I}\left(\theta \in A^{(z)}(X_k)\right) \cdot p_k(\theta), \end{aligned}$$

where the last line is true because $\boldsymbol{Z}_k(X_k)$ is a deterministic function of $\boldsymbol{\theta}$ and X_k. Normalizing to ensure the density integrates to one gives the result. □

The Noise Channel Assumptions allow us to easily update the prior on $\boldsymbol{\theta}$. As we will see next, they also allow us to easily express the conditions required to maximize one-step entropy reduction.

4.3 Posterior Entropy

We focus on how we select the alternatives we offer to the user. First, we need to choose a metric to evaluate the effectiveness of each question. One option is to

use a measure of dispersion of the posterior distribution of $\boldsymbol{\theta}$, and the objective is to decrease the amount of spread as much as possible with every question. Along these lines, we elect to use differential entropy for its tractability.

For a probability density p, the differential entropy of p is defined as

$$H(p) = \int_{\Theta} -p(\theta) \log_2 p(\theta)\, d\theta.$$

For the entirety of this thesis, all logarithms are base-2, implying that both Shannon and differential entropy are measured in bits. Because we ask the user multiple questions, it is important to incorporate the previous response history $\mathcal{Y}_k$ when considering posterior entropy. Let H_k be the entropy operator at time epoch k such that $H_k(\boldsymbol{\theta}) = H(\boldsymbol{\theta} \,|\, \mathcal{Y}_k)$, which takes into account all of the previous observation history $\mathcal{Y}_k$. Occasionally, when looking at the performance of a policy π, we would want to randomize over all such histories. This is equivalent to the concept of *conditional entropy*, with $H^{\pi}(\boldsymbol{\theta} \,|\, \boldsymbol{\mathcal{Y}}_k) = \mathbb{E}^{\pi}\left[H_k(\boldsymbol{\theta})\right]$.

Throughout the thesis, we represent discrete distributions as vectors. Accordingly, define $\Delta^m = \{u \in \mathbb{R}^m : \sum_z u^{(z)} = 1,\, u \geq 0\}$ to be the set of discrete probability distributions over m alternatives. For a probability distribution $u \in \Delta^m$, we define $h(u)$ to be the Shannon entropy of that discrete distribution, namely

$$h(u) = \sum_{z \in \mathbb{Z}} -u^{(z)} \log_2 u^{(z)}.$$

Here, we consider discrete probability distributions over the alternatives we offer, which is why distributions u are indexed by $z \in \mathbb{Z}$.

Since stochastic matrices are be used to model some noise channels, we develop similar notation for matrices. Let $\Delta^{m \times m}$ denote the set of $m \times m$ row-stochastic matrices. Similarly to how we defined the Shannon entropy of a vec-

tor, we define $h(P)$ as an m-vector with the Shannon entropies of the rows of P as its components. In other words,

$$h\left(P\right)^{(z)} = \sum_{y \in \mathbb{Y}} -P^{(zy)} \log_2 P^{(zy)}.$$

An important concept in information theory is *mutual information*, which measures the entropy reduction of a random variable when conditioning on another. It is natural to ask about the relationship between the information gain of $\boldsymbol{\theta}$ and that of $\boldsymbol{Z}_k(X_k)$ after observing signal $\boldsymbol{Y}_k(X_k)$. Mutual information in this context is defined as

$$I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = H_k(\boldsymbol{\theta}) - H_k(\boldsymbol{\theta} \mid \boldsymbol{Y}_k(X_k)). \tag{4.3}$$

One critical property of mutual information is that it is symmetric, or in other words, $I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{\theta})$ (see Cover 1991, p. 20). In the context of our model, this means that observing signal $\boldsymbol{Y}_k(X_k)$ gives us the same amount of information about linear classifier $\boldsymbol{\theta}$ as would observing the linear classifier would provide about the signal. This is one property we exploit throughout this work, since the latter case only depends on the noise channel, which by assumption does not change over time. We show in Theorem 4.3.1 below that the Noise Channel Assumptions allow us to determine how the noise channel affects the posterior entropy of linear classifier $\boldsymbol{\theta}$.

The first identity, given by (4.4), says that the noise provides an additive effect with respect to entropy, particularly because the noise does not depend on $\boldsymbol{\theta}$ itself. The second identity, given by (4.5), highlights the fact that $\boldsymbol{Y}_k(X_k)$ provides the same amount of information on the linear classifier $\boldsymbol{\theta}$ as it does on the true answer $\boldsymbol{Z}_k(X_k)$ for a given question. This means that the entropy of both $\boldsymbol{\theta}$ and $\boldsymbol{Z}_k(X_k)$ are reduced by the same number of bits when asking question X_k.

Intuitively, asking the question that would gain the most clarity from a response would also do the same for the underlying linear classifier. This is formalized in Theorem 4.3.1 below.

Theorem 4.3.1. *The following information identities hold under the Noise Channel Assumptions for all time epochs k. The first is the* ***Noise Separation Equality****, namely*

$$H_k(\boldsymbol{\theta} \,|\, Y_k(X_k)) = H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Z}_k(X_k)) + H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{Y}_k(X_k)), \tag{4.4}$$

and the ***Noise Channel Information Equality****, given by*

$$I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = I(\boldsymbol{Z}_k(X_k); \boldsymbol{Y}_k(X_k)), \tag{4.5}$$

where the latter term does not depend on response history $\mathcal{Y}_k$.

Proof. Using the symmetry of mutual information,

$$H_k(\boldsymbol{\theta} \,|\boldsymbol{Y}_k(X_k)) - H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Y}_k(X_k), \boldsymbol{Z}_k(X_k)) = H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{Y}_k(X_k)) - H(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{\theta}, \boldsymbol{Y}_k(X_k)).$$

Further, we know $H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Y}_k(X_k), \boldsymbol{Z}_k(X_k)) = H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Z}_k(X_k))$ because $\boldsymbol{Y}_k(X_k)$ and $\boldsymbol{\theta}$ are conditionally independent given $\boldsymbol{Z}_k(X_k)$. Also, since $\boldsymbol{Z}_k(X_k)$ is a function of $\boldsymbol{\theta}$ and X_k, it must be that $H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{\theta}, \boldsymbol{Y}_k(X_k)) = 0$. Putting these together gives us the first identity. To prove the second identity, we use the fact that

$$H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Z}_k(X_k)) + H_k(\boldsymbol{Z}_k(X_k)) = H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{\theta}) + H_k(\boldsymbol{\theta}).$$

Again, $H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{\theta}) = 0$ because $\boldsymbol{Z}_k(X_k)$ is a function of $\boldsymbol{\theta}$ and X_k. This yields $H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Z}_k(X_k)) = H_k(\boldsymbol{\theta}) - H_k(\boldsymbol{Z}_k(X_k))$. Substitution into the first identity gives us

$$H_k(\boldsymbol{\theta}) - H_k(\boldsymbol{\theta} \,|\, \boldsymbol{Y}_k(X_k)) = H_k(\boldsymbol{Z}_k(X_k)) - H_k(\boldsymbol{Z}_k(X_k) \,|\, \boldsymbol{Y}_k(X_k)),$$

which is (4.5), by definition of mutual information. Finally, by the Noise Channel Assumptions, signal $\boldsymbol{Y}_k(X_k)$ is conditionally independent of history $\mathcal{Y}_k$ given $\boldsymbol{Z}_k(X_k)$, and therefore, $I_k(\boldsymbol{Z}_k(X_k); \boldsymbol{Y}_k(X_k)) = I(\boldsymbol{Z}_k(X_k); \boldsymbol{Y}_k(X_k))$. □

The entropy pursuit policy is one that maximizes the reduction in entropy of the linear classifier, namely $I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = H_k(\boldsymbol{\theta}) - H_k(\boldsymbol{\theta} \mid \boldsymbol{Y}_k(X_k))$, at each time epoch. We leverage the results from Theorem 4.3.1 to find conditions on questions that maximally reduce entropy in the linear classifier $\boldsymbol{\theta}$. However, we first need to introduce some more notation.

For a noise channel parameterized by $f = \{f^{(z)} : z \in \mathbb{Z}\}$, let φ denote the function on domain Δ^m defined as

$$\varphi(u\,; f) = H\left(\sum_{z\in\mathbb{Z}} u^{(z)} f^{(z)}\right) - \sum_{z\in\mathbb{Z}} u^{(z)}\, H\left(f^{(z)}\right). \tag{4.6}$$

We will show in Theorem 4.3.2 that (4.6) refers to the reduction in entropy from asking a question, where the argument $u \in \Delta^m$ depends on the question. We define the channel capacity over noise channel f, denoted $C(f)$, to be the supremum of φ over this domain, namely

$$C(f) = \sup_{u\in\Delta^m} \varphi(u\,; f), \tag{4.7}$$

and this denotes the maximal amount of entropy reduction at every step. These can be similarly defined for a discrete noise channel. For a noise channel parameterized by transmission matrix P, we define

$$\varphi(u\,;\, P) = h(u^T P) - u^T h(P), \tag{4.8}$$

and $C(P)$ is correspondingly the supremum of $\varphi(\cdot\,;\, P)$ in its first argument. In Theorem 4.3.2 below, we show that $\varphi(u\,;\, f)$ is precisely the amount of entropy over linear classifiers $\boldsymbol{\theta}$ reduced by asking a question with respective predictive distribution u under noise channel f.

Theorem 4.3.2. *For a given question $X \in \mathbb{X}^m$, define $u_k(X) \in \Delta^m$ such that $u_k^{(z)}(X) = \mu_k\left(A^{(z)}(X)\right)$ for all $z \in \mathbb{Z}$. Suppose that the Noise Channel Assumptions*

hold. Then for a fixed noise channel parameterized by $f = \{f^{(z)} : z \in \mathbb{Z}\}$,

$$I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = \varphi\left(u_k(X_k)\,;\, f\right). \tag{4.9}$$

Consequently, for all time epochs k*, we have*

$$\sup_{X_k \in \mathbb{X}^m} I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) \leq C(f), \tag{4.10}$$

and there exists $u_* \in \Delta^m$ *that attains the supremum. Moreover, if there exists some* $X_k \in \mathbb{X}^m$ *such that* $u_k(X_k) = u_*$*, then the upper bound is attained.*

Proof. We first use (4.5) from Theorem 4.3.1, namely that $I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) = I_k(\boldsymbol{Z}_k(X_k); \boldsymbol{Y}_k(X_k))$. We use the fact that mutual information is symmetric, meaning that the entropy reduction in $\boldsymbol{Z}_k(X_k)$ while observing $\boldsymbol{Y}_k(X_k)$ is equal to that in $\boldsymbol{Y}_k(X_k)$ while observing $\boldsymbol{Z}_k(X_k)$. Putting this together with the definition of mutual information yields

$$\begin{aligned}
I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)) &= I_k(\boldsymbol{Z}_k(X_k); \boldsymbol{Y}_k(X_k)) \\
&= H_k(\boldsymbol{Y}_k(X_k)) - H_k(\boldsymbol{Y}_k(X_k) \,|\, \boldsymbol{Z}_k(X_k)) \\
&= H\left(\sum_{z \in \mathbb{Z}} \mathbb{P}_k(\boldsymbol{Z}_k(X_k) = z)\, f^{(z)}\right) - \sum_{z \in \mathbb{Z}} \mathbb{P}_k(\boldsymbol{Z}_k(X_k) = z)\, H(f^{(z)}) \\
&= H\left(\sum_{z \in \mathbb{Z}} \mu_k\left(A^{(z)}(X_k)\right)\, f^{(z)}\right) - \sum_{z \in \mathbb{Z}} \mu_k\left(A^{(z)}(X_k)\right)\, H(f^{(z)}),
\end{aligned}$$

which is equal to $\varphi(u_k(X_k)\,; f)$, where $u_k^{(z)}(X_k) = \mu_k\left(A^{(z)}(X_k)\right)$. Therefore, the optimization problem in (4.10) is equivalent to

$$\sup_{X_k \in \mathbb{X}^m} \varphi\left(u_k(X_k)\,; f\right).$$

Since $\{u_k(X) : X \in \mathbb{X}^m\} \subseteq \Delta^m$, we can relax the above problem to

$$\sup_{u \in \Delta^m} \varphi(u\,; f).$$

It is known that mutual information is concave in its probability mass function (see Cover 1991, p. 31), and strictly concave when the likelihood functions $f^{(z)}$ differ on a set of positive measure. Thus, for a fixed noise channel f, $\varphi(\cdot\,;\,f)$ is concave on Δ^m, a compact convex set, implying an optimal solution u_* exists and the optimal objective value $C(f) > 0$ is attained. Further, if we can construct some $X_k \in \mathbb{X}^m$ such that $\mu_k\left(A^{(z)}(X_k)\right) = u_*^{(z)}$ for every $z \in \mathbb{Z}$, then the upper bound is attained. □

We have shown that entropy reduction of the posterior of $\boldsymbol{\theta}$ depends only on the implied predictive distribution of a given question and structure of the noise channel. If we are free to fabricate alternatives to achieve the optimal predictive distribution, then we reduce the entropy of the posterior by a fixed amount $C(f)$ at every time epoch. Perhaps the most surprising aspect of this result is the fact that the history $\mathcal{Y}_k$ plays no role in the *amount* of entropy reduction, which is important for showing that entropy pursuit is an optimal policy for reducing entropy over several questions.

In practice, one can usually ask more than one question, and it is natural to ask if there is an extension that gives us a bound on the posterior entropy after asking several questions. Using the results in Theorem 4.3.2, we can derive an analogous lower bound for this case.

Corollary 4.3.3. *For a given policy $\pi \in \Pi$, we can write the entropy of linear classifier $\boldsymbol{\theta}$ after K time epochs as*

$$H(\boldsymbol{\theta}) - H^{\pi}\left(\boldsymbol{\theta} \,|\, \boldsymbol{\mathcal{Y}}_K\right) = \mathbb{E}^{\pi}\left[\sum_{k=1}^{K} \varphi(u_k(X_k)\,;\,f)\right], \tag{4.11}$$

and a lower bound for the differential entropy of $\boldsymbol{\theta}$ after asking K questions is given

below by

$$\inf_{\pi \in \Pi} H^{\pi}(\boldsymbol{\theta} \,|\, \boldsymbol{\mathcal{Y}}_K) \geq H(\boldsymbol{\theta}) - K \cdot C(f). \tag{4.12}$$

Further, if for a given policy π and history $\mathcal{Y}_k$ indicates that comparative question X_k should be posed to the user, then the lower bound is attained if and only if $u_k(X_k) = u_*$, with u_* as defined in Theorem 4.3.2. Thus, entropy pursuit is an optimal policy.

Proof. Using the information chain rule, we can write the entropy reduction for a generic policy $\pi \in \Pi$ as

$$\begin{aligned} H(\boldsymbol{\theta}) - H^{\pi}(\boldsymbol{\theta} \,|\, \boldsymbol{\mathcal{Y}}_K) &= I^{\pi}\left(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}_K\right) \\ &= \sum_{k=1}^{K} \mathbb{E}^{\pi}\left[I_k\left(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k)\right)\right] \leq K \cdot C(f), \end{aligned}$$

where the last inequality comes directly from Theorem 4.3.2, and the upper bound is attained if and only if $u_k(X_k) = u_*$ for every $k = 1, 2, \ldots, K$. This coincides with the entropy pursuit policy. □

Essentially, Corollary 4.3.3 shows that the greedy entropy reduction policy is, in fact, the optimal policy over any time horizon. However, there is still an important element that is missing: how can we ensure that there exists some alternative that satisfies the entropy pursuit criteria? We address this important concern in Section 4.3.3.

4.3.1 Optimality Conditions for Predictive Distribution

Because of the properties of entropy, the noise channel function φ has a lot of structure. We use this structure to find conditions for a non-degenerate opti-

mal predictive distribution u_* as well as derive sensitivity results that allow the optimality gap of a close-to-optimal predictive distribution to be estimated.

Before we prove structural results for the channel equation φ, some more information theoretic notation should be introduced. Given two densities $f^{(i)}$ and $f^{(j)}$, the *cross entropy* of these two densities is defined as

$$H\left(f^{(i)}, f^{(j)}\right) = \int_{\mathbb{Y}} -f^{(i)}(y) \log_2 f^{(j)}(y)\, dy.$$

Using the definition of cross entropy, the Kullback-Leibler divergence between two densities $f^{(i)}$ and $f^{(j)}$ is defined as

$$\mathrm{KL}\left(f^{(i)} \,\middle\|\, f^{(j)}\right) = H(f^{(i)}, f^{(j)}) - H(f^{(i)}).$$

Kullback-Leibler divergence is a tractable way of measuring the difference of two densities. An interesting property of Kullback-Leibler divergence is that for any densities $f^{(i)}$ and $f^{(j)}$, $\mathrm{KL}(f^{(i)} \| f^{(j)}) \geq 0$, with equality if and only if $f^{(i)} = f^{(j)}$ almost surely. Kullback-Leibler divergence plays a crucial role the first-order information for the channel equation φ.

We now derive results that express the gradient and Hessian of φ in terms of the noise channel, which can either be parameterized by f in the case of a density, or by a fixed transmission matrix P in the discrete noise channel case. For these results to hold, we require the cross entropy $H(f^{(i)}, f^{(j)})$ to be bounded in magnitude for all $i, j \in \mathbb{Z}$, which is an entirely reasonable assumption.

Lemma 4.3.4. *For a fixed noise channel characterized by $f = \{f^{(z)} : z \in \mathbb{Z}\}$, if the cross entropy terms $H(f^{(i)}, f^{(j)})$ are bounded for all $i, j \in \mathbb{Z}$, then the first and second*

partial derivatives of φ with respect to u are given by

$$\frac{\partial \varphi(u\,;\,f)}{\partial u^{(z)}} = \mathrm{KL}\left(f^{(z)} \,\middle\|\, \sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}\right) - \xi$$
$$\frac{\partial^2 \varphi(u\,;\,f)}{\partial u^{(z)}\,\partial u^{(w)}} = -\xi \int_{\mathbb{Y}} \frac{f^{(z)}(y)\, f^{(w)}(y)}{\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}(y)}\, dy,$$

where $\xi = \log_2 e$, and $\mathrm{KL}(\cdot \,\|\, \cdot)$ is the Kullback-Leibler Divergence.

In particular, if a discrete noise channel is parameterized by transmission matrix P, the gradient and Hessian matrix of φ can be respectively expressed as

$$\nabla_u \varphi(u\,;\,P) = -P \log_2 \left(P^T u\right) - h(P) - \xi e$$
$$\nabla_u^2 \varphi(u\,;\,P) = -\xi\, P \left(\mathrm{diag}\left(u^T P\right)\right)^{-1} P^T,$$

where the logarithm is taken component-wise.

Proof. We first prove the result in the more general case when the noise channel is parameterized by f. From the definition of φ,

$$\varphi(u\,;\,f) = \int_{\mathbb{Y}} -\left(\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}(y)\right) \log_2 \left(\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}(y)\right)\, dy - \sum_{i\in\mathbb{Z}} u^{(i)} H(f^{(i)}).$$

Since $t \mapsto -\log t$ is convex, by Jensen's inequality, $H(f^{(z)}, \sum_i u^{(i)} f^{(i)}) \leq \sum_i u^{(i)} H(f^{(z)}, f^{(i)})$, which is bounded. By the Dominated Convergence Theorem, we can switch differentiation and integration operators, and thus,

$$\begin{aligned}\frac{\partial}{\partial u^{(z)}} \varphi(u\,;\,f) &= \int_{\mathbb{Y}} -f^{(z)}(y) \log_2 \left(\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}(y)\right)\, dy - \xi - H(f^{(z)}) \\ &= \mathrm{KL}\left(f^{(z)} \,\middle\|\, \sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}\right) - \xi.\end{aligned}$$

Concerning the second partial derivative, Kullback-Leibler divergence is always non-negative, and therefore, Monotone Convergence Theorem again allows us to switch integration and differentiation, yielding

$$\frac{\partial^2 \varphi(u\,;\,f)}{\partial u^{(z)}\,\partial u^{(w)}} = -\xi \int_{\mathbb{Y}} \frac{f^{(z)}(y)\, f^{(w)}(y)}{\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}(y)}\, dy.$$

For the discrete noise channel case, the proof is analogous to above, using Equation (4.8). Vectorizing yields

$$\begin{aligned}\nabla_u \varphi(u\,;P) &= -P\left(\log_2(P^T u) + \xi e\right) - h(P) \\ &= -P\log_2\left(P^T u\right) - h(P) - \xi e.\end{aligned}$$

Similarly, the discrete noise channel analogue for the second derivative is

$$\frac{\partial^2 \varphi(u\,;P)}{\partial u^{(z)} \partial u^{(w)}} = -\xi \sum_{y\in\mathbb{Y}} \frac{P^{(zy)} P^{(wy)}}{\sum_{i\in\mathbb{Z}} u^{(i)} P^{(iy)}},$$

and vectorizing gives us the Hessian matrix. □

One can now use the results in Lemma 4.3.4 to find conditions for an optimal predictive distribution for a noise channel parameterized either by densities $f = \{f^{(z)} : z \in \mathbb{Z}\}$ or transmission matrix P. There has been much research on how to find the optimal predictive distribution u_* given a noise channel, as in Gallager (1968). Generally, there are two methods for finding this quantity. The first relies on solving a constrained concave maximization problem by using a first-order method. The other involves using the Karush-Kuhn-Tucker conditions necessary for an optimal solution (see Gallager 1968, p. 91 for proof).

Theorem 4.3.5 (Gallager)**.** *Given a noise channel parameterized by $f = \{f^{(z)} : z \in \mathbb{Z}\}$, the optimal predictive distribution u_* satisfies*

$$\mathrm{KL}\left(f^{(z)} \,\middle\|\, \sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}\right) \begin{cases} = C(f) & u_*^{(z)} > 0 \\ < C(f) & u_*^{(z)} = 0, \end{cases}$$

where $C(f)$ is the channel capacity.

The difficulty in solving this problem comes from determining whether or not $u_*^{(z)} > 0$. In the context of preference elicitation, when fixing the number of

offered alternatives m, it is critical for every alternative to contribute to reducing uncertainty. However, having a noise channel where $u_*^{(z)} = 0$ implies that it is more efficient to learn without offering alternative z.

To be specific, we say that a noise channel parameterized by $f = \{f^{(z)} : z \in \mathbb{Z}\}$ is *admissible* if there exists some $f_* \in \operatorname{Int}(\operatorname{Hull}(f))$ such that for all $z \in \mathbb{Z}$,

$$\operatorname{KL}\left(f^{(z)} \,\middle\|\, f_*\right) = C$$

for some $C > 0$. Otherwise, we say the noise channel is *inadmissible*. Admissibility is equivalent the existence of a predictive distribution $u_* > 0$ where all m alternatives are used to learn a user's preferences. For pairwise comparisons, any noise channel where $f^{(1)}$ and $f^{(2)}$ differ on a set of non-zero Lebesgue measure is admissible. Otherwise, for $m > 2$, there are situations when $u_*^{(z)} = 0$ for some $z \in \mathbb{Z}$, and Lemma 4.3.6 provides one of them. In particular, if one density $f^{(z)}$ is a convex combination of any of the others, then the optimal predictive distribution will always have $u_*^{(z)} = 0$.

Lemma 4.3.6. *Suppose the noise channel is parameterized by densities $f = \{f^{(z)} : z \in \mathbb{Z}\}$, and its corresponding optimal predictive distribution is u_*. If there exists $\lambda^{(i)} \geq 0$ for $i \neq z$ such that $\sum_{i \neq z} \lambda^{(i)} = 1$ and $f^{(z)}(y) = \sum_{i \neq z} \lambda^{(i)} f^{(i)}(y)$ for all $y \in \mathbb{Y}$, then $u_*^{(z)} = 0$.*

Proof. Suppose $f^{(z)} = \sum_{i \neq z} \lambda^{(i)} f^{(i)}$. Take any $u \in \Delta^m$ such that $u^{(z)} > 0$. We will construct a $\bar{u} \in \Delta^m$ such that $\bar{u}^{(z)} = 0$ and $\varphi(\bar{u}\,;\, f) > \varphi(u\,;\, f)$. Define $\bar{u}$ as

$$\bar{u}^{(i)} = \begin{cases} u^{(i)} + \lambda^{(i)} u^{(z)} & i \neq z \\ 0 & i = z. \end{cases}$$

It is easy to verify that $\sum_i \bar{u}^{(i)} f^{(i)} = \sum_i u^{(i)} f^{(i)}$. But since entropy is strictly

concave, we have $H\left(f^{(z)}\right) > \sum_{i \neq z} \lambda^{(i)} f^{(i)}$. Consequently,

$$\begin{aligned}
\varphi(u\,;\,f) &= H\left(\sum_{i\in\mathbb{Z}} u^{(i)} f^{(i)}\right) - \sum_{i\in\mathbb{Z}} u^{(i)} H\left(f^{(i)}\right) \\
&= H\left(\sum_{i\neq z} \bar{u}^{(i)} f^{(i)}\right) - \sum_{i\neq z} u^{(i)} H(f^{(i)}) - u^{(z)} H(f^{(z)}) \\
&< H\left(\sum_{i\neq z} \bar{u}^{(i)} f^{(i)}\right) - \sum_{i\neq z} u^{(i)} H(f^{(i)}) - u^{(z)} \sum_{i\neq z} \lambda^{(i)} H(f^{(i)}) \\
&= H\left(\sum_{i\neq z} \bar{u}^{(i)} f^{(i)}\right) - \sum_{i\neq z} \bar{u}^{(i)} H(f^{(i)}) = \varphi(\bar{u}\,;\,f),
\end{aligned}$$

and therefore, one can always increase the objective value of φ by setting $u^{(z)} = 0$. □

Of course, there are other cases where the predictive distribution u_* is not strictly positive for every $z \in \mathbb{Z}$. For example, even if one of the densities is an *approximate* convex combination, the optimal predictive distribution would likely still have $u_*^{(z)} = 0$. In general, there is no easy condition to check whether or not $u_* > 0$. However, our problem assumes m is relatively small, and so it is simpler to find u_* and confirm the channel is admissible. In the case of a discrete noise channel, Shannon and Weaver (1948) gave an efficient way to do this by solving a relaxed version of the concave maximization problem, provided that the transmission matrix P is invertible.

Theorem 4.3.7 (Shannon)**.** *For a discrete noise channel parameterized by a non-singular transmission matrix P, let*

$$v = \frac{\exp\left(-\xi^{-1} P^{-1} h(P)\right)}{e^T \exp\left(-\xi^{-1} P^{-1} h(P)\right)}, \tag{4.13}$$

where the exponential is taken component-wise. If there exists $u > 0$ such that $u^T P = v^T$, then $u \in \mathrm{Int}(\Delta^m)$ is the optimal predictive distribution, meaning that

$\nabla_u \varphi(u\,;P) = \beta e$ *for some* $\beta \in \mathbb{R}$*, and* $\varphi(u_*\,;P) = C(P)$*, and the noise channel is admissible. Otherwise, then there exists some* $z \in \mathbb{Z}$ *such that* $u^{(z)} = 0$*, and the noise channel is inadmissible.*

Proof. Using (4.8) and Lagrangian relaxation,

$$\begin{aligned}\sup_{u:\, e^T u = 1} \varphi(u\,;P) &= \sup_{u:\, e^T u = 1} h(u^T P) - u^T h(P) \\ &= \sup_{u \in \mathbb{R}^m} \inf_{\lambda \in \mathbb{R}} h(u^T P) - u^T h(P) - \lambda \left(e^T u - 1\right).\end{aligned}$$

Differentiating with respect to u and setting equal to zero yields

$$-P \log_2 \left(P^T u\right) - h(P) + \xi e - \lambda e = 0,$$

and since P is invertible,

$$-\log_2 \left(P^T u\right) = P^{-1} h(P) + (\lambda - \xi) e,$$

since $Pe = e$ for all stochastic matrices P. Algebra yields

$$\begin{aligned}P^T u &= \exp\left(-\xi^{-1} P^{-1} h(P) + (\lambda/\xi - 1) e\right) \\ &= \Lambda \cdot \exp\left(-\xi^{-1} P^{-1} h(P)\right),\end{aligned}$$

where $\Lambda = \exp(\lambda/\xi - 1)$ is some positive constant. We require $e^T u = 1$, and if $u^T P = v^T$, it must be that

$$u^T e = u^T P e = v^T e,$$

implying that $e^T u = 1$ if and only if $e^T v = 1$. Hence, Λ is a normalizing constant that allows $v^T P = 1$. Thus, we can set v as in (4.13), and now it is clear that $v \in \Delta^m$. We can invert P to find an explicit form for u, but $P^{-T} v$ is only feasible for the original optimization problem if it is non-negative. However, if there exists some $u \in \Delta^m$ such that $u^T P = v^T$, then the optimal solution to the relaxed problem is feasible for the original optimization problem, proving the theorem.

□

If there does not exist some $u \geq 0$ that satisfied $u^T P = v^T$ for v defined in (4.13), then the non-negativity constraint would be tight, and $u_*^{(z)} = 0$ for some $z \in \mathbb{Z}$. In this case, the noise channel is inadmissible, because it implies asking the optimal question under entropy pursuit would assign zero probability to one of the alternatives being the model consistent answer, and thus posits a question of strictly less than m alternatives to the user.

The condition of P being non-singular has an enlightening interpretation. Having a non-singular transmission matrix implies there would be no two distinct predictive distributions for $\boldsymbol{Z}_k(X_k)$ that yield the same predictive distribution over $\boldsymbol{Y}_k(X_k)$. This is critical for the model to be identifiable, and prevents the previous problem of having one row of P being a convex combination of other rows. The non-singular condition is reasonable in practice: it is easy to verify that matrices in the form $P = \alpha I + (1 - \alpha) v e^T$ for some $v \in \Delta^m$ is invertible if and only if $\alpha > 0$. Transmission matrices of this type are fairly reasonable: with probability α, the user selects the "true response," and with probability $(1 - \alpha)$, the user selects from discrete distribution v, regardless of $\boldsymbol{Z}_k(X_k)$. The symmetric noise channel is a special case of this. In general, if one models $P = \alpha I + (1-\alpha)S$, where S is an $m \times m$ stochastic matrix, then P is non-singular if and only if $-\alpha/(1 - \alpha)$ is not an eigenvalue of S, which guarantees that P is invertible when $\alpha > 1/2$. Nevertheless, regardless of whether or not P is singular, it is relatively easy to check the admissibility of a noise channel, and consequently conclude whether or not it is a good modeling choice for the purpose of preference elicitation.

4.3.2 Sensitivity Analysis

In reality, we cannot always fabricate alternatives so that the predictive distribution is exactly optimal. In many instances, the set of alternatives $\mathbb{X}$ is finite. This prevents us from choosing an X_k such that $u_k(X_k) = u_*$ exactly. But if we can find a question that has a predictive distribution that is sufficiently close to optimal, then we can reduce the entropy at a rate that is close to the channel capacity. Below, we elaborate on our definition of sufficiently close by showing φ is strongly concave, using the Hessian to construct quadratic upper and lower bounds on the objective function φ.

Theorem 4.3.8. *If there exists $u_* \in \Delta^m$ such that $u_* > 0$ and $\varphi(u_* ;\, f) = C(f)$ (i.e., if the noise channel is admissible), then there exist constants $0 \leq r(f) \leq R(f)$ such that*

$$r(f) \cdot \|u - u_*\|^2 \leq C(f) - \varphi(u ;\, f) \leq R(f) \cdot \|u - u_*\|^2.$$

Further, suppose transmission matrix P encoding a discrete noise channel is non-singular, and has minimum probability $\kappa_1 = \min_{zy} P^{(zy)} > 0$, maximum probability $\kappa_2 = \max_{zy} P^{(zy)}$, channel capacity $C(P)$ and distribution u_* such that $\varphi(u_* ; P) = C(P)$. If $u_* > 0$, we have

$$\frac{\xi}{2\kappa_2} \left\|(u - u_*)^T P\right\|^2 \leq C(P) - \varphi(u ; P) \leq \frac{\xi}{2\kappa_1} \left\|(u - u_*)^T P\right\|^2$$

for all $u \in \Delta^m$, with $\xi = \log_2 e$.

Proof. The (z, w) component of $-\nabla^2 \varphi(\cdot ;\, f)$ is lower bounded by

$$\int_{\mathbb{Y}} \frac{f^{(z)}(y) f^{(w)}(y)}{\sum_{i \in \mathbb{Z}} u^{(i)} f^{(i)}(y)} \, dy \geq \frac{1}{\max_{i \in \mathbb{Z},\, y \in \mathbb{Y}} f^{(i)}(y)} \int_{\mathbb{Y}} f^{(z)}(y) f^{(w)}(y) \, dy, \tag{4.14}$$

since the denominator can be upper bounded. Let M denote the $m \times m$ matrix with its (z, w) component equal to the right-most term in (4.14) above. Since

it can be written as a Gram matrix for an integral product space, M is positive semi-definite, and it is clear that $M \preceq -\nabla^2\varphi(u\,;\,f)$ for all $u \in \Delta^m$. Correspondingly, let $r(f)$ be the smallest eigenvalue of M.

For an upper bound, we employ a different approach. Let $q_R(u) = C - (R/2)\,\|u - u_*\|^2$ denote the implied quadratic lower bound to φ. It is clear that $q_R(u) \geq 0$ if and only if $\|u - u^*\| \leq \sqrt{2C/R}$. Since φ is a non-negative function, we only need to find R so that q_R is a lower bound when $q_R(u) > 0$. Consider

$$\begin{aligned} \inf_{R} \quad & R \\ \text{s.t.} \quad & q_R(u) \leq \varphi(u\,;\,f) \quad \forall u:\ \|u - u_*\| < \sqrt{2C/R}. \end{aligned}$$

The problem is feasible since $\nabla^2\varphi$ is continuous about u_*, and hence, there exists an R sufficiently large such that q_R is a lower bound of φ in a small neighborhood around u_*. The problem is obviously bounded since the optimal value must be greater than $r(f)$. Now let $R(f)$ denote the optimal value to the problem above. Taylor expanding about u_* yields

$$r(f) \cdot \|u - u_*\|^2 \leq C(f) - \varphi(u\,;\,f) + \nabla_u\,\varphi(u_*\,;\,f)^T(u - u_*) \leq R(f) \cdot \|u - u_*\|^2.$$

But since $u_* > 0$, optimality requires $\nabla_u\,\varphi(u_*\,;\,f) = \beta e$ for some $\beta \in \mathbb{R}$. Since u and u_* are both probability distributions,

$$\nabla_u\,\varphi(u_*\,;\,f)^T(u - u_*) = \beta e^T(u - u_*) = 0,$$

and hence the lower and upper bounds hold.

The proof for the discrete noise channel case is similar, with the exception being that we can easily find constants that satisfy the quadratic lower and upper bounds of the optimality gap $C(P) - \varphi(u_*\,;\,P)$. We observe the elements of $u^T P$ are lower bounded by $\kappa_1 = \min_{zy} P_{zy}$ and upper bounded by $\kappa_2 = \max_{zy} P_{zy}$.

Therefore, for all $u \in \Delta^m$,

$$\xi\kappa_2^{-1}\, PP^T \preceq \nabla_u^2\, \varphi(u\,;P) \preceq \xi\kappa_1^{-1}\, PP^T.$$

Lemma 4.3.4 implies that $\nabla_u \varphi(u_* \mid P) = \beta e$ since $u_* > 0$. Thus, Taylor expansion about u_* yields

$$\frac{\xi}{2\kappa_1} \left\|(u-u_*)^T P\right\|^2 \leq C(f) - \varphi(u;P) + \nabla_u\, \varphi(u_*;P)^T(u-u_*) \leq \frac{\xi}{2\kappa_2} \left\|(u-u_*)^T P\right\|^2.$$

Lastly, since both u_* and u are distributions, their components sum to one, implying $\nabla_u\, \varphi(u\,;\, P)^T(u-u_*) = 0$. The result directly follows. □

This gives us explicit bounds on the entropy reduction in terms of the L_2 distance of a question's predictive distribution from the optimal predictive distribution. In theory, this allows us to enumerate through all questions in $\mathbb{X}^m$ and select that whose predictive distribution is closest to optimal, although this is difficult when the size of $\mathbb{X}$ is large.

Symmetric Noise Channel

A symmetric noise channel is a special case of a discrete noise channel, where the transmission matrix entries only depend on whether or not $y = z$. There are many instances where in a moment of indecision, the user can select an alternative uniformly at random, especially when she does not have a strong opinion on any of the presented alternatives. A symmetric noise channel useful for modeling situations when m is relatively small; if m is large, with the offered alternatives being presented as a list, the positioning in the list might have an effect on the user's response. However, if the number of alternatives in the comparative question is small, the ordering should not matter, and a symmetric noise channel would be a reasonable modeling choice.

One way to parameterize a symmetric noise channel is by representing the transmission matrices as $P_\alpha = \alpha I + (1-\alpha)(1/m)\, ee^T$, where e is a vector of all ones, and $\alpha \in [0,1]$. There are other scenarios including symmetric noise channels that allow $P^{(zy)} > P^{(zz)}$ for $y \neq z$, but these situations would be particularly pessimistic from the perspective of learning, so we opt to exclude these noise channels from our definition. Since $\varphi(\cdot\,;\, P_\alpha)$ is concave and now symmetric in its first argument, choosing $u_*^{(z)} = 1/m$ for every $z \in \mathbb{Z}$ is an optimal solution. Thus, we want to choose the question X_k so that the user is equally likely to choose any of the offered alternatives.

In the case of symmetric noise, we can easily calculate the channel capacity using (4.8), yielding

$$C(P_\alpha) = \log_2 m - h\left(\alpha e^{(1)} + (1-\alpha)\,(1/m)\, e\right), \tag{4.15}$$

where $e^{(1)}$ is an m-vector with its first component equal to one, and all others equal to zero. The concavity of h gives a crude upper bound for the channel capacity, namely $C(P_\alpha) \leq \alpha \log_2 m$. Comparatively, under no noise, one can reduce the entropy of the posterior of the linear classifier by $\log_2 m$ bits at every time epoch. There is an intuitive explanation for this result. With noise level α, we only observe the model-consistent response with probability α at each step. Even under the best case scenario of knowing which responses were model-consistent and which were a random draw, the expected number of bits of reduced posterior entropy at each step would only be $\alpha \log_2 m$. In fact, the expected entropy reduction in reality is lower than this because we do not know which responses are informative of linear classifier $\boldsymbol{\theta}$.

Because the symmetric noise channel is a special case of a discrete noise channel, we can leverage the results from Theorem 4.3.8 to derive symmetric

noise channel sensitivity bounds.

Corollary 4.3.9. *Suppose we have a symmetric noise channel parameterized by P_α, where $P_\alpha = \alpha I + (1-\alpha)(1/m)\, ee^T$, implying that $u_*^{(z)} = 1/m$ for all z. Then*

$$\frac{\xi\alpha^2}{2\left(\alpha + (1-\alpha)(1/m)\right)} \left\|u - u_*\right\|^2 \;\leq\; C(P_\alpha) - \varphi(u\,; P_\alpha) \;\leq\; \frac{\xi\alpha^2}{2(1-\alpha)(1/m)} \left\|u - u_*\right\|^2$$

for all $u \in \Delta^m$.

Proof. We start with the bounds from Theorem 4.3.8 and further refine. The off-diagonal entries of P_α, by our parameterization of symmetric noise channel, are its smallest elements, and therefore, $\kappa_1 = (1-\alpha)(1/m)$. Similarly, the diagonal entries of P_α are the largest elements, and so $\kappa_2 = \alpha + (1-\alpha)(1/m)$. Lastly, one can easily verify $(u - u_*)^T P = \alpha\,(u - u_*)^T$. □

We return to the symmetric noise channel case in Section 4.3.3, where we show that in the theoretical case of allowing fabrication of alternatives, a subset of alternatives can always be constructed to achieve a uniform predictive distribution regardless of the prior, and hence the optimal rate of entropy reduction can always be achieved.

4.3.3 Selection of Alternatives from the Continuum

Now that we have results relating the predictive distribution $u_k(X_k)$ to the entropy reduction in the linear classifier $\boldsymbol{\theta}$, we now explore how we can appropriately choose alternatives X_k at every time epoch that yield a desirable predictive distribution.

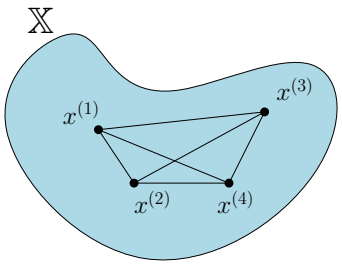


Figure 4.1: The continuum regime for alternative set $\mathbb{X}$. The document vectors $x^{(i)}$ can be chosen so that any direction $x^{(i)} - x^{(j)}$ can be achieved for all possible combinations.

We first focus on the easier case where we can construct alternatives to ask any comparative questions we desire. For a set of m alternatives $(x^{(1)}, \ldots, x^{(m)}) = X \in \mathbb{X}^m$ and a prior probability measure μ, the characteristic polytopes $A^{(1)}(X), \ldots, A^{(m)}(X)$ determine the predictive probabilities. Each set $A^{(z)}(X)$ composed of constraints $\theta^T \left(x^{(z)} - x^{(i)}\right) \geq 0$ for $i \neq z$ (ignoring strictness vs. non-strictness of inequalities). Thus, for the set of alternatives $\mathbb{X}$ to have full expressiveness with respect to our model, one must be able to choose alternatives so that $x^{(i)} - x^{(j)}$ can take any direction in $\mathbb{R}^d$. A reasonable and sufficient condition for the interior of $\mathbb{X}$ to be non-empty. When this is the case, we can always choose alternatives such that the relative direction between any two can take any value. This is what we refer to as the continuum regime.

In most practical situations, the set of alternatives is finite, and such construction is not possible. However, this assumption is more mathematically tractable and allows us to give conditions for when we can ask questions that yield a desirable predictive distribution, and consequently maximize entropy reduction. We return to the more realistic assumption of a finite alternative set later in Section 5.4.

Consider using pairwise comparisons, i.e., when $m = 2$. Is it true that re-

gardless of the noise channel and the prior distribution of $\boldsymbol{\theta}$ that we can select a question X_k that achieves the optimal predictive distribution $u_k(X_k)$? A simple example proves otherwise. Suppose *a priori*, the linear classifier $\boldsymbol{\theta}$ is normally distributed with zero mean and an identity covariance matrix. Because the distribution is symmetric about the origin, regardless of the hyperplane we select, exactly $1/2$ of the probabilistic mass lies on either side of the hyperplane. This is the desirable outcome when the noise channel is symmetric, but suppose this were not the case. For example, if the noise channel required $2/3$ of the probabilistic mass on one side of the hyperplane, there is no way to achieve this.

This issue is related to a certain metric called *halfspace depth*, first defined by Tukey (1975) and later refined by Donoho and Gasko (1992). The halfspace depth at a point $\eta \in \mathbb{R}^d$ refers to the minimum probabilistic mass able to be partitioned to one side of a hyperplane centered at η. In this thesis, we only consider the case where the cutting plane is centered at the origin, and need only to consider the case where $\eta = 0$. Hence, let

$$\delta(\mu_k) = \inf_{v \neq 0} \mu_k \left(\left\{ \theta : \theta^T v \geq 0 \right\} \right). \tag{4.16}$$

In our previous example, the halfspace depth of the origin was equal to $1/2$, and therefore, there were no hyperplanes that could partition less than $1/2$ of the probabilistic mass on a side of a hyperplane.

The question now is whether we can choose a hyperplane such that $u_k^{(z)}(X_k) = u_*^{(z)}$ for any $u_*^{(z)} \in [\delta(\mu_k), 1 - \delta(\mu_k)]$. We first prove an intuitive result regarding the continuity of probabilistic mass of a halfspace with respect to the cutting plane. One can imagine rotating a hyperplane about the origin, and since the probability measure has a density with respect to Lebesgue measure, there will not be any sudden jumps in probabilistic mass on either side of the

hyperplane.

Lemma 4.3.10. *If probability measure μ is absolutely continuous with respect to Lebesgue measure, then the mapping $v \mapsto \mu\left(\{\theta \in \Theta : \theta^T v \geq 0\}\right)$ is continuous.*

Proof. Suppose we have a sequence $(v_j : j \geq 0)$ in $\mathbb{R}^d \setminus \{0\}$ such that $v_j \to v$. The functions $\mathbb{I}(\{\theta : \theta^T v_j \geq 0\})$ converge to $\mathbb{I}(\theta : \theta^T v \geq 0\})$ almost surely. Taking expectations and using Dominated Convergence Theorem gives the result. □

Lemma 4.3.10 enables us to find conditions under which we can ask a question X_k that yields a desirable predictive distribution $u_k(X_k)$. In particular, Corollary 4.3.11 uses a variant of the intermediate value theorem.

Corollary 4.3.11. *Suppose $u_* > 0$ and $\mathrm{Int}(\mathbb{X}) \neq \varnothing$. Then there exists $X_k = (x_1, x_2) \in \mathbb{X}^2$ such that $u_k(X_k) = u_*$ if and only if $\max u_* \leq 1 - \delta(\mu_k)$.*

Proof. Take any $v \in C = \{w \in \mathbb{R}^d : \|w\| = 1\}$, where $\mu_k\left(\{\theta : \theta^T v \geq 0\}\right) = \delta(\mu_k)$. Now let $v' = -v$, and since μ_k is absolutely continuous with respect to Lebesgue measure, $\mu_k(\theta^T v' \geq 0) = \mu_k(\theta^T v' > 0) = 1 - \delta(\mu_k)$. Also, C is connected, and $w \mapsto \mu_k(\{\theta : \theta^T w \geq 0\})$ is a continuous mapping: it follows that the image of any path from v to v' must also be connected. But the image is a subset of the real line, and therefore must be an interval. Lastly, C is a compact set, implying that the endpoints of this interval are attainable, and so the image of any such path is equal to $[\delta(\mu_k), 1 - \delta(\mu_k)]$.

To recover the two alternatives, first select a vector $w \in \mathbb{R}^d$ such that $\mu_k(\{\theta : \theta^T w \geq 0\}) = u^{(1)}$. Choose $x^{(1)} \in \mathrm{Int}(\mathbb{X})$, and subsequently choose $x^{(2)} = x^{(1)} - cw$, where $c > 0$ is a positive scalar that ensures $x^{(2)} \in \mathbb{X}$. Finally, let $X_k = \left(x^{(1)}, x^{(2)}\right)$.

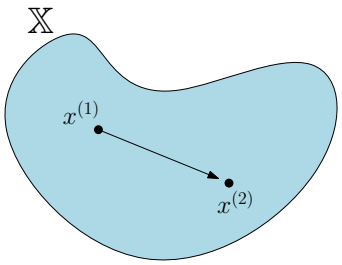


Figure 4.2: Selection of alternatives $x^{(1)}$ and $x^{(2)}$. Since $x^{(1)}$ lies in the interior of $\mathbb{X}$, it is always possible to choose $x^{(2)}$ so that $x^{(1)} - x^{(2)}$ has the same direction as v.

To prove the converse statement, suppose $\max\{u_*\} > 1-\delta(\mu_k)$. Then by definition of halfspace depth, $\min\{u_*\} \notin \left\{\mu_k(\{\theta : \theta^T v \geq 0\}) : v \neq 0\right\}$. Thus, there does not exist a hyperplane that can separate $\mathbb{R}^d$ into two halfspaces with probabilistic mass u_*. □

Can we draw a similar conclusion if we offer more more than two alternatives at each time epoch? The mass partition problem becomes increasingly complex when greater than two alternatives are included. Since the sets $A^{(z)}(X)$ correspond to convex polyhedral cones, the problem becomes that of finding a partition of m convex polyhedral cones, or a polyhedral m-fan as it is known in the computational geometry literature, that attains the prescribed probabilistic mass u_*. There are a number of results pertaining to convex equipartitions and extensions of the Borsuk-Ulam Theorem, most notably the Ham Sandwich Theorem. Despite this, to the best of our knowledge, there is no result for general mass partitions of convex polyhedral m-fans in the computational geometry literature. For this reason, we prove such a result here: that one can construct a polyhedral m-fan with the corresponding predictive distribution u_* if the measure μ is such that $\max\{u_*\} < 1 - \delta(\mu)$.

Unlike the previous case that focused on pairwise comparisons, the inequal-

ity is strict. One of the reasons this is the case is because of the specific structure of the polyhedral cones in our problem. Since $A^{(z)}(X)$ corresponds to the linear classifier in which the dot product with alternative z is maximal, these polyhedral cones cannot be halfspaces unless the predictive probability for some alternative equals zero, which we do not allow. Thus, we enforce the additional constraint that each $A^{(z)}$ is a *salient* convex polyhedral cone, meaning that it does not contain a linear subspace.

To prove the result, we first show the result in the case of two dimensions: constructing the polyhedral m-fan, then deriving the feature vectors for the corresponding alternatives. This result is then generalized to the case of any dimension by using a projection argument.

Lemma 4.3.12. *Suppose $d = 2$ and $m > 2$. If $\max\{u_*\} < 1 - \delta(\mu)$, then there exists a two-dimensional polyhedral m-fan characterized by polyhedral cones $(A^{(z)} : z \in \mathbb{Z})$ such that $\mu(A^{(z)}) = u_*^{(z)}$ for all $z \in \mathbb{Z}$.*

Proof. Without loss of generality we can assume $\|\theta\| = 1$, and in the case of two dimensions that is equivalent to θ being parameterized by the interval $[0, 2\pi)$ on the unit circle. For an interval $\mathcal{I}$ measuring angles in radians, let

$$\text{Cone}(\mathcal{I}) = \left\{ \begin{pmatrix} r\cos\eta \\ r\sin\eta \end{pmatrix} : \eta \in \mathcal{I},\, r > 0 \right\}.$$

Accordingly, let μ^C be a measure defined on the unit circle such that $\mu^C(\mathcal{I}) = \mu\left(\text{Cone}(\mathcal{I})\right)$ for every Lebesgue-measurable interval on $[0, 2\pi)$. This implies that

$\delta(\mu^C) = \delta(\mu)$. For radian angles $\eta^{(1)} < \eta^{(2)} < \cdots < \eta^{(m+1)} = \eta^{(1)} + 2\pi$, we define

$$B^{(z)} = \begin{cases} [\eta^{(1)}, \eta^{(2)}] & z = 1 \\ (\eta^{(z)}, \eta^{(z+1)}] & z = 2, \ldots, m-1 \\ (\eta^{(m)}, \eta^{(m+1)}) & z = m \end{cases}$$

for all $z \in \mathbb{Z}$. The asymmetry with respect to sets being $B^{(z)}$ closed or open is due to the definition of $A^{(z)}$ in (4.1). For each $B^{(z)}$ to correspond to a convex set strictly contained in a halfspace, we require $\eta^{(z+1)} - \eta^{(z)} < \pi$. Our objective is to appropriately select the angles $(\eta^{(z)} : z \in \mathbb{Z})$ so that $\mu^C(B^{(z)}) = u^{(z)}$. It suffices to consider only two cases.

Case 1: $\max\{u_*\} < \delta(\mu)$

This is the simpler case, since all the probabilities from the predictive distribution are strictly smaller than *any* halfspace measure. Arbitrarily choose $\eta^{(1)}$. Now we want to choose $\eta^{(2)} \in \left(\eta^{(1)}, \eta^{(1)} + \pi\right)$ so that the interval contains prescribed measure $u_*^{(1)}$. The function $\eta^{(2)} \mapsto \mu^C\left([\eta^{(1)}, \eta^{(2)}]\right)$ is monotonically increasing, continuous, and takes values on $(0, \delta(\mu)]$. Since $u_*^{(z)} \leq \max u_* \leq \delta(\mu_k)$, the Intermediate Value Theorem allows us to choose $\eta^{(2)}$ so the interval has measure $u_*^{(1)}$. Continue in this way until all such angles $\eta^{(z)}$ are attained.

Case 2: $\max\{u_*\} \in [\delta(\mu), 1 - \delta(\mu))$

Here, it is necessary to define the set $B^{(z)}$ corresponding to the largest predictive probability u_* first, and that with the smallest second. Without loss of

generality, suppose $u_*^{(1)} = \max u_*$ and $u_*^{(2)} = \min u_*$. Then choose $\eta^{(1)}$ such that

$$\mu^C\left([\eta^{(1)}, \eta^{(1)} + \pi]\right) \in (\max u_*\,,\ \max u_* + \min u_*)\,. \tag{4.17}$$

This is possible because

$$(\delta(\mu), 1 - \delta(\mu)) \cap (\max u_*\,,\ \max u_* + \min u_*) \neq \varnothing,$$

due to the assumptions that $\max u_* \in [\delta(\mu), 1 - \delta(\mu))$ and $\min u_* > 0$. Now define $\eta^{(2)}$ such that $\mu^C\left[\eta^{(1)}, \eta^{(2)}\right] = \max u_*$.

Now we define the interval corresponding to $\min u_*$ directly adjacent. Suppose $\mu^C(\eta^{(2)}, \eta^{(2)} + \pi] > \min u_*$. Then by the Intermediate Value Theorem, there exists some $\eta^{(3)}$ such that $\mu^C(\eta^{(2)}, \eta^{(3)}] = \min u_*$. Otherwise, suppose that $\mu^C(\eta^{(m+1)} - \pi, \eta^{(m+1)}) > \min u_*$. Again, by the Intermediate Value Theorem, we can find $\eta^{(m)}$ less than π radians from $\eta^{(m+1)}$ such that $\mu^C(\eta^{(m)}, \eta^{(m+1)}) = \min u_*$.

We claim that these are the only two possibilities. By way of contradiction, suppose that neither of these scenarios are true; in other words,

$$\mu^C[\eta^{(2)}, \eta^{(2)} + \pi] \leq \min u_*$$
$$\mu^C[\eta^{(m+1)} - \pi, \eta^{(m+1)}] \leq \min u_*.$$

We can decompose these intervals into non-overlapping parts. Define

$$a = \mu^C(\eta^{(2)} + \pi, \eta^{(1)} + 2\pi]$$
$$b = \mu^C(\eta^{(2)}, \eta^{(1)} + \pi]$$
$$c = \mu^C(\eta^{(1)} + \pi, \eta^{(2)} + \pi).$$

Suppose that $\max\{a, b\} + c \leq \min u_*$. The measure of the union of the three intervals $1 - \max u_* = a + b + c$, which implies $1 - \max u_* \leq \min u_* + \min\{a, b\}$.

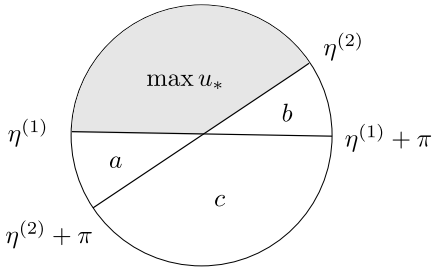


Figure 4.3: Diagram showing the distribution of probabilistic mass partitioned in unit circle.

Finally, since the smallest component of u_* must be smaller in magnitude than the sum of the other non-maximal components,

$$\max\{a, b\} + c \leq \min u_* \leq 1 - \max u_* - \min u_* \leq \min\{a, b\},$$

implying among other things that $b = \min u_*$ in this scenario. However, this is a contradiction, since we originally chose $\eta^{(1)}$ such that $b + c < \max u_* + \min u_*$ due to (4.17). Therefore, this scenario is not possible, and we can always find an interval with probabilistic mass strictly greater than $\min u_*$ directly adjacent to an interval with maximal probabilistic mass.

In all cases, we have defined the first two intervals, and the remaining unallocated region of the unit circle is strictly contained in an interval of width less than π radians. Thus, one can easily define a partition as in Case 1, and every subsequent interval would necessarily have to have length strictly less than π radians. To recover the convex cones, let $A^{(z)} = \mathrm{Cone}\left(B^{(z)}\right)$ for every $z \in \mathbb{Z}$, and it is clear that $A^{(z)}$ contains the desired probabilistic mass. □

Lemma 4.3.12 gives a way to construct polyhedral fans with the desired probabilistic mass. We are interested in finding a set of alternatives that represents this polyhedral fan, and this is exactly what Theorem 4.3.13 does in the

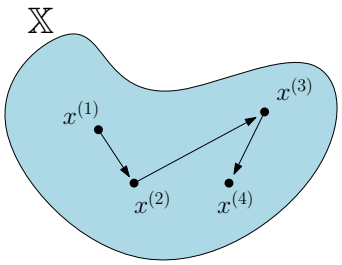


Figure 4.4: Iterative selection of alternatives. Since each $x^{(z)}$ is in the interior of $\mathbb{X}$, it is always possible to select $x^{(z+1)}$ to maintain a specific direction for $x^{(z+1)} - x^{(z)}$.

two-dimensional case. The critical condition required is for the set of alternatives $\mathbb{X}$ to have non-empty interior.

Theorem 4.3.13. *Suppose $d = 2$ and $m > 2$. Then given a measure μ that is absolutely continuous with respect to Lebesgue measure and an optimal predictive distribution u_*, if $\mathrm{Int}(\mathbb{X}) \neq \varnothing$ and $\max u_* < 1 - \delta(\mu)$, then there exists $X \in \mathbb{X}^m$ such that $u(X) = u_*$.*

Proof. First, use Lemma 4.3.12 to construct a polyhedral fan with the correct probabilistic weights. Using the angles $\eta^{(1)}, \ldots, \eta^{(m)}$ constructed in the Lemma, we can define separating hyperplanes $v^{(1)}, \ldots, v^{(m)}$ by setting $v^{(z)} = \left(-\sin \eta^{(z)}, \cos \eta^{(z)}\right)$. Then we have

$$\bar{A}^{(z)} = \left\{ \theta : \begin{array}{c} \theta^T v^{(z)} > 0 \\ \theta^T v^{(z+1)} \leq 0 \end{array} \right\}.$$

The goal now is to define the alternatives. First, choose $x^{(1)} \in \mathrm{Int}(\mathbb{X})$. Now define $x^{(z+1)} = x^{(z)} + c^{(z+1)} v^{(z+1)}$, where $c^{(z+1)} > 0$ is a positive scaling that ensures $x^{(z+1)} \in \mathrm{Int}(\mathbb{X})$ if $x^{(z)} \in \mathrm{Int}(\mathbb{X})$. Now we can equivalently write

$$\bar{A}^{(z)} = \left\{ \theta : \begin{array}{c} \theta^T \left(x^{(z)} - x^{(z-1)}\right) > 0 \\ \theta^T \left(x^{(z)} - x^{(z+1)}\right) \geq 0. \end{array} \right\}.$$

Let $X = (x^{(1)}, \ldots, x^{(m)})$. It remains to show that $A^{(z)}(X) = \bar{A}^{(z)}$. Because $A^{(z)}(X)$ has the same linear inequalities as $\bar{A}^{(z)}$, it is clear that $A^{(z)}(X) \subseteq \bar{A}^{(z)}$ for all z. Now suppose there exists some $\theta \in \bar{A}^{(z)}$. Since $(\bar{A}^{(z)} : z \in \mathbb{Z})$ is a partition of $\mathbb{R}^2$, it is clear that $\theta \notin A^{(z')}$ for $z' \neq z$, and thus, $\theta \notin A^{(z')}(X)$. Since $\left(A^{(z)}(X) : z \in \mathbb{Z}\right)$ is also a partition of $\mathbb{R}^2$, it must be that $\theta \in A^{(z)}(X)$. This directly implies $\bar{A}^{(z)} = A^{(z)}(X)$, and so $u^{(z)}(X) = \mu(A^{(z)}) = u_*^{(z)}$. □

Theorem 4.3.13 shows that in the case of two dimensions, a set of m alternatives can be generated to ensure that the entropy of the posterior distribution of $\boldsymbol{\theta}$ maximally decreases. This result can be generalized to arbitrary dimension by selecting a two dimensional subspace and leveraging the previous result.

Theorem 4.3.14. *Suppose* $\mathrm{Int}(\mathbb{X}) \neq \varnothing$ *and* $u_* > 0$. *If* $\max u_* < 1 - \delta(\mu_k)$, *then there exists* $X_k = (x^{(1)}, x^{(2)}, \ldots, x^{(m)}) \in \mathbb{X}^m$ *such that* $u_k(X_k) = u_*$. *Further, if* $\max u_* > 1 - \delta(\mu_k)$, *then finding such a question is not possible.*

Proof. We begin by proving the last claim of the theorem. Since any $A^{(z)}(X_k)$ can be contained by a halfspace centered at the origin, and since all such halfspaces have probabilistic mass less than or equal to $1 - \delta(\mu_k)$, then we must have $\mu_k(A^{(z)}(X)) \leq 1 - \delta(\mu_k)$ for every $z \in \mathbb{Z}$ and for every $X_k \in \mathbb{X}^m$.

Now we show the main result of the theorem. There exists some $\bar{\beta} \in \mathbb{R}^m \setminus \{0\}$ such that $\mu_k\left(\{\theta : \theta^T \bar{\beta} \geq 0\}\right) = \delta(\mu_k)$, since μ_k has density p_k and is continuous. Let $\mathcal{H} = \{\theta : \theta^T \bar{\beta} = 0\}$ denote the hyperplane. Now choose a two-dimensional subspace L such that $L \perp \mathcal{H}$. For $\nu \in L$, define density p_k^L as

$$p_k^L(\nu) = \int_{\omega \in L^\perp} p_k(\nu + \omega)\, \lambda_{d-2}(d\omega),$$

where λ_{d-2} is $(d-2)$-Lebesgue measure, and let μ_k^L denote measure induced by

density p_k^L. For $\beta \in L$, we have

$$\begin{aligned}
\mu_k^L\left(\{\nu \in L:\, \nu^T\beta \geq 0\}\right) &= \int_{\nu\in L:\, \nu^T\beta\geq 0} p_k^L(\nu)\, \lambda_2(d\nu) \\
&= \int_{\nu\in L:\, \nu^T\beta\geq 0} \int_{\omega\in L^\perp} p_k(\nu+\omega)\, \lambda_{m-2}(d\omega)\, \lambda_2(d\nu) \\
&= \int_{(\nu,\omega)\in(L\times L^\perp):\, (\nu+\omega)^T\beta\geq 0} p_k(\nu+\omega)\, \lambda_d(d\nu \times d\omega) \\
&= \int_{\theta:\, \theta^T\beta\geq 0} p_k(\theta)\, \lambda(d\theta) = \mu_k(\{\theta:\, \theta^T\beta \geq 0\}).
\end{aligned}$$

Thus, μ_k^L is consistent with μ_k. In particular, $\mu_k^L(\{\theta:\, \theta^T\bar{\beta} \geq 0\}) = \delta(\mu_k)$, and thus $\delta(\mu_k^L) \leq \delta(\mu_k)$, meaning we can use the previous Theorem 4.3.13 to find an appropriate comparative question.

In particular, let γ_1 and γ_2 denote two orthogonal d-vectors that span L, and let $\Gamma \in \mathbb{R}^{d\times 2}$ contain γ_1 and γ_2 as its columns. To use the Theorem 4.3.13, we pass $\mu_k^L \circ \Gamma$, and to convert the resulting question $X_k^L = (x^{(1)}, \ldots, x^{(m)})$ back into d-dimensional space, take $X_k = (\Gamma x^{(1)}, \ldots, \Gamma x^{(m)})$. □

Theorem 4.3.14 provides one possible construction for a question X_k that gives a desirable predictive distribution, although there may be others. However, it is clear that if the halfspace depth $\delta(\mu_k)$ is too large, it will not always be possible to find a question that can yield the optimal predictive distribution, even if we can construct questions in the continuum. But while it may not be possible to *maximally* reduce the entropy of the posterior distribution, we may choose a question X_k that can still reduce entropy by a constant amount at each time epoch.

We conclude the section by showing that entropy in the linear classifier $\boldsymbol{\theta}$ can be reduced linearly, even if not optimally.

Theorem 4.3.15. *Suppose* $\mathbb{X} = \mathbb{R}^d$*, and let* $\boldsymbol{\sigma}_k = (\max\{u_*\} - (1 - \delta(\boldsymbol{\mu}_k)) + \epsilon)^+$*. Then the following upper bound holds when* $m = 2$ *for* $\epsilon = 0$ *and when* $m > 2$ *for arbitrarily small* $\epsilon > 0$*.*

$$K \cdot C(f) - \sup_{\pi} \mathbb{E}^{\pi}\left[I(\boldsymbol{\theta}; \mathcal{Y}_k)\right] \leq r(f)\left(1 + \frac{1}{m-1}\right)\sum_{k=1}^{K}\mathbb{E}^{\pi}[\boldsymbol{\sigma}_k^2]$$
$$\leq K \cdot r(f)\left(1 + \frac{1}{m-1}\right)\left((\max\{u_*\} - 1/2 + \epsilon)^+\right)^2.$$

Proof. We start with the case when $m > 2$. Fix any small $\epsilon > 0$. Let $z' = \arg\max\{u_*\}$. We write equality because in the cases where $\boldsymbol{\sigma}_k > 0$, the maximum component is unique; otherwise, when $\boldsymbol{\sigma}_k = 0$, the choice of z' is irrelevant. We construct an "approximate predictive distribution" $\bar{u}_k$ such that

$$\bar{\boldsymbol{u}}_k^{(z)} = \begin{cases} u_*^{(z)} - \boldsymbol{\sigma}_k & z = z' \\ u_*^{(z)} + \boldsymbol{\sigma}_k/(m-1) & z \neq z' \end{cases}$$

This new vector $\bar{\boldsymbol{u}}_k$ is the projection of u_* onto the set $\{u \in \Delta^m : \max\{u\} \leq (1-\delta(\mu_k))-\epsilon\}$. This "approximate predictive distribution" is chosen to minimize the L_2 distance from optimal u_*, and therefore maximize entropy reduction.

One can show that $\max\{\bar{u}_k\} < 1-\delta(\mu_k)$, and $\|\bar{u}_k - u_*\|^2 \leq \sigma_k^2\,(1 + 1/(m-1))$. Now we can construct $\bar{X}_k$ such that $u_k(\bar{X}_k) = \bar{u}_k$ at every step, which is possible by Theorem 4.3.14 since $\bar{u} > 0$ and $\max\{\bar{u}_k\} < 1 - \delta(\mu_k)$. Now we use Theo-

rem 4.3.8 to show

$$\begin{aligned} K \cdot C(f) - \sup_{\pi} I^{\pi}(\boldsymbol{\theta}; \mathcal{Y}_k) &\leq \sum_{k=1}^{K} \left(C(f) - \mathbb{E}_k^{\pi} \left[\varphi(u_k(\bar{X}_k)\,;\, f) \right] \right) \\ &= \sum_{k=1}^{K} \left(C(f) - \mathbb{E}_k^{\pi} \left[\varphi(\bar{\boldsymbol{u}}_{\boldsymbol{k}}\,;\, f) \right] \right) \\ &\leq \sum_{k=1}^{K} r(f)\, \mathbb{E}_k^{\pi} \left[\|\bar{\boldsymbol{u}}_{\boldsymbol{k}} - u_*\|^2 \right] \\ &= r(f) \left(1 + \frac{1}{m-1} \right) \sum_{k=1}^{K} \mathbb{E}_k^{\pi} \left[\boldsymbol{\sigma}_k^2 \right]. \end{aligned}$$

And since $1 - \delta(\mu_k) \geq 1/2$, it follows that $\sigma_k \leq (\max\{u_*\} - 1/2 + \epsilon)^+$, regardless of μ_k. The proof is analogous for the $m = 2$ case: the only change required is to set $\epsilon = 0$, because by Corollary 4.3.11, we can find a question if $\max\{u_*\} \leq 1 - \delta(\pi_k)$, where the inequality need not be strict. □

Putting Theorem 4.3.15 together with Corollary 4.3.3 shows that if the alternative set $\mathbb{X}$ has non-empty interior, the expected differential entropy of linear classifier $\boldsymbol{\theta}$ can be reduced at a linear rate, and this is optimal up to a constant factor.

Finally, recall in Section 4.3.2 we defined the case of a symmetric noise channel. There, $u_*^{(z)} = 1/m$ for all $z \in \mathbb{Z}$. In the pairwise comparison case, $\max u_* = 1/2 \leq 1 - \delta(\mu)$ for all measures μ. In the multiple comparison case, $\max u_* = 1/m < 1/2 \leq 1 - \delta(\mu)$ for all measures μ. Thus, regardless of m, in the continuum setting, a set of alternatives can always be constructed to achieve a uniform predictive distribution, and therefore optimally reduce posterior entropy.

4.4 Misclassification Error

The entropy pursuit policy itself is intuitive, especially when the noise channel is symmetric. However, differential entropy as a metric for measuring knowledge of the user's preferences is not intuitive. One way to measure the extent of our knowledge about a user's preferences is testing ourselves using a randomly chosen question and estimating the answer after observing a response from the user. This probability we get the answer wrong called misclassification error.

Specifically, we sequentially ask questions X_k and observe signals $\boldsymbol{Y}_k(X_k)$ at time epochs $k = 1, \ldots, K$. After the last question, we are then posed with an evaluation question. The evaluation question will be an n-way comparison between randomly chosen alternatives, where n can differ from m. Denote the evaluation question as $\boldsymbol{S}_K \in \mathbb{X}^n$, where a particular evaluation question $S_K = \left(s^{(1)}, \ldots, s^{(n)}\right)$. The evaluation question is chosen at random according to some unknown distribution. Denote the model-consistent answer as $\boldsymbol{W}_K(S_K) = \min\left\{\arg\max_{w \in \mathbb{W}} \boldsymbol{\theta}^T s_w\right\}$, where the minimum serves as a tie-breaking rule. The goal is to use history $\boldsymbol{\mathcal{Y}}_K$ and the question $\boldsymbol{S}_K$ to predict $\boldsymbol{W}_K(S_K)$. Let $\hat{W}_K$ denote the candidate answer that depends on the chosen evaluation question response history. Then our goal for the adaptive problem is to find a policy that minimizes

$$\mathcal{E}_K^{\pi} = \mathbb{E}^{\pi}\left[\inf_{\hat{W}_K \in \mathbb{W}} \mathbb{P}\left(\boldsymbol{W}_K(\boldsymbol{S}_K) \neq \hat{W}_K \,\middle|\, \boldsymbol{\mathcal{Y}}_K, \boldsymbol{S}_K\right)\right], \tag{4.18}$$

and one can do this by adaptively selecting the best question X_k that will allow us to learn enough about the user's preferences to correctly answer evaluation question $\boldsymbol{S}_k$ with high certainty. Let $\mathcal{E}_K^* = \inf_{\pi} \mathcal{E}_K^{\pi}$ be the misclassification error under the optimal policy, assuming it is attained.

We make several reasonable assumptions on the dependence of $\boldsymbol{S}_K$ and $\boldsymbol{W}_K(\boldsymbol{S}_K)$ with the model-consistent response $\boldsymbol{Z}_k(X_k)$ and signal $\boldsymbol{Y}_k(X_k)$ from the learning question X_k.

Evaluation Question Assumptions. *For evaluation question $\boldsymbol{S}_K = S_K$ and corresponding model-consistent answer $\boldsymbol{W}_K(S_K)$, we assume*

- *Evaluation question $\boldsymbol{S}_K = S_K$ is selected randomly from $\mathbb{X}^n$, independent from all else, and*
- *For all such questions S_K, signal $\boldsymbol{Y}_k(X_k)$ and model-consistent answer $\boldsymbol{W}_K(S_K)$ are conditionally independent given $\boldsymbol{Z}_k(X_k)$ for all $k = 1, \ldots, K$.*

In practice, solving the fully adaptive problem is intractable, and instead, one can use a knowledge gradient policy to approach this problem. This is equivalent to solving a greedy version of the problem where we are evaluated at every step. In other words, after observing signal $\boldsymbol{Y}_k(X_k)$, we are posed with answering a randomly selected evaluation question $\boldsymbol{S}_k$, with no concern about any future evaluation. Every question in the sequence $(\boldsymbol{S}_k : k = 1, \ldots, K)$ is selected i.i.d. and follows the Evaluation Question Assumptions. The knowledge gradient policy chooses X_k such that at every time epoch k it solves

$$\mathcal{E}_k^{KG} = \inf_{X_k \in \mathbb{X}^m} \mathbb{E}^{KG} \left[\inf_{\hat{W}_k \in \mathbb{W}} \mathbb{P}\left(\boldsymbol{W}_k(S_k) \neq \hat{W}_k \,\middle|\, \boldsymbol{\mathcal{Y}}_k,\, \boldsymbol{S}_k \right) \right].$$

Obviously, $\mathcal{E}_k^{KG} \geq \mathcal{E}_k^*$ for all k, since knowledge gradient cannot perform strictly better than the fully adaptive optimal policy. It would be beneficial to know how wide the gap is, and this can be done by finding a lower bound on the optimal misclassification error. In Section 4.4.1, we work towards this goal by analyzing the knowledge gradient objective.

4.4.1 Characterizing the Knowledge Gradient Policy

We would like to be able to analyze the solution to the knowledge gradient policy, but first, further notation must be defined. For a particular assessment question $S \in \mathbb{X}^n$, we define the sets

$$B^{(w)}(S) = \left\{ \theta \in \Theta : \begin{array}{ll} \theta^T s^{(w)} \geq \theta^T s^{(i)} & \forall i > w \\ \theta^T s^{(w)} > \theta^T s^{(i)} & \forall i < w \end{array} \right\}. \tag{4.19}$$

Similar to $\{A^{(z)}(X) : z \in \mathbb{Z}\}$, the sets $\{B^{(w)}(S) : w \in \mathbb{W}\}$ characterizes a partition of Θ and defines the sets of linear classifiers that correspond to a particular answer to the assessment S. We also want to generalize the definition of $u(X)$ to be able to handle joint and conditional distributions in matrix form. Accordingly, let $U_k(X_k, S_k) \in \Delta^{m \times n}$ be defined by its components, with

$$U_k^{(zw)}(X_k, S_k) = \mu_k \left(A^{(z)}(X_k) \cap B^{(w)}(S_k) \right). \tag{4.20}$$

Similarly, define $U_k(X_k \,|\, S_k) \in \Delta^{n \times m}$ to be

$$U_k^{(wz)}(X_k \,|\, S_k) = \mu_k \left(A^{(z)}(X_k) \,|\, B^{(w)}(S_k) \right), \tag{4.21}$$

denoting the matrix of conditional probabilities, with row w of $U_k(X_k \,|\, S_k)$ denoting probability vectors in Δ^m that are conditioned on the event $\boldsymbol{W}_k(S_k) = w$. These matrices will allow us to more easily represent different objective functions and equations throughout the rest of the chapter.

Now we are ready to present a more closed-form objective function corresponding to the one-step knowledge gradient policy, both for the cases of a general noise channel and a discrete noise channel.

Theorem 4.4.1. *Suppose we have a noise channel parameterized by* $f = \{f^{(z)} : z \in$

$\mathbb{Z}\}$. *Then the knowledge gradient policy can be calculated as*

$$\mathcal{E}_k^{KG} = 1 - \sup_{X_k \in \mathbb{X}^m} \mathbb{E}\left[\int_{\mathbb{Y}} \max_w \sum_{z \in \mathbb{Z}} f^{(z)}(y)\, U_k^{(zw)}(X_k, \boldsymbol{S}_k)\, dy\right], \tag{4.22}$$

and in the case of a discrete noise channel parameterized by transmission matrix P,

$$\mathcal{E}_k^{KG} = 1 - \sup_{X_k \in \mathbb{X}^m} \mathbb{E}\left\|U_k^T(X_k, \boldsymbol{S}_k) \cdot P\right\|_{\infty,1} \tag{4.23}$$

where $\|\cdot\|_{p,q}$ denotes the entry-wise $L_{p,q}$ norm of the given matrix, with norms being performed in that order.

Proof. Our goal is to minimize

$$\mathcal{E}_k = \mathbb{E}_k\left[\min_w \mathbb{P}_k\left(\boldsymbol{W}_k(S_k) \neq w \,\middle|\, \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k = S_k\right)\right]. \tag{4.24}$$

Using the properties of conditional probability, this is equivalent to minimizing

$$\mathcal{E}_k = 1 - \mathbb{E}\left[\int_{\mathbb{Y}} \max_{w \in \mathbb{W}} \sum_{z \in \mathbb{Z}} f^{(z)}(y)\, \mathbb{P}_k\left(\boldsymbol{Z}_k(X_k) = z, \boldsymbol{W}_k(S_k) = w \mid \boldsymbol{S}_k = S_k\right)\, dy\right],$$

which is equal to (4.22). For a discrete noise channel, (4.24) is equivalent to

$$\begin{aligned}\mathcal{E}_k &= 1 - \mathbb{E}\left[\sum_{y \in \mathbb{Y}} \max_{w \in \mathbb{W}} \sum_{z \in \mathbb{Z}} P^{(zy)} \cdot \mathbb{P}_k\left(\boldsymbol{Z}_k(X_k) = z, \boldsymbol{W}_k(S_k) = w \mid \boldsymbol{S}_k = S_k\right)\right], \\ &= 1 - \mathbb{E}\left[\sum_{y \in \mathbb{Y}} \max_{w \in \mathbb{W}} \left(U_k^T(X_k\, \boldsymbol{S}_k) \cdot P\right)^{(wy)}\right],\end{aligned}$$

The outer operations, taking a sum and calculating a row-wise maximum, is exactly the definition of a $L_{\infty,1}$ norm of a matrix. This leaves us with (4.23), completing the proof. □

Theorem 4.4.1 above shows exactly what is needed to compute the objective for the knowledge gradient policy. Even with a one-step policy, the objective is difficult to compute, mostly due to the computation of the joint distribution

matrix $U_k(\cdot,\cdot)$. The problem is that pairs of queries are compared, and the fact that matrix has mn components means the probabilistic mass for mn convex polytopes need to be estimated. Worse yet, this quantity needs to be computed for many such pairs of queries and averaged accordingly. It is clear that any benefit gained from using a knowledge gradient policy is paid by a dramatically larger computational budget.

This begs the question: how exactly does a knowledge gradient policy differ from entropy pursuit, and is there a computationally efficient way of improving entropy pursuit that provides the benefits of a knowledge gradient policy? Information theory gives us a way answer these questions, and in the next sections, we will bridge the theory between knowledge gradient objectives and entropy based objectives.

4.4.2 An Interactive Approach

It would be helpful to have an analogue to Theorem 4.3.1 so we can relate the posterior Shannon entropy of the answer $\boldsymbol{W}(S)$ of evaluation question S to the answer $\boldsymbol{Z}_k(X_k)$ of initial question X_k. It turns out that information content in observing signal $\boldsymbol{Y}_k(X_k)$ to infer answer $\boldsymbol{W}_k$ is related to a concept in information theory called *interaction information*. In the context of this model, for a model consistent answer $\boldsymbol{Z}_k$, observed response $\boldsymbol{Y}_k$, evaluation question S and true answer $\boldsymbol{W}(S)$, Interaction Information denotes the difference

$$\begin{aligned} I_k(\boldsymbol{W}(S);\boldsymbol{Y}_k;\boldsymbol{Z}_k) &= I_k(\boldsymbol{W}(S);\boldsymbol{Y}_k) - I_k(\boldsymbol{W}(S);\boldsymbol{Y}_k \mid \boldsymbol{Z}_k) \\ &= I_k(\boldsymbol{Y}_k;\boldsymbol{Z}_k) - I_k(\boldsymbol{Y}_k;\boldsymbol{Z}_k \mid \boldsymbol{W}(S)) \\ &= I_k(\boldsymbol{Z}_k;\boldsymbol{W}_k) - I_k(\boldsymbol{Z}_k;\boldsymbol{W}(S) \mid \boldsymbol{Y}_k). \end{aligned}$$

Similarly, we define Conditional Interaction Information as

$$I_k(\boldsymbol{W}(\boldsymbol{S}); \boldsymbol{Y}_k; \boldsymbol{Z}_k \mid \boldsymbol{S}) = \mathbb{E}\left[I_k(\boldsymbol{W}(\boldsymbol{S}); \boldsymbol{Y}_k; \boldsymbol{Z}_k \mid \boldsymbol{S} = S)\right].$$

Interaction information tells us the relationship between three random variables in terms of the redundancy in information content. In general, this quantity can be positive or negative. If the interaction information between three random variables is negative, then one does not learn as much from an observation when already knowing the outcome of another. This is the more natural and relevant case in the context of misclassification error.

In particular, the goal is to ask questions so that the observations can provide the maximum amount of information on the answer to an unknown evaluation question. Theorem 4.4.2 decomposes this problem into an equivalent formulation using interaction information, for which we seek to maximize the amount of redundancy between the chosen questions X_k and the unknown evaluation question $\boldsymbol{S}_k$.

Theorem 4.4.2. *Under the Noise Channel Assumptions and Evaluation Question Assumptions, we have*

$$I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k) = I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k) \mid \boldsymbol{S}_k) \leq I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k)). \tag{4.25}$$

Proof. We first claim that $I_k(\boldsymbol{W}_k(S_k); Y_k(X_k) \mid \boldsymbol{Z}_k(X_k), \boldsymbol{S}_k = S_k) = 0$ under the assumptions. This is because $\boldsymbol{Y}_k(X_k)$ and $\boldsymbol{W}(S)$ are conditionally independent given $\boldsymbol{Z}_k(X_k)$, and this directly implies that quantity is equal to zero by the

definition of conditional mutual information (Cover 1991, p. 22). Now we see

$$
\begin{aligned}
&I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k) \\
&\quad = I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k) - I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{Z}_k(X_k), \boldsymbol{S}_k) \\
&\quad = I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k) \mid \boldsymbol{S}_k) \qquad (4.26) \\
&\quad = I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k)) - I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k) \mid \boldsymbol{W}_k(\boldsymbol{S}_k),\ \boldsymbol{S}_k), \qquad (4.27)
\end{aligned}
$$

where (4.36) denotes the interaction information between random variables $\boldsymbol{W}_k(S_k)$, $\boldsymbol{Z}_k(X_k)$, and $\boldsymbol{Y}_k(X_k)$. In particular, the equality of particular importance is

$$
I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k) = I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k)) - I_k(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k) \mid \boldsymbol{W}_k(\boldsymbol{S}_k),\ \boldsymbol{S}_k), \tag{4.28}
$$

resulting from the properties of interaction information. Now since $\boldsymbol{S}_k$ is independent of $\boldsymbol{Y}_k(X_k)$ and $\boldsymbol{Z}_k(X_k)$, we have $I(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k)) = I(\boldsymbol{Y}_k(X_k); \boldsymbol{Z}_k(X_k) \mid \boldsymbol{S}_k)$, and the equality in (4.25) directly follows. The inequality is because the last term in (4.28) is non-negative, due to the properties of mutual information. □

As previously mentioned, interaction information does not have to be non-negative. Here, the equality in (4.25) implies that the interaction information is non-negative since $I_k(\boldsymbol{W}(\boldsymbol{S}); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S})$ is always non-negative. This means that when we ask question X_k, observing signal $\boldsymbol{Y}_k(X_k)$ yields less information when we also know the true answer $\boldsymbol{W}(S)$ to another question S, an intuitive result. We use Theorem 4.4.2 to relate $I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k)$ to $I_k(\boldsymbol{\theta}; \boldsymbol{Y}_k(X_k))$.

4.4.3 Lower Bound on Misclassification Error

We now would like to relate misclassification error to the entropy of the posterior distribution of the linear classifier $\boldsymbol{\theta}$. Theorem 4.4.3 shows that regardless of the estimator $\hat{W}_k$, one cannot reduce misclassification error without bound unless the posterior entropy of $\boldsymbol{\theta}$ is reduced as well. This is due to an important tool in information theory called Fano's Inequality.

Theorem 4.4.3. *For any policy π, a lower bound for the misclassification error under that policy is given by*

$$\mathcal{E}_k^{\pi} \geq \frac{H(\boldsymbol{W}_k(\boldsymbol{S}_k) \,|\, \boldsymbol{S}_k) - I^{\pi}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}_k) - 1}{\log_2 n}.$$

Proof. Suppose we have a fixed question $\boldsymbol{S}_k = S_k$, and let $\hat{W}_k$ be any estimator of $\boldsymbol{W}_k(S_k)$ that is a function of history $\mathcal{Y}_k$ and known assessment question S_k. By Fano's inequality, (Cover 1991, p. 39), we have

$$\mathbb{P}_k(\boldsymbol{W}_k(S_k) \neq \hat{W}_k \,|\, \boldsymbol{S}_k = S_k) \geq \frac{H_k(\boldsymbol{W}_k(S_k) \,|\, \boldsymbol{S}_k = S_k) - 1}{\log_2 n}. \tag{4.29}$$

Taking an expectation over possible assessment questions and past history yields

$$\mathbb{E}^{\pi}\left[\mathbb{P}_k(\boldsymbol{W}_k(\boldsymbol{S}_k) \neq \hat{W}_k \,|\, \boldsymbol{S}_k)\right] \geq \frac{H^{\pi}(\boldsymbol{W}_k(\boldsymbol{S}_k) \,|\, \boldsymbol{\mathcal{Y}}_k,\, \boldsymbol{S}_k) - 1}{\log_2 n}, \tag{4.30}$$

where the right side holds because of the definition of conditional entropy. Now

we use the upper bound on $I_k(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k)$ from Theorem 4.4.2 to show

$$\begin{aligned}
H^\pi(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{\mathcal{Y}}_k,\, \boldsymbol{S}_k) &= H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - I^\pi(\boldsymbol{W}_k(\boldsymbol{S}_k); \boldsymbol{\mathcal{Y}}_k \mid \boldsymbol{S}_k) \\
&= H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - \mathbb{E}^\pi \left[\sum_{\ell=1}^{k} I_\ell(\boldsymbol{W}_\ell(\boldsymbol{S}_\ell); \boldsymbol{Y}_\ell(X_\ell)) \right] \\
&\leq H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - \mathbb{E}^\pi \left[\sum_{\ell=1}^{k} I_\ell(\boldsymbol{Z}_\ell(X_\ell); \boldsymbol{Y}_\ell(X_\ell)) \right] \\
&= H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - \mathbb{E}^\pi \left[\sum_{\ell=1}^{k} I_\ell(\boldsymbol{\theta}; \boldsymbol{Y}_\ell(X_\ell)) \right] \\
&= H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - I^\pi(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}_k),
\end{aligned}$$

where the penultimate equality is from Theorem 4.3.1. Thus, we get

$$\mathbb{E}^\pi \left[\mathbb{P}_k(\boldsymbol{W}_k(\boldsymbol{S}_k) \neq \hat{W} \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k) \right] \geq \frac{H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - I^\pi(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}_k) - 1}{\log_2 n}, \quad (4.31)$$

and the result follows. □

The bound does not provide any insight if $H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{\mathcal{Y}}_k), \boldsymbol{S}_k) < 1$ since the lower bound would be negative. This is most problematic when $n = 2$, in which case, the Shannon entropy of $\boldsymbol{W}_k$ is bounded above by one bit. However, if the conditional entropy of $\boldsymbol{W}_k(S_k)$ after observing signal $\boldsymbol{Y}_k(X_k)$ is still significantly large, the misclassification error will not be reduced past a certain threshold.

There are some interesting conclusions that can be drawn from the lower bound. First, $H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k)$ can be viewed as a constant that describes the problem complexity, representing the expected entropy of evaluation question $\boldsymbol{S}_k$. The lower bound is a linear function with respect to the mutual information of linear classifier $\boldsymbol{\theta}$ and the observation history $\boldsymbol{\mathcal{Y}}_k$.

We can use this result to bound both the knowledge gradient policy and the fully adaptive optimal policy from below. Corollary 4.4.4 below leverages

Theorem 4.4.3 to estimate the optimality gap of knowledge gradient from the optimal policy.

Corollary 4.4.4. *Under noise channel f with channel capacity $C(f)$, the optimal misclassification error under the optimal policy after asking k comparative questions is bounded by*

$$\frac{H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{S}_k) - C(f) \cdot k - 1}{\log_2 n} \leq \mathcal{E}_k^* \leq \mathcal{E}_k^{KG}.$$

Of course, there is a fairly significant gap in the lower bound, since the misclassification errors are non-negative, and yet the lower bound is linear. The gap comes from the second inequality in (4.25), and this upper bound essentially throws out the redundant information about possible evaluation questions learned by previous user responses. Nonetheless, it tells us that posterior entropy reduction is necessary for misclassification error reduction.

4.4.4 Upper Bound for Misclassification Error

We have shown that if the posterior entropy of linear classifier θ is significantly large, then we cannot hope to reduce the misclassification error beyond a given threshold. The converse is not necessarily true: differential entropy can be unboundedly negative, so we cannot realistically expect a differential entropy upper bound on a probability. Nevertheless, we show that misclassification error can be bounded from above by $H(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k)$.

Theorem 4.4.5. *Suppose we define the estimator $\hat{W}$ as follows:*

$$\hat{W} = \arg\max_w \mathbb{P}_k(\boldsymbol{W}_k(\boldsymbol{S}_k) = w \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k). \tag{4.32}$$

Then we have

$$\mathbb{E}_S\left[\mathbb{P}_k\left(\boldsymbol{W}_k \neq \hat{W} \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k\right)\right] \leq \frac{1}{2} H(\boldsymbol{W}_k \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k).$$

Proof. For $u \in \Delta^n$, define $f(u) = 1 - \max_{z\in\mathbb{Z}} u^{(z)}$. We first show

$$f(u) \leq \frac{1}{2} h(u). \tag{4.33}$$

We first write f as a piecewise-linear function, namely

$$f(u) = \begin{cases} 1 - u^{(z)} & u \in M^{(z)}, \end{cases}$$

where $\{M^{(z)}\}$ is a partition of Δ^n. For $I \subset \mathbb{Z}$, define e_I such that

$$e_I^{(z)} = \begin{cases} 1/|I| & z \in I \\ 0 & z \notin I. \end{cases}$$

It is easy to see that $M^{(z)} = \text{Hull}\left(\{e_I : z \in I,\, I \subset \mathbb{Z}\}\right)$. We first want to show that (4.33) holds for all vertices of $M^{(z)}$. Suppose we consider $I \subset \mathbb{Z}$ such that $|I| = k$. Then $f(e_I) = 1 - 1/k$, and $h(e_I) = \log_2 k$. We would like to find the smallest constant L such that

$$1 - 1/k \leq L \cdot \log_2 k$$

for all $k = 1, 2, \ldots, n$. The smallest choice for the constant is $L = 1/2$, which attains the bound at $k = 2$. Therefore, for all vertices u of $M^{(z)}$, we have $f(u) \leq (1/2)\, h(u)$.

Now suppose $u \in M^{(z)}$. On each $M^{(z)}$, we see f is linear. Thus, there exists weights $\lambda_I \geq 0$ such that $\sum_I \lambda_I = 1$ and

$$f(u) = \sum_{I:\, z\in I,\, I\subset\mathbb{Z}} \lambda_I\, f(e_I) \leq \sum_{I:\, z\in I,\, I\subset\mathbb{Z}} \lambda_I\, h(e_I) \leq h(u),$$

with the last statement holding because of the concavity of h. Because we arbitrarily chose $M^{(z)}$, (4.33) holds for all $u \in \Delta^n$.

Now we see that for fixed $\boldsymbol{Y}_k(X_k) = y$ and $\boldsymbol{S}_k = S_k$, using (4.33),

$$\begin{aligned}\mathbb{P}(\boldsymbol{W}_k \neq \hat{W} \mid \boldsymbol{Y}_k(X_k) = y, \boldsymbol{S}_k = S_k) &= 1 - \max_w \mathbb{P}(\boldsymbol{W}_k = w \mid \boldsymbol{Y}_k(X_k) = y, \boldsymbol{S}_k = S_k) \\ &\leq \frac{1}{2} H(\boldsymbol{W}_k \mid \boldsymbol{Y}_k(X_k) = y, \boldsymbol{S}_k = S_k),\end{aligned}$$

and taking expectations of both sides with respect to $\boldsymbol{S}_k$ and $\boldsymbol{Y}_k(X_k)$ yields the result. □

Theorem 4.4.5 shows that minimizing $H_k(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k)$ will also minimize the misclassification error. The proof demonstrates that the upper bound is tight when there is a two-way tie for selecting the most likely response to a given question S_k, with no probabilistic mass on the other $n-2$ alternatives. The worst-case for this upper bound is a uniform predictive distribution, in which case, the upper bound can be as high as $(1/2)\log_2 n$ and can easily grow larger than one with high n.

With Theorems 4.4.3 & 4.4.5, we have shown that the misclassification error is bounded above and below by linear functions of $H_k(\boldsymbol{W}_k(\boldsymbol{S}_k) \mid \boldsymbol{Y}_k(X_k), \boldsymbol{S}_k)$. We have also showed a similar lower bound in terms of the posterior entropy of $\boldsymbol{\theta}$, but there is no corresponding upper bound. This suggests that if the objective is to minimize misclassification error, then entropy pursuit can work, but its effectiveness is heavily reliant on the structure of alternative space $\mathbb{X}^m$.

4.4.5 Interaction-based Entropy Policy

Since maximizing $I(\boldsymbol{W}_k(\boldsymbol{S}_k)\,;\,\boldsymbol{Y}_k(X_k)\,|\,\boldsymbol{S}_k)$, is equivalent to minimizing misclassification error, it would be beneficial to analyze policies that attempt to minimize this different entropy metric. First, we show below in Corollary 4.4.6 that maximizing this quantity is equivalent to maximizing a penalized version of our original notion of entropy.

Corollary 4.4.6. *For a given comparative question $X_k \in \mathbb{X}^m$,*

$$I_k(\boldsymbol{W}_k(\boldsymbol{S}_k)\,;\,\boldsymbol{Y}_k(X_k)\,|\,\boldsymbol{S}_k) = \varphi\left(u(X_k)\,;\,f\right) - \mathbb{E}\left[\varphi\left(U_k^{(\boldsymbol{W}_k,\cdot)}\left(X_k\,|\,\boldsymbol{S}_k\right)\,;\,f\right)\right]. \quad (4.34)$$

Proof. Starting with (4.28) from Theorem 4.4.2, we have

$$I_k(\boldsymbol{W}_k(\boldsymbol{S}_k);\boldsymbol{Y}_k(X_k)\,|\,\boldsymbol{S}) = I_k(\boldsymbol{Y}_k(X_k);\boldsymbol{Z}_k(X_k)) - I_k(\boldsymbol{Y}_k(X_k);\boldsymbol{Z}_k(X_k)\,|\,\boldsymbol{W}_k(\boldsymbol{S}_k),\,\boldsymbol{S}_k). \quad (4.35)$$

Now we use the definition of φ from Theorem 4.3.2 to express both terms on the right side of the (4.35) above. It is immediate from Theorem 4.3.2 that

$$I_k(\boldsymbol{Y}_k(X_k)\,;\,\boldsymbol{Z}_k(X_k)) = \varphi(u_k(X_k)\,;\,f).$$

The second term of (4.35) is merely a variant of the first. By the definition of conditional mutual information (Cover 1991, p. 23), we have

$$\begin{aligned} &I_k\left(\boldsymbol{Y}_k(X_k)\,;\,\boldsymbol{Z}_k(X_k)\,|\,\boldsymbol{W}_k(\boldsymbol{S}_k),\,\boldsymbol{S}_k\right) \\ &\quad= \mathbb{E}_k\left[I_k\left(\boldsymbol{Y}_k(X_k)\,;\,\boldsymbol{Z}_k(X_k)\,|\,\boldsymbol{W}_k(\boldsymbol{S}_k) = W_k,\,\boldsymbol{S}_k = S_k\right)\right], \quad (4.36) \\ &\quad= \mathbb{E}_k\left[\varphi\left(U_k^{(\boldsymbol{W}_k,\cdot)}\left(X_k\,|\,\boldsymbol{S}_k\right)\,;\,f\right)\right], \quad (4.37) \end{aligned}$$

where the expression on the inside of the expectation in (4.36) is only the same mutual information calculation performed, except where the posterior $p_k(\cdot)$ is

restricted to a particular polytope $B^{(w)}(S_k)$. Thus, (4.36) is equivalent to (4.37), where $U_k(X_k \mid S_k)$ is substituted in place of $u_k(X_k)$ to reflect the influence of the assessment question on the predictive distribution, and the row of this matrix is selected at random according to $u_k(S_k)$. □

Corollary 4.4.6 provides intuition on how an interaction-based policy differs from an entropy pursuit policy. The objective functions in both cases are nearly identical, with the exception of the second term of (4.34), which serves as a penalty function. The purpose of the penalty is to ensure that a query X_k remains informative of possible other questions S_k. In fact, the penalty inside the expectation is zero exactly when the matrix $U_k(X_k \mid S_k)$ is a binary, stochastic matrix, implying that $\boldsymbol{Z}_k(X_k)$ is fully correlated $\boldsymbol{W}_k(S_k)$. Otherwise, as the correlation between these random variables weaken, the penalty term grows larger. And taking the expectation across assessment questions $\boldsymbol{S}_k$ implies that the penalty term encourages asking the user queries that provide a lot of information for answers to other queries.

The difficulty remains in evaluating the interaction-based objective. Below, Theorem 4.4.7 provides a relatively simple approximation to the interaction-based entropy objective defined in (4.34).

Theorem 4.4.7. *The interaction-based objective from* (4.34) *can be approximated by*

$$I_k(\boldsymbol{W}_k(\boldsymbol{S}_k)\,;\,\boldsymbol{Y}_k(X_k) \mid \boldsymbol{S}_k) = \left\langle \operatorname{Cov}\left(U_k^{(\boldsymbol{W}_k(\boldsymbol{S}_k),\cdot)}(X_k \mid \boldsymbol{S}_k)\right)\,,\,\nabla_u^2\varphi(u_* \mid f)\right\rangle + O\left(\sum_{w\in\mathbb{W}} \mathbb{E}_k\left\|U_k^{(w,\cdot)}(X_k \mid \boldsymbol{S}_k) - u_*\right\|^3\right), \tag{4.38}$$

where $\langle\cdot,\cdot\rangle$ denotes the trace inner product. Further, an approximate interaction-based objective is given below by

$$\sup_{X_k\in\mathbb{X}^m} \left\langle \operatorname{Cov}\left(U_k^{(\boldsymbol{W}_k(\boldsymbol{S}_k),\cdot)}(X_k \mid \boldsymbol{S}_k)\right)\,,\,\nabla_u^2\varphi(u_* \mid f)\right\rangle. \tag{4.39}$$

Proof. Starting with (4.34), we will use a second order Taylor expansion on both terms. For a generic $u \in \Delta^m$, we have

$$\varphi(u\,;\,f) = \varphi(u_*) - \frac{1}{2}(u-u_*)^T \nabla_u^2 \varphi(u_*\,;\,f)(u-u_*) + O(\|u-u_*\|^3), \tag{4.40}$$

since $\nabla_u \varphi(u_*\,;\,f) = \beta e$ for some constant $\beta \in \mathbb{R}$, and $e^T(u-u_*) = 0$ since all probability distributions sum to unity. Now, fix assessment question $\boldsymbol{S}_k = S_k$ and true assessment answer $\boldsymbol{W}_k(S_k) = W_k$. If we look at the difference of this expression for the two different arguments $u_k(X_k)$ and $U_K^{(W_k(S_k),\cdot)}(X_k \,|\, S_k)$, we see (omitting the cubic remainder terms)

$$\begin{aligned}
&\varphi(u_k(X_k)\,;\,f) - \varphi(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k)\,;\,f) \\
&= -\frac{1}{2}\left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(u_k(X_k)-u_*\right) \\
&\quad + \frac{1}{2}\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_*\right) \\
&= -\frac{1}{2}\left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(u_k(X_k)-u_*\right) \\
&\quad + \frac{1}{2}\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right) \\
&\quad + \frac{1}{2}\left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(u_k(X_k)-u_*\right) \\
&\quad + \left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right) \\
&= \frac{1}{2}\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right) \\
&\quad + \left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right).
\end{aligned}$$

Consider the last term of the last line, and now allow $\boldsymbol{W}_k(S_k)$ to be random. Taking an expectation,

$$\begin{aligned}
&\mathbb{E}_k\left[\left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\left(U_k^{(\boldsymbol{W}_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right)\right] \\
&\qquad = \left(u_k(X_k)-u_*\right)^T \nabla_u^2 \varphi(u_*\,;\,f)\,\mathbb{E}_k\left[U_k^{(\boldsymbol{W}_k(S_k),\cdot)}(X_k \,|\, S_k) - u_k(X_k)\right] = 0,
\end{aligned}$$

since $\mathbb{E}_k\left[U_k^{(\boldsymbol{W}_k(S_k),\cdot)}(X_k \,|\, S_k)\right] = u_k(X_k)$, and the expectation operator is linear.

Putting everything together, we have

$$\begin{aligned}
&\mathbb{E}_k\left[\varphi(u\,;\,f) - \varphi\left((U_k^{(\boldsymbol{W}_k(\boldsymbol{S}_k),\cdot)}(X_k \mid \boldsymbol{S}_k)\right)\,;\,f)\right] \\
&= \frac{1}{2}\,\mathbb{E}_k\left[\left(U_k^{(W_k(S_k),\cdot)}(X_k \mid S_k) - u_k(X_k)\right)^T \nabla_u^2\varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \mid S_k) - u_k(X_k)\right)\right] \\
&\quad + O\left(\sum_{w\in\mathbb{W}} \mathbb{E}_k\left\|U_k^{(w,\cdot)}(X_k \mid S_k) - u_*\right\|^3\right). \qquad (4.41)
\end{aligned}$$

Lastly, the first term can be rewritten as

$$\begin{aligned}
&E_k\left[\left(U_k^{(W_k(S_k),\cdot)}(X_k \mid S_k) - u_k(X_k)\right)^T \nabla_u^2\varphi(u_*\,;\,f)\left(U_k^{(W_k(S_k),\cdot)}(X_k \mid S_k) - u_k(X_k)\right)\right] \\
&\qquad = \left\langle \mathrm{Cov}\left(U_k^{(W_k(S_k),\cdot)}(X_k \mid S_k)\right)\,,\,\nabla_u^2\varphi(u_*\,;\,f)\right\rangle. \qquad (4.42)
\end{aligned}$$

The result immediately follows. □

The approximation policy in (4.39) is simpler to calculate, in that it only involves deriving the Hessian matrix at the optimal predictive distribution u_* only once, and then maximizing an inner product. The major computational hurdle is effectively estimating the covariance matrix for a given question, which is a difficulty similar to that of the true interaction-based objective as well as the knowledge gradient. Calculating joint distributions for pairs of queries would be a computationally enormous undertaking. This would involve estimating the probabilistic volumes of all convex partitions $\{A^{(z)}\}$ corresponding to individual queries, as well as the intersection of such partitions. Some estimation can be done with subsampling, but doing this for all such combinations is intractable.

That being said, concept behind the interactive-based objective is just as important as the policy it implies. Adding a simple penalty term that quantifies how queries interrelate has the potential to greatly improve the quality of questions asked to the user. Future research can and should delve into variants of

penalty functions that can sufficiently approximate the interaction-based objective, without having to rely on heavily combinatorial calculations. Ideally, a distance function that measures difference in angles between hyperplanes may be sufficient to capture the extent to which a pair of queries are similar.

4.5 Conclusion

In this chapter, we analyze the problem of eliciting a given user's preferences by adaptively querying the user with choice-based questions. We formulate this problem in a sequential active learning setting, where a user's preferences are governed by an unknown linear classifier, and the observed responses are perturbed by noise. We assume the underlying observation model where noise does not depend on the underlying preferences. Under this regime, we show that the differential entropy of the posterior distribution of this linear classifier can be reduced linearly with respect to the number of questions posed. Further, there exists an optimal predictive distribution that allows this optimal linear rate to be attained. We provide sensitivity results that show the entropy reduction is close to maximal when the actual predictive distribution of a given question is close to optimal in L_2 distance.

On the problem of appropriately choosing the alternatives: when the set of alternatives has non-empty interior, we provide a construction to find a question that achieves the linear lower bound to a constant multiplicative factor, and exactly for predictive distributions when $\max\{u_*\} = 1/2$ for pairwise comparisons or $\max\{u_*\} < 1/2$ for multi-way comparisons. When the set of alternatives is large but finite, we have demonstrated through simulation experiments that

one can find questions that consistently yield a linear decrease in differential entropy, and this rate is reasonably close to optimal.

In addition to focusing on differential entropy, we consider misclassification error as an alternative metric that more intuitively captures the knowledge one has for a user's preferences. Using Fano's inequality, a classic result in the field of information theory, we show the performance of the optimal policy with respect to this metric is bounded below by a linear function in posterior entropy, suggesting a relationship between entropy-based and misclassification error-based policies. Our computational results in the next chapter largely confirm this, as the entropy pursuit policy and the knowledge gradient policy perform similarly in a variety of scenarios. For this reason, and the fact that the knowledge gradient requires a significantly larger computational budget, entropy pursuit is preferred for adaptive choice-based active preference learning.

Although this work assumes that the number of alternatives m is constant with respect to time, this can be relaxed with a word of caution. From the perspective of entropy, it is always beneficial to increase m, which can be misleading. Thus, if m is allowed to vary with time, one should not use entropy pursuit to choose m, and should use another method to select the number of alternatives to present to the user. This may be done by fixing a static sequence m_k in advance, or the parameter could be adjusted adaptively by another policy in tandem with entropy pursuit. Both approaches would most likely require extensive precomputation, since the geometry of the space of alternatives would heavily affect any policy governing m. Similar is the case of when a suitable prior for the user is not known. In practice, this would also dictate the need for a preprocessing step, perhaps fitting a Gaussian mixture to a population of

estimated linear classifiers (Chen and Frazier 2016, see). Regardless, this motivates the use of entropy pursuit in adaptive choice-based preference elicitation, as well as the study of its effectiveness using historical user responses and experimentation.

REFERENCES

[1] Nir Ailon. "An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity". In: *Journal of Machine Learning Research* 13.Jan (2012), pp. 137–164.

[2] José M Bernardo. "Expected information as expected utility". In: *The Annals of Statistics* (1979), pp. 686–690.

[3] Eric Brochu, Tyson Brochu, and Nando de Freitas. "A Bayesian interactive optimization approach to procedural animation design". In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association. 2010, pp. 103–112.

[4] Bangrui Chen and Peter Frazier. "The Bayesian Linear Information Filtering Problem". In: *arXiv preprint arXiv:1605.09088* (2016).

[5] Thomas M Cover. *Elements of information theory*. John Wiley, 1991.

[6] David L Donoho and Miriam Gasko. "Breakdown properties of location estimates based on halfspace depth and projected outlyingness". In: *The Annals of Statistics* (1992), pp. 1803–1827.

[7] Daria Dzyabura and John R Hauser. "Active machine learning for consideration heuristics". In: *Marketing Science* 30.5 (2011), pp. 801–819.

[8] Johannes Fürnkranz and Eyke Hüllermeier. "Pairwise preference learning and ranking". In: *European conference on machine learning*. Springer. 2003, pp. 145–156.

[9] Robert G Gallager. *Information theory and reliable communication*. Vol. 2. Springer, 1968.

[10] Paul E Green and Venkatachary Srinivasan. "Conjoint analysis in consumer research: issues and outlook". In: *Journal of consumer research* 5.2 (1978), pp. 103–123.

[11] Neil Houlsby et al. "Bayesian active learning for classification and preference learning". In: *arXiv preprint arXiv:1112.5745* (2011).

[12] Bruno Jedynak, Peter I Frazier, Raphael Sznitman, et al. "Twenty questions with noise: Bayes optimal policies for entropy loss". In: *Journal of Applied Probability* 49.1 (2012), pp. 114–136.

[13] Dennis V Lindley. "On a measure of the information provided by an experiment". In: *The Annals of Mathematical Statistics* (1956), pp. 986–1005.

[14] Jordan J Louviere, David A Hensher, and Joffre D Swait. *Stated choice methods: analysis and applications*. Cambridge University Press, 2000.

[15] David JC MacKay. "Information-based objective functions for active data selection". In: *Neural computation* 4.4 (1992), pp. 590–604.

[16] Sebastián Maldonado, Ricardo Montoya, and Richard Weber. "Advanced conjoint analysis using feature selection via support vector machines". In: *European Journal of Operational Research* 241.2 (2015), pp. 564–574.

[17] Robert D Nowak. "The geometry of generalized binary search". In: *Information Theory, IEEE Transactions on* 57.12 (2011), pp. 7893–7906.

[18] Denis Saure and Juan Pablo Vielma. "Ellipsoidal Methods for Adaptive Choice-Based Conjoint Analysis". In: *preprint SSRN 2798984* (2016).

[19] William W Cohen Robert E Schapire and Yoram Singer. "Learning to order things". In: *Advances in Neural Information Processing Systems* 10.451 (1998), p. 24.

[20] Claude E Shannon and Warren Weaver. *The mathematical theory of communication*. University of Illinois press, 1948.

[21] Olivier Toubia, John Hauser, and Rosanna Garcia. "Probabilistic polyhedral methods for adaptive choice-based conjoint analysis: Theory and application". In: *Marketing Science* 26.5 (2007), pp. 596–610.

[22] Olivier Toubia, John R Hauser, and Duncan I Simester. "Polyhedral methods for adaptive choice-based conjoint analysis". In: *Journal of Marketing Research* 41.1 (2004), pp. 116–131.

[23] John W Tukey. "Mathematics and the picturing of data". In: *Proceedings of the international congress of mathematicians*. Vol. 2. 1975, pp. 523–531.

[24] Jie Yu, Peter Goos, and Martina Vandebroek. "A comparison of different Bayesian design criteria for setting up stated preference studies". In: *Transportation Research Part B: Methodological* 46.7 (2012), pp. 789–807.

CHAPTER 5

IMPLEMENTATION AND EMPIRICAL STUDY OF CONJOINT LEARNING

5.1 Introduction

Previously in Chapter 4, we established a model that allowed us to derive theory that characterizes the content of an informative query. This Chapter focuses on implementing the components of this model and evaluating them in various settings.

Perhaps one of the most significant challenges in a Bayesian model is the need to sample from a non-trivial posterior distribution. This is critical for any implementation of the learning policies in our model, and it is the basis for some of our modeling decisions. As a result, Section 5.2 is dedicated to efficiently sampling from posterior distributions generated by our noise model, as well as providing methodology for being able to efficiently generate a new sampler when updating or changing a posterior distribution.

Section 5.4, we provide numerical results demonstrating that entropy pursuit performs similarly to an alternative algorithm that greedily minimizes misclassification error. This is shown in a variety of scenarios and across both metrics. Taking into account the fact that entropy pursuit is far more computationally efficient than the alternative algorithm, we conclude that entropy pursuit should be preferred in practical applications.

Finally, in Section 5.5, we focus on studying an empirical example that uses historical user interactions with real alternatives, and attempt to passively learn

both linear classifiers for 45,000 users *and* the parameters of a noise channel simultaneously using sampling techniques developed earlier in the chapter.

5.2 Sampling via Hit-and-Run

One of the main difficulties is estimating the probabilistic mass contained in a given halfspace. To do this, we will sample from the probability distribution and calculate the proportion of the sample that lies in the halfspace. We can use statistical theory to show that given a sufficiently large sample size, the estimate is "close" to the true amount of probabilistic mass contained within the halfspace.

We pose an algorithm that can sample from a class of prior distributions that contain the priors p_n at every time epoch n. We assume that p_0 is a mixed Gaussian prior: for $\ell = 1, \ldots, L$, we have weights q_ℓ (with $\sum_\ell q_\ell = 1$) that weight the different multivariate Gaussian distributions, characterized by mean μ_ℓ and covariance matrix Σ_ℓ.

At every time epoch $k = 1, \ldots, K$, the update involves the halfspaces induced by the alternatives X_k. If the user selects alternative $Y_k = y_k$, we take the assumed noise channel into account to reweight the prior p_k uniformly over specified polytopes $\{A_k^{(z)}(X_k) : z \in \mathbb{Z}\}$. After this, the entire probability space is re-normalized, yielding p_{k+1}. Thus, we want a sampling algorithm that can efficiently sample over a mixed normal prior, updated with upweighted or downweighted polytopes over the space. Section 5.2 offers a way to sample over such posterior distributions in an efficient way.

The method we pose is to use a hit-and-run sampler, as specified in Bélisle, Romeijn, and Smith (1993). Starting at some $\theta_k \in \Theta$, the sampler selects a direction $v \in \mathbb{R}^d$ uniformly at random, which can be done by drawing from a d-dimensional multivariate Normal distribution and normalizing. We then construct a line $\{\theta_k + r\, v : \ r \in \mathbb{R}\}$ that is centered on original point θ_k and spans the direction d through the entirety of $\mathbb{R}^d$. The sampler then considers the probability distribution p_n conditioned on that line, and selects θ_{k+1} randomly under the conditional distribution. Using this construction guarantees that the proposed point will always be accepted. The algorithm seems simple, but we need to tailor it to the distributions in our given class.

First, we want to find the distribution of a single Gaussian, conditioned on lying on a line in $\mathbb{R}^d$. We can express this distribution in closed form. Interestingly enough, the distribution of a multivariate Gaussian conditioned on a one-dimensional subspace is, in fact, a univariate Gaussian distribution.

Lemma 5.2.1. *Suppose that $G \sim$ Normal(μ, Σ) and that $\mathcal{I} = \{\theta + r\, v : \ r \in \mathbb{R}\}$, where $\theta \in \mathbb{R}^d$ specifies an initial point on $\mathcal{I}$ and $v \in \mathbb{R}^d$ specifies a direction.*

Further, suppose a random point G lies on the line $\mathcal{I}$, and we can write $G = \theta + Rv$. Then for such points G that lie on this line, R is a univariate normal random variable such that

$$R \sim \textit{Normal}\left(-\frac{v^T\Sigma^{-1}(\theta-\mu)}{v^T\Sigma^{-1}v}, \frac{1}{v^T\Sigma^{-1}v}\right). \tag{5.1}$$

Proof. We know that the density function of G is proportional to

$$g(w) \propto \exp\left(-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)\right).$$

Now let $v = \theta + r\,d$, and consider the quantity in the exponential. We have

$$\begin{aligned} g(\theta + r\,v) &\propto \exp\left(-\frac{1}{2}\left(\theta + r\,v - \mu\right)^T \Sigma^{-1}(\theta + r\,v - \mu)\right) \\ &= \exp\left(-\frac{1}{2} r^2\, v^T \Sigma^{-1} v - r\, v^T \Sigma^{-1}(\theta - \mu) - \frac{1}{2}(\theta - \mu)^T \Sigma^{-1}(\theta - \mu)\right) \\ &\propto \exp\left(-\frac{v^T \Sigma^{-1} v}{2}\left(r + \frac{v^T \Sigma^{-1}(\theta - \mu)}{v^T \Sigma^{-1} v}\right)^2\right), \end{aligned}$$

which can be seen by completing the square in terms of r. We can normalize this quantity with respect to r, and find that it is a normal distribution, with the desired mean and variance. □

We can use this result to sample from a single Gaussian distribution or even a mixed normal. But now we want to take the likelihood into account.

Given previous choices between alternatives, the user's response creates a set of polytopes $\{P_\beta\}$ that partition $\mathbb{R}^d$. Momentarily ignoring the prior, all points lying within the same polytope are uniformly distributed, but each polytope itself has a different probabilistic weight. Thus, if we were to draw a line $\mathcal{I} \subset \mathbb{R}^d$, then each portion $\mathcal{I} \cap P_\beta$ has the same probabilistic weight as P_β, and the probability that a point $t \in \mathcal{I}$ falls on that portion of the line is proportional to that probabilistic weight, multiplied by the prior probability of lying in $\mathcal{I} \cap P_\beta$.

Putting these two observations together, we can randomly generate a point on a line, according some distribution in our specified class. We first randomly select one of the segments $\mathcal{I} \cap P_\beta$. Then we generate a mixed Gaussian random variable, conditional on it lying in the specified interval. We can do this by choosing one of the Gaussian distributions in the mix with a certain probability.

Theorem 5.2.2. *Suppose M is the distribution of the mixed Gaussian with mixing proportions $\{q_\ell : \ell = 1, \ldots, L\}$ for Gaussians $\{N_\ell : \ell = 1, \ldots, L\}$. Further, let*

$\mathcal{I} = \{\theta + rv : r \in \mathbb{R}\}$ denote a line centered at $\theta \in \mathbb{R}^d$ for some unit direction $v \in \mathbb{R}^d$.

Further, we parameterize a point M lying on line $\mathcal{I}$ as $M = \theta + Rv$ for some value $R \in \mathbb{R}$. Then for such points $M \in \mathcal{I}$, random variable R takes a univariate mixed Gaussian distribution such that its density g_R is defined as

$$g_R(t) = \sum_{\ell=1}^{L} \lambda_\ell \, g_{R_\ell}(t), \tag{5.2}$$

where g_{R_ℓ} denotes the radial densities defined in Lemma 5.2.1 with the corresponding hyperparameters (μ_ℓ, Σ_ℓ) for all $\ell = 1, \ldots, L$, and the weights λ_ℓ are defined by

$$\lambda_\ell \propto q_\ell \sqrt{\frac{\det(\Sigma_\ell^{-1})}{v^T \Sigma_\ell^{-1} v}}. \tag{5.3}$$

Proof. We know that for each Gaussian density g_{N_ℓ} for all ℓ, there exists a constant $c_\ell > 0$ such that

$$g_{N_\ell}(t) = c_\ell \cdot g_{R_\ell}(t) \tag{5.4}$$

for all $t \in \mathcal{I}$. In fact, using Lemma 5.2.1 and the density of multivariate Normal distributions, it is relatively straightforward to verify that

$$c_\ell = \frac{(2\pi)^{-d/2} \, \det(\Sigma_\ell)^{-1/2}}{(2\pi)^{-1/2} \, (1/v^T \Sigma_\ell^{-1} v)^{-1/2}} \tag{5.5}$$

for all $\ell = 1, \ldots, L$. Now using the definition of M, we have for $t \in \mathcal{I}$

$$\begin{aligned} g_R(t) &\propto g_M(t) \\ &= \sum_{\ell=1}^{L} q_\ell \, g_{N_\ell}(t) \\ &= \sum_{\ell=1}^{L} q_\ell c_\ell \, g_{R_\ell}(t), \end{aligned}$$

showing that $\lambda_\ell \propto q_\ell c_\ell$, since g_{R_ℓ} integrates to one over $\mathcal{I}$. Finally, the constants in the numerator and denominator can be ignored because they are shared by all such $\{\lambda_\ell : \ell = 1, \ldots, L\}$. □

Therefore, we can select Gaussian distribution ℓ with probability distribution proportional to $\{\lambda_\ell\}$ defined in (5.3), then generate a Gaussian conditional on $N_\ell \in \mathcal{I}$.

5.3 Hot Starts for Samplers

The hit-and-run sampler outlined above is efficient but requires an initial point. If the initial point is drawn from the corresponding stationary distribution, then all subsequent points drawn from the sampler will be generated according to that stationary distribution as well. There are situations where the distribution from which we sample will need to be updated with an additional observation or a new parameter. In this case, we can intelligently sample from the old distribution to find an initial point with the correct updated distribution. This concept is referred to as using a hot start, and it is necessary for efficient sampling.

We present hot start techniques for two instances, both of which allowing stationarity to be preserved when updating or adjusting the prior distribution currently being sampled. In Section 5.3.1 addresses hot starting when updating the prior distribution with a new observation. Section 5.3.2 tackles the hot start problem when changing the parameters governing the noise channel.

5.3.1 Hot Starts for Posterior Updates

Every time the prior is updated, the initial starting point for its internal sampler should be updated as well. If this new starting point is chosen so that it

is distributed according to the updated posterior distribution, then there is no need for a burn-in period to "warm-up" the sampler. The problem is that we can only sample points from the prior, and not the posterior. But we can use rejection sampling to find such an initial point.

Suppose we can sample from prior distribution with density p_k, and suppose we have likelihood density $\ell(y|\theta)$ for some observation y. The posterior can be calculated as

$$p_{k+1}(\theta) = \frac{\ell(y|\theta)p_k(\theta)}{\int_\Theta \ell(y|\theta)p_k(\theta)\,d\theta}.$$

In most circumstances, the denominator of this expression is difficult to compute. Instead, we would like to draw from the existing sampler corresponding to p_k, and accept this point with a certain probability. This is the premise behind rejection sampling (Casella, Robert, and Wells 2004). In this scheme if we wanted to sample from distribution g using distribution f, we find a constant M such that $p_{k+1}(\theta)/p_k(\theta) \leq M$ for all θ. With this, sample θ from f, and sample $U \sim \text{Uniform}[0,1]$ independently. Then we accept θ if $U \leq g(\theta)/\left(M\, f(\theta)\right)$. Otherwise, we reject θ and repeat the process again. This guarantees that θ has the correct distribution. For maximal efficiency, one must choose this constant M to be as tight as possible. The unconditional acceptance probability is equal to $1/M$, which implies that the mean number of draws until acceptance is equal to M.

Below in Theorem 5.3.1, we provide a rejection sampling scheme that allows one to find a starting point for the posterior sampler by sampling from the prior.

Theorem 5.3.1. *Suppose the prior for random variable θ at time epoch k is defined as p_k. Further, suppose one observes signal y_{k+1} at the next time epoch. Accordingly, the*

prior is updated as

$$p_{k+1}(\theta \mid y_{k+1}) \propto p_k(\theta) \cdot \ell(y_{k+1} \mid \theta),$$

where $\ell(y_{k+1} \mid \theta)$ *denotes the likelihood of observing* y_{k+1} *given* θ*. Now consider the following sampling scheme.*

Algorithm 5.3.1 Hot start routine for posterior updates.

1: $U \leftarrow 1$
2: $u \leftarrow 0$
3: **while** $U > u$ **do**
4: Draw $\theta \sim p_k$
5: Draw $U \sim \text{Uniform}[0, 1)$
6: $u \leftarrow \dfrac{\ell(y_{k+1} \mid \theta)}{\sup_{\theta'} \ell(y_{k+1} \mid \theta')}$
7: **end while**
8: **return** θ

Then the resulting sample has the correct distribution, with $\theta \sim p_{k+1}(\cdot \mid y_{k+1})$*.*

Proof. We use the rejection sampling framework from above. A simple choice for M is

$$\frac{p_{k+1}(\theta \mid y_{k+1})}{p_k(\theta)} = \frac{\ell(y_{k+1} \mid \theta)}{\int_\Theta \ell(y_{k+1} \mid \theta) p_k(\theta)\, d\theta} \leq \frac{\sup_\theta \ell(y_{k+1} \mid \theta)}{\int_\Theta \ell(y_{k+1} \mid \theta) p_k(\theta)\, d\theta}, \tag{5.6}$$

which depends on observation y_{k+1}. This gives us the desired ratio

$$\frac{p_{k+1}(\theta \mid y_{k+1})}{M \cdot p_k(\theta)} = \frac{\ell(y_{k+1} \mid \theta)}{\sup_{\theta'} \ell(y_{k+1} \mid \theta')}, \tag{5.7}$$

and we accept or reject accordingly. □

The benefit of Theorem 5.3.1 is that the conditional acceptance probability only depends on the relative likelihoods of seeing observation y, and does not depend on the prior. This makes calculating the ratio extremely efficient in most

circumstances. The drawback is that this sampling scheme does not work efficiently if there exist signals y such that the likelihood is extremely low for a given θ. If that is the case, M can be extremely large. Fortunately, in cases where the likelihood and prior distributions are mutually absolutely continuous, this routine allows one to hot start a sampler for the posterior using a sample from the prior. This sampling scheme works well in the case of a discrete noise channel, where all signals can occur with positive probability. We outline this case below in Corollary 5.3.2.

Corollary 5.3.2. *Suppose we are using the bit-flip noise model and can write the likelihood as* $\ell(y|\theta) = \sum_{z\in\mathbb{Z}} f^{(z)}(y)\mathbb{I}(\theta \in A^{(z)})$*, where* $\{A^{(z)} : z \in \mathbb{Z}\}$ *is a partition of* $\mathbb{R}^d$*. Accordingly, let* $z_k(\theta)$ *denote the unique value* $z \in \mathbb{Z}$ *such that* $\theta \in A_k^{(z_k(\theta))}$*. Consider the following sampling scheme, where*

Algorithm 5.3.2 Hot start for posterior update under bit-flip noise.

1: $U \leftarrow 1$
2: $u \leftarrow 0$
3: **while** $U > u$ **do**
4: Draw $\theta \sim p_k$
5: Draw $U \sim \text{Uniform}[0, 1)$
6: $u \leftarrow \dfrac{f^{(z)}(y)}{\sup_{i\in\mathbb{Z}} f^{(i)}(y)}$
7: **end while**
8: **return** θ

Then the resulting sample has the correct distribution, with $\theta \sim p_{k+1}(\cdot \mid y_{k+1})$*. Further, suppose that the noise channel is a discrete symmetric noise channel parameterized by* α*, such that* $P_\alpha = \alpha I + (1-\alpha)(1/m)ee^T$*. Consider the sampling scheme*

Algorithm 5.3.3 Hot start for posterior update under symmetric noise channel.

$U \leftarrow 1$
$u \leftarrow (1-\alpha)/(m\alpha)$
while $U > u$ **do**
  Draw $\theta \sim p_k$
  Draw $U \sim \text{Uniform}[0, 1)$
  if $y_{k+1} = z_{k+1}(\theta)$ **then**
    $u \leftarrow 1$
  end if
end while
return θ

Then the sample has the correct distribution and $\theta \sim p_{k+1}$.

Proof. The first part of the claim is obvious from Theorem 5.3.1. To derive the second sampling scheme, we only substitute $f^{(z)}(y_{k+1}) = P^{(zy_{k+1})}$, and notice that

$$\frac{p_{k+1}(\theta \mid y_{k+1})}{M \cdot p_k(\theta)} = \begin{cases} \dfrac{f^{(z)}(y_{k+1})}{\sup_{i \in \mathbb{Z}} f^{(i)}(y_{k+1})} & \theta \in A^{(z)}. \end{cases} \tag{5.8}$$

Further, the largest elements of stochastic transmission matrix P are on the diagonal, so the denominator is easy to find. If $y_{k+1} = z_{k+1}(\theta)$, the candidate θ will be accepted with probability one, since we assume the diagonal entries of P are larger than those on the rest of the row. The rest follows from algebra. □

Corollary 5.3.2 provides a way of sampling from a prior distribution to efficiently find a starting point for a new posterior sampler, especially in the case of a discrete symmetric noise channel. It is relatively straightforward to use the same technique for other discrete transmission matrices. If one can sample according to the prior distribution p_0, one can use the accept-reject routines to ensure that samplers from any posterior starts with a point already in the correct stationary distribution. Hence, stationarity is always maintained, and no

burn period is required for these samplers.

5.3.2 Hot Starts for Adjusting Noise Parameters

There will be instances when a posterior has been updated with many observations but one wants to sample according to a posterior using a different set of noise parameters. This will be important when we use a Gibbs sampler in Section 5.3.3 to simultaneously estimate a noise channel and the linear classifier for many users. It is also relevant when the adaptive learning algorithm needs to be tuned for instances where it learns too passively or aggressively. In this scenario, we can use a similar rejection sampling technique to find a starting point for the new sampler by using the old one, and this is detailed in Theorem 5.3.3 below.

Theorem 5.3.3. *Suppose we make observations under a bit-flip noise channel parameterized by likelihood functions* $\{f^{(z)} : z \in \mathbb{Z}\}$*, such that* $\ell(y \,|\, \theta) = \sum_{z \in \mathbb{Z}} f^{(z)}(y)\, \mathbb{I}(\theta \in A_k^{(z)}$*. Let* $z_k(\theta)$ *denote the unique value* $z \in \mathbb{Z}$ *such that* $\theta \in A_k^{(z_k(\theta))}$*. Suppose we can express the posterior of linear classifier* θ *after* K *observation as*

$$p_K(\theta \,|\, \mathcal{Y}_K \,;\, f) = D_f \cdot p_0(\theta) \prod_{k=1}^{K} f^{(z_k(\theta))}(y_k) \tag{5.9}$$

for some normalizing constant $D_f > 0$*. Now suppose the noise channel is changed to* $\{g^{(z)} : z \in \mathbb{Z}\}$*, and consider the sampling scheme below.*

Algorithm 5.3.4 Hot start for noise channel update.

1: $U \leftarrow 1$
2: $u \leftarrow 0$
3: **while** $U > u$ **do**
4: Draw $\theta \sim p_{K,f}$
5: Draw $U \sim \text{Uniform}[0, 1)$
6: $u \leftarrow \prod_{k=1}^{K} \frac{g^{(z_k(\theta))}(y_k)/f^{(z_k(\theta))}(y_k)}{\sup_{z\in\mathbb{Z}} g^{(z)}(y_k)/f^{(z)}(y_k)}$
7: **end while**
8: **return** θ

Then $\theta \sim p_K(\theta \,|\, \mathcal{Y}_K \,;\, g)$.

Proof. First, we consider the ratio

$$\frac{p_K(\theta \,|\, \mathcal{Y}_K \,;\, g)}{p_K(\theta \,|\, \mathcal{Y}_K \,;\, f)} = \frac{D_g \cdot p_0(\theta) \prod_{k=1}^{K} g^{(z_k(\theta))}(y_k)}{D_f \cdot p_0(\theta) \prod_{k=1}^{K} f^{(z_k(\theta))}(y_k)} \tag{5.10}$$

and find an upper bound for it. A relatively simple upper bound that depends on $\mathcal{Y}_K$ is given by

$$M = (D_g/D_f) \cdot \prod_{k=1}^{K} \sup_{z\in\mathbb{Z}} g^{(z)}(y_k)/f^{(z)}(y_k). \tag{5.11}$$

Therefore, the normalized ratio is given by

$$\frac{p_K(\theta \,|\, \mathcal{Y}_K \,;\, g)}{M\, p_K(\theta \,|\, \mathcal{Y}_K \,;\, f)} = \prod_{k=1}^{K} \frac{g^{(z_k(\theta))}(y_k)/f^{(z_k(\theta))}}{\sup_{z\in\mathbb{Z}} g^{(z)}(y_k)/f^{(z)}(y_k)}, \tag{5.12}$$

and we accept or reject accordingly. □

Theorem 5.3.3 uses the likelihood ratio of the new and old noise channels to determine whether accept or reject a candidate point θ. This makes it much easier to hot start a new sampler with a different noise channel. Similar to the previous hot start methods, however, there are still some drawbacks. The worst-case likelihood ratios need to be bounded, and using this method when K is

large can be challenging. In fact, the expected number of iterations until acceptance can be exponentially decreasing in K. Nevertheless, we can apply this to discrete noise channels, and this is highlighted below by Corollary 5.3.4.

Corollary 5.3.4. *Let two discrete noise channels be parameterized by transmission matrices P and Q, and consider the sampling scheme below.*

Algorithm 5.3.5 Hot start for discrete noise channel update.

$U \leftarrow 1$
$u \leftarrow 0$
while $U > u$ **do**
  Draw $\theta \sim p_K(\cdot \,|\, \mathcal{Y}_K \,;\, P)$
  Draw $U \sim \text{Uniform}[0, 1)$
  $u \leftarrow \prod_{k=1}^{K} \dfrac{Q^{(z_k(\theta), y_k)} / P^{(z_k(\theta), y_k)}}{\sup_{z \in \mathbb{Z}} Q^{(z, y_k)} / P^{(z, y_k)}}$
end while
return θ

Then the generated vector has the desired distribution, with $\theta \sim p_K(\cdot \,|\, \mathcal{Y}_K \,;\, Q)$. Further, in the case of two symmetric noise channels where $P = \alpha I + (1-\alpha)(1/m)ee^T$ and $Q = \beta I + (1-\beta)(1/m)ee^T$, consider the following sample scheme.

Algorithm 5.3.6 Hot start for symmetric noise channel update.

$U \leftarrow 1$
$u \leftarrow 0$
while $U > u$ **do**
  Draw $\theta \sim p_K(\cdot \,|\, \mathcal{Y}_K \,;\, P)$
  Draw $U \sim \text{Uniform}[0, 1)$
  $u \leftarrow R(\alpha, \beta)^{N(\alpha, \beta, \mathcal{Y}_K, \theta)}$
end while
return θ

where $R(\alpha, \beta)$ is the ratio

$$R(\alpha, \beta) = \frac{\min\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}, \frac{1-\beta}{1-\alpha}\right)}{\max\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}, \frac{1-\beta}{1-\alpha}\right)} \tag{5.13}$$

and $N(\alpha, \beta, \mathcal{Y}_K, \theta)$ *is defined by*

$$N(\alpha, \beta, \mathcal{Y}_K, \theta) = \begin{cases} \sum_{k=1}^{K} \mathbb{I}(y_k = z_k(\theta)) & \beta < \alpha \\ K - \sum_{k=1}^{K} \mathbb{I}(y_k = z_k(\theta)) & \beta > \alpha. \end{cases} \tag{5.14}$$

Then $\theta \sim p_K(\cdot \,|\, \mathcal{Y}_K \,;\, Q)$.

Proof. The first half of the claim is evident by letting $f^{(z)}(y) = P^{(zy)}$ and $g^{(z)}(y) = Q^{(zy)}$ and applying Theorem 5.3.3. The second half of the claim is a special case of the first. It is easy to verify that if $\beta > \alpha$, then the elements on the diagonal of Q are larger than those of P, and the off-diagonal elements of Q are smaller than those of P. In this case, the ratio from (5.12) is equal to

$$\prod_{k=1}^{K} \frac{Q^{(z_k(\theta), y_k)} / P^{(z_k(\theta), y_k)}}{\sup_{z \in \mathbb{Z}} Q^{(z, y_k)} / P^{(z, y_k)}} = \frac{\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}\right)^{\sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)} \left(\frac{1-\beta}{1-\alpha}\right)^{K - \sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)}}{\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}\right)^{K}}$$

$$= \left(\frac{\left(\frac{1-\beta}{1-\alpha}\right)}{\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}\right)} \right)^{K - \sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)}.$$

Otherwise, if $\beta < \alpha$, the opposite is true, and similarly,

$$\prod_{k=1}^{K} \frac{Q^{(z_k(\theta), y_k)} / P^{(z_k(\theta), y_k)}}{\sup_{z \in \mathbb{Z}} Q^{(z, y_k)} / P^{(z, y_k)}} = \frac{\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}\right)^{\sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)} \left(\frac{1-\beta}{1-\alpha}\right)^{K - \sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)}}{\left(\frac{1-\beta}{1-\alpha}\right)^{K}}$$

$$= \left(\frac{\left(\frac{\beta + (1-\beta)(1/m)}{\alpha + (1-\alpha)(1/m)}\right)}{\left(\frac{1-\beta}{1-\alpha}\right)} \right)^{\sum_{k=1}^{K} \mathbb{I}(z_k(\theta) = y_k)}.$$

Combining both of these completes the proof. □

Corollary 5.3.4 above shows how to hot start a new sampler when learning from observations perturbed by a discrete noise channel. From (5.14), it is clear

that $N(\cdot, \cdot, \mathcal{Y}_K, \cdot) \leq K$, which implies that the conditional acceptance probability decreases exponentially as a function of the number of observations K. This can be compounded when α and β are further apart, as shown by the definition of $R(\alpha, \beta)$ in (5.13). However, if the changes in the noise channel are small, Corollary 5.3.4 shows that it is possible to efficiently hot start a new sampler using a new noise channel with the same observations, maintaining the invariant of stationarity.

5.3.3 Learning the Noise Channel with Gibbs Sampling

In cases for numerical studies and other situations, it is desirable to estimate parameters for the noise channel by looking at the interactions of many users with historical queries. The main issue is that the noise channel dictates how the observations affect the updating of the prior. The ideal solution involves learning both the noise channel and the linear classifiers for many users *simultaneously*.

Learning all such quantities at once can be computationally intensive, so we only consider the case of estimating a discrete noise channel. In this scenario, one approach to tackling this massive learning problem is using a Gibbs Sampler. The way to do this involves only two sampling steps. The first involves sampling the noise channel from some posterior q that depends on the user interaction history as well as the current linear classifiers $\{\theta_{j-1}(u) : u = 1, \ldots, U\}$. The second step is to sample new linear classifiers using the updated noise channel P_j. Since we start with a sampler conditioned on noise channel P_{j-1}, sampling a new linear classifier is done by the rejection sampling step given in Corollary 5.3.4. This process is outlined in Algorithm 5.3.7 below.

Algorithm 5.3.7 Joint Estimation Gibbs Sampler

1: **for** samples $j = 1, \ldots, J$ **do**
2: $\quad P_j \leftarrow q(\cdot \mid \{\theta_{j-1}(u) : u = 1, \ldots, U\})$
3: $\quad$ **for** users $u = 1, \ldots, U$ **do**
4: $\quad\quad \theta_j(u) \sim p_K(\cdot \mid \mathcal{Y}_K ; P_j)$ via rejection sampling from $p_K(\cdot \mid \mathcal{Y}_K ; P_{j-1})$
5: $\quad$ **end for**
6: **end for**

Some important questions are how to model the prior $q(\cdot)$ and how to sample from it. The answer to both of these depend on the class of transmission matrices one wants to consider. For the simple case of symmetric noise channel, there is only a single, one-dimensional parameter to estimate. The noise parameter α can have a non-informative Beta prior. In this case, user interactions can be treated as conditionally independent given the noise channel P_α, and therefore, the observations $\mathbb{I}(z_k(\theta(u)) = y_k(u))$ are independent across users u and queries k. Therefore, one can update the prior q, and this posterior is also Beta-distributed. Sampling a new noise channel is now as simple as sampling from a Beta posterior.

Of course, the modeling of noise channels can be much more involved. One can model each row of transmission matrix P as having a Dirichlet prior, and in the same manner as above, this prior can be updated after comparing the sampled linear classifiers $\theta(u)$ against the results from the previous queries.

Lastly, this endeavor can still be computationally arduous, even when restricting ourselves to discrete noise channels. However, there is a significant benefit to using a Gibbs sampler when user interactions are conditionally independent. Once the current noise channel P_j has been sampled, the sampling in lines 3–5 of Algorithm 5.3.7 can be done in parallel. This greatly accelerates the process of sampling from many users and alleviates the main computational

bottleneck of the algorithm. We will use this Gibbs sampler in Section 5.5 to simultaneously learn the linear classifiers of many users as well as the noise channel itself.

5.4 Simulated Computational Results

In the following subsections, we present computational results from simulated responses using vectorizations of real alternatives. Section 5.4.1 discusses our approach and methodology for the numerical experiments, and Section 5.4.2 gives the results of the computational studies and provides insights regarding the performance of the entropy pursuit and knowledge gradient policies.

5.4.1 Methodology

As an alternative space, we use the 13,108 academic papers on arXiv.org from the condensed matter archive written in 2014. The information retrieval literature is rife with different methods on how to represent a document as a vector, including bag of words, term frequency inverse document frequency (Salton and McGill 1986), and word2vec (Goldberg and Levy 2014), along with many others (for an overview of such methods, see Raghavan and Schütze 2008). In practice, the method for vectorizing the alternatives is critical; if the vectors do not sufficiently represent the alternatives, any recommendation system or preference elicitation algorithm will have trouble. For the numerical experiments, we elected to use a vector representation derived from Latent Dirichlet Allocation (LDA) as described by Blei, Ng, and Jordan (2003). The resulting

feature vectors are low-dimensional and dense. Since we cannot compute the posterior distribution analytically, we resort to sampling instead, and the low-dimensional LDA vectors allow for more efficient sampling.

With any method that utilizes Bayesian inference, it is important to have enough structure that allows for an efficient sampling scheme from the resulting posterior distributions. The benefit of having the simple update of up-weighting and down-weighting polytopes is that the sampling scheme becomes quite easy. We use a hit-and-run sampler as described earlier that chooses a direction uniformly from the unit sphere, then samples from the one-dimensional conditional distribution of the next point lying on that line. Now, re-weighting polytopes turns into re-weighting line segments. If it is to easy sample points conditionally on this line, hit-and-run is an efficient way of sampling. We use a multivariate normally distributed prior because it allows for both computational tractability for sampling from this conditional distribution as well as a natural representation of prior information.

To select the hyperparameters for the prior, we sample academic papers and fit a multivariate normal distribution to this sample. Assuming users' linear classifiers have the same form and interpretation as a vector representation is not reasonable in general. However, in the case of academic papers, authors are also readers, and so the content in which the users are interested is closely related to the content they produce. Therefore, in this situation, it is reasonable to assume that a user's parameterization of preferences lives in the same space as the parameterization of the feature set. This is not necessarily the case for other types of alternatives, and even if it were, using feature vectors to model preference vectors may not be the best choice. That being said, there are many

ways to initialize the prior. If one has a history of past user interaction with alternatives, one could estimate the linear preference vector for each user using an expectation maximization scheme, and fit a mixed normal prior to the empirical distribution of estimated linear classifiers, as done by Chen and Frazier (2016). Since the focus here is to compare the performance of the two algorithms of interest, our choice for initializing the prior is sufficient.

5.4.2 Cross-Metric Policy Comparison

We first compare the entropy pursuit and knowledge gradient policies while varying the number of presented alternatives. Due to the large set of alternatives, it is computationally intractable to choose questions that optimally follow either policy, so alternatives are subsampled from $\mathbb{X}$ and we approximate both policies using the alternatives from the subsample. If N alternatives are subsampled, then the approximate entropy pursuit policy requires exhaustively optimizing over combinations of alternatives (permutations if the noise channel is not symmetric), and hence will require maximizing over $\binom{N}{m}$ subsets. On the other hand, the knowledge gradient policy requires comparing $\binom{N}{m}$ informative questions X with $\binom{N}{n}$ assessment questions S, and thus requires estimating $\binom{N}{m}\binom{N}{n}$ quantities. Already, this implies that if the computational budget per question is fixed for both algorithms, one can afford a polynomially larger subsample for entropy pursuit than for knowledge gradient. For example, in the case where $m = n = 2$, a computational budget that allows a subsample of $N = 15$ alternatives for the knowledge gradient policy would allow the entropy pursuit policy a subsample size of $N' = 149$. However, rather than fixing a computational budget for both policies at each step, we allow both policies the

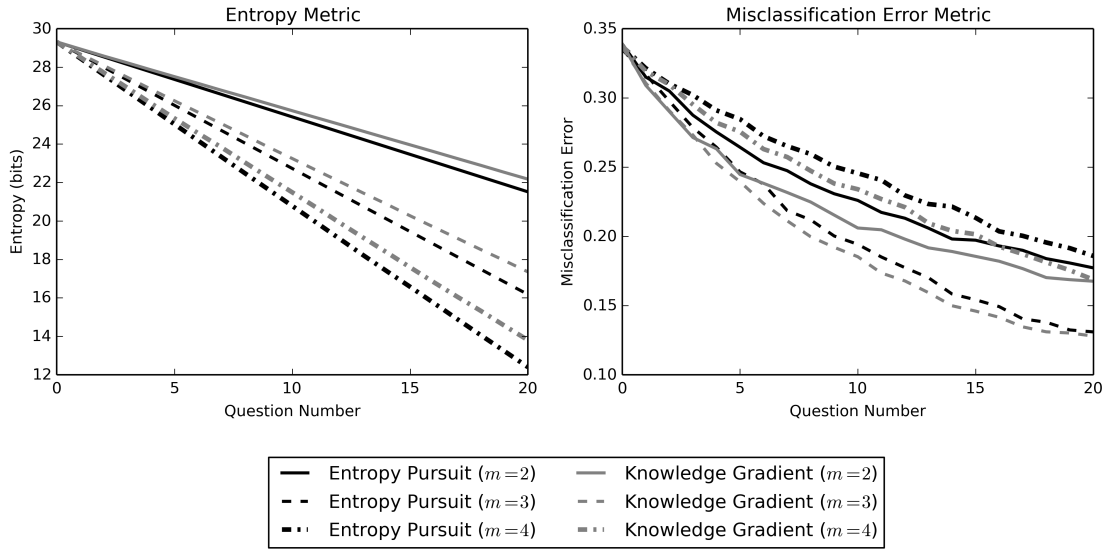


Figure 5.1: Comparison of average performance of the entropy pursuit and knowledge gradient policies under a symmetric noise channel ($\alpha = 0.7$), simulated and averaged with 100 sample paths. Estimates are accurate to ± 0.007 for misclassification error and ± 0.06 bits for entropy.

same number of subsamples, setting $N = 15$ for both policies and all sets of parameters. We do this to allow for a more straightforward comparison of the two policies, although further computational studies should study their performance under a fixed computational budget. Lastly, the numerical study in this chapter fixes $n = 2$. We make this decision because any larger values of n will make the computations prohibitively expensive, and it is not clear that larger values of n will provide any additional benefit.

Figure 5.1 compares the entropy pursuit and knowledge gradient policies by varying m and fixing other parameters to reflect a low-noise, low prior information scenario. As expected, each algorithm performs better on their respective metrics for a fixed number of provided alternatives m. However, a more surprising conclusion is the similarity in performance of the two algorithms for any fixed m for *both metrics*. This suggests that the price to pay for switching

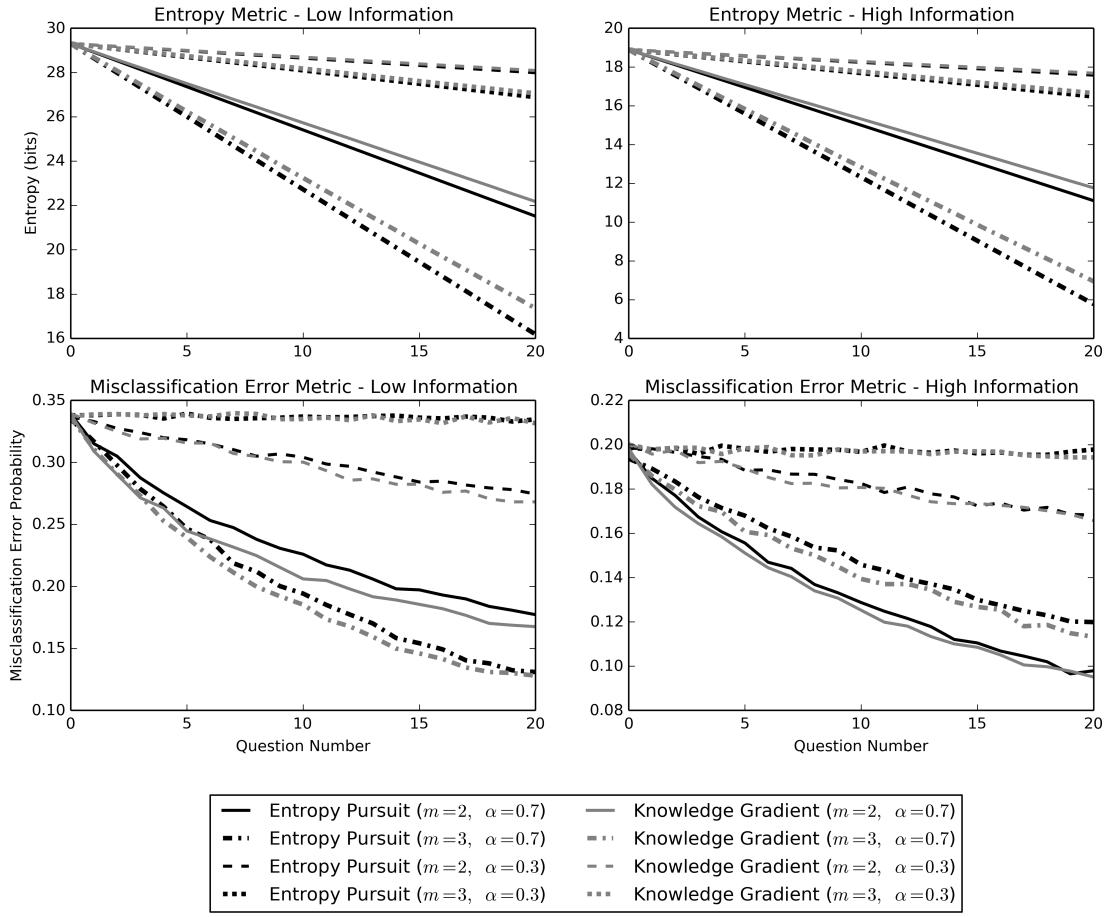


Figure 5.2: Comparison of the entropy pursuit and knowledge gradient policies under a symmetric noise channel for various levels of noise and prior information, simulated and averaged with 100 sample paths. Estimates are accurate to ± 0.007 for misclassification error and ± 0.06 bits for entropy.

from the knowledge gradient policy to the entropy pursuit policy is small compared to the gain in computational efficiency. In fact, if the computational budget for each question were fixed, one would be able to subsample many more alternatives to compute the entropy pursuit policy compared to the knowledge gradient policy, and it is very likely the former would out-perform the latter in this setting. To see if this occurrence takes place in other scenarios, such as those with higher noise and a more informative prior, one can consult Figure 5.2. Again, for all the different parameter settings, both policies perform similarly.

Another interesting aspect of the computational results are the effects of the parameters on the performance of the two policies. Differential entropy predictably decreases faster when more alternatives are presented to the user. In the case of a symmetric noise channel, increasing m only increases the channel capacity for a fixed noise level α. From the perspective of minimizing posterior entropy, this makes sense because offering more alternatives at each time epoch should theoretically allow one to refine the posterior distribution faster. However, in reality, the noise channel most likely varies with the number of offered alternatives m, where the quality of the noise channel degrades as m grows. In the most extreme example, offering too many alternatives to the user will result in a phenomenon called "decision paralysis," where the user's responses will not contain useful information about her preferences. In this case, the model is not capturing the added uncertainty, and focusing on posterior entropy as a performance metric may be misleading.

In contrast, the knowledge gradient policy captures this intuition, since pairwise comparisons decrease misclassification error faster than three-way comparisons in the cases of high noise or a highly informative prior. In fact, three-way comparisons only prevail in a low-noise, low prior information scenario, which is fairly optimistic. Both policies under three-way comparisons were aggressive, and in the high-noise case, they fail to learn anything at all about the user's preferences. In practice, it will be necessary to estimate parameters for the noise channel in order to choose the correct value of m. For now, it suffices to say that pairwise comparisons are robust and reliable.

5.5 Empirical Computational Results

Even though the simulated computations provide much insight into the effectiveness of the policy, they do not quantify the extent to which the model fits reality. There are many questions that the computational efforts in Section 5.4 do not answer. The problem lies in the simulated user responses, which may not be a realistic choice model. As a result, we will take a more empirical approach to evaluate the effectiveness of the our model. This includes testing the response model by using recorded interactions with existent alternatives, as well as testing the computational efficiency of our estimation procedure.

This effort is detailed throughout the rest of the chapter. In Section 5.5.1, we characterize the data set used in the empirical evaluation and delve into steps to clean this data. Section 5.5.2 provides the algorithmic approach used to estimate linear classifiers and noise parameters, as well as validating the estimated data. Lastly, we discuss these computational findings and their implications in Section 5.5.3.

5.5.1 Empirical Data Set

Again, we use academic papers from arXiv.org as a set of alternatives. Specifically, we consider papers from astrophysics (`astro-ph`) category published in the year of 2016. As before, we use Blei, Ng, and Jordan (2003) to generate 10-dimensional LDA vectors for these papers, which assigns a probability distribution on topics, and we take a log-odds transform to generate vectors that take values on $\mathbb{R}^{10}$. We choose LDA vectors because our model favors represen-

tations that are dense and low-dimensional, due to the geometric nature of the inference routine involved.

Unless an experiment or application is created to adaptively ask the desired queries to specific users, any interactions the user has with the website will not necessarily be in the desired form of a comparison between multiple alternatives. However, user interactions stemming from alternatives presented in a list can be interpreted as multiway comparisons between surrounding alternatives (Joachims et al. 2007), and one can successfully use such comparisons to learn a user's preferences (Radlinski and Joachims 2007). We take this approach, interpreting a user interaction with a list as a comparative query. The immediate drawback is that we cannot pose a particular query to a user, implying that we cannot test the effectiveness of policies such as entropy pursuit or knowledge gradient. Nevertheless, we use these interactions to passively learn linear classifiers and parameters of the noise model.

There are many ways to derive a consideration set from a user's interaction with alternatives presented in a list. Our method is to consider each selected paper as a pairwise comparison with the previous one. The logic behind this decision is that the user most likely saw the previous item before clicking on the selected paper, and therefore made an implicit comparison between the two, favoring the one most recently selected. If the first paper from the list is selected, then this is treated as a pairwise comparison with the second paper in the list. Although this seems unintuitive at first, there is research to suggest that a user's eyes focus on the first and second ranked object in a list with roughly equal time (Granka, Joachims, and Gay 2004), and therefore is likely to consider both simultaneously. There are many variations of this that address problems like

how to generate comparisons when adjacent alternatives are selected. We opt for this modeling choice because of its simplicity and computational ease, but it would be beneficial for future numerical experiments to try other methods of generating consideration sets.

On arXiv.org, for each category, we consider user interactions in the "new" list, which present papers that are published on the arXiv website that day. We identify a user by a hashed cookie ID, and this allows us to look at interactions that take place on multiple days. The interactions track which papers were selected on the `astro-ph` new list. To prevent the inclusion of bots that select every document on the list, we strain out users who ever select more than 10 academic papers in a single day.

We consider interactions throughout the year of 2016. This interaction data is split into a training portion, taking place from January 1 through June 30, and a validation portion, taking place from July 1 through December 31. The training portion of the data includes all users, excluding strained bots, that interacted with these new lists, totaling to 45,099 users. The validation portion only considers interactions from users that are present in the training portion, which includes 5,253 users. Even though there are a large number of users for which we do not have validation data, we keep them in the training data to better learn the noise channel.

From Figures 5.3 & 5.4, we can see that almost half of the training sample only interacted with the list only once, and virtually all users in the training sample have no more than 20 interactions total with the `astro-ph` new list. However, there is a non-trivial number of users who, in the course of one day, select more than one academic paper on average, and many of these types of

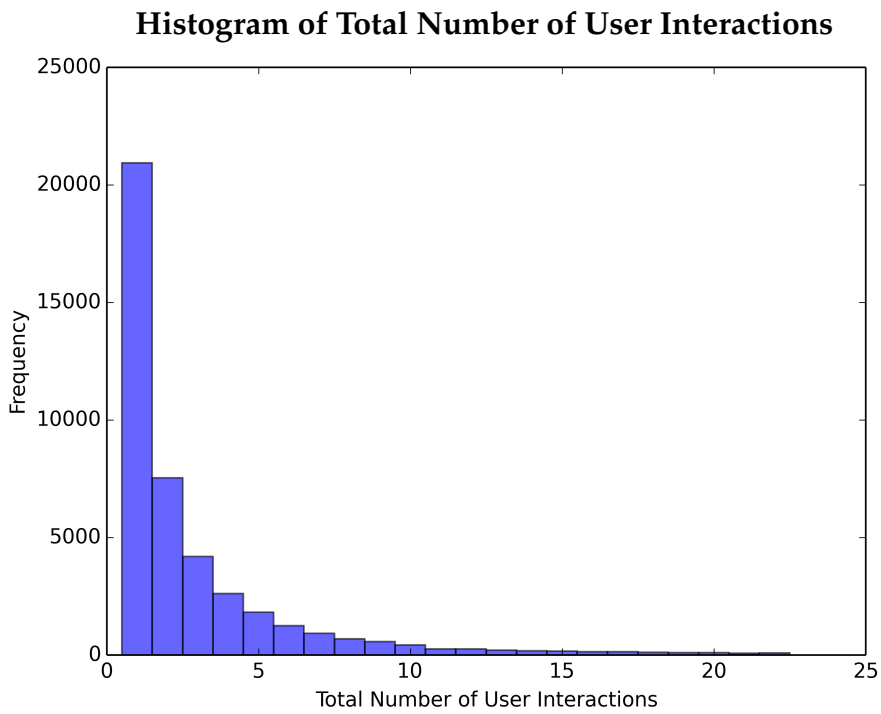


Figure 5.3: A histogram of the total number of papers selected by individual users from the `astro-ph` new list from January 1 to June 30, 2016.

users visit again.

The goal is to use the history of the training users to predict their linear classifiers θ and to estimate the parameters of the noise channel.

5.5.2 Algorithmic Approach

Because the learning involves looking at a past history all at once, we are forced to learn passively and sample from the resulting posteriors. However, this gives us the opportunity to learn the linear classifiers while simultaneously learning parameters of the noise model, using the methodology presented in Section 5.3.3. Afterwards, we evaluate the quality of our estimates by using an

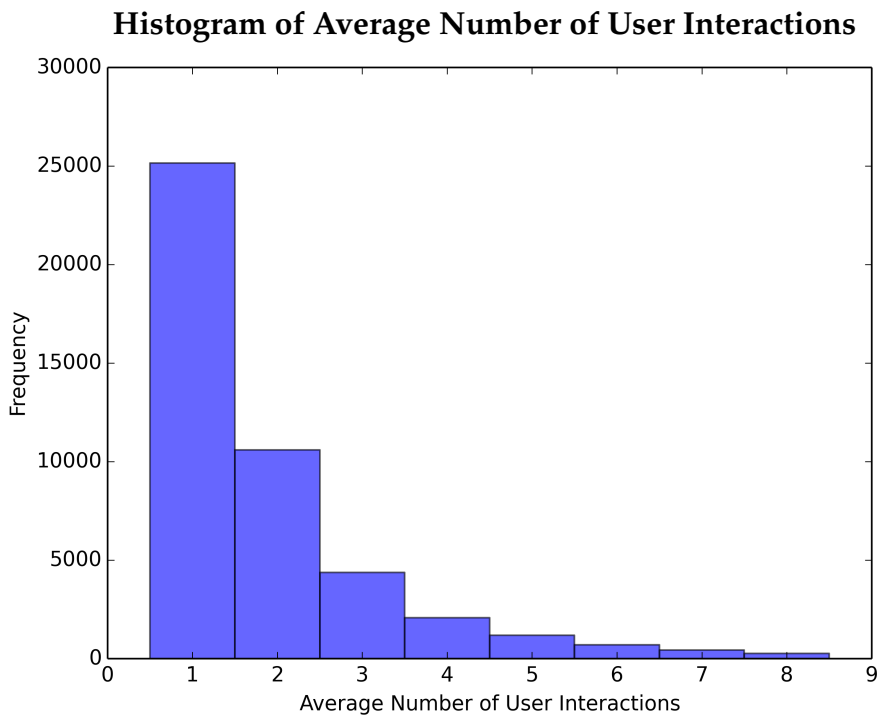


Figure 5.4: A histogram of the average number of papers per session selected by individual users from the `astro-ph` new list from January 1 to June 30, 2016.

entirely different metric for the validation portion of our observations.

To keep things relatively simple, we only attempt learning a symmetric noise channel for the case $m = 2$, corresponding to the previous section when we committed to focusing on pairwise comparisons. In particular, we are learning the parameter α that defines the transmission matrix $P_\alpha = \alpha I + (1-\alpha)(1/2)ee^T$. To reiterate, having $\alpha = 1$ implies that the user always responds to comparative queries by choosing the academic paper that maximizes the dot product with her linear classifier. On the other hand, having $\alpha = 0$ corresponds to the case where the user responds uniformly at random to every query. Loosely speaking, larger the estimated values of α imply more observations that are explained by the utility model. We naively assign a non-informative prior so that $\alpha \sim$

Beta$(1, 1)$.

The first step is to generate the posterior distributions of θ for all 45,099 users in our training set, using each of their interactions. Each user starts with the same multivariate Normal prior, fit to the average document vector and a scaled-up covariance matrix, similarly to Section 5.4. Then for each paper they clicked on the list, we derive their consideration set, and equivalently the comparative question $X \in \mathbb{X}^2$ implied by their click. The corresponding posterior is then updated with the response. Each posterior object maintains a starting point for a sampler, and when it is updated with a new observation, the hot start routine outlined in Section 5.3.1. This allows there to always exist a "warm" starting vector for a sampler if one must be created.

Now we sample according to Algorithm 5.3.7. As mentioned in the corresponding section, the process is greatly accelerated if the sampling step for users in done in parallel, since it would otherwise take hundreds of thousands of sequential hit-and-run proposals just to complete one iteration. We ran the Gibbs sampler for 1000 iterations, acquiring 1000 samples of α as well as 1000 linear classifiers θ for all 45,099 training users. These samples are not independent, and are affected by autocorrelation, as is the case with most Gibbs samplers.

Out-of-sample validation is necessary to evaluate the effectiveness of the model to predict real life outcomes and not merely simulated ones. To evaluate the effectiveness of a set of linear classifiers, we first observe that each θ implies a preferential order over alternatives. And even though a user's choices may be contaminated by noise, the user's response to a query is consistent with this ordering with probability α. Therefore, evaluating this preferential ordering is a viable way to evaluate the performance of our sampled linear classifiers.

We use a metric called discounted cumulative gain (DCG), used to evaluate relevance of alternative presented in an ordered list. Suppose, for a given set of N alternatives presented in an ordered list ℓ, that $\left\{r^{(i)} : i = 1, \ldots, N\right\}$ denotes a list of binary variables indicating whether or not alternative i has been clicked. Then one defines the sum

$$\text{DCG}(\ell) = \sum_{i=1}^{N} \frac{r^{(\ell(i))}}{\log_2(i+1)}.$$

Intuitively, the DCG of a particular ranked list of alternatives is larger when the alternatives more relevant to the user are placed further towards the top of the list. Otherwise, discounting plays a factor, and the opportunity cost of generating a list where relevant alternatives are further down becomes larger.

There are issues when one wants to compare two DCG scores, but finds that a different number of alternatives were clicked in both cases. One way to handle this is to normalize the score by dividing by the maximum attainable score. This is precisely the definition of normalized DCG (nDCG). Mathematically speaking,

$$\text{nDCG}(\ell) = \frac{\text{DCG}(\ell)}{\max_{\ell'} \text{DCG}(\ell')},$$

where the maximum is taken over all permutations of the initial list. Now all nDCG scores fall between zero and one, and can easily be compared with each other.

The way we use nDCG is as follows: each θ implies a ranking of alternatives ℓ_θ, and the nDCG of each of these rankings can be calculated for a particular "new" list on `astro-ph`. But since θ is random, we can compute an *expected* nDCG score for each user, quantifying the extent to which the ranking implied by the sampled classifiers are supported by the real interactions of validation

users. In other words, for every user, we are estimating

$$\mathbb{E}\left[\text{nDCG}(\ell_{\boldsymbol{\theta}})\right] \tag{5.15}$$

with the sample average. Even though the sampled linear classifiers $\theta(u)$ for all users u suffers from a degree of autocorrelation, we include the entire sample in the sample average estimate, with the exception of a burn period (which we detail in the next section). In the following section, we explore the computational results from the passive learning experiment.

5.5.3 Results

The two components to the results are the noise estimation and the nDCG evaluation. Unlike the linear classifiers for the thousands of users, parameter α is one-dimensional and provides a succinct representation of the quality of fit provided by those linear classifiers. Accordingly, Figure 5.5 provides a traceplot for α to show the progression of the Gibbs sampler. Among other things, it shows that the α samples reached a steady state at around 200 iterations, which suggests incorporating a burn period for inference. This may seem odd, since we constructed our hot-start mechanisms to always preserve stationarity. However, this stationarity is conditional on the noise channel, and if the α parameter has not reached a stationary distribution, then the unconditional $\theta(u)$ samples will not have reached a stationary distribution. As a result, the loss in (5.15) will be estimated using a sample average using the last 800 samples of the user. One can see that the chain mixed well once it reached steady-state, albeit in a relatively small band 0.43–0.47. Both of these statements are good news, indicating the sampling mechanism works when presented with real data.

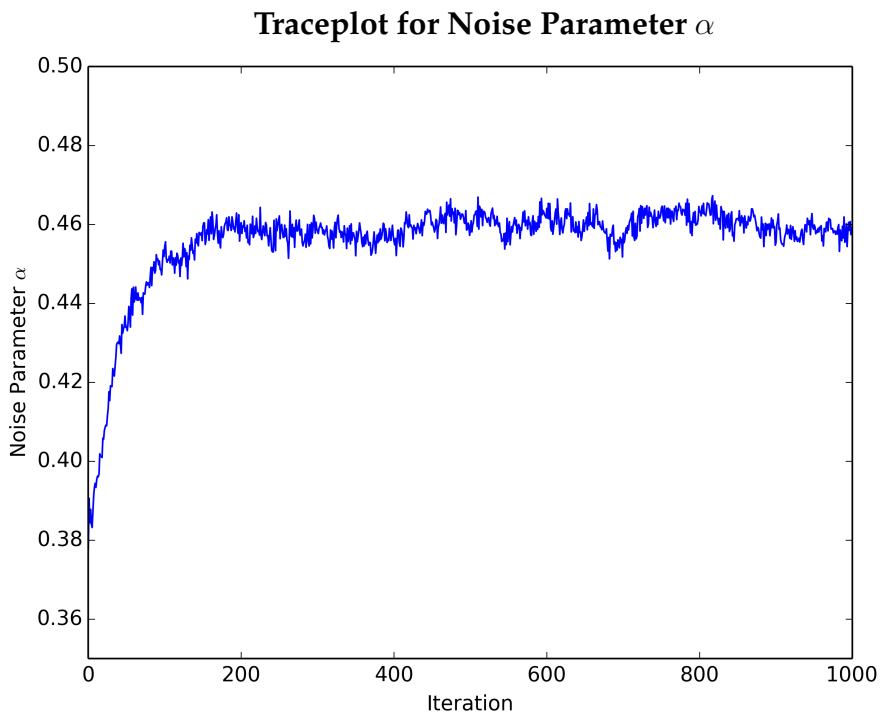


Figure 5.5: Traceplot of noise parameter α from the Gibbs sampler.

Using a burn period of 200 iterations, we can take a look at the distribution of α. Figure 5.6 shows a histogram of the sampled values of α. The distribution is bell-shaped and has relatively little skew. In particular, we calculated a sample mean of 0.4595 and a sample standard deviation of 0.0029. This implies that the user responses are not perturbed by noise roughly 46% of the time.

Now we turn to the nDCG validation results. Unfortunately, the validation results are not as promising. Figure 5.7 shows that the distribution of nDCG among users is fairly dispersed and heavily skewed right. In particular, we compute a mean of 0.3126, a median of 0.2938, and a standard deviation of 0.100. Strictly speaking, this indicates that the learning capability of our model only achieved 31% of the maximum DCG value. But this is not a very useful explanation. To a provide reference, the nDCG achieved when choosing randomly

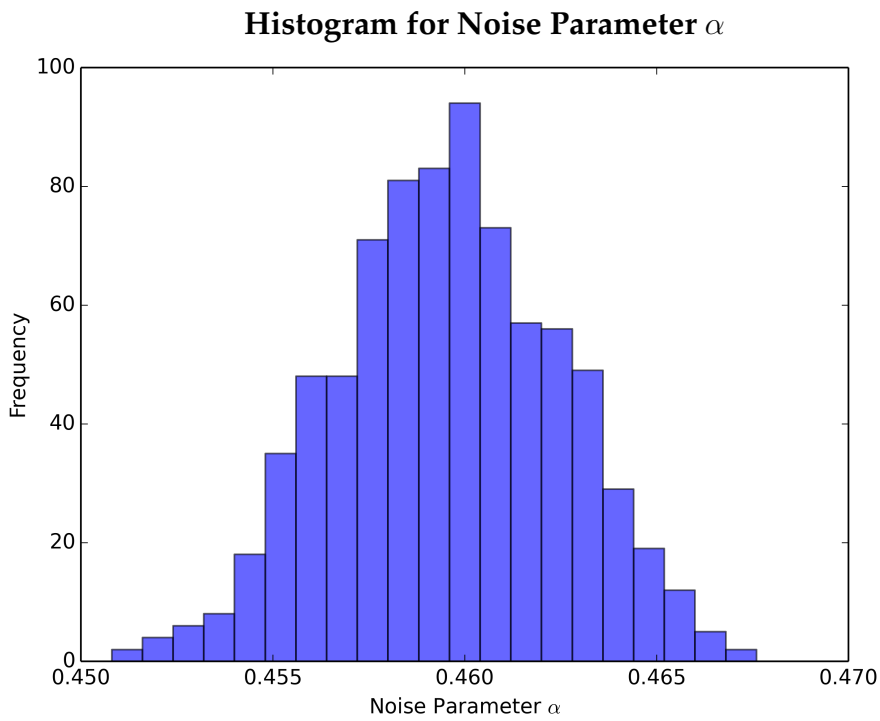


Figure 5.6: Histogram of noise parameter α from the Gibbs sampler, with burn period of 200 iterations.

from a list of 40 alternatives, which is the length of a typical "new" `astro-ph` list, is about 0.277. Roughly speaking, we can qualify the predictive capability of our linear classifiers as slightly better than flipping a coin. In fact, using the inverse of nDCG, $t \mapsto 2^{1/t} - 1$, suggests that a user selecting one paper from a list would search below the ninth ranked paper to find something relevant for herself (by plugging the mean into the convex inverse and using Jensen's inequality). This is fairly discouraging for the model's predictive abilities at large.

Delving further into the data, one may think this phenomenon is due to the large time periods involved in training or validating. However, looking at Figure 5.8, this performance does not seem to change significantly with respect to the date of the observations. But this time series raises a seemingly contradictory statistic: the average nDCG for a given day among all users is 0.3687, which

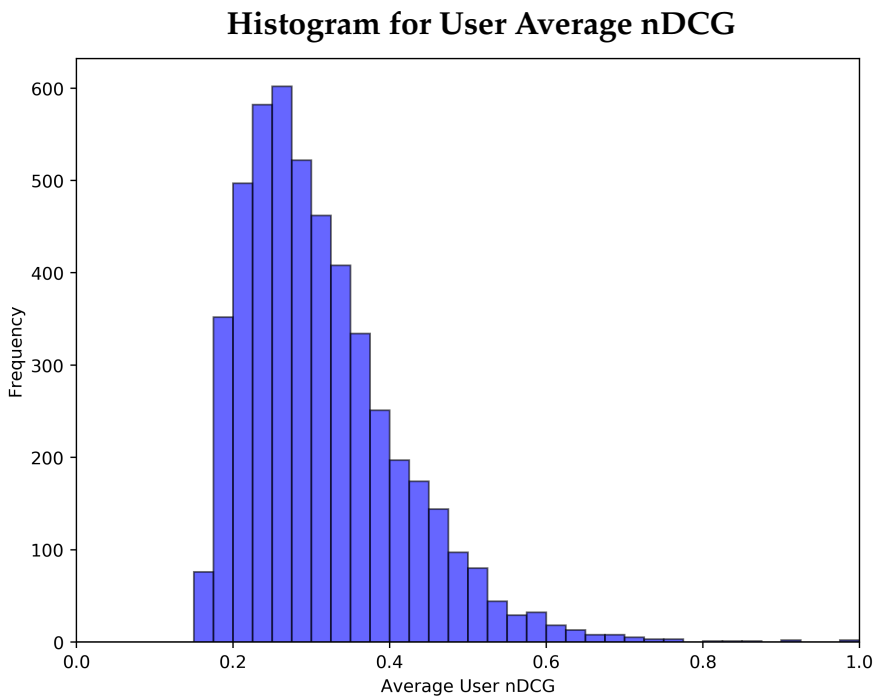


Figure 5.7: Histogram of expected nDCG (using a 200 iteration burn period for sampled linear classifiers) for all users.

is different from the previous value. This is the result of something subtle: the nDCG of more frequent arXiv users is implicitly weighted higher in the second average than in the first.

To confirm our theory, we consider Figure 5.9, which shows a scatterplot comparing the number of user clicks with the `astro-ph` new list with the average nDCG for that particular user. Even with the large cluster of users where information is limited, there is a slight positive trend when the number of user clicks increases. The issue at hand is explaining why this trend is that weak.

There are many dynamics that can explain this. One issue is that most users tend to look no further than the top ten listed academic papers in the "new" list, especially when the results span across multiple pages. This is generally true for many search engines and scenarios in which alternatives are presented

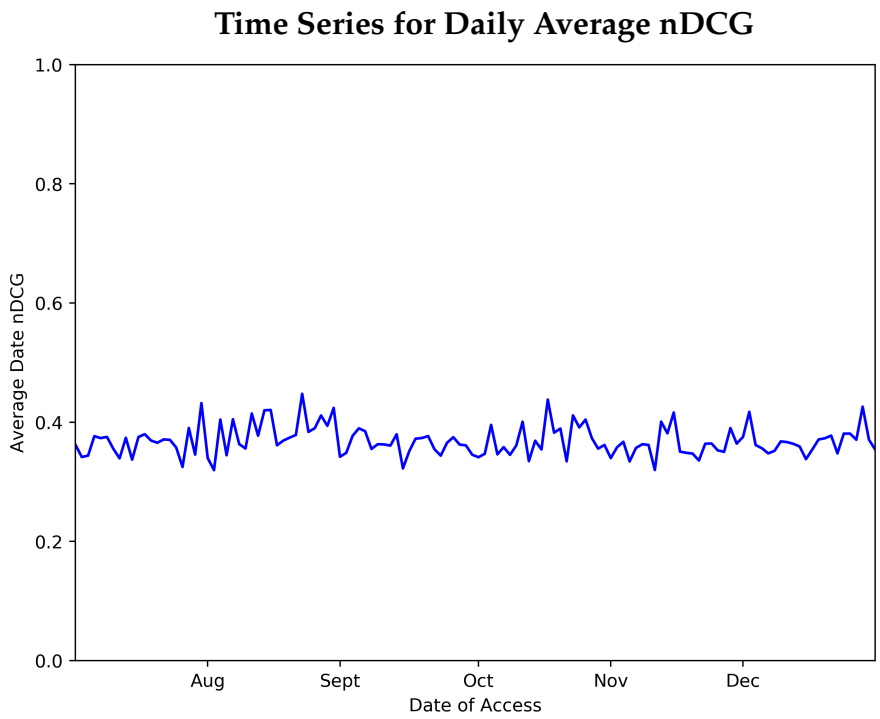


Figure 5.8: Time series of averaged daily nDCG (using a 200 iteration burn period for sampled linear classifiers) by user from July 1 through December 31, 2016.

in a list format. But for the arXiv "new" and "recent" lists, the papers are presented in order of publication date, regardless of user. This implies that papers a user might deem more relevant may appear on the last page of the "new" list, which may prevent users from interacting with relevant alternatives. This effect should be explored in subsequent data analysis, most likely by excluding alternatives that appear too far down the original list.

Any number of modeling choices can affect our ability to learn from previous user interactions. Our choice of vector representation may not be the best choice for the model. Perhaps future investigation should find the vectorization that best captures differences between alternatives, and evaluate these choices with nDCG or different metrics. The parameterization of noise channels

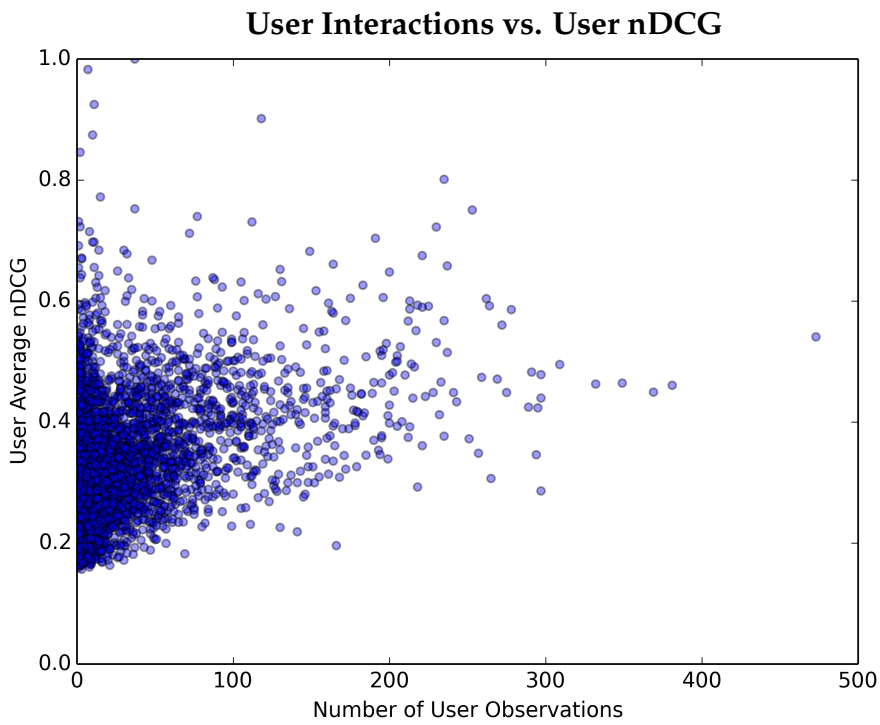


Figure 5.9: Scatterplot comparing the number of user interactions with the "new" `astro-ph` to the expected nDCG of the user.

may not be complex enough to explain how users choose alternatives from a list, and perhaps the order in which the alternatives are presented is important. This is closely related to our method of deriving consideration sets from interactions in lists. For this particular passive experiment, we only assumed that a user selection implied a single comparison between the selected paper and the paper listed immediately before. Again, further numerical studies should consider other methods for constructing consideration sets and evaluate their performance, and ideally should conclude the extent to which preference information can or cannot be learned from a list format.

One of the most significant sacrifices made in this statistical study is giving up the ability to adaptively pose questions to a single user. To a substantial degree, the weakness in explanatory power of the model is the cost of not asking

informative queries. Randomly constructed queries can provide redundant and unhelpful information in the context of previous observations. Again, there is nothing that promotes relevant papers to the top of the list, and therefore, the knowledge gained from a user interaction is not as informative. And then there is the question of policy performance, which we attempt to answer previously in Section 5.4. We concluded that the entropy pursuit policy performs similarly to the knowledge gradient policy in terms of learning efficiency, while being much more computationally feasible. However, the users are the ultimate authorities on the relevance and learning power of a given querying policy, and without the ability to control the queries, we will not be able to conclusively determine the effectiveness of certain policies or the model as a whole.

There is a point at which not being able to pose our own comparative queries to the user affects our ability to learn, and future studies should attempt to find where that point is. Our model shows a moderate amount of promise, but in order to gauge its effectiveness in adaptive online learning, future endeavors in this area should focus on testing an application that measures the benefit of asking Bayes-optimal direct-comparison questions versus observing answers to random ones.

5.6 Conclusion

In this chapter, we develop methodology to efficiently implement the components necessary to use the theory in Chapter 4 in real applications, and study the effectiveness of this model in two different scenarios. In the first scenario, we show that the entropy pursuit policy performed similarly to the knowledge

gradient, even when the computational effort required for the latter is much more massive. The second scenario is much different in nature, in that we give up the chance to adaptively ask questions in exchange for real historical data. In applying the Gibbs sampler used to simultaneously estimate linear classifiers and the noise channel, we see that the sampler itself performs well, but the predictive strength of the generated vectors is moderate at best.

The question of how the selected hyperparameters of the model— the number of offered alternatives m, the structure of the noise channel, the vectorization of alternatives, for instance—affect the quality of the predictions is still highly relevant. But perhaps the most pertinent question remains the exact strength of adaptiveness in the context of this model, and whether or not it can be done in a computationally efficient manner. This begs the need for future research in such problems to combine the scenarios of Section 5.4 5.5 into a single study, developing an application to test these query policies for many different users. We conjecture that there exists a penalized form of entropy pursuit that can deliver both prediction power while remaining computationally feasible for a live user. Nevertheless, the work in this chapter demonstrates that the model developed in the previous chapter is not merely confined to the realm of theory, but rather can be implemented in an efficient fashion.

REFERENCES

[1] Claude JP Bélisle, H Edwin Romeijn, and Robert L Smith. "Hit-and-run algorithms for generating multivariate distributions". In: *Mathematics of Operations Research* 18.2 (1993), pp. 255–266.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent Dirichlet Allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[3] George Casella, Christian P Robert, and Martin T Wells. "Generalized accept-reject sampling schemes". In: *Lecture Notes-Monograph Series* (2004), pp. 342–347.

[4] Bangrui Chen and Peter Frazier. "The Bayesian Linear Information Filtering Problem". In: *arXiv preprint arXiv:1605.09088* (2016).

[5] Yoav Goldberg and Omer Levy. "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method". In: *arXiv preprint arXiv:1402.3722* (2014).

[6] Laura A Granka, Thorsten Joachims, and Geri Gay. "Eye-tracking analysis of user behavior in WWW search". In: *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2004, pp. 478–479.

[7] Thorsten Joachims et al. "Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search". In: *ACM Transactions on Information Systems (TOIS)* 25.2 (2007), p. 7.

[8] Filip Radlinski and Thorsten Joachims. "Active exploration for learning rankings from clickthrough data". In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2007, pp. 570–579.

[9] P Raghavan and H Schütze. *Introduction to information retrieval*. 2008.

[10] Gerard Salton and Michael J McGill. "Introduction to modern information retrieval". In: (1986).