The Thesis Committee for Srilakshmi Pattabiraman
certifies that this is the approved version of the following thesis:

Finding good enough coins under symmetric and asymmetric information

APPROVED BY

SUPERVISING COMMITTEE:

Supervisor: ____________________________
Constantine Caramanis

____
Alexandros G. Dimakis

Finding good enough coins under symmetric and asymmetric information

by

Srilakshmi Pattabiraman

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2017

Acknowledgments

I would like to take this opportunity to express my sincere gratitude to Prof. Constantine Caramanis for his time, and patience. If not for the several insightful discussions over the last two years, my growth would have been bleak. I would also like to thank Prof. Alex Dimakis for his support.

Many thanks to Prof. de Veciana, Prof. Ren, Prof. Caramanis, Prof. Dimakis, Prof. Vikalo, Prof. Hanasusanto, Prof. Zitkovic, Prof. Nikolova, Prof. Price, Prof. Bakolas, and Prof. Walker for I have benefitted immensely from their courses during my time here.

Abstract

Finding good enough coins under symmetric and asymmetric information

Srilakshmi Pattabiraman, MSE
The University of Texas at Austin, 2017

Supervisor: Constantine Caramanis

We study the problem of returning m coins with biases above 0.5. These good enough coins that are returned by the agent should be acceptable to the authority by meeting the authority's Family Wise Error Rate constraint. We design adaptive algorithms that invoke Sequential Probability Ratio Test to find these good enough coins. We consider scenarios that differ in terms of the information available about the underlying Bayesian setting. The symmetry or asymmetry of the underlying setup, *i.e.*, the difference between what the agent and the authority know about the underlying prior and the support, presents different challenges. We also make notes on the algorithms' sample complexity.

Table of Contents

List of Figures

Chapter 1

Introduction

We consider the problem of finding m coins (amongst infinitely many) with biases greater than 0.5 under the constraint that the probability of any coin being biased 0.5 or below is less than some pre-specified value α^*. In particular, we consider a setting wherein a player is tasked with finding k coins with biases greater than 0.5 and getting it approved from a non adversarial, independent party whom we shall refer to as the authority. The authority approves of the set of discovered coins if its Family Wise Error Rate (FWER) is below some pre-specified level α^*. The central idea in this work is an adaptive strategy that uses the Sequential Probability Ratio Test (SPRT) (as described in [1]). We consider three settings wherein the information available to the player and the authority is different, describe our algorithms for these settings and compute the upper bound on the algorithms' sample complexities.

1.1 Related Work

Statistics, Learning Theory and Machine Learning communities house extensive literature on Hypothesis Testing [2, 3, 4], Anomaly Detection [5], Family Wise Error Rate (FWER) (and closely related False Discovery Rate

(FDR))[6, 7, 8] and Multi-arm Bandits Problems [9, 10, 11, 12] . [1, 13] are the seminal works of Wald and Chernoff (respectively) on sequential analysis upon which several works are built (including ours). Pure exploration problems in the multi-arm bandits literature are closely related to our work. Some of the recent works in the well studied Top arm(s) and closely related problems -the problem of finding the arm(s) with the largest (or close to the largest) mean - include [14], [15], [16] and [17]. Thresholding bandits problems studied in [18], [19] require algorithms to return all arms or a finite subset of arms that are above a certain threshold. Most of these works consider a setting wherein the number of arms is finite and sample complexity required to correctly (up to a certain precision) identify the top arm(s) is quantified. Another work that closely relates to the top arm problem is [20] which describes the optimal adaptive strategy for the problem of finding the most biased coin in the fewest flips. However, our setting involves infinite coins (arms), a prior on the biases of the coins, an FWER constraint and a requirement to get the biased coin(s) approved by the authority. The possible asymmetry of the underlying setting is another distinction. Together, they make our setting unique.

1.2 Outline

Chapter 2 describes our problem setting formally. Chapter 3 reviews the closely related pure exploration literature, and also provides supplementary expository notes. In Chapter 4, we outline some key ideas used throughout this text. Chapter 5 considers the setting where both the agent, and the authority know all parameters of the underlying model. Chapter 6 discusses the setting wherein the agent is endowed with complete information of the support and the prior on it, while the authority has no such information. Chapter 7 deals with the setting where both the agent and the authority know the support, but neither of them are aware of the prior. Chapter 8 summarizes the text and discusses the limitations therein.

Chapter 2

Problem Formulation

Consider an infinite set of coins. We consider the problem of returning $m \geq 1$ coins with bias > 0.5 such that their collective FWER is guaranteed to be below α^*, while performing the fewest number of flips. The bias of the coins, $\{p_i\}_{i\in[k]}$ ($[k] = \{1...k\}$), are distributed identically, and independent (i.i.d.) of other coins. The *a priori* probability that the Bernoulli parameter of a coin is p_i is denoted by π_i. We shall assume that $\pi_i > \epsilon_\pi, \forall i \in [k]$, where ϵ_π is some known positive constant.

The number of tosses T performed in expectation by a strategy A is given by $\mathbb{E}[T|A]$, where the expectation is jointly over the prior and any randomness in the strategy A. A incurs a cost

$$\mathrm{R}(A) = M\mathbb{1}_{\mathrm{FWER}(A)>\alpha^*} + 1\mathbb{E}[T|A]$$

where M is an arbitrarily large positive constant. Thus, the objective is to find the minimizer of the following program:

$$\begin{aligned} \underset{A\in\mathcal{A}}{\arg\min} \quad & \mathbb{E}[T|A] \\ \text{subject to} \quad & \mathrm{FWER}(A) <= \alpha^* \end{aligned}$$

where $\mathcal{A}$ is the set of all randomized strategies. Since an arbitrarily large cost is incurred if the FWER constraint is violated, we have the following definition:

Definition 2.0.1. An algorithm A is *admissible* if its FWER is below the pre-specified requirement α^*.

Definition 2.0.2. An algorithm A is δ-weakly *admissible* if it contains FWER below $\alpha^* + \delta$.

Definition 2.0.3. A strategy A^* is *asymptotically optimal* if

$$\lim_{\alpha^* \to 0} \frac{\mathrm{R}(A^*)}{\inf_{A \in \mathcal{A}} \mathrm{R}(A)} = 1.$$

In a similar spirit, we define the following:

Definition 2.0.4. An algorithm A whose expected FWER is below α^* is *unbiased*, i.e., for an *unbiased* algorithm A, $\mathbb{E}[\mathrm{FWER}(A)] \leq \alpha^*$. A is δ-*weakly unbiased* if its expected FWER is below $\alpha^* + \delta$, *i.e.*, $\mathbb{E}[\mathrm{FWER}(A)] \leq \alpha^* + \delta$.

Chapter 3

Background

3.1 Pure Exploration Review

We shall review some related algorithms in the Pure Exploration of Multi Arm Bandits literature. We do not intend this section to be exhaustive; nor do we intend it to be rigorous. Our aim is to convey qualitatively the focus of the pure exploration multi-arm bandit literature, and its distinction thereof from our problem.

Consider the setting wherein a finite number of arms are present. Let A represent this set of arms. Each arm $a \in A$ returns 1 or 0 when pulled with probability μ_a or $1 - \mu_a$ respectively. Identifying the top arm, *i.e.*, the arm $a^* \leftarrow \arg\max_{a \in A} \mu_a$ has been of extreme interest. To that end, we now highlight two algorithms from [15] to illustrate the core ideas used to address such questions, and the nature of guarantees provided therein. [15, Definition 1] is presented here for completeness.

Definition 3.1.1. An algorithm is a (ϵ, δ)-PAC algorithm for the multi armed bandit with sample complexity T, if it outputs an ϵ-optimal arm, a', with probability at least $1 - \delta$, when it terminates, and the number of time steps the algorithm performs until it terminates is bounded by T .

A simple algorithm samples each arm a finite number of times, and outputs the arm with the highest empirical average. Algorithm 1 specifies the number of pulls of each arm. Theorem 1 describes the PAC bound on the algorithm. The theorem is an immediate consequence of Hoeffding's concentration inequality and union bound. Note that this algorithm is non-adaptive, in that every arm is pulled the same number of times irrespective of the promise of its posterior.

Algorithm 1 Naive

```
procedure NAIVE (ϵ > 0, δ > 0)
    Foreach arm a ∈ A
        Sample arm a   (4/ϵ^2) log(2n/δ) times
        p̂_a ← average reward of arm a
    return arg max_{a∈A} {p̂_a}
```

Theorem 1. *The algorithm Naive* (ϵ, δ) *is an* (ϵ, δ)*-PAC algorithm with arm sample complexity* $\mathcal{O}\left(\frac{n}{\epsilon^2}\log\left(\frac{n}{\delta}\right)\right)$

Algorithm 2 improves over the simple Naive algorithm by sampling (in successive rounds) the arms that are promising. In every round, a set of arms are pulled a finite number of times. Arms that are promising, in particular, whose empirical means are above the median, are propagated to the next round. Thus, at every round about half of the current arms are eliminated. Theorem 2 gives the PAC guarantee for the algorithm. Note that Algorithm 2 brings down the sample complexity to $\mathcal{O}\left(\frac{n}{\epsilon^2}\log\left(\frac{1}{\delta}\right)\right)$ from $\mathcal{O}\left(\frac{n}{\epsilon^2}\log\left(\frac{n}{\delta}\right)\right)$.

Algorithm 2 Median Elimination

procedure MEDIAN ELIMINATION $(\epsilon > 0, \delta > 0)$
  $S_1 = A, \epsilon_1 = \frac{\epsilon}{4}, \delta_1 = \frac{\delta}{2}, l = 1$
  while $|S_l| > 1$ **do**
    Sample $a \in S_l$ $l = \frac{4}{\epsilon_l^2} \log\left(\frac{3}{\delta_l}\right)$ time
    $\hat{p}_a^l \leftarrow$ average reward of arm $a \in S_l$
    $m_l \leftarrow$ median of the empirical rewards $\{\hat{p}_a^l\}_{a \in S_l}$
    $S_{l+1} = S_l \setminus \{a : \hat{p}_a < m_l\}$
    $\epsilon_{l+1} \leftarrow \frac{3}{4}\epsilon_l$, $\delta_{l+1} \leftarrow \frac{1}{2}\delta_l$, $l \leftarrow l + 1$
  return $\arg\max_{a \in A}\{\hat{p}_a\}$

Theorem 2. *The algorithm Median Elimination* (ϵ, δ) *is an* (ϵ, δ)*-PAC algorithm with arm sample complexity* $\mathcal{O}\left(\frac{n}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$.

A natural extension of this problem is the *top m arms* problem. As the name suggests, under the setting of the *top arm* problem, we ask for an algorithm that returns the m arms that have the largest empirical means, *i.e.*, we ask the algorithm to return the set S wherein $\{S \subset A : |S| = m, \text{ and } \forall i \in S, \forall j \in (A \backslash S) : \mu_i \geq \mu_j\}$. In [16], three algorithms are designed that parallel the ones designed in [15]. We now briefly discuss these algorithms. Algorithm 3 parallels Algorithm 1, in that every arm is pulled a fixed number of times and the set of arms that have the highest empirical means are returned. Theorem 3 gives the PAC bound for the algorithm. Henceforth, we let T_n represent the set of all arms.

Algorithm 3 Direct

procedure DIRECT (m, n, ϵ, δ)

  Foreach arm $a \in T_n$

    Sample a $l = \frac{2}{\epsilon^2} \log\left(\frac{n}{\delta}\right)$ times.

    $\hat{p}_a \leftarrow$ average reward of arm a

  return $\{S \subset T_n : |S| = k, \text{ and } \forall i \in S, \forall j \in (T_n \backslash S) : \hat{p}_i \geq \hat{p}_j\}$

Theorem 3. *Direct(m, n, ϵ, δ) is (ϵ, m, δ)-optimal with sample complexity* $\mathcal{O}\left(\frac{n}{\epsilon^2} \log\left(\frac{n}{\delta}\right)\right)$.

Algorithm 4 improves over Algorithm 3 by calling the Median Elimination algorithm to find m arms successively. Theorem 4 gives its corresponding PAC guarantee.

Algorithm 4 Incremental

procedure INCREMENTAL (m, n, ϵ, δ)

  $S_1 \leftarrow \{\}, R_1 \leftarrow T_n, l \leftarrow 1$

  while $l \leq m$ **do**

    $a^{'} \leftarrow$ Median Elimination$(R_l, \epsilon, \frac{\delta}{m})$

    $S_{l+1} \leftarrow S_l \cup a^{'}, R_{l+1} \leftarrow R_l - a^{'}, l \leftarrow l + 1$

  return S_{m+1}

Theorem 4. *Incremental(m, n, ϵ, δ) is (ϵ, m, δ)-optimal with sample complexity* $\mathcal{O}\left(\frac{mn}{\epsilon^2} \log\left(\frac{m}{\delta}\right)\right)$.

Note that Algorithm 4's logrithmic dependence is over the number of arms to be returned and not the total number of arms. When $m = o(n)$, this is a significant improvement over Algorithm 3. However, the information about the pulls in the Median Elimination subroutine is only used to determine the best arm in that stage. This information could be used to eliminate less promising arms, thereby reducing sample complexity. Algorithm 5 formalizes this intuition, and Theorem 5 provides its PAC gaurantee.

Algorithm 5 Halving

---

procedure HALVING ($\epsilon > 0, \delta > 0$)
  $R_1 = T_n, \epsilon_1 = \frac{\epsilon}{4}, \delta_1 = \frac{\delta}{2}, l = 1$
  while $l \leq \log\left(\frac{n}{m}\right)$ **do**
    Sample $a \in R_l$ $\frac{2}{\epsilon_l^2} \log\left(\frac{3m}{\delta_l}\right)$ times
      $\hat{p}_a^l \leftarrow$ average reward of arm $a \in R_l$
      $m_l \leftarrow$ median of the empirical rewards $\{\hat{p}_a^l\}_{a \in S_l}$
    Find $R_l^{'}$ such that $|R_l^{'}| = \max\left(\left[\frac{|R_l|}{2}\right], m\right)$, and $\forall i \in R_l, \forall j \in (R_l \backslash R_l^{'}) : \hat{p}_i \geq \hat{p}_j$
    $R_{l+1} = R_l^{'} \backslash \{a : \hat{p}_a < m_l\}$
    $\epsilon_{l+1} \leftarrow \frac{3}{4}\epsilon_l$, $\delta_{l+1} \leftarrow \frac{1}{2}\delta_l$, $l \leftarrow l + 1$
  return $\left[R_{\log\left(\frac{n}{m}\right)+1}\right]$

Theorem 5. *Incremental(m, n, ϵ, δ) is (ϵ, m, δ)-optimal with sample complexity* $\mathcal{O}\left(\frac{mn}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$.

Another interesting pure exploration problem is that of the Thresholding Bandits. Given a set of n arms, the task is to design an algorithm that returns all (or a set of) arms whose means are above a certain threshold τ. Amongst other papers, this problem is also studied in [19]. [19] also provides valuable discussion about the gaps between the best known lower and upper bounds in the fixed confidence and the fixed budget settings. Anytime Parameter-free Thresholding Algorithm 6 is a tight fixed budget algorithm that provides a guarantee on the probability of incorrectly returning a suboptimal arm. The algorithm leverages upon the lower bound. The authors show that the bottle neck of the problem is $\sqrt{T_k(t)}\hat{\Delta}_k(t)$, wherein $\hat{\Delta}_i(s) := \hat{\Delta}_i(s)^{\tau,\epsilon}(s) = |\hat{\mu}_i(s) - \tau| + \epsilon$, with $\hat{\mu}_i(t) = \frac{1}{T_i(t)} \sum_{s=1}^{T_i(t)} X_{i,s}$, and $T_i(t) = \sum_{s=1}^{t} \mathbb{I}_{I_s = i}$. Hence, by forcing $\sqrt{T_k(t)}\hat{\Delta}_k(t)$ to be equal across all arms, the algorithm claims its optimality.

Algorithm 6 Anytime Parameter-free Thresholding

```
procedure NAIVE (τ, ϵ)
    Pull each arm once. t ← K
    while t < T do
        Pull arm I_t = arg min_{k≤K} √(T_k(t)) Δ̂_k(t)
    return Ŝ_τ = {k : μ̂_k(T) ≥ τ}
```

Perhaps the setting that comes closest to ours is the one considered in [20]. The authors discuss a Bayesian setting with infinite coins wherein each coin is either *heavy* (in particular, the bias of the coin is $p+\epsilon$) with probability α, and not heavy (the bias of the coin is $p - \epsilon$) with probability $1 - \alpha$. The task is to design an algorithm that finds a heavy coin in the fewest number of flips. The Algorithm 7 presented therein considers a setting with n coins, and a single absorbing layer. The authors prove optimality of the algorithm using tools from Markov games.

Algorithm 7 Likelihood-Toss

```
procedure LIKELIHOOD-TOSS (α, δ, n)
    L_i ← 1 ∀i ∈ [n]
    while L_i < (1−α)(1−δ)/(αδ) ∀i ∈ [n] do
        Toss coin i* = arg max{L_i : i ∈ [n]}. Break ties arbitrarily.
        b_{i*} ← 1_{Heads}
        L_{i*} ← L_{i*} ((p+ϵ)/(p−ϵ))^{b_{i*}} ((1−p−ϵ)/(1−p+ϵ))^{1−b_{i*}}
    return coin with maximum L_i
```

3.2 Supplementary Reading

We provide some expository material that might aid readability of this text.

3.2.1 Sequential Probability Ratio Test

Consider the problem of distinguishing between two hypotheses $H_0 = p_0$ (null) and $H_1 = p_1$ (alternate). The SPRT designed by [1] provides an elegant solution for the optimal number of samples required to distinguish between the hypotheses. For a given acceptance threshold (α, β) for $(\mathbb{P}_F, \mathbb{P}_M)$ respectively, SPRT designs corresponding rejection and acceptance thresholds $(b, a) = (\log(\frac{1}{\alpha}), \log(\beta))$ for the null hypotheses. The test essentially analyses the behavior of the log-likelihood random walk S_N. If $S_N \geq b$, then the null hypothesis is rejected, and if $S_N \leq a$, then the null is accepted. Thus,

$$N = \inf_n \{n : S_n \leq a \textbf{ or } S_n \geq b\},$$

where $b > 0 > a$, denotes the stopping time of the random walk. Expected number of samples required to accept or reject the null hypotheses is an immediate consequence of Wald's lemma. To that end, we now state the lemma (without proof).

Lemma 1. *Let $\zeta_{i i \in \mathbb{N}}$ be a sequence of independently and identically distributed random variables such that $\mathbb{E}[|\zeta_0|] = \mu, \mu < \infty$, and τ be a stopping time such* *that* $\mathbb{E}[\tau] < \infty$. *Then,*

$$\mathbb{E}\left[\sum_{i=0}^{\tau} \zeta_i\right] = \mathbb{E}[\tau]\mu.$$

The stopping time N for the log-likelihood random walk is finite with probability 1. In particular, the probability that N exceeds any finite positive integer k under the null (alternate) hypotheses, *i.e.*, $\mathbb{P}_0[N > k](\mathbb{P}_1[N > k])$, is bounded above by $\frac{\rho^k}{\sqrt{e^a}} \left(\frac{\rho^k}{\sqrt{e^{-b}}}\right)$, (where $\rho < 1$ is the Bhattacharyya coefficient for the distribution pair (p_0, p_1)) which decays to 0 exponentially fast as $k \to \infty$.

We now quantify the expected stopping time under the null hypothesis, $E_0[N]$.

$$\mathbb{E}_0[S_N] \overset{(a)}{=} \mathbb{E}_0[N]\mathbb{E}_0\left[\log\left(\frac{p_1(Y_i)}{p_0(Y_i)}\right)\right], \tag{3.1}$$

$$\mathbb{E}_0[N] = \frac{\mathbb{E}_0[S_N]}{\mathbb{E}_0\left[\log\left(\frac{p_1(Y_i)}{p_0(Y_i)}\right)\right]}, \tag{3.2}$$

$$\mathbb{E}_0[N] \overset{(b)}{=} \frac{(1-\alpha)a + \alpha b}{-D(p_0||p_1)}. \tag{3.3}$$

(a) follows from Wald's lemma. (b) combines (i) $\mathbb{E}[S_N] = \mathbb{P}_0[S_N \geq b]\mathbb{E}_0[S_N|S_N \geq b] + \mathbb{P}_0[S_N \leq a]\mathbb{E}_0[S_N|S_N \leq a] = \alpha b + (1-\alpha)a$, and (ii) $\mathbb{E}_0\left[\log\left(\frac{p_1(Y_i)}{p_0(Y_i)}\right)\right] = -D(p_0||p_1)$, where $-D(p_0||p_1)$ is the KL divergence between the distributions p_0, p_1.

Similarly,

$$\mathbb{E}_1[N] = \frac{(1-\beta)a + \beta b}{D(p_1||p_0)}. \tag{3.4}$$

3.2.2 Generalized SPRT

We briefly describe the Generalized SPRT algorithm, and present its salient results here. Let $\{\zeta_i\}_{i\in\mathbb{N}}$ be a set of random variables following a density f with respect to a dominant measure μ. Generalized SPRT is an algorithm that distinguishes between two families of hypothesis,

$$H_0 : f \in \{g_\theta : \theta \in \Theta\},$$
$$H_1 : f \in \{h_\gamma : \gamma \in \Gamma\},$$

where g_θ, h_γ are density functions with respect to the dominant measure μ. To avoid singularity issues, the densities are to be mutually absolutely continuous for all θ, γ. For $\{\zeta_i\}_{i\in[n]}$, the generalized likelihood statistic is given by

$$L_n = \frac{\max_{\gamma\in\Gamma} \prod_{k'=1}^{n} f_\gamma(\zeta_{k'})}{\max_{\theta\in\Theta} \prod_{k'=1}^{n} f_\theta(\zeta_{k'})}.$$

For two positive numbers $\exp(-a), \exp(b)$ such that $\exp(-a) < \exp(b)$, stopping time τ is given by

$$\tau = \inf_{n\in\mathbb{N}} \{L_n \geq \exp(b) \textbf{ or } L_n \leq \exp(a)\}.$$

The Generalized SPRT sequentially uses samples until the L_n process stops. (The L_n process *stops* when $L_\tau \in [\exp(b), \infty) \cup (0, \exp(-a)]$.)
Under the technical conditions A1, A2, and A3, Theorem 2.2 in [21] states

$$\sup_{\theta\in\Theta} \log \mathbb{P}_\theta(L_\tau > \exp(b)) \sim -b, \quad \sup_{\gamma\in\Gamma} \log \mathbb{P}_\gamma(L_\tau < \exp(-a)) \sim -a, \quad \text{as } a, b \to \infty.$$

Further, Theorem 2.3 in [21] states that under the setting of Theorem 2.2, the expected stopping time admits the approximation

$$\mathbb{E}_\theta[\tau] \sim \frac{a}{\inf_\gamma \in \Gamma} D_\theta(\theta||\gamma), \quad \mathbb{E}_\gamma[\tau] \sim \frac{b}{\inf_\theta \in \Theta} D_\gamma(\gamma||\theta), \quad \text{as } a, b \to \infty,$$

where $D_\nu(g||h)$ is the KL divergence between the densities g, h under the measure ν.

Chapter 4

Preliminaries

4.1 Notation

In the setting where $k = 2$, *i.e.*, when only two types of coins are present in the urn, let us assume that the two possible biases of the coins (p_0, p_1), and the prior (π_0, π_1) over them are known. In particular, we assume $p_0 = 0.5$, and $p_1 > 0.5$. Following terminology from the statistical detection theory literature, we call the p_0 coin *null*, and the p_1 coin *alternate*. $X_1^{T_j}$ denotes the set of random variables $\{X_1, X_2, \ldots, X_{T_j}\}$, wherein each X_i takes a value in $\{0, 1\}$ according to the outcome of the i^{th} toss. For an algorithm A, $A(X_1^{T_j})$ denotes that A uses $X_1^{T_j}$ to decide upon a p_j coin. As evident, T_j is the stopping time for A under p_j. H_j denotes the hypothesis that a coin's bias is p_j. We denote by $\mathbb{P}_F = \mathbb{P}_0(A(X_1^{T_0}) = H_1)$ the probability of false positive (probability of declaring a null coin as an alternate), and by $\mathbb{P}_M = \mathbb{P}_1(A(X_1^{T_1}) = H_0)$ the probability of missed detection (probability of declaring an alternate as a null). The partial sum of the log-likelihood random walk up to *time* N is given by S_N, *i.e.*, $S_N = \sum_{i=1}^{N} \log\left(\frac{p_1(X_i)}{p_0(X_i)}\right)$.

4.2 Finding one p_1 coin

In order to find one p_1 coin, we run SPRT test sequentially until we declare a coin as an alternate. The expected sample size for this procedure is illustrated in Figure 4.1, and is calculated below.

$$\begin{aligned} S = \quad & \pi_0[(1-\alpha)(\mathbb{E}_0[N|S_N \leq a] + S) + \alpha\mathbb{E}_0[N|S_N \geq b]] \quad + \\ & \pi_1[\beta(\mathbb{E}_1[N|S_N \leq a] + S) + (1-\beta)\mathbb{E}_1[N|S_N \geq b]] \\ = \quad & \frac{\pi_0\mathbb{E}_0[N] + \pi_1\mathbb{E}_1[N]}{1-\pi_0(1-\alpha)-\pi_1\beta} \end{aligned}$$

where $\mathbb{E}_0[N], \mathbb{E}_1[N]$ are as in (3.3), (3.4) respectively. Hence:

$$S = \frac{\pi_0\frac{(1-\alpha)a+\alpha b}{-D(p_0||p_1)} + \pi_1\frac{(1-\beta)b+\beta a}{D(p_1||p_0)}}{1-\pi_0(1-\alpha)-\pi_1\beta}$$

The Bayesian setting implies that every coin must be tested with the same (α, β). Every coin being examined is essentially put through a hypothesis test that is a function of its flips alone since the distribution of the biases are independent. In addition, since the biases are distributed identically, different (α, β) pairs offer no advantage; in other words, assignment of different (α, β) pairs to coins would be of help only if the biases were not iid.

Let us suppose that the M^{th} coin is the first to be declared biased and all

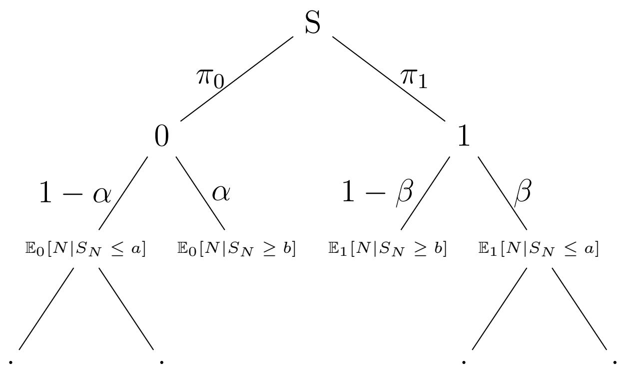

Figure 4.1: Sample complexity of sequential-SPRT for $k = 2$: To calculate S, we note that the process is memoryless every time it picks a coin. If a p_0 coin is picked, with probability α is it declared significant and the process terminates, and with probability $1-\alpha$ the coin is declared a null and another coin is picked. Similarly, when a p_1 coin is picked, with probability β the coin is declared a null and the process continues, and with probability $1-\beta$ the coin is declared significant and the process terminates. p_0, p_1 coins are picked with probability π_0, π_1 respectively.

preceding $M-1$ coins were declared unbiased.

$$\begin{aligned}
&\mathbb{P}(\text{coin M was unbiased}|\text{coin M was declared biased})\\
&\overset{(a)}{=} \frac{\mathbb{P}(\text{coin M was unbiased, declared biased})}{\mathbb{P}(\text{coin M was declared biased})}\\
&\overset{(b)}{=} \frac{\mathbb{P}(\text{coin M was unbiased})\mathbb{P}(\text{coin M was declared biased}|\text{M was unbiased})}{\mathbb{P}(\text{coin M was declared biased})}\\
&\overset{(c)}{\leq} \frac{\pi_0 \rho^{M-1}\alpha}{\rho^{M-1}(1-\rho)}\\
&= \alpha\frac{\pi_0}{\pi_0\alpha + (1-\beta)\pi_1} \quad \overset{(d)}{=} \alpha^*
\end{aligned}$$

$(a), (b)$ are Bayes' rule, in (c), ρ is the probability of a coin not being declared biased $\rho = \mathbb{E}[1_{\text{not declared bias}}] = \mathbb{E}[\mathbb{E}[1_{\text{not declared bias}}]|\text{bias of the coin}] = (1-\alpha)\pi_0 + \beta\pi_1$, and (d) shall follow from the definitions of α and β. The minimizers of the following program achieve the minimum sample complexity. The objective of the program is the expected number of tosses required to prove a coin, and (4.2) is the FWER constraint.

$$\underset{\alpha,\beta\in[0,0.5)}{\text{minimize}} \quad \frac{\pi_0 \frac{(1-\alpha)a+\alpha b}{-D(p_0||p_1)} + \pi_1 \frac{(1-\beta)b+\beta a}{D(p_1||p_0)}}{1-\pi_0(1-\alpha)-\pi_1\beta} \tag{4.1}$$

$$\text{subject to} \quad \alpha\frac{\pi_0}{\pi_0\alpha+(1-\beta)\pi_1} \leq \alpha^* \tag{4.2}$$

$$b = \log(\frac{1}{\alpha}), \quad a = \log(\beta) \tag{4.3}$$

The unconstrained objective is minimized at the right boundary. Thus (4.2) is an active constraint that governs selection of the parameters α, β. Small values of α^* forces α to be small. Thus constraint (4.2) is approximately $\alpha\frac{\pi_0}{(1-\beta)\pi_1} \leq \alpha^*$, which implies $\alpha = \alpha^*\frac{(1-\beta)\pi_1}{\pi_0}$ is a consistent assignment. In particular, the optimal α is within a constant factor of $\alpha^*\frac{\pi_1}{\pi_0}$. Solving (4.1) with this assignment would give an optimal solution. If, in addition to α, β were also small, then the LHS of constraint (4.2) $\sim \alpha\frac{\pi_0}{\pi_1}$. Thus, $\alpha = \alpha^*\frac{\pi_1}{\pi_0}$ and $\beta = \alpha^*$ is a consistent assignment. Since the $\mathbb{E}_0[N]$ and $\mathbb{E}_1[N]$ are optimal, so is the objective $\frac{\pi_0\mathbb{E}_0[N]+\pi_1\mathbb{E}_1[N]}{1-\pi_0(1-\mathbb{P}_F)-\pi_1\mathbb{P}_M}$.

Further, we wish to draw attention to the fact that although the problem explicitly constraints probability of false positives only, by demanding a p_1 coin, the probability of miss is also implicitly constrained. In other words, β is not a free variable. Small values of α pushes the algorithm towards not declaring coins alternates. However, the objective of the algorithm is to find a significant coin with the least number of flips, and this asks for some significant power, thereby constraining β. This is the reason why β is constrained to take values in $[0, 0.5)$, and not $[0, 1]$.

Chapter 5

Symmetric, Complete Information

Let us look at the scenario wherein both the agent and the authority are endowed with complete knowledge of the support and the prior over it. In this section we design the "(m)-successive" SPRT algorithm to find m alternates that collectively meet the FWER constraint.

Let H_0 denote the null hypothesis class and H_1, the alternate.

$$H_0 : p \in (0, 0.5]$$

$$H_1 : p \in (0.5, 1)$$

where p is the bias of the coin. Before we present the general algorithm, we illustrate it with $k = 3$ and $p_2 > p_1 > p_0 = 0.5$, and their respective priors π_2, π_1 and π_0. Thus, p_0 coin is our null and p_1, p_2 are the alternates. For $m = 1$, the task is to design an algorithm that, in expectation, requires minimum number of samples to declare a coin significant, and contains $\mathbb{P}_F$ at level α^*. It is easy to see that if we happen to pick a p_2 coin, the least sample complexity in expectation corresponds to that given by SPRT for the p_2 coin. However, if we pick a p_1 coin, it is optimal to commit to proving the coin only if, in expectation, the tosses required to prove this coin is less than the tosses required to *find* and *prove* a p_2 coin. This idea lies at the core of the following algorithm, and it also emphasizes why the problem does not fit readily in the setting of binary hypothesis testing.

5.1 Algorithm for Illustration

Let $L_n^{ij} = \frac{\prod_{k'=1}^{n} p_i(Y_{k'})}{\prod_{k'=1}^{n} p_j(Y_{k'})}$ and $S_n^{ij} = \log[\frac{\prod_{k'=1}^{n} p_i(Y_{k'})}{\prod_{k'=1}^{n} p_j(Y_{k'})}]$ denote the likelihood ratio and the log-likelihood ratio of the observations (with respect to hypotheses that the coins are p_i, p_j) respectively. Algorithm 8 illustrates the optimal procedure to find a biased coin.

Algorithm 8

```
1: procedure SEARCH(p_2, p_1, π_2, π_1, α*, x ≤ 0, k, m = 1)
2:     a ← log(β*)    b ← log(2Cπ_0 k / (α* π'_0))    ▷ β*, C are solutions of generalized program (4.1)
3:     while j < m do
4:         Pick a coin and keep flipping until S_i^10 < a or S_i^20 > b or S_i^10 > b or
           {S_i^12 < x and (b − S_i^10)/D(p_1||p_0) > b/D(p_2||p_0) + (π_1/π_2) S_i^21/(−D(p_1||p_2))}}    ▷ i is the stopping time
5:         if S_i^20 < a and S_i^10 < a then
6:             Pick another coin continue
7:         if S_i^20 > b or S_i^10 > b then
8:             j ← j + 1    continue    ▷ Alternate
9:         if S_i^21 < x and
10:            (b − S_i^10)/D(p_1||p_0) > b/D(p_2||p_0) + (π_1/π_2) S_i^21/(−D(p_1||p_2)) + (π_0/π_2) a/(−D(p_1||p_0))    then
11:            Pick another coin
12:    return Alternates
```

Line 4 stopping condition: $S_i^{10} < a$ **or** $S_i^{20} > b$ **or** $S_i^{10} > b$ **or** $\{S_i^{12} < x$ **and** $\frac{b - S_i^{10}}{D(p_1||p_0)} > \frac{b}{D(p_2||p_0)} + \frac{\pi_1}{\pi_2}\frac{S_i^{21}}{(-D(p_1||p_2))}\}\}$

Line 10 condition: $\frac{b - S_i^{10}}{D(p_1||p_0)} > \frac{b}{D(p_2||p_0)} + \frac{\pi_1}{\pi_2}\frac{S_i^{21}}{(-D(p_1||p_2))} + \frac{\pi_0}{\pi_2}\frac{a}{-D(p_1||p_0)}$

Note that in line 10 of Algorithm 8, $S_i^{21} < 0$. The question of whether or not to pick another coin and start tossing that instead arises only if the coin at hand seems to be a p_1 coin. If we decide to pick another coin when

S_i^{21} crosses some level y from above, then, because of the Bayesian setup of the problem, we must reject every subsequent (seemingly) p_1 coin at the same level y. Thus, S, the number of tosses required to search for and prove a p_2 coin when the partial sums of the walks update exactly by their drift μ, is given by:

$$S = \frac{b}{D(p_2||p_1)} + \frac{\pi_1}{\pi_2}\frac{x}{-D(p_1||p_2)} + \frac{\pi_0}{\pi_2}\frac{a}{-D(p_1||p_0)} \tag{5.1}$$

The LHS of line 10 in Algorithm 8 is the projected tosses required in expectation to prove the (seemingly) p_1 coin in hand and the RHS is S for the present level S^{21}.

We remark that, devoid of other absorbing levels, all good particles are absorbed at barrier $b > 0$ under a positive drift $\mu > 0$ with probability 1. The claim is an immediate consequence of Martingale Convergence Theorem which is stated below.

Theorem 6. *Let* $\{\zeta_i\}_{i\in\mathbb{N}}$ *be a martingale such that* $\sup_{i\in\mathbb{N}} \mathbb{E}[|\zeta_i|] < \infty$. *Then, there exists a random variable* ζ *such that* $\zeta_i \to \zeta$ *almost surely.*

We now proceed to prove the claim that all good particles are absorbed at b almost surely under a positive drift. Let Y_i represent $\log\left(\frac{p_1(Y_i)}{p_0(Y_i)}\right)$. Thus,

$$\mathbb{E}_1[Y_i] = D(p_1||p_0).$$

We note that $\sup_{i\in\mathbb{N}} \mathbb{E}[|Y_i|] = \mathbb{E}[|Y_i|] < \infty$. Let $Z_i = Y_i - D(p_1||p_0)$. Martingale convergence theorem states that $\sum_{i=0}^{n-1} Z_i$ converges. Also,

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=0}^{n-1} Z_i\right] = \frac{1}{n}\left[n\mathbb{E}[Y_i] - nD(p_1||p_1)\right] = 0.$$

Strong law of large numbers states that $\sum_{i=0}^{n-1} Y_i - nD(p_1||p_0) \to 0$ almost surely. Therefore $\sum_{i=0}^{n-1} Y_i$ crosses any positive level from below almost surely. This proves our claim.

If one were to pick a p_2 coin and toss until declaration (*i.e.*, assume that only one absorbing barrier is present at b, and wait till the particle gets absorbed), the expected number of tosses is $\frac{b}{D(p_2||p_0)}$. Introducing other absorbing barriers decreases this quantity as it terminates the walks of all particles that cross a from above around a[1]. This is true for the other two terms as well. Hence, Equation (5.1) is an upper bound on the number of tosses required to search and prove a coin.

When a p_0 coin is tossed a finite number of times, there exists a positive probability for the maximum likelihood estimator's outcome to be p_1 or p_2 (and likewise for other coins as well). Accounting for all possibilities[2], we arrive at the following expression for the sample complexity S, which is also illustrated in the Figure 5.1.

[1]More precisely, the random walks are terminated at $a + o(1)$.

[2]*Rejecting* a coin at a certain level x induces a probability of miss and ν and η capture them.

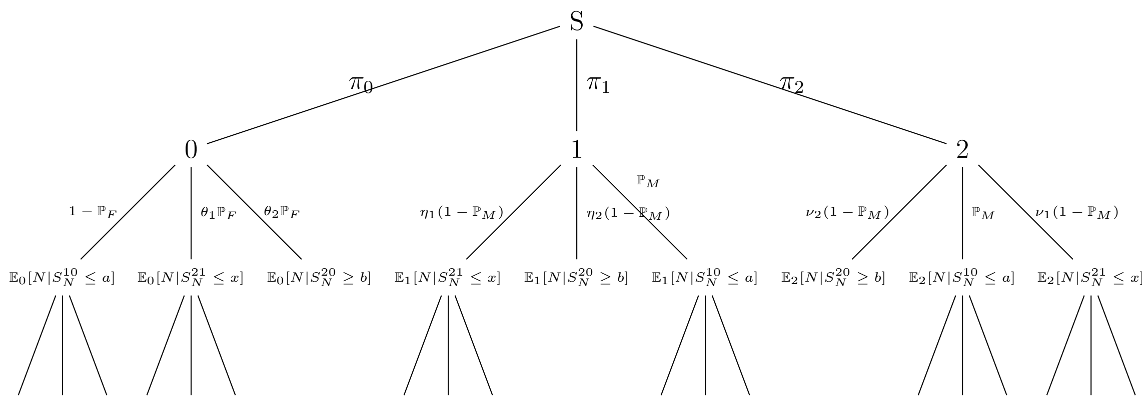

Figure 5.1: Sample complexity of sequential-SPRT for $k = 3$: As in the $k = 2$ case, to calculate S, we note that the process is memoryless every time it picks a coin. If a p_0 coin is picked, with probability $\theta_2 \mathbb{P}_F$ is it declared an alternate, bagged, and the process terminates; with probability $1 - \mathbb{P}_F$ the coin is declared a null and another coin is picked, and with probability $\theta_1 \mathbb{P}_F$, the coin is thought to be a p_1 coin, and without declaring it either way we proceed to pick another coin. Similarly, for a p_1 or a p_2 coin picked, with some probability the coin is declared a null, or thought to be a p_1 coin, and the process continues; and with some probability, the coin is declared as an alternate and the process terminates. p_0, p_1, p_2 coins are picked with probability π_0, π_1, π_2 respectively.

$$\begin{aligned}
S = &\pi_0[(1-\mathbb{P}_F)(\mathbb{E}_0[N|S_N^{10} \leq a] + S) + \mathbb{P}_F\theta_1(\mathbb{E}_0[N|S_N^{21} \leq x] + S) + \mathbb{P}_F\theta_2\mathbb{E}_0[N|S_N^{20} \geq b]] \\
&+ \pi_1[\mathbb{P}_M(\mathbb{E}_1[N|S_N^{10} \leq a] + S) + (1-\mathbb{P}_M)\eta_1(\mathbb{E}_1[N|S_N^{21} \leq x] + S) + (1-\mathbb{P}_M)\eta_2\mathbb{E}_1[N|S_N^{20} \geq b]] \\
&+ \pi_2[\mathbb{P}_M(\mathbb{E}_2[N|S_N^{10} \leq a] + S) + (1-\mathbb{P}_M)\nu_1(\mathbb{E}_2[N|S_N^{21} \leq x] + S) + (1-\mathbb{P}_M)\nu_2\mathbb{E}_2[N|S_N^{20} \geq b]]
\end{aligned}$$

$$S = \frac{\pi_0\mathbb{E}_0[N] + \pi_1\mathbb{E}_1[N] + \pi_2\mathbb{E}_2[N]}{\mathrm{Dr}} \tag{5.2}$$

$$\mathrm{Dr} = 1 - \pi_0(1-\mathbb{P}_F+\theta_1\mathbb{P}_F) - \pi_1(\mathbb{P}_M+\eta_1(1-\mathbb{P}_M)) - \pi_2(\mathbb{P}_M+(1-\mathbb{P}_M)\nu_1) \tag{5.3}$$

where

$$\mathbb{E}_0[N] = (1-\mathbb{P}_F)\frac{a}{-D(p_0||p_1)} + \theta_1\mathbb{P}_F\frac{\mathbb{E}_0[S_N^{10}|S_N^{21} \leq x]}{-D(p_0||p_1)} + \theta_2\mathbb{P}_F\frac{\mathbb{E}_0[S_N^{10}|S_N^{20} \geq b]}{-D(p_0||p_1)} \tag{5.4}$$

$$\mathbb{E}_1[N] = \eta_1(1-\mathbb{P}_M)\frac{x}{-D(p_1||p_2)} + \eta_2(1-\mathbb{P}_M)\frac{\mathbb{E}_1[S_N^{21}|S_N^{20} \geq b]}{-D(p_1||p_2)} + \mathbb{P}_M\frac{\mathbb{E}_1[S_N^{21}|S_N^{10} \leq a]}{-D(p_1||p_2)} \tag{5.5}$$

$$\mathbb{E}_2[N] = \nu_2(1-\mathbb{P}_F)\frac{b}{D(p_2||p_0)} + \mathbb{P}_M\frac{\mathbb{E}_2[S_N^{20}|S_N^{10} \leq a]}{D(p_2||p_0)} + \nu_1(1-\mathbb{P}_M)\frac{\mathbb{E}_2[S_N^{20}|S_N^{21} \leq x]}{D(p_2||p_0)} \tag{5.6}$$

Lemma 2. *Let S^* denote the optimal sample complexity, and $S_{(5.1)}$ denote the sample complexity of Algorithm 8 as given by* (5.1)*. The excess number of expected of tosses performed by algorithm 8, $|S^* - S_{(5.1)}|$, is upper bound by a function that decreases with decreasing values of parameters $\beta^*, \log(1/|x|)$ used in SPRT.*

Proof. We begin by noting that (5.1) is only an upper bound on the projected number of tosses. Our aim is to show that this upper bound is not too far away for small values of the parameters. From (5.4) we know

$$
\begin{aligned}
&\frac{a}{-D(p_0||p_1)} - \mathbb{E}_0[N] \\
&= \frac{a}{-D(p_0||p_1)} - \left((1-\mathbb{P}_F)\frac{a}{-D(p_0||p_1)} + \theta_1\mathbb{P}_F\frac{\mathbb{E}_0[S_N^{10}|S_N^{21} \leq x]}{-D(p_0||p_1)} + \theta_2\mathbb{P}_F\frac{\mathbb{E}_0[S_N^{10}|S_N^{20} \geq b]}{-D(p_0||p_1)}\right) \\
&= \theta_1\mathbb{P}_F\frac{a - \mathbb{E}_0[S_N^{10}|S_N^{21} \leq x]}{-D(p_0||p_1)} + \theta_2\mathbb{P}_F\frac{a - \mathbb{E}_0[S_N^{10}|S_N^{20} \geq b]}{-D(p_0||p_1)} \\
&\overset{(a)}{\leq} \mathbb{P}_F\frac{a-b}{(-D(p_0||p_1))} \\
&\leq 2\mathbb{P}_F\frac{\max\{|a|, b\}}{D(p_0||p_1)}.
\end{aligned} \tag{5.7}
$$

where (a) follows from two observations - $a \leq S_n^{10} \leq b$, and $\theta_1 + \theta_2 = 1$. Similarly, from (5.5), we infer

$$\frac{x}{-D(p_1||p_2)} - \mathbb{E}_1[N] \leq (\eta_2 + \mathbb{P}_M\eta_1)\frac{x}{-D(p_1||p_2)}. \tag{5.8}$$

Also, assuming that $\exists\lambda > 0$ for all trajectories such that if $S_N^{10} \approx a$ then S_N^{20} is at most λa, then

$$\frac{b}{D(p_2||p_0)} - \mathbb{E}_0[N] \leq (\nu_1 + \nu_2\mathbb{P}_M)\frac{b - \lambda a}{D(p_2||p_0)}. \tag{5.9}$$

Together, (5.7), (5.8), and (5.8) show that for small values of the parameters, the projected number of tosses is also small. □

We make some further remarks about the sample complexity for $m = 1$. We now quantify the upper bound on number of tosses required to declare a coin. In expectation, every p_1 coin is rejected the first time it is tossed or tossed until proven.[3] In that, we draw attention to the fact that for any fixed

[3]Analysis is done for $x = 0$. When $x > 0$ we obtain an additive factor of $\frac{\pi_1}{\pi_2}\frac{x}{-D(p_2||p_1)}$ instead of $\frac{\pi_1}{\pi_2}$.

p_0, p_2, π there exists p_1^* such that $\forall p_1 < p_1^*$, it is optimal to search and prove a p_2 coin, and $\forall p_1 \geq p_1^*$, it is optimal to flip the coin till it crosses b from below. In the first case, *i.e.*, when flipping the p_1 coin is sub-optimal, $\mathbb{E}[S_1^{21}] < 0$. Thus the coin, in expectation, is rejected after the first toss. Otherwise, in expectation, it is tossed until S_N^{10} crosses b from below. The sample complexity is computed separately for the two cases:

- Case 1: Every p_1 coin is rejected after the first toss.

  Let $S^{'}$ be the tosses required to prove one coin.

$$S^{'} = \pi_0 \left[\frac{a}{-D(p_0||p_1)} + S^{'} \right] + \pi_1 \left[\frac{-D(p_2||p_1)}{-D(p_2||p_1)} + S^{'} \right] + \pi_2 \frac{b}{D(p_2||p_0)}$$

$$S^{'} = \frac{\pi_0}{\pi_2} \frac{a}{-D(p_0||p_1)} + \frac{\pi_1}{\pi_2} + \frac{b}{D(p_2||p_0)}$$

- Case 2: Every p_1 coin is proved.

$$S^{'} = \pi_0 \left[\frac{a}{-D(p_0||p_1)} + S^{'} \right] + \pi_1 \left[\frac{b}{D(p_1||p_0)} \right] + \pi_2 \frac{b}{D(p_2||p_0)}$$

$$S^{'} = \frac{\pi_0}{1-\pi_0} \frac{a}{-D(p_0||p_1)} + \frac{\pi_1}{1-\pi_0} \frac{b}{D(p_1||p_0)} + \frac{\pi_2}{1-\pi_0} \frac{b}{D(p_2||p_0)}$$

Lemma 3. *When $m > 1$, Algorithm 8 performs at most $C_1 m \log(m)$ tosses more than the optimal,*
where $C_1 \in \left\{ \frac{1}{D(p_2||p_0)}, \frac{1}{1-\pi_0} \left[\frac{\pi_1}{D(p_1||p_0)} + \frac{\pi_2}{D(p_2||p_0)} \right] \right\}$.

Proof. Let us first look at the natural lower bound for the problem:

$$\begin{aligned}\text{FWER} &= \mathbb{P}(\text{at least one of the coins returned is biased})\\ &\geq \mathbb{P}(\text{coin i is biased}|\text{i was returned}) \quad (\forall i \in [k])\\ &= \max_i \mathbb{P}(\text{coin i is biased}|\text{i was returned})\end{aligned}$$

$\mathbb{P}_F$ of every coin can *at best* be α^* as any other α^* would violate the FWER constraint. Since the biases are i.i.d., and the setting is Bayesian, the procedure that performs the least number of flips to declare m coins significant is SPRT performed m times successively. We shall call this the (m) successive-SPRT. Consider the case where the p_1 coin is rejected after its first toss. The asymptotic expected number of tosses to declare m coins, $\mathbb{E}(N_m^\alpha)$ is:

$$\mathbb{E}(N_m^\alpha) \geq \left[\frac{\pi_0}{\pi_2}\frac{\log(\beta)}{-D(p_0||p_1)} + \frac{\pi_1}{\pi_2} + \frac{\log(\frac{2}{\alpha})}{D(p_2||p_0)}\right] m$$

The asymptotic upper bound on the expected number of tosses for declaring m coins, $\mathbb{E}(N_m^{\alpha/m})$, using the above algorithm:

$$\mathbb{E}(N_m^{\alpha/m}) \leq \left[\frac{\pi_0}{\pi_2}\frac{\log(\beta)}{-D(p_0||p_1)} + \frac{\pi_1}{\pi_2} + \frac{\log(\frac{2m}{\alpha})}{D(p_2||p_0)}\right] m$$

□

Lemma 4. *Algorithm 8 is admissible, i.e., FWER(A)* $\leq \alpha^*$.

Proof. Let Algorithm 8's probability of declaring a null coin as biased be denoted by $\mathbb{P}_F^A$. Let $\exp(b) = B$

$$\begin{aligned}
\mathbb{P}_0\{L_N^{i0} \geq B\} &= \sum_{n=1}^{\infty} \mathbb{P}_0(\{N = n\} \cap \{L_N^{i0} \geq B\}) \\
&= \sum_{n=1}^{\infty} \int_{\{N=n\}\cap\{L_N^{i0}\geq B\}} \prod_{k=1}^{n} p_0(y_k) \quad d\mu(y_1)...d\mu(y_n) \\
&= \sum_{n=1}^{\infty} \int_{\{N=n\}\cap\{L_N^{i0}\geq B\}} \frac{\prod_{k=1}^{n} p_i(y_k)}{L_N^{i0}} \quad d\mu(y_1)...d\mu(y_n) \\
&\leq \frac{1}{B} \sum_{n=1}^{\infty} \int_{\{N=n\}\cap\{L_N^{i0}\geq B\}} \prod_{k=1}^{n} p_i(y_k) \quad d\mu(y_1)...d\mu(y_n) \\
&\leq \frac{1}{B} \\
&= \frac{\alpha}{2}
\end{aligned}$$

And by union bound, we have:

$$\mathbb{P}_F^A \leq \mathbb{P}_0\{L_N^{10} \geq B\} + \mathbb{P}_0\{L_N^{20} \geq B\} \leq \alpha$$

Let us suppose coins indexed $\{M_i\}_{i\in[k]}$ were declared biased and all others (with indices $< M_k$, $\notin \{M_i\}_{i\in[k-1]}$) were declared unbiased.

$$\begin{aligned}
&\mathbb{P}(\exists \text{ an unbiased coin in } \{M_i\}_{i\in[m]} | \{M_i\}_{i\in[m]} \text{were declared biased}) \\
&\overset{(a)}{\leq} \sum_{i=1}^{m} \mathbb{P}(M_i \text{ was unbiased coin} | \{M_i\}_{i\in[m]} \text{were declared biased}) \\
&\overset{(b)}{=} \sum_{i=1}^{m} \frac{\pi_0 \rho^{M_m - m}(1-\rho)^{m-1}\alpha}{\rho^{M_m - m}(1-\rho)^m} \quad \overset{(c)}{=} \alpha^*
\end{aligned}$$

(a) is by union bound; in (b), the variables represent the same quantities as mentioned earlier and (c) follows from definitions. $\square$

Having conveyed the core ideas, we now extend the algorithm to a more general setting. We assume a prior density f_{prior} over the biases of the coins defined over $(0, 0.5] \cup [0.5+\epsilon, 1)$. We utilize Generalized Sequential Probability Ratio Test (Generalized SPRT) as described in [21], and show that its optimality translates to our setting as well. We now tailor Generalized SPRT to our problem; henceforth, we let L_n denote the generalized likelihood ratio

$$L_n = \frac{\max_{\gamma \in \Gamma} \prod_{k'=1}^{n} p_\gamma(y_{k'})}{\max_{\delta \in \Delta} \prod_{k'=1}^{n} p_\delta(y_k^{'})}$$

wherein Γ corresponds to the "alternate interval" $[0.5+\epsilon, 1]$ and Δ corresponds to the "null interval" $[0, 0.5]$.

5.2 Complete Algorithm

A recursion tree, similar to the one illustrated previously, provides us an expression for sample complexity that accounts for low probability events. However, asymptotically, as given by line 16 in Algorithm 9 suffices.

Lemma 5. *For small values of α^*, Algorithm 9 is admissible, i.e.,*

$$\mathbb{P}[\exists \textit{ an unbiased coin in } \{M_i\}_{i\in[k]} | \{M_i\}_{i\in[k]} \textit{ were declared biased }] \leq \alpha^*, \ \textit{ as } \alpha^* \to 0.$$

Proof. Let $\exp(b) = B$. An upper bound for the probability of false positive under coin with bias δ^*, $\mathbb{P}_{\delta^*}\{L_N \geq B\}$ is obtained through steps similar to the

[5] ζ can be thought of as a risk-aversion factor. In the case of discrete support, ζ is the gap between two corresponding alphabets.

[5] 13: When the support is discrete, the distribution is not altered as ζ contains no mass.

Algorithm 9

procedure SEARCH($f_{prior}, \alpha^*, \zeta, m = 1$)[4]

  $b \leftarrow \log(\frac{C\pi_0 m}{\pi_0' \alpha^*})$, $a \leftarrow \log(\beta^*)$ ▷ $\pi_0 = \int_\Delta f_{prior} d\mu(\delta)$

  while $j < m$ **do**

    Pick a coin and keep flipping until $S_i < a$ **or** $S_i > b$ **or** $\{S_i > 0$ **and** easier to declare a coin with larger bias$\}$

    if $S_i < a$ **then**

      Declare null **continue**

    if $S_i > b$ **then**

      $k \leftarrow k + 1$ **continue** ▷ Alternate

    if $S_i > 0$ **then**

      $\hat{\theta} \leftarrow \max_{\gamma \in \Gamma} \prod_{k=1}^{i} p_\gamma(y_k)$

      $\Theta \leftarrow [0.5 + \epsilon, \hat{\theta}]$

      $\Gamma \leftarrow [\hat{\theta} + \zeta, 1]$

      Redistribute ζ mass on the Θ interval uniformly [5]

      $L_i^{\hat{\theta}+\zeta,\hat{\theta}} \leftarrow \frac{\max_{\gamma \in \Gamma} \prod_{k=1}^{i} p_\gamma(y_k)}{\max_{\theta \in \Theta} \prod_{k=1}^{i} p_\theta(y_k)}$

      $S_i^{\hat{\theta}+\zeta,\hat{\theta}} \leftarrow \log(L_i^{\hat{\theta}+\zeta,\hat{\theta}})$

      if $\frac{b - S_i}{D(\hat{\theta}||0.5 - \frac{\epsilon}{2})} > \mathbb{E}_\gamma[\frac{b}{D(\gamma||0.5 - \frac{\epsilon}{2})}] + \frac{\pi_1}{\pi_2}\mathbb{E}_\theta[\frac{S_i^{\hat{\theta}+\zeta,\hat{\theta}}}{-D(\theta||\hat{\theta}+\zeta)}] + \frac{\pi_0}{\pi_2}\mathbb{E}_\delta[\frac{a}{-D(\delta||0.5 + \frac{\epsilon}{2})}]$ **then**

        continue

▷ $\pi_1 = \int_\Theta f_{prior} d\mu(\theta)$; $\pi_2 = \int_\Gamma f_{prior} d\mu(\gamma)$

  return Alternates

ones outlined in the proof of Lemma 4.

$$\begin{aligned}
\mathbb{P}_{\delta^*}\{L_N \geq B\} &= \sum_{n=1}^{\infty} \mathbb{P}_{\delta^*}(\{N=n\} \cap \{L_N \geq B\}) \\
&= \sum_{n=1}^{\infty} \int_{\{N=n\}\cap\{L_N \geq B\}} \prod_{k=1}^{n} p_{\delta^*}(y_k) \quad d\mu(y_1)...d\mu(y_n) \\
&\leq \sum_{n=1}^{\infty} \int_{\{N=n\}\cap\{L_N \geq B\}} \frac{\prod_{k=1}^{n} p_{\gamma^*_{mle}}(y_k)}{[\frac{\prod_{k=1}^{n} p_{\gamma^*_{mle}}(y_k)}{\prod_{k=1}^{n} p_{\delta^*_{mle}}(y_k)}]} \quad d\mu(y_1)...d\mu(y_n) \\
&\leq \frac{1}{B}
\end{aligned}$$

$$\begin{aligned}
\int_{\Delta} f_{prior} \mathbb{P}_{\delta^*}\{L_N \geq B\} d\mu(\delta) &\leq \frac{1}{B} \int_{\Delta} f_{prior} d\mu(\delta) \\
&= \frac{\pi_0}{B} \\
&\leq \alpha
\end{aligned}$$

$$\begin{aligned}
&\mathbb{P}(\exists \text{ an unbiased coin in } \{M_i\}_{i\in[m]} | \{M_i\}_{i\in[m]} \text{were declared biased}) \\
&\overset{(a)}{\leq} \sum_{i=1}^{m} \mathbb{P}(M_i \text{ was unbiased coin} | \{M_i\}_{i\in[m]} \text{were declared biased}) \\
&\overset{(b)}{=} \sum_{i=1}^{m} \frac{\pi_0 \rho^{M_m - m}(1-\rho)^{m-1}\alpha}{\rho^{M_m - m}(1-\rho)^m} \\
&= \frac{m\pi_0\alpha}{1-\rho} \\
&\overset{(c)}{=} \alpha^*
\end{aligned}$$

Since $\mathbb{E}[1_{L_N \geq B} | \Delta] = \int_{\Delta} \frac{f_{prior}}{\pi_0} \mathbb{P}_{\delta^*}\{L_N \geq B\} d\mu(\delta)$, the claim follows. □

Chapter 6

Asymmetric Information

The first player is now endowed with complete information, and the authority has no *a priori* knowledge about the prior. Since we can employ Algorithm 8 and benefit from its properties if only the authority were convinced of the mass on the null, our approach is to find an optimal strategy with which we could convince the authority of our null-alternate prior (π_0, π_0^c) ahead of the coin testing/proving phase. Let's suppose that the authority would accept the null-alternate prior (π_0, π_0^c) if the first player could verify it at least up to an l_1 distance of δ. Furthermore, let's suppose that the authority would accept the hypothesis that there is no mass on an interval $(\tilde{\theta}, \tilde{\gamma})$ if it were to be shown that at least $1 - \vartheta$ mass is on $(\tilde{\theta}, \tilde{\gamma})^c$. In order to verify the null-alternate prior, we could design a binary hypothesis testing problem with the null interval $[0, \bar{\theta}]$ and the alternate interval $[\underline{\gamma}, 1]$, where $\bar{\theta}, \underline{\gamma}$ are the end points of the null and the alternate intervals that the second player is aware of. Let $\mathbb{P}_M = \mathbb{P}_F = c_1$. The average number of tosses required to classify a coin is $\sum_{\theta \in \Theta} \pi_\theta \frac{-\log(c_1)}{|\mathbb{E}_\theta \log \frac{p_{\tilde{\theta}}}{p_\theta}|} + \sum_{\gamma \in \Gamma} \pi_\gamma \frac{-\log(c_1)}{|\mathbb{E}_\gamma \log \frac{p_{\tilde{\gamma}}}{p_\gamma}|}$. Perhaps, we could do better by assuming a shorter null interval $[0, p]$ and a shorter alternate $[q, 1]$ interval, while ensuring that all coins in $(p, \bar{\theta})$ get classified as null coins, and all coins in $(\underline{\gamma}, q)$ get classified as alternates. If the gap between the null and the alternate

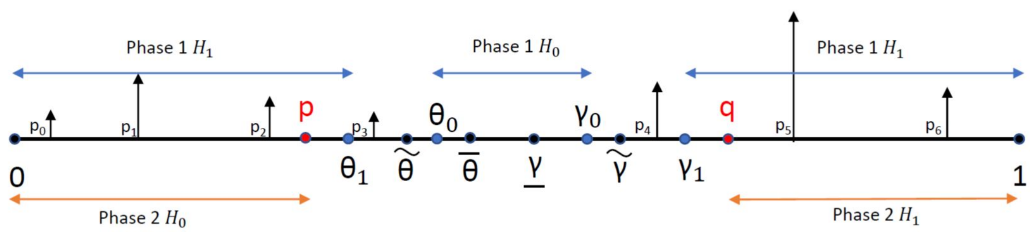

Figure 6.1: Null and Alternate intervals for Phase 1 and Phase 2

intervals were to be much larger than the known $(\bar{\gamma}, \underline{\delta})$, it could be beneficial to verify (a portion) the gap before proceeding to find the optimal p, q.

We design a three-phase algorithm that accounts for the tradeoffs between proving the largest gap and designing the fastest test in expectation. Phase 1 verifies that there is no mass on $(\tilde{\theta}, \tilde{\gamma})$ interval. Phase 2 verifies the null-alternate prior (π_0, π_0^c). Phase 1 and Phase 2 invoke Algorithm 10 to verify their respective hypothesis. Their null and alternate intervals is illustrated in figure 6.1. Phase 3 calls Algorithm 9 in order to find and return the biased coins. The parameters required for the algorithms are the optimizers of the following program.

$$\underset{n_1, c_1, n_2, c_2, \delta}{\text{minimize}} \quad n_1 g_1(c_1) + n_2 g_2(c_2) + N(\boldsymbol{\pi}, \boldsymbol{p}) \tag{6.1}$$

$$\text{subject to} \quad \sqrt{\frac{2}{\pi n_1}} + \frac{4\sqrt{2}}{n_1^{\frac{3}{4}}} + 2c_1 \leq \vartheta \tag{6.2}$$

$$\sqrt{\frac{2}{\pi n_2}} + \frac{4\sqrt{2}}{n_2^{\frac{3}{4}}} + 2c_2 \leq \delta \tag{6.3}$$

$n_1 g_1(c_1)$ denotes the number of tosses (on average) required to verify that no coin is in the gap $(\tilde{\theta}, \tilde{\gamma})$.

$$g_1(c_1) = \sum_{\theta \in \Theta} \pi_\theta \frac{-\log c_1}{|\mathbb{E}_\theta \log \frac{p_{\theta_0}}{p_\theta}|} + \sum_{\gamma \in \Gamma} \pi_\gamma \frac{-\log c_1}{|\mathbb{E}_\gamma \log \frac{p_{\gamma_0}}{p_\gamma}|}$$

where γ_0, θ_0 are the maximizers of the following program:

$$\text{maximize} \quad \sum_{\theta \in \Theta} \pi_\theta \frac{1}{|\mathbb{E}_\theta \log \frac{p_{\tilde{\theta}}}{p_\theta}|} + \sum_{\gamma \in \Gamma} \pi_\gamma \frac{1}{|\mathbb{E}_\gamma \log \frac{p_{\tilde{\gamma}}}{p_\gamma}|} - \left[\sum_{\theta \in \Theta} \pi_\theta \frac{1}{|\mathbb{E}_\theta \log \frac{p_{\theta_0}}{p_\theta}|} + \sum_{\gamma \in \Gamma} \pi_\gamma \frac{1}{|\mathbb{E}_\gamma \log \frac{p_{\gamma_0}}{p_\gamma}|} \right] \tag{6.4}$$

$$\begin{aligned}
\text{subject to} \quad & \epsilon \leq \theta_1 \leq \theta_0 \leq \bar{\theta} \\
& 1 - \epsilon \geq \gamma_1 \geq \gamma_0 \geq \underline{\gamma} \\
& \forall i \in (\epsilon, \tilde{\theta}) \quad \mathbb{E}_i \log \frac{p_{\theta_0}}{p_{i \wedge \theta_1}} \leq 0 \\
& \exists \theta' \in (\theta_0, \gamma_0) : \forall i \in (\tilde{\theta}, \theta') \quad \mathbb{E}_i \log \frac{p_{i \vee \theta_0}}{p_{\theta_1}} \geq 0 \\
& \forall i \in (\theta', \tilde{\gamma}) \quad \mathbb{E}_i \log \frac{p_{i \wedge \gamma_0}}{p_{\gamma_1}} \geq 0 \\
& \forall i \in (\tilde{\gamma}, 1 - \epsilon) \quad \mathbb{E}_i \log \frac{p_{\gamma_0}}{p_{i \vee \gamma_1}} \leq 0
\end{aligned}$$

$g_2(c_2)$ denotes the average number of tosses performed to update the prior once.

$$g_2(c_2) = \sum_{\theta \in \Theta} \pi_\theta \frac{-\log c_2}{|\mathbb{E}_\theta \log \frac{p_q}{p_\theta}|} + \sum_{\gamma \in \Gamma} \pi_\gamma \frac{-\log c_2}{|\mathbb{E}_\gamma \log \frac{p_\gamma}{p_p}|}$$

where p, q are the maximizers of the following program:

$$\underset{p,q}{\text{maximize}} \quad \sum_{\theta\in\Theta} \pi_\theta \frac{1}{|\mathbb{E}_\theta \log \frac{p_\gamma}{p_\theta}|} + \sum_{\gamma\in\Gamma} \pi_\gamma \frac{1}{|\mathbb{E}_\gamma \log \frac{p_\gamma}{p_{\bar\theta}}|} - \left[\sum_{\theta\in\Theta} \pi_\theta \frac{1}{|\mathbb{E}_\theta \log \frac{p_q}{p_\theta}|} + \sum_{\gamma\in\Gamma} \pi_\gamma \frac{1}{|\mathbb{E}_\gamma \log \frac{p_\gamma}{p_p}|} \right] \tag{6.5}$$

$$\text{subject to} \quad \epsilon \le p \le \tilde\theta$$

$$1 - \epsilon \ge q \ge \tilde\gamma$$

$$\forall i \in (p, \tilde\theta): \quad \mathbb{E}_i \log \frac{p_q}{p_p} \le 0$$

$$\forall j \in (\tilde\gamma, q): \quad \mathbb{E}_j \log \frac{p_q}{p_p} \ge 0$$

Algorithm 10 Asymmetric Information

```
procedure π₀Estimation (Θ, Γ, p, q, n)
    j ← 0                                              ▷ coins are indexed by j
    π̃ ← 0   T₀ ← 0   T₁ ← 0
    while j ≤ n do
        Pick a coin and flip till {S_i ≤ log c or S_i ≥ −log c}
            where S_i = log( max_{γ∈Γ} ∏_{l=1}^{i} p_γ(X_l) / max_{θ∈Θ} ∏_{l=1}^{i} p_θ(X_l) )
        j ← j + 1
        if S_i ≥ −log c then
            Declare H₀    T₀ ← T₀ + 1
        if S_i ≤ log c then
            Declare H₁    T₁ ← T₁ + 1
    return(π̃ ∝ (T₀, T₁))
```

Note that the total number of tosses required in expectation is the sum of the expected number of tosses required in each phase. After we verify the null-alternate prior, the number of tosses required in expectation is the number of tosses that Algorithm 9 would require. This quantity $N(\boldsymbol{\pi}, \boldsymbol{p})$ does not depend on the optimizing parameters, and is included in the objective for clarity. Thus, (6.1) is the quantity that we want to minimize. We have assumed that the authority would accept the hypothesis that there is no mass on an interval $(\tilde{\theta}, \tilde{\gamma})$ if it were to be shown that atleast $1 - \vartheta$ mass is on $(\tilde{\theta}, \tilde{\gamma})^c$. In particular, to us it means that the "prior" on the gap that we infer from Algorithm 10 in Phase 1 can be at most ϑ. In other words, the l_1 separation between the prior $\boldsymbol{\pi}$ and the prior learned through Algorithm 10 $\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}$, $l_1(\tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}) < \vartheta$. Consider the problem of learning the prior when sampling symbols, *i.e.*, every sample is a support alphabet as against 0 or 1. Let $\tilde{\boldsymbol{\pi}}$ denote the prior that can be learnt under this "perfect" alphabet model. Using triangle inequality, we know,

$$\mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}, \boldsymbol{\pi})] \leq \mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}, \tilde{\boldsymbol{\pi}})] + \mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}, \boldsymbol{\pi})].$$

We use the result of [22, Lemma 7], to substitute for $\mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}, \boldsymbol{\pi})]$, and worst case reasoning to substitute for $\mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}, \tilde{\boldsymbol{\pi}})]$. Thus, we arrive at (6.2). Similarly, we have assumed that the authority would accept the null-alternate prior (π_0, π_0^c) if the player could verify it at least up to an l_1 distance of δ. We arrive at constraint (6.3) through the same line of reasoning as used to obtain constraint (6.2). Algorithm 10 invoked twice in succession, first to verify the gap and next to verify the mass on the null, followed by Algorithm 8 9 is admissible contingent on verification of the prior. Since the prior is verified in expectation, this procedure is unbiased.

Lemma 6. *For $\vartheta, \delta > 0$, and $m = 1$, Algorithm 10 invoked twice in succession, first to verify the gap and next to verify the mass on the null, followed by*

Algorithm 8 spends the minimum number of tosses in expectation (over the excess of Algorithm 8) amongst all unbiased algorithms that perform hypothesis testing.

Proof. Note that minimizers of Program 6.1 are independent of the FWER requirements α^*. $\delta, \vartheta > 0$ result in finite $n_1 g_1(c_1) + n_1 g_1(c_1)$. Thus, for small values of α^*, $N(\boldsymbol{\pi}, \boldsymbol{p})$ dominates, and hence the claim. □

Lemma 7. *For $\vartheta, \delta > 0$, and when $m > 1$, Algorithm 10 invoked twice in succession, first to verify the gap and next to verify the mass on the null, followed by Algorithm 8 spends at most $C_3 m \log(m)$ tosses (over the excess of Algorithm 8) more than the optimal asymptotically in expectation.*

Finally, we note that instead of verifying the null-alternate prior, we could verify an upper bound on the null, and prove the coins with respect to a higher threshold. This could be beneficial in some situations. Our program can be modified to accommodate for this flexibility as well.

Chapter 7

Symmetric, Incomplete Information

Since we do not know the prior, we first learn the prior using Algorithm 11. Let c denote the maximum misclassification probability while learning the prior, i.e., $\mathbb{P}_i(\text{coin clasified as j}) \leq c \quad \forall i \neq j$.

Algorithm 11 uses the deterministic DGFi policy in [23], and for completeness we briefly describe the problem, its setting, and the policy here.

Consider K cells, M of which are informative, *i.e.*, *signals*, and the rest noise. When a cell i is probed at time j, a random variable ζ_i^j is generated according to distribution f_i if the signal is present, otherwise ζ_i^j is generated according to the distribution g_i. It is assumed that the (f_i, g_i) pairs are known for all cells. The paper addresses the problem of finding a policy that is optimal with respect to a Bayes' risk as defined in [23]. The DFGi policy as described in [23] is an asymptotically optimal policy with respect to the Bayes' risk. For $M = 1$, the policy asks to probe cells that rapidly increase the difference between the largest log-likelihood ratio (LLR), and the second largest LLR. When this gap, $\Delta S = \{\text{largest LLR} - \text{second largest LLR}\} \geq -\log(c)$, the process is terminated, and the cell whose LLR is the largest is declared as the one with signal. It is proved that the probability of incorrectly declaring a noisy cell is at most c.

Algorithm 11 uses the deterministic DGFi policy in [23] with $K = 3$, S_{i0} as the signals g_i's and deterministic 0 process as the alternate f_i for all i. We, therefore, wait for the most likely walk (S_{20}, S_{10} or 0) to be $-\log(c)$ away from the second most likely walk before classifying the coin and updating the prior. (Of course, we are not concerned about the probing feature, since we are compute S_{i0} for all i.) Let $g(c)$ denote the average number of tosses performed to update the prior once in Algorithm 11. Assume (w.l.o.g) $p_i < p_{i+1} \quad \forall i, i+1 \in [k]$.

$$g(c) = \sum_{i \in [k]} \pi_i \frac{c}{D(p_i)}$$

$$\text{where } D(p_i) = \begin{cases} D(p_0||p_1), & \text{if } i = 0, \\ D(p_k||p_{k-1}), & \text{if } i = k, \\ \min\{D(p_i||p_{i-1}), D(p_i||p_{i+1})\}, & \text{otherwise.} \end{cases}$$

Algorithm 11 Subroutine: Learning a Noisy Prior

procedure L.SEARCH(p_2, p_1, p_0, c, n)
  $j \leftarrow 0$ ▷ coins are indexed by j
  $\tilde{\boldsymbol{\pi}} \leftarrow \mathbf{0}$
  Pick a coin $j \leftarrow 1$
  while $j \leq n$ **do**
    Keep flipping the coins till $\{S_i^{21} \leq \log c, S_i^{10} \geq 0\}$ **or** $\{S_i^{21} \geq -\log c, S_i^{20} \geq 0\}$ **or** $\{S_i^{10} \leq \log c\}$
    if $S_i^{10} < \log c$ **then**
      $\tilde{\pi}_0 \leftarrow \tilde{\pi}_0 + 1$ $j \leftarrow j + 1$ **continue**
    if $S_i^{20} - S_i^{10} > -\log c, S_i^{20} \geq 0$ **then**
      $\tilde{\pi}_2 \leftarrow \tilde{\pi}_2 + 1$ $j \leftarrow j + 1$ **continue**
    if $S_i^{20} - S_i^{10} < -\log c, S_i^{10} \geq 0$ **then**
      $\tilde{\pi}_1 \leftarrow \tilde{\pi}_1 + 1$ $j \leftarrow j + 1$ **continue**
  $\hat{\boldsymbol{\pi}} \propto \tilde{\boldsymbol{\pi}}$ ▷ $\hat{\pi}$ is a noisy empirical frequency vector
  return $\hat{\boldsymbol{\pi}}$

Algorithm 12 Min-Max setting

procedure L.SEARCH($p_2, p_1, p_0, \alpha^*, \hat{\boldsymbol{\pi}}, n$)
  $n \leftarrow 0$ ▷ n is the number of coins declared alternates
  $a \leftarrow \log(\beta^*)$ $b \leftarrow \log(\frac{2C_4\hat{\pi}_0 m}{\alpha^* \hat{\pi}_0{}'})$
  Pick a coin
  while $n < m$ **do**
    Invoke algorithm 8 to find a biased coin
    $n \leftarrow n + 1$

Let $f(\delta)$ denote the number of tosses required to complete the task with $\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}$ that is utmost δ away from $\boldsymbol{\pi}$ in l_1. Let us assume we do know the prior.

If we were to choose a very small c, we might spend a lot of flips learning the prior, if we choose a large c, we risk learning a prior too far from the ground truth. We should also explore enough (n) coins such that the central limit theorem kicks in, all while trying to minimize the number of flips spent so far and $f(\delta)$. Program (7.1) is designed to handle the said trade-offs.

$$\underset{n,c,\delta}{\text{minimize}} \quad ng(c) + f(\delta) \tag{7.1}$$

$$\text{subject to} \quad \alpha^* \left(\frac{1 + \frac{\delta}{2\pi_0}}{1 - \frac{\delta}{2\pi_1}} \right) \leq \bar{\alpha^*} + \alpha^* \tag{7.2}$$

$$\sqrt{\frac{2(k-1)}{\pi n}} + \frac{4k^{\frac{1}{2}}(k-1)^{\frac{1}{4}}}{n^{\frac{3}{4}}} + (k-1)c \leq \delta \tag{7.3}$$

If the priors were accurate, then, we could claim the following:

Lemma 8. *Algorithm 12 is $\bar{\alpha^*}$-weakly unbiased, i.e., with arguments of program* (7.1) *as parameters and $m = 1$, Algorithm 12 contains the expected FWER below $\bar{\alpha^*} + \alpha^*$. Also, Algorithm 12 performs optimal number of tosses in expectation (over the excess of Algorithm 8).*

Proof. To verify the claim, note the following:

- The expected FWER when the l_1 distance from the prior is at most δ, is $\alpha^* \frac{1+\delta/(2\pi_1)}{1-\delta/(2\pi_0)}$. This accounts for (7.2). Thus, Algorithm 12 contains the worst case FWER below $\bar{\alpha^*}$.

- Algorithm 11 learns a prior $\tilde{\boldsymbol{\pi}}$. It's probability of misclassifying an i coin as a j coin,i.e. $\mathbb{P}_i(S_{j0} - S_{l0} \geq -\log(c)) \leq (k-1)c$ (where $l \neq j, i$). Hence:

$$\pi_i(1-(k-1)c) \leq \mathbb{P}(\tilde{H}_i) \leq \pi_i(1-c) + c$$

The worst case l_1 separation between the prior $\boldsymbol{\pi}$ and the learnable $\tilde{\boldsymbol{\pi}}$ - $l_1(\tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}) < (k-1)c$. Using triangle inequality, followed by [22, Lemma 7]:

$$\mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}, \boldsymbol{\pi})] \leq \mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}_{\boldsymbol{l}}, \tilde{\boldsymbol{\pi}})] + \mathbb{E}[l_1(\tilde{\boldsymbol{\pi}}, \boldsymbol{\pi})]$$

This accounts for (7.3).

□

Lemma 9. *Under the conditions of the previous lemma, with $m > 1$, the Algorithm 12 performs additional $C_4 m \log m$ tosses in expectation (over the excess of Algorithm 8).*

Lemma 8 and Lemma 9 were based on the assumption that we knew the prior ($g(c)$ is defined in terms of it). Since the prior is unknown, we turn the two stage algorithm into a three stage algorithm with the first stage being a burn-in stage to learn the prior. We first fit an uniform distribution over the alphabet, i.e., assume alphabet in the support has equal probability mass. We now solve program (7.1). Using c thus obtained, we run Algorithm 11 till we learn the prior up to a sufficient l_1 distance.

Using the learnt prior, we then solve (7.1) again. We now use the parameters obtained to crank Algorithm 12. Note that the number of tosses performed in the burn-in stage is some number bounded from above. For any specific c, n we can find an upper bound on the expected l_1 distance between the true prior and that learnt. Thus, Lemma (8) and lemma (9) would still hold with different constants.

Chapter 8

Conclusion

The problem of finding good enough coins was studied. Algorithms were proposed so as to enable the agent to return good coins acceptable to the authority under symmetrical and asymmetrical information settings.

In our text, we assume the knowledge of the underlying support. Designing algorithms without this assumption is a natural future direction. Another direction of interest is along finding tighter lower bounds under generalized settings.

Bibliography

[1] A. Wald. Sequential tests of statistical hypotheses. *Ann. Math. Statist.*, 16(2):117–186, 06 1945.

[2] Sirin Nitinawarat, George K Atia, and Venugopal V Veeravalli. Controlled sensing for multihypothesis testing. *IEEE Transactions on Automatic Control*, 58(10):2451–2464, 2013.

[3] Mohammad Naghshvar and Tara Javidi. Active m-ary sequential hypothesis testing. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 1623–1627. IEEE, 2010.

[4] Mohammad Naghshvar, Tara Javidi, et al. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703–2738, 2013.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[6] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the royal statistical society. Series B (Methodological)*, pages 289–300, 1995.

[7] Dean P Foster and Robert A Stine. α-investing: a procedure for sequential control of expected false discoveries. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(2):429–444, 2008.

[8] Adel Javanmard and Andrea Montanari. On online control of false discovery rate. *arXiv preprint arXiv:1502.06197*, 2015.

[9] Chun-Hung Chen, Donghai He, Michael Fu, and Loo Hay Lee. Efficient simulation budget allocation for selecting an optimal subset. *INFORMS Journal on Computing*, 20(4):579–595, 2008.

[10] Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *COLT-23th Conference on Learning Theory-2010*, pages 13–p, 2010.

[11] Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648, 2004.

[12] Lloyd W Koenig and Averill M Law. A procedure for selecting a subset of size m containing the l best of k independent normal populations, with applications to simulation. *Communications in Statistics-Simulation and Computation*, 14(3):719–734, 1985.

[13] Herman Chernoff. *Sequential Analysis and Optimal Design.*

[14] Jean-Yves Audibert and Sébastien Bubeck. Best Arm Identification in Multi-Armed Bandits. In *COLT - 23th Conference on Learning Theory - 2010*, page 13 p., Haifa, Israel, June 2010.

[15] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *J. Mach. Learn. Res.*, 7:1079–1105, December 2006.

[16] Shivaram Kalyanakrishnan and Peter Stone. Efficient selection of multiple bandit arms: Theory and practice. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 511–518, USA, 2010. Omnipress.

[17] Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pages 227–234, USA, 2012. Omnipress.

[18] Subhojyoti Mukherjee, K. P. Naveen, Nandan Sudarsanam, and Balaraman Ravindran. Thresholding bandits with augmented UCB. *CoRR*, abs/1704.02281, 2017.

[19] Andrea Locatelli, Maurilio Gutzeit, and Alexandra Carpentier. An optimal algorithm for the thresholding bandit problem. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1690–1698. JMLR.org, 2016.

[20] Karthekeyan Chandrasekaran and Richard M. Karp. Finding the most biased coin with fewest flips. *CoRR*, abs/1202.3639, 2012.

[21] Xiaoou Li, Jingchen Liu, and Zhiliang Ying. Generalized sequential probability ratio test for separate families of hypotheses. *Sequential analysis*, 33(4):539–563, 2014.

[22] Sudeep Kamath, Alon Orlitsky, Dheeraj Pichapati, and Ananda Theertha Suresh. On learning distributions from their samples. In *COLT*, pages 1066–1100, 2015.

[23] Boshuang Huang, Kobi Cohen, and Qing Zhao. Sequential active detection of anomalies in heterogeneous processes. *arXiv preprint arXiv:1704.00766*, 2017.

[24] Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648, 2004.