Optimizations for Election Tabulation Auditing

by

Mayuri Sridhar

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2019

Author .....
Department of Electrical Engineering and Computer Science
February 2019

Certified by .....
Ronald L. Rivest
MIT Institute Professor
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chairman, Department Committee on Graduate Theses

Optimizations for Election Tabulation Auditing

by

Mayuri Sridhar

Submitted to the Department of Electrical Engineering and Computer Science
on February 2019, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this thesis, we explore different techniques to improve the field of election tabulation audits. In particular, we start by discussing the open problems in statistical election tabulation audits and categorizing these problems into three main sections – audit correctness, flexibility, and efficiency.

In our first project, we argue that Bayesian audits provide a more flexible framework for a variety of elections than RLAs. Thus, we initially focus on analyzing their statistical soundness. Furthermore, we design and implement optimization techniques for Bayesian audits which show an increase in efficiency on synthetic election data.

Then, motivated by empirical feedback from audit teams, we focus on workload estimation for RLAs. That is, we note that audit teams often want to finish the audit in a single round even if it requires sampling a few additional ballots. Hence, for the second project, we design software tools which can make initial sample size recommendations with this in mind.

For our largest project, we focus on approximate sampling. That is, we argue that approximate sampling would provide an increase in efficiency for RLAs and suggest a particular sampling scheme, k-cut. We explore the usability of k-cut by providing and analyzing empirical data on single cuts. We argue that for large k, the model will converge to the uniform distribution exponentially quickly. We discuss simple mitigation procedures to make any statistical procedure work with approximate sampling and provide guidance on how to choose k. We also discuss usage of k-cut in practice, from pilot audit experiences in Indiana and Michigan, which showed that k-cut led to a significant real-life increase in efficiency.

Thesis Supervisor: Ronald L. Rivest
Title: MIT Institute Professor

Acknowledgments

Thank you to Prof. Ronald Rivest for supervising my work over the past year. Between weekly meetings in-person and constantly answering my questions (both research-related and not) over email, he has been an incredibly supportive mentor. He was always willing to spend an hour at a whiteboard discussing the details of a proof or discussing how to write graduate school applications. But, most of all, thank you to Prof. Rivest for teaching me how exciting research can be. This year has been busy and overwhelming at times, but he taught me how to enjoy the entire experience. With his help, it has also been one of my best years yet.

When I was starting my M. Eng. in January, I did not know anything about the field of election security or the people involved. Over the past year, I have learned that the people in the field are passionate about democracy and really welcoming. Their help has been crucial to getting my research used in practice and I cannot thank them enough. In particular, thank you to Profs. Jay Bagga and Bryan Byers from BSU for helping me pilot k-cut for the first time in Marion County, Indiana in May 2018. Thank you as well to the Marion County Clerk's Office for their trials and feedback. Thank you to Liz Howard from the Brennan Center of Justice and the county clerks in Rochester Hills, Lansing, and Kalamazoo for helping us pilot k-cut in Michigan in December 2018. Thank you to Miguel Nunez from the Rhode Island Board of Elections for helping us pilot k-cut in Rhode Island in January 2019. Their support gave me the opportunity to see k-cut used as part of an auditing procedure and measure real-life increases in efficiency which was incredibly satisfying.

I would also like to thank the software and audit teams for all the audits I've attended. In particular, thank you to John McCarthy and Mark Lindeman from Verified Voting, John Marion from Common Cause, Jennifer Morrell from Democracy Works, and Zara Perumal. Every person on this list has taken time from his or her busy schedule to answer millions of my questions, patiently and repeatedly. Thank you.

Thank you to my friends: Sharmeen Sayed Dafedar, Asmita Jana, Nora Kelsall,

Alan Samboy, Barbara Zinn, and Amin Manna, to name a few. Thank you for reading my papers, thank you for bringing me coffee when I was working on my proofs, thank you for making me laugh when I was stressed out. This thesis would not have been completed without you all.

Thank you to my family. My mom, dad, and sister have supported my work in every way. My dad has always read every version of any paper that I have written, including all the versions of this thesis. My mom has spent so much time listening to me talk about the problems that I run into, giving me advice, and repeatedly assuring me that everything would work out. My sister has helped me on many levels, from formatting my equations properly to making me laugh when I'm stuck and having a bad day. Thank you all so much.

There are so many other people who have helped me over the past year, without whom this work would not have been completed - any list that I try to make has been hopelessly incomplete. So, for all those who I haven't named so far, thank you so much for your support throughout the past year. Thank you to everyone who welcomed me into the field with open arms. Thank you for answering my questions and for giving me feedback. Thank you for listening to my ideas and helping me refine them to the point of usability. But, most of all, thank you for spending your time helping me. It's been a wonderful year.

Lastly, I would like to thank the Center for Science of Information (CSoI), an NSF Science and Technology Center, for supporting this work under grant agreement CCF-0939370.

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Overview of Election Tabulation Auditing

1.1 Introduction

The correctness of elections is a hot topic today since elections are the foundation of our democracy. Many citizens (from voters to election officials) are worried about their local, state, and national election results being accurate and free of interference, whether that interference is due to their voting machines being hacked or errors in the ballot handling process. In particular, we would like to be able to verify that a contest's results are accurate quickly and with minimum labor. To aid this effort, several states are moving towards the use of paper ballots to guarantee a voter-verified paper trail for the election. That is, the ground truth results of the election are based on a physical piece of paper that the voter has seen and verified; the voting machine has no chance of editing the paper when the contest results are tabulated. Thus, even if the voting machines are hacked, we can still look at the paper to find the actual results.

In addition to paper ballots, many states have process-oriented auditing procedures in place. For instance, states like Michigan have complex chain-of-custody procedures from the tabulator to the ballot storage containers to guarantee that no ballots are misplaced. In addition, some state laws include recount margins; if the

margin in a race is extremely close, then all the ballots in the race are recounted to guarantee that the outcome is correct.

Some states are also moving towards requiring post-election tabulation audit procedures to show that the results of an election were correct. Previously, standard auditing techniques included examining a fixed percentage of ballots for a particular election, voting machine, or town. For instance, we might decide to look at 1% of the votes in a specific county in New York. In this technique, humans look at a single ballot and compare their interpretation of it to what the machine labeled it. If there are too many issues, we can escalate all the way up to requiring a hand-count for a contest.

During this audit, we assume that we have originals of all the paper ballots that were cast during the election. As a simple example, let us assume that we have a contest with two candidates Alice and Bob. At the end of the night, the voting machines report that 10,000 ballots were cast and Alice won the election with 70% of the votes. In general, the audit should verify this result if Alice won the election. To check whether Alice won, we choose a random sample of ballots and have audit teams manually interpret the votes on the chosen ballots. Let us assume that we choose a sample size of 10 ballots. If there are 7,000 votes reported for Alice, we expect that there will be approximately 7 ballots in our sample having a vote for Alice. Perhaps it is alright if there are only 6 ballots with a vote for Alice - we are randomly sampling and we expect some variance. However, what should happen if there are no ballots in our sample for Alice?

In this case, we can escalate the audit. In particular, there is some probability that we were very unlucky and happened to choose only ballots with votes for Bob. Statistically speaking, this is very unlikely. Thus, we can sample some more ballots; let us assume we sample another 50 ballots. Now, let us say that we see that there are 42 ballots with a vote for Alice and 18 ballots with a vote for Bob. This might be enough evidence that Alice won the contest; if so, the audit would be complete. However, if there are 30 ballots with a vote for Alice and 30 ballots with a vote for Bob, then we may have to escalate further. We can repeat this process and keep

increasing the number of ballots we sample. Ideally, the goal of a post-election audit is to provide statistical confidence that the reported results are correct – in our case, that Alice truly won the election. If we cannot provide this confidence, then the audit can escalate to a full hand recount of the ballots to possibly upset the reported result.

Recently, there has been a movement towards risk-limiting audits (RLAs) which are a specific family of auditing techniques that we will describe in the next section.

1.2 Related Work and Resources on RLAs

RLAs, pioneered by Lindeman and Stark [11], are audits that are based on statistical strategies In original auditing techniques, we would require looking at the same number of ballots in contests that have a huge margin as in contests that are very close. This fixed-percentage does not provide any statistical assurance about the outcome of the election. That is, if we expect that Alice won 70% of the votes but there were only 5 votes for Alice in our sample, what does that mean? When do we need to increase the sample size and when can we stop the audit?

RLAs answer these questions by providing a "risk limit guarantee". That is, the risk limit is the maximum probability that the audit will fail to escalate to a full hand-count, given that the contest results are wrong [11]. We can stop the audit when our risk limit is satisfied – that is, the audit is complete when the sample provides sufficiently strong evidence that the contest's reported result is correct, where the required "strength" of the evidence depends on our risk limit. Generally, in elections with large margins, we can satisfy the audit with a small number of ballots. Moreover, RLAs are easily configurable - we can change the risk limit based on the state's policies. Given a risk limit and the margin of the contest, we can calculate the expected initial number of ballots we need to sample to verify the outcome, if the reported margins are correct. Based on the results of the initial sample, we can decide whether the audit has satisfied the risk limit stopping condition or we can escalate, perhaps all the way to a full hand count.

Throughout this thesis, we will refer to this family of procedures as frequentist

audits. For further details about frequentist audits, see [5, 9, 10, 11, 17, 18].

1.3 Bayesian Audits

Another class of audits, designed by Rivest et al. [18], are Bayesian audits . Bayesian audits are also statistically based and also require a variable number of sampled ballots, based on the margin of the elections. Bayesian audits are a more flexible alternate framework. These audits work using simulations and can easily be adapted to different voting paradigms. The Bayesian audit with an upset probability limit of α, proceeds in three major stages:

1. Get a random sample of size k. The value for k can be based on the reported results and typical values range from 20 to 500. (**sample**)

2. Given this sample, run many simulations to estimate what the rest of the ballots which were not sampled look like (**restore**)

3. If the reported winner wins less than $1-\alpha\%$ of the time over all the simulations, sample more ballots (**escalate**)

Malagon et al. [13] provide a succinct explanation of Bayesian audits. We note that the flexibility of Bayesian audits comes largely from the "restore" step. In particular, the Bayesian audit uses simulations to model the population of cast ballots based on the sample. This framework does not rely on the intrinsic details of the "winner" function and can be easily extended to handle other voting methods such as ranked-choice voting.

This thesis discusses both RLAs and Bayesian audits, as well as the pros and cons of each.

1.4 Notation and Relevant Terminology

We use notation and election-related terminology that we introduce here and use throughout the thesis.

Notation. We let $\ln(x)$ denote the natural logarithm of x, and let $\lg(x)$ denote the base-two logarithm of x.

We let $\gamma(x)$ denote the gamma function. For positive, integral x, $\gamma(x) = (x-1)!$.

We let $[n]$ denote the set $\{0, 1, \ldots, n-1\}$, and we let $[a, b]$ denote the set $\{a, a+1, \ldots, b-1\}$.

We let $\mathcal{U}[n]$ denote the uniform distribution over the set $[n]$. In $\mathcal{U}[n]$, the "$[n]$" may be omitted when it is understood to be $[n]$, where n is the number of ballots in the stack. We let $\mathcal{U}[a, b]$ denote the uniform distribution over the set $[a, b]$. If $X \sim \mathcal{U}[n]$, then

$$Pr[X = i] = \mathcal{U}[n](i) = 1/n \text{ for } i \in [n] \ .$$

Thus, $\mathcal{U}$ denotes the uniform distribution on $[n]$. For the continuous versions of the uniform distribution: we let $\overline{\mathcal{U}}(0, 1)$ denote the uniform distribution over the real interval $(0, 1)$, and let $\overline{\mathcal{U}}(a, b)$ denote the uniform distribution over the interval (a, b). These are understood to be probability densities, not discrete distributions. The "$(0, 1)$" may be omitted when it is understood to be $(0, 1)$. Thus, $\overline{\mathcal{U}}$ denotes the uniform distribution on $(0, 1)$.

We let $VD(p, q)$ denote the variation distance between probability distributions p and q; this is the maximum, over all events E, of

$$Pr_p[E] - Pr_q[E].$$

Election Terminology. The term "ballot" here refers to a single piece of paper on which the voter has recorded a choice for each contest for which the voter is eligible to vote. One may refer to a ballot as a "card." Multi-card ballots are not discussed in this thesis.

Audit types. Lindeman et al. [11] describe two kinds of post-election tabulation audits: *ballot-polling* audits, and *ballot-comparison* audits. In a ballot-polling audit, the auditor pulls randomly selected ballots until the sample size is large enough to provide sufficient statistical assurance about the contest outcome. In a ballot-

comparison audit, the auditor samples random ballots, compares them to the ballot's electronic cast-vote record (CVR) and the risk is based on the number and type of discrepancies between the paper ballot and the CVR. In general, ballot-comparison audits are significantly more efficient than ballot-polling audits, particularly for RLAs. We note that both Bayesian audits and RLAs can be divided into these two categories and are procedurally the same. The primary difference between RLAs and Bayesian audits is the stopping condition of the audits.

1.5 Overview of Thesis

Chapter 2 starts by discussing the open problems in election tabulation audits. We categorize these problems into three main sections – audit correctness, flexibility, and efficiency – and discuss the details of each category. We also discuss the specific problems that our thesis focuses on.

In Chapter 3, we analyze the statistical soundness of Bayesian audits and prove that the probability of simulating the exactly correct results increases with the sample size in expectation. This helps justify the choice of Bayesian audits as a good candidate for a statistical audit procedure.

In Chapter 4, we discuss optimization techniques for Bayesian audits. We describe two optimization techniques that we designed and implemented, and give their results on synthetic election data.

In Chapter 5, we focus on workload estimation for RLAs. We design tools which can make initial sample size recommendations for RLAs, to guarantee that the audit will finish in a single round, with high probability.

In Chapter 6, we introduce the idea of approximate sampling to increase the efficiency of audits in practice. Here, we design a particular approximate sampling scheme k-cut and analyze its efficiency compared to counting-based techniques.

In Chapter 7, we explore the usability of k-cut by providing and analyzing empirical data on single cuts. We also discuss metrics to measure the convergence rate of k-cut.

In Chapter 8, we argue that for large k, the model will converge to the uniform distribution exponentially quickly, with minimal assumptions on the single cut distribution.

In Chapter 9, we discuss a simple mitigation procedure for making approximate sampling compatible with RLAs – sample tally mitigation. We prove that little mitigation is required for plurality RLAs, however, we also discuss drawbacks of using this technique.

In Chapter 10, we provide a general mitigation procedure – risk limit adjustment – for making any statistical procedure work with any approximate sampling procedure. Then, based on our empirical data, we analyze the risk limit adjustment required for k-cut and suggest values of k to use in practice.

In Chapter 11, we discuss usage of k-cut in practice, including how to choose values for k and dealing with multiple stacks of ballots. We also provide timing data from pilot audit experiences with k-cut in Indiana and Michigan.

In Chapter 12, we suggest future problems to explore and summarize our contributions.

Chapter 2

Open Problems in Auditing

I spent the first few months of my research exploring different problems related to making statistical election tabulation audits work in practice. The three main categories of problems that I identified and explored over the past year were audit correctness, audit flexibility, and audit efficiency. In this chapter, I provide an overview of each of these categories and identify possible areas to explore.

2.1 Correctness of Audits

Statistical election tabulation audits can be broadly classified as frequentist audits or Bayesian audits. In the frequentist approach, as described by Lindeman and Stark [11], the risk is defined as the probability that, if the true outcome of the contest did not match the reported result, the audit would not detect the issue. That is, the frequentist risk measurement represents a worst-case bound on the probability of accepting an incorrect outcome.

By contrast, computing the Bayesian upset probability relies on simulations. In particular, the Bayesian model assumes that the true population of all the ballots is similar to the sample we draw; that is, the sample is "representative" of the population. If this is true, we can create a variety of "test" populations through simulations, using methods such as Polya's Urn. Then, for each test population, we compute the winner. The Bayesian upset probability is defined as the percentage of simulations

where someone other than the reported winner wins in the test population. In practice, we use the Dirichlet-Multinomial model to generate our "test" populations which provides a significant increase in efficiency over the Polya's Urn technique. The hyperparameters for the Dirichlet-Multinomial simulations are the sample tally vector with some additional pseudocounts.

Bayesian audits have a variety of applications, as described in Chapter 1. However, the statistical properties of Bayesian upset probabilities have not been explored very much. For instance, before our work, we did not know of any tools to estimate the expected number of ballots required to satisfy a Bayesian audit with an upset probability limit of α and a margin of m. Furthermore, we also do not know the relationship between the Bayesian upset probability and the risk limit of an RLA.

We would like to prove that Bayesian audits satisfy certain statistical properties. For simplicity, we start by showing that as the sample size increases, the probability of our test population being exactly correct increases monotonically in expectation. Intuitively, this shows that the "restore" step of the audit has a higher chance of generating the correct population of ballots as the sample size increases. We hope to use this work to develop a stronger understanding of the correctness and convergence rates of Bayesian audits.

2.2 Flexibility of Audits

We note that for plurality or majority contests, the RLA definition works well. Work done by Lindeman et al. [11, 12] shows how to calculate the risk for these contests with single or multiple winners. However, this work does not easily extend into more complex voting methods.

For instance, consider ranked-choice voting, where voters fill in a preferential ballot. In particular, instead of voting for Alice or Bob, a voter would fill out a preference list of candidates. A voter could claim that his/her first choice is Alice, his/her second choice is Bob and his/her last choice is Charlie. Computing the winner for these contests is quite tricky. In the instant-runoff model, the candidate with the

least first-choice votes is eliminated. Then, they are removed from every ranking in every ballot and the process is repeated until two candidates remain, where it becomes a simple majority contest. This style of voting (with further procedural steps included) is used for primary elections in Maine. However, it is tricky to identify the "margins" of a race with ranked-choice voting, which makes it tricky to design a risk-limiting audit. Blom et al. [4] have done research in this area to combine techniques from risk-limiting audits for plurality elections into a format for ranked-choice voting for instant-runoff voting. But, we note that the combining techniques introduce a large overhead in complexity and still do not generalize easily for other voting techniques. To the best of our knowledge, developing risk-limiting audits which are independent of the details of the voting procedure is currently still an open problem.

One approach is to instead use Bayesian audits. Since the Bayesian audit simply involves replicating the votes on the sample ballots and computing the winner, the only requirement for the Bayesian framework is is the availability of a "social choice function" that computes the winner of a test population. However, as mentioned previously, the statistical properties of the Bayesian upset probability have not been thoroughly explored. Thus, the policy decisions around choosing a target upset probability are perhaps less straightforward.

In my thesis, I do not explore applications for RLAs to more complex voting methods. However, I mention this area of work to emphasize the importance of developing a deeper understanding of Bayesian audits, since they are currently the only statistically-based audit technique for these voting methods.

2.3 Efficiency of Audits

Finally, for audits to be useful in practice, we would like to be able to measure and optimize their efficiency. In particular, Stark's website [20] provides both RLA ballot polling and ballot comparison tools which can estimate an initial sample size for a given contest, based on the reported margins of the contest. However, these estimates

are based on the expected sample tallies, assuming the reported margins are accurate.

First, we wanted to better understand the workload during an audit. For example, we found that election officials often preferred to sample a few extra ballots in the first round and finish in a single round, rather than sampling fewer ballots in the first round and then requiring escalation. Based on this insight, we wanted to explore providing a more general workload estimation tool, which could be used to choose a sample size, so that the audit will complete in a single round with high probability. The required high probability could be chosen by the auditor, based on how much work they are willing to do in the first round.

We also wanted to explore how to allocate workload in complex multi-jurisdiction audits. In particular, if there are multiple strata and a total required sample size s, we can allocate samples to different strata at different rates. In practice, this could be based on the efficiency or the margins in the different strata. This can be phrased as an optimization problem, where we are trying to minimize the total workload of the auditor while satisfying the required risk limit or bound on the upset probability. We model the total workload as the total number of ballots that need to be sampled, although in more complex cases, this could be a weighted sum.

In my thesis, I explore and implement some initial optimization algorithms for workload estimation and sample size allocation. These algorithms are implemented in the planner module of Rivest's Bayesian audit support program [16]. Moreover, I also develop some tools to estimate initial sample sizes where the audit will complete within a single round with high probability.

Part II

Analyzing Bayesian Audits

Chapter 3

Properties of Bayesian Audits

This chapter discusses the statistical soundness of Bayesian audits. In particular, we will prove that Bayesian audits have a form of monotonicity; that is, we show that as the sample size increases, our probability of simulating the exactly correct actual tally increases steadily in expectation. This shows that the Bayesian simulations are a good candidate to use for statistical election audits. We will discuss further extensions to our work that can be used to relate Bayesian upset probabilities to the risk limit of an RLA.

3.1 Problem Description

Statistical properties of Bayesian audits including their correctness and convergence rates have not been thoroughly explored. We would like to prove that the Bayesian audits have similar statistically sound properties to RLAs, although the Bayesian upset probability and the RLA's risk limit are fundamentally different measurements.

As a first step, we wanted to prove that as we sample more ballots, the probability of restoring the exactly correct population tally will be non-decreasing. Intuitively, the restoration process follows the ideas in Polya's Urn. In particular, let us assume that we have 7 votes for Alice and 3 votes for Bob in our sample, and there are 20 unsampled ballots left. Our urn starts out with a single vote for Alice and a single vote for Bob; these are pseudocounts for when we have no votes for a candidate. We

add all the votes in our sample to our urn which now contains 8 votes for Alice and 4 for Bob. Then, in a single run of our restore operation we randomly choose a ballot from the urn - let us say that the ballot contains a vote for Bob. Then, we add in 2 ballots for Bob into our urn. We repeat this operation until there are 32 ballots in our urn, remove the ballots corresponding to the pseudocounts, and compute the winner. The theorem we prove shows that the probability of the population tally being exactly the same as the actual votes increases with the sample size for almost all possible initial sample sizes.

For efficiency, we use the Dirichlet Multinomial distribution for our simulations instead of using an urn. Work done by Marshall and Olkins [14] shows that this distribution approximates the "urn-draw-replace" process well. We apply the uniform prior of a vote per candidate as a hyperparameter α_i; that is, α_i is set to 1 before seeing any votes for candidate i.

3.2 Proof of Monotonicity

To prove our theorem, we first define a few key terms. Let us assume that we have m candidates in a race. Through our audit procedure, we have obtained a sample s, an m-dimensional vector $s_1..s_m$ where $\sum_{i=1}^{m} s_i = z$, and s_i represents the number of votes in our sample for candidate i. The actual real tally is a vector X, where x_i represents the number of total votes for candidate i and $\sum_{i=1}^{m} x_i = n$, the total number of ballots cast in the contest. We use a prior of one vote per candidate to start with, so $\alpha_i = s_i + 1$. The α vector and total number of unsampled ballots $(n - z)$ are input to the Dirichlet-Multinomial distributions which produces sample "extensions." We want to show that the probability of generating $u = X - s$ (we refer to this as the "non-sample" tally) by sampling from the Dirichlet-Multinomial distribution increases with z in expectation.

Theorem 1 (Monotonicity) *In expectation, when we draw a new ballot, the prob-*

ability of generating the exact correct remaining data increases if

$$m \leq (n-z)^2\Big[1 + \frac{\sum_{i=1}^{m} u_i\Big(\sum_{a\neq i}(s_a+1)\Big)\prod_{k\neq i}(s_k+1)}{(\sum_{j=1}^{m} u_j)\prod_{j=1}^{m}(s_j+1)}\Big]$$

where m is the number of candidates, n is the total number of ballots cast, z is the size of the sample, s_i for $i \in [1,m]$ is the number of votes for candidate i in the sample and u_i for $i \in [1,m]$ is the correct non-sample tally.

Proof:

Without loss of generality, we assume that the sample s does not include all the ballots to ensure that drawing a new ballot is well-defined.

For any given sample s, we can define $s^{'}(j)$ as the sample s with one additional ballot for a particular candidate j. In particular, we have an m-dimensional vector $s^{'}(j)$, where $s^{'}(j)_i = s_i$ for all $i \neq j$ and $s^{'}(j)_j = s_j + 1$. Thus, $\sum_{i=1}^{m} s^{'}(j)_i = z+1$. Then, we can compute the probability of generating u given s compared to the probability of generating of generating $u' = X - s^{'}(j)$ given $s^{'}(j)$. We show that the probability of generating the correct non-sample tally increases in expectation as the number of ballots in the sample increases.

We can then calculate the probability mass function of generating u' given $s^{'}(j)$ and u given s. We note that the PMF of the Dirichlet function for a given set of α-values is as follows:

$$PMF(u|\alpha) = \frac{t!\Gamma(\sum \alpha_k)}{\Gamma(t+\sum \alpha_k)}\prod_{k=1}^{m}\frac{\Gamma(u_k+\alpha_k)}{(u_k)!\Gamma(\alpha_k)}$$

where $t = \sum u_i$ [23].

Thus, plugging in our values gives us that

$$PMF(u|s) = \frac{(n-z)!\Gamma(z+m)}{\Gamma(n+m)}\prod_{a=1}^{m}\frac{\Gamma(s_a+u_a+1)}{u_a!\Gamma(s_a+1)}$$

$$PMF(u'|s^{'}(j)) = \frac{(n-z-1)!\Gamma(z+m+1)}{\Gamma(n+m)}\frac{\Gamma(s_j+u_j+1)}{(u_j-1)!\Gamma(s_j+2)}\prod_{a=1,a\neq j}^{m}\frac{\Gamma(s_a+u_a+1)}{u_a!\Gamma(s_a+1)}$$

Using these expressions, we can calculate the difference in PMF which we want to show is non-negative. We note that $\Gamma(n) = (n-1)!$ for all positive integral n.

$$PMF(u'|s^{'}(j)) - PMF(u|s)$$
$$= \Big(\frac{(n-z-1)!\Gamma(z+m)\Gamma(x_j+1)}{\Gamma(n+m)(u_j-1)!\Gamma(s_j+1)} \prod_{a=1,a\neq j}^{m} \frac{\Gamma(x_a+1)}{(u_a)!\Gamma(s_a+1)}\Big)\Big[\frac{z+m}{s_j+1} - \frac{n-z}{u_j}\Big]$$

We note that, assuming $n \geq z+1$, the first term is positive. Thus, we can say that

$$PMF(u'|s^{'}(j)) - PMF(u|s) = C\Big[\frac{z+m}{s_j+1} - \frac{n-z}{u_j}\Big]$$

for some $C > 0$.

We note that, on an individual ballot level, this implies that drawing a ballot for candidate j only increases the probability of generating the correct non-sample tally if we satisfy the inequality

$$\frac{z+m}{s_j+1} - \frac{n-z}{u_j} > 0$$

However, we want to calculate the expected change in PMF by considering all possible values for j. Thus, we know that

$$\mathbb{E}[\Delta PMF] = \sum_{j=1}^{m} \Pr[\text{draw a ballot for candidate } j](PMF(u'|s^{'}(j)) - PMF(u|s)).$$

We note that the probability of drawing a ballot for candidate j when our current sample tally is s is $\frac{u_j}{n-z}$. Thus,

$$\mathbb{E}[\Delta PMF] = \sum_{j=1}^{m} \frac{u_j}{n-z} C\Big(\frac{z+m}{s_j+1} - \frac{n-z}{u_j}\Big)$$
$$= \sum_{j=1}^{m} \Big(\frac{u_j C(z+m)}{(n-z)(s_j+1)} - C\Big)$$

We want to find when this quantity is non-negative. In particular, we note that

$$\mathbb{E}[\Delta PMF] \geq 0$$

$$\iff \sum_{j=1}^{m} \Big(\frac{u_j C(z+m)}{(n-z)(s_j+1)} - C \Big) \geq 0$$

$$\iff \Big[\frac{z+m}{n-z} \sum_{j=1}^{m} \Big(\frac{u_j}{s_j+1} \Big) \Big] - m \geq 0$$

$$\iff (z+m) \sum_{j=1}^{m} \frac{u_j}{s_j+1} \geq \frac{m}{n-z}$$

We can then apply Lemma 1 (proven below) to the term $\sum_{j=1}^{m} \frac{u_j}{s_j+1}$. In particular, we note that

$$\sum_{j=1}^{m} \frac{u_j}{s_j+1} \geq \frac{q \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j+1)} \qquad \forall q \leq Q$$

for

$$Q = 1 + \frac{\sum_{i=1}^{m} u_i \Big(\sum_{a \neq i} (s_a+1) \Big) \prod_{b \neq i} (s_b+1)}{(\sum_{j=1}^{m} u_j) \prod_{j=1}^{m} (s_j+1)}$$

We note that the inequality is tight when $q = Q$.

Plugging in the results from Lemma 1 gives us that

$$\mathbb{E}[\Delta PMF] \geq 0$$

$$\iff (z+m) \frac{Q * \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j+1)} \geq \frac{m}{n-z}$$

However, we note that $\sum_{j=1}^{m} (s_j+1) = z+m$ and $\sum_{j=1}^{m} u_j = (n-z)$. Plugging this in tells us that this holds for all

$$m \leq (n-z)^2 Q$$

. ■

This implies that as long as there are enough non-sample ballots (ballots which were cast which have not been sampled yet), the probability of generating the exactly

correct non-sample tally increases in expectation. We note that Q is always greater than or equal to 1. For completeness, we prove the Q-Lemma as well.

Lemma 1 (Q-Lemma)

$$\sum_{j=1}^{m} \frac{u_j}{s_j + 1} \geq \frac{q \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j + 1)} \qquad \forall q \leq Q$$

where

$$Q = 1 + \frac{\sum_{i=1}^{m} u_i \Big(\sum_{a \neq i} (s_a + 1) \Big) \prod_{b \neq i} (s_b + 1)}{(\sum_{j=1}^{m} u_j) \prod_{j=1}^{m} (s_j + 1)},$$

s_i and u_i for $i \in [1, m]$ are non-negative.

Proof: This proof is mostly by algebraic manipulation. In particular, we note that

$$\sum_{j=1}^{m} \frac{u_j}{s_j + 1} \geq \frac{q \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j + 1)}$$

$$\iff \frac{q \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j + 1)} \leq \sum_{j=1}^{m} \frac{u_j}{s_j + 1}$$

$$\iff \frac{q \sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} (s_j + 1)} \leq \frac{\sum_{j=1}^{m} (u_j \prod_{a=1, a \neq j}^{m} (s_a + 1))}{\prod_{j=1}^{m} (s_j + 1)}$$

where we have gotten the right-hand side in the form of a single fraction, with a common denominator. We can now cross multiply to get:

$$\iff q(\sum_{j=1}^{m} u_j) \prod_{j=1}^{m} (s_j + 1) \leq \sum_{j=1}^{m} (s_j + 1) \big[\sum_{j=1}^{m} u_j \prod_{a=1, a \neq j}^{m} (s_a + 1) \big]$$

$$\iff q \leq \frac{\sum_{j=1}^{m} (s_j + 1) \Big[\sum_{i=1}^{m} (u_i) \prod_{a \neq i} (s_a + 1) \Big]}{(\sum_{j=1}^{m} u_j) \prod_{j=1}^{m} (s_j + 1)}$$

Then, we note that we can split the numerator on specific values of j. In particular, we can split our first term, the sum over $s_j + 1$, based on whether $i = j$. Using this,

we can write our bound on q as:

$$q \leq \frac{\left[\sum_{i=1}^{m}(u_i)\prod_{a\neq i}(s_a+1)*(s_i+1)\right]+\left[\sum_{i=1}^{m}(u_i)\sum_{a\neq i}(s_a+1)\prod_{b\neq i}(s_b+1)\right]}{(\sum_{j=1}^{m}u_j)\prod_{j=1}^{m}(s_j+1)}$$

$$q \leq \frac{\left[\sum_{i=1}^{m}(u_i)\prod_{a}(s_a+1)\right]}{(\sum_{j=1}^{m}u_j)\prod_{j=1}^{m}(s_j+1)}+\frac{\left[\sum_{i=1}^{m}(u_i)\sum_{a\neq i}(s_a+1)\prod_{b\neq i}(s_b+1)\right]}{(\sum_{j=1}^{m}u_j)\prod_{j=1}^{m}(s_j+1)}$$

However, we note that the first term cancels out entirely! This gives us our desired bound of

$$q \leq 1+\frac{\sum_{i=1}^{m}u_i\Big(\sum_{a\neq i}(s_a+1)\Big)\prod_{b\neq i}(s_b+1)}{(\sum_{j=1}^{m}u_j)\prod_{j=1}^{m}(s_j+1)}$$

■

3.3 Takeaways and Extensions

Thus, we have proven that as the sample size increases, our probability of simulating the exactly correct actual tally increases in expectation for almost all possible sample sizes. A simple sufficient, though not necessary, condition for this bound to hold is when

$$m \leq (n-z)^2\,,$$

since Q is always at least 1. This implies that the inequality will only stop holding for the last $\sqrt{m}$ ballots in the audit.

Thus, in most stages of the audit process, the probability of restoring the exactly correct unsampled data tally increases in expectation with the sample size. In practice, this means that the bound holds for all but the last ballot in races with up to 4 candidates. Even with races with more candidates, our simulations show that the bound appears to hold for all but the last ballot. Thus, we have shown that in almost all the stages of the Bayesian audit, the simulations get more accurate (with respect to our metric) monotonically. In general, since we expect that $m << n$, it would be feasible to require a full hand-count if we reach the last $\sqrt{m}$ ballots without satisfying

our upset probability limit.

Extensions of this work could be used to show that Bayesian audits have other good statistical properties. That is, we have shown that the probability of generating the exact correct "non-sample" tally increases monotonically. We would further like to show that as the sample size increases, the winner in the simulations is likely to be the actual correct winner. In particular, we would like to find a closed-form bound on the probability of generating a winner i, for any given sample tally. We might be able to combine this result with the probability of generating sample tallies with uniformly random sampling, to relate the Bayesian upset probability and the risk limit of an RLA.

Another extension we would like to explore is the rate of convergence of the audits. That is, we want to look at the variance of the restored ballots as the sample size increases. Using this, we would like to show that as we increase the sample size, the probability of generating the correct overall winner increases quickly. If we can find a tight approximation for this, we can use this to choose the appropriate sample size for an audit based on the reported margins. This could be used to estimate the workload of a Bayesian audit and make recommendations on how each round of an audit should proceed in practice.

Part III

Workload Estimation and Optimization

Chapter 4

Optimization of Audits

In this chapter, we will discuss the framework for optimization techniques for minimizing audit workload. We only discuss Bayesian audits in this chapter and we outline techniques for workload estimation as well as optimization. In particular, we define a naive framework, where an audit proceeds in rounds and we sample a constant number of ballots in each round. We test this technique out on a sample election and calculate the required sample size before the audit is complete. Then, we define two new techniques – the random walk approach and the Robbins-Monro discrete optimization approach. For each technique, we define escalation techniques where we require a variable number of sampled ballots in each round. We show that both these techniques require a smaller number of ballots before the audit is complete on our sample election. These techniques are implemented in Rivest's Bayesian audit support program [16].

4.1 Problem Description

To analyze and optimize the efficiency of Bayesian audits, we would like to see how the auditing process works in a variety of cases. For instance, often contests span several counties where we can sample from each county at different rates. We note that the audit proceeds in rounds. At each round where the upset probability limit is not satisfied, we have to choose which counties to sample from and how much to

sample from each one. Ideally, we want to sample enough ballots so that the audit is likely to complete in the next round - however, we are trying to minimize the total work that needs to be done; hence, auditing more ballots than necessary to satisfy the upset probability stopping condition is suboptimal.

After abstracting away a lot of the real-life logistics of these problems, we can consider a simple form of the problem. In particular, we can define a function $f(x)$, which takes as input a collection of ballots. We denote z as our sample size, which is the size of the vector x. Let n be the total number of ballots cast in the contest. Its return value represents the Bayesian upset probability for this particular set of ballots. We cannot measure $f(x)$ directly for a given set of ballots. However, we can run a simulation to restore the remaining ballots and find out whether a given simulation supports the reported winner or not. Ideally, given a particular set of ballots, we want to estimate how many ballots we should sample in the next stage of the audit so that $f(x)$ for the new set of ballots will be approximately our upset probability limit.

4.2 Naive Approach

The naive approach, that was initially implemented in Rivest's Bayesian audit support program, would sample the same number of ballots at each stage of the audit [16]. In particular, the default setting of this would sample 40 ballots from each stratum in each round of the audit. After the additional ballots were sampled, the tool would measure the Bayesian upset probability. If the audit's stopping condition was satisfied, then the audit would stop. If not, another 40 ballots per stratum would be sampled.

We experimented with new approaches in a specific simulated election. In this election, we had a contest for Mayor over two strata. The first stratum had no cast vote records (CVRs) and could only perform a ballot-polling audit. In this stratum, there were 8,000 reported votes for Alice and 2,000 reported votes for Bob with no CVRs. The second stratum had CVRs and could perform a ballot-comparison audit. In the second stratum, there were 48,000 reported votes for Alice and 52,000 reported

votes for Bob. There were no discrepancies - all the reported votes and actual votes matched.

For the Polya's urn simulations in a ballot-comparison audit, our "ballots" are in the form (reported vote, actual vote). Thus, a vote for Bob that is reported for Alice is different than a vote for Bob that is reported for Bob. In our "restore" procedures, if we see many votes for Bob that were reported for Alice in our sample, then the ballots in our model populations will also have a lot of votes of this form. However, in practice, we assume that the voting machines are likely to be accurate. Thus, we start our urn off with pseudo-counts of 50 ballots of the form (Alice, Alice) and (Bob, Bob). Then, we add an additional pseudocount of 0.5 for votes of the form (Alice, Bob) and (Bob, Alice). This represents our prior that the voting machines are highly likely to be accurate but occasionally make mistakes. For the ballot-polling audit, we start off with pseudocounts of one ballot apiece for (Missing, Alice) and (Missing, Bob) since we have no reported votes.

Using this model, the naive approach requires a sample of 320 ballots to satisfy the stopping condition of a 5% Bayesian upset probability.

4.3 Random Walk Approach

If we ran a large number of simulations, we can estimate $f(x)$ quite accurately for any given value of x, but this is quite inefficient. In particular, for any vector x_0 which represents our current sample tally, the number of possible extensions x_* that we could produce from x_0 is exponential in $n - z$. For each extension, we would need many simulations to calculate $f(x_*)$ and find the appropriate sample size to escalate to.

Thus, we consider a more efficient approach based on random walks. In particular, we start with a sample tally s which is a vector, where each element s_i of the vector represents the number of votes for candidate i in the sample and the total sample size is z. Furthermore, we define n as the total number of ballots, α as the risk limit and u as the default number of ballots to sample next. Then, we use Dirichlet Multinomial

to extend our sample s to a sample of size $z + u$ - we can think of this as simulating what would happen if we sampled an additional u ballots.

Then, we treat our extension as a sample and "restore" the remaining $n - z - u$ ballots and compute the winner k times where k is a hyperparameter that we tune. Typical values of k used in our simulations ranged from 1 to 6. If all k simulations have the actual winner be the reported winner, we decrease u by 1 with some probability l; if not, we increase u by 1 with some probability r.

Ideally, we want to choose l and r so our random walk will converge on the value of our risk limit. By the definition of the Bayesian upset probability, we know that when $f(x) = \alpha$, a simulation will return the incorrect winner with probability α; ideally, at this x, we want to increase or decrease u with equal probability so we converge at this point.

Our probability of decreasing the number of ballots becomes the probability that all k winners are correct, which is $(1 - \alpha)^k$ multiplied by l. Similarly, the probability of increasing the number of ballots becomes the probability that at least one of the k winners is wrong, which is $1 - (1 - \alpha)^k$ multiplied by r. This gives us that the ratio of $r : l$ should be

$$\frac{(1 - \alpha)^k}{1 - (1 - \alpha)^k}$$

. Thus, we can run thousands of steps of this random walk and we expect to converge approximately on the value of u which gives us an upset probability of α.

We note that we can extend this procedure to contests that span multiple counties. That is, we can sample at a different rate in each county to increase efficiency. To do this, we can choose a county (using round robin or more complex heuristics) and run a random walk to determine how many additional ballots to sample in that county. We can repeat this process over all the counties to design a sampling plan for the next stage of the audit.

We implemented this functionality in the planner module of the Bayesian audit support program with varying values of k. We tested this on the same election described in the previous section.

We can show that on test cases, the escalation with varying sample sizes proves to be more efficient and required fewer sample ballots to reach the appropriate upset probability. For instance, if we choose $k = 3$ and run 70 iterations of a random walk in each escalation, the required sample size goes down from 320 ballots to 305 ballots.

In the future, we would like to extend the hyperparameter search to see how much more efficiency this technique could add. However, we consider these results a promising start.

4.4 Robbins-Monro Optimization

We also implemented a discrete, multi-dimensional version of the Robbins-Monro optimization algorithm [19] developed by Hill [7].

This works by optimizing over the loss function $|f(x) - \alpha|$ which is minimized when $f(x)$ is exactly α. In general, we optimize this algorithm by following the technique outlined by Hill which provides a framework for optimizing over different counties at the same time. Intuitively, we run gradient descent on the function $|f(x) - \alpha|$ by approximating it with a continuous piecewise-linear function.

If exact measurements were available for our function at any given value of x, then we could find the minimum value by running gradient descent on our continuous approximation. However, as previously mentioned, this would be computationally inefficient. Thus, instead we assume that we only have access to noisy estimates of $|f(x) - \alpha|$ for any given value of x. We can obtain these estimates through a few simulated extensions with the Dirichlet-Multinomial model. We denote these noisy estimates as $g(x)$. To account for the noisy measurements, we estimate the gradient at x using finite differences instead of vanilla gradient descent. That is, we calculate a random vector δ where each δ_i is a random Bernoulli variable which takes the value 1 with probability 0.5. Then, we perturb our vector x and compute $g(x_+)$ and $g(x_-)$ where $x_+ = x + \delta$ and $x_- = x - \delta$. We can use $g(x_+) - g(x_-)$ (with required normalization) as our estimate for our gradient and use it to determine our direction [7].

For our step sizes, we use Robbins-Monro step sizes of $a_k = (k+1)^q$ where $q < 1$, to ensure that the noise is averaged out. In particular, we update x as follows:

$$x_{i+1} = x_i - a_k(g(x_+) - g(x_-)).$$

As a default, we run 100 trials to estimate $g(x)$.

We have found that choosing $q = \frac{-1}{3}$ or $\frac{-1}{2}$ provides promising results. Our default parameters use $q = \frac{-1}{3}$. We run our random walk for a default of 10 steps before choosing our new value of x. For simplicity, we note that we change x at the same rate in each jurisdiction; however, this does not always have to hold.

Our results showed some improvements on the random walk approach, with regards to minimizing the overall number of ballots in elections across several counties. We ran the same test election, using this optimization technique, with

$$a_k = \frac{1}{(k+1)^{1/3}},$$

and the default 10 steps per iteration. Similar to the random walk technique, the required sample size decreased from 320 ballots to 306 ballots.

4.5 Takeaways and Extensions

We have shown that both our optimization techniques – the random walk approach and discrete Robbins-Monro optimization – quickly show an improvement in the required workload for a sample election. We note that our results are preliminary; we have not explored many other sample elections or run a grid search to find the appropriate hyperparameters. However, our initial results were quite promising.

We would like to extend this work by running many more rounds of testing to better understand when these optimization techniques are useful. Furthermore, we would like to define a more complex definition of "workload," instead of solely analyzing the number of ballots. We note that, in our experience, many counties prefer to sample a few extra ballots and have the audit complete in a single round with high

probability. We would like to integrate these intuitions to form a better cost function to optimize over.

Chapter 5

Work Estimation for Audits

This chapter explores work estimation tools for RLAs to provide a easy-to-use tool for election auditors to use. This tool is designed to guarantee that the audits terminate in a single round with high probability. In particular, we analyze both ballot polling and ballot comparison risk limiting audits. We provide Jupyter notebooks which calculate the initial required sample size for each audit to guarantee that the audit completes in a single round with high probability.

5.1 Ballot Polling Workload Estimation

We follow the structure outlined by Lindeman et al. [12]to estimate the required sample size for an audit. In particular, we assume there are m candidates in a race, each of whom have a reported voteshare s_i, for i in the range $[1, m]$. For simplicity, we ignore t, a tolerance factor for RLAs.

The RLA procedure for a reported winner w is as follows:

- Initialize T = 1
- If the ballot is for the winner, multiply T by $2s_w$
- Else, if it is valid for anyone else, multiply T by $2(1 - s_w)$
- Stop when T is greater than $\frac{1}{\alpha}$

Thus, we choose a sample size z to guarantee that after z ballots, we satisfy the stopping condition in expectation. Assuming there are z_w votes for the reported winner in the sample, the value of T will become

$$T = (2s_w)^{z_w}(2(1-s_w))^{z-z_w}.$$

Solving this for z, where we assume that $z_w = c * z$ tells us:

$$\begin{aligned}
&T > \frac{1}{\alpha} \\
&\iff (2s_w)^{z_w}(2(1-s_w))^{z-z_w} > \frac{1}{\alpha} \\
&\iff z_w \ln(2s_w) + (z - z_w)\ln(2(1-s_w)) > \ln(\frac{1}{\alpha}) \\
&\iff z(c\ln(2s_w) + (1-c)\ln(2(1-s_w))) > \ln(\frac{1}{\alpha}) \\
&\iff z > \frac{\ln(\frac{1}{\alpha})}{c\ln(2s_w) + (1-c)\ln(2(1-s_w))}
\end{aligned}$$

In expectation, we expect $c = s_w$. This makes the expected sample size

$$z > \frac{\ln(\frac{1}{\alpha})}{s_w\ln(2s_w) + (1-s_w)\ln(2(1-s_w))}$$

This provides a similar bound to Lindeman's work [12] with a minor additive factor difference, where Lindeman allows for the fact that the final value of T typically exceeds $\frac{1}{\alpha}$. For simplicity, we use our bound as the baseline and note that all our extensions can also include this additive factor for safety.

We note that in the final steps in solving for z, we assume that $z_w = c * z$ which is only true in expectation. We would like to calculate the required sample size for finishing in one round with high probability.

To do this, first we note that the number of votes produced for the reported winner, assuming the reported voteshare is exactly s_w is a binomial distribution where there are z trials and each trial has a probability s_w of success. In practice, the actual voteshare is not necessarily exactly s_w. So, in our simulations, we allow for a

parameter representing the actual voteshare and a separate parameter for the reported voteshare, which is used to update the value for T in the ballot-polling audit.

For example, let us assume that the reported voteshare for the winner in the election is 70% and we are quite confident that the actual voteshare for the winner was at least 65%. Then, we can tell our tool that we know that the actual voteshare for the winner was 65%, which can be used to predict the initial sample size in a more conservative manner. The reported voteshare of 70% will be used to calculate the risk of the audit. If we are confident in the reported voteshares being accurate, then these parameters can take the same value.

Due to the nature of the sequential probability test that the RLA uses, if the actual and reported voteshares are significantly different, then there is a significant probability of escalating to a full hand count. That is, if the actual margin is not more than half the reported margin, then the average sample number is undefined. In this case, the audit has a positive probability of escalating to a full hand-count even when the results are accurate.

Thus, we can use the binomial CDF to find a lower bound on the number of votes that we will see in a sample of size z, with probability $1 - \epsilon$. An example plot for an actual voteshare of 70%, and varying sample sizes, with $\epsilon = 0.05$ is shown in Figure 5-1.

We denote the minimum fraction of votes in our sample for the winner as c^*. Thus, we can plug in this value of c^* into our bound above to guarantee that our audit completes in a single round with probability at least $1-\epsilon$. That is, we choose z to guarantee that

$$z(c^* \ln(2s_w) + (1 - c^*) \ln(2(1 - s_w))) > \ln(\frac{1}{\alpha}) .$$

Again, we note that this is only well-defined when c^* values are close to expectations to guarantee that z is not negative. The code to calculate initial sample sizes with high probability is provided in Appendix A.

For a simple example, we note that if we run an audit with a 5% risk limit,

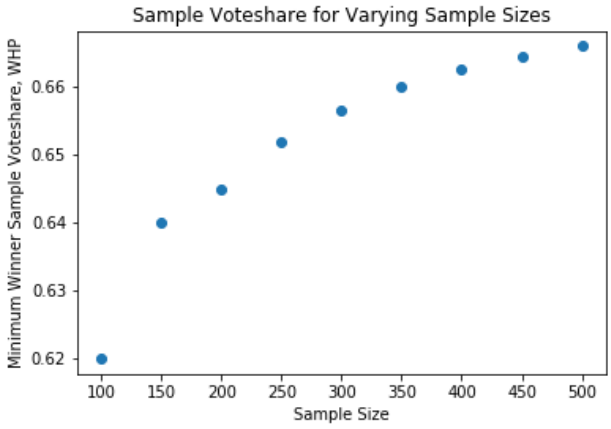


Figure 5-1: Minimum Sample Voteshare With High Probability
Minimum voteshare in a sample, with at least 95% probability, for varying sample sizes, given an actual voteshare of 70% in the population.

where the winner's reported voteshare is 70%, the winner's actual voteshare is 65%, and $\epsilon = 0.2$, then our tool suggests sampling 183 ballots in the first round to complete the audit in the first round with at least 80% probability. We note that due to the way BRAVO works, if our actual voteshares and reported voteshares are not that similar, the number of ballots increases very quickly. However, this is not due to the workload estimation tool, but rather the nature of the probability test.

5.2 Ballot Comparison Workload Estimation

Stark [21] outlined the risk calculation for simple ballot comparison RLAs. Our simulations follow his outline and focus on estimating the required sample size for a ballot-comparison audit.

We assume there are m candidates in a race. Let s_i denote the reported vote share for candidate i and n denote the total number of ballots cast. We define z as the sample size drawn for the audit. We can define V as the smallest reported margin (ex. 0.1) between the reported winner and a runner-up. In particular, if s_w is the

voteshare for the reported winner, then we can define V as

$$\min_{i \neq w}(s_w - s_i)\,.$$

We can denote γ as the inflation factor in the audit. We note that we require $\gamma > 1$. From Stark's work, we know that larger values of γ increase the initial sample size but require less expansion if there are more overstatements than expected. For our simulations, we choose $\gamma = 1.01$.

Thus, the P-value for the ballot comparison audit is

$$P = (1 - 1/U)^z * (1 - 1/(2\gamma))^{-o_1} * (1 - 1/\gamma)^{-o_2}$$

where $U = \frac{2\gamma}{V}$ and there are o_1 single-vote overstatements and o_2 two-vote overstatements.

For our sample size calculations, we note that the number of single-vote overstatements o_1 is a binomial random variable with $p = r_1$, where r_1 is the rate of one-vote overstatements and m trials. Similarly o_2 is a binomial random variable with $p = r_2$, where r_2 is the rate of two-vote overstatements and m trials. Using these numbers, we can calculate for a given sample size the maximum number of 1 and 2 vote overstatements we will see with probability at least $(1-\epsilon)$. Given these values, we can choose z to guarantee that the audit will complete in the first round with high probability.

For simplicity, we choose $\gamma = 1.01$ and a minimum margin of 0.1. Following the structure of Stark's work, we choose a single vote overstatement rate of 0.5%, a double vote overstatement rate of 0.1%, and a risk limit of 5% [21]. We would like our audit to complete within the first round with probability at least 90%. That is, there is at most a 10% chance of our audit escalating if our values for r_1 and r_2 are accurate. Our tool suggests a sample size of 72 ballots for these settings.

In Figure 5-2, we can plot the change in the required sample size for fixed values of ϵ, γ, r_1, and r_2, as a function of the minimum margin in the election.

We can see an exponential decay of the number of samples required as the minimum margin value increases, which matches our expectations. We find similar graphs

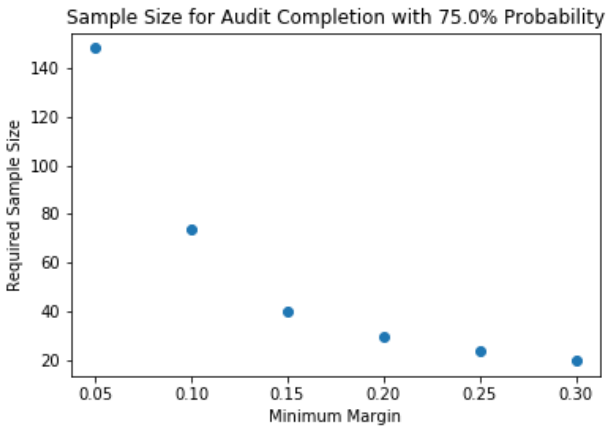


Figure 5-2: Minimum Ballot-Comparison Sample Estimates
Required sample size decays as the minimum margin increases, for fixed values of $\epsilon = 0.25$, $\gamma = 1.01$, $r_1 = 0.5\%$, and $r_2 = 0.1\%$.

for varying values of ϵ, although the required number of ballots for small values of ϵ and a small margin grow extremely quickly. For instance, a minimum margin of 5% and $\epsilon = 0.05$ would require an initial sample size over 600 ballots. The code to calculate initial sample sizes with high probability is provided in Appendix B.

Part IV

Approximate Sampling and k-cut

Chapter 6

Introduction to Approximate Sampling

In this chapter, we will introduce approximate sampling, the major project that I worked on to improve the efficiency of the sampling process for post-election audits. Here, we begin by discussing related sampling techniques for RLAs. We introduce our approximate sampling technique k-cut and discuss the increase in efficiency from using k-cut compared to previous counting-based techniques.

Throughout this chapter, we use notation (including $[n]$ and $\mathcal{U}$) which is defined in Section 1.4.

6.1 Related Work

The goal of RLAs are to provide assurance that the reported results of the contest are correct; that is, they agree with the results that a full hand-count would reveal. To do this, the auditor draws ballots uniformly at random one at a time from the set of all cast paper ballots, until the sample of ballots provides enough assurance that the reported outcomes are correct. As previously discussed, an RLA takes as input a "risk-limit" α (like 0.05), and ensures that if a reported contest outcome is incorrect, then this error will be detected and corrected with probability at least $1 - \alpha$.

This work explores a novel method for drawing a sample of the cast paper ballots.

The new method may often be more efficient than standard methods. However, it has a cost: ballots are drawn in a way that is only "approximately uniform." This paper also provides ways of compensating for such non-uniformity.

There are two standard approaches for drawing a random sample of cast paper ballots:

1. **[ID-based sampling]** Print on each scanned cast paper ballot a unique identifying number (ballot ID numbers). Draw a random sample of ballot ID numbers, and retrieve the corresponding ballots.

2. **[Position-based sampling]** Give each ballot an implicit ballot ID equal to its position in a canonical listing of all ballot positions. Then proceed as with method (1).

These methods work well, and are guaranteed to produce random samples, assuming the counting involved in retrieving the ballots is perfect.

In practice, auditors use software, like Stark's website [22], which takes in a ballot manifest as input and produces the random sample of ballot ID numbers. In this software, it is typically assumed that sampling is done without replacement.

However, finding even a single ballot using these sampling methods can be tedious and awkward in practice. For example, given a random sample of ID numbers, one may need to count or search through a stack of ballots to find the desired ballot with the right ID or at the right position. Moreover, typical auditing procedures assume that there are no mistakes when finding the ballots for the sample. Yet, this seems to be an unreasonable assumption - if we require a sample size of 1,000 ballots, for instance, it is likely that there are a few "incorrectly" chosen ballots along the way, due to counting errors. In fact, Goggin et al. [6] have shown that counting is an imperfect technique. In the literature about RLAs, there is no way to correct for these mistakes.

6.2 Problem Definition

Our goal is to simplify the sampling process.

In particular, we want to define a general framework for compensating for "approximate sampling" in RLAs. Our framework of approximate sampling can be used to measure and compensate for human error rate while using ID-based or position-based sampling. Moreover, we also define a simpler approach for drawing a random sample of ballots which does not rely on counting at all. Our technique is simple and easy to iterate on and may be of particular interest when the stack of ballots to be drawn from is large. We define mitigation procedures to account for the fact that the sampling technique is no longer uniformly random.

The problem to be solved is:

> How can one select a single ballot (approximately) at random from a given stack of n ballots?

This section presents the "k-cut" sampling procedure for doing such sampling. The k-cut procedure does not need to know the size n of the stack, nor does it need any auxiliary random number generators or technology.

We assume that the collection of ballots to be sampled from is in the form of a stack. These may be ballots stored in a single box or envelope after scanning. One may think of the stack of ballots as being similar to a deck of cards. When the ballots are organized into *multiple* stacks, sampling is slightly more complex—see Chapter 11.

For now we concentrate on the single-stack case. We imagine that the size n of the stack is 25–800 or so.

The basic operation for drawing a single ballot is called "k-cut and pick," or just "k-cut." This method does k cuts then draws the ballot at the top of the stack.

To make a single cut of a given stack of n paper ballots:

- Cut the stack into two parts: a "top" part and a "bottom" part.
- Switch the order of the parts, so what was the bottom part now sits above the top part. The relative order of the ballots within each part is preserved.

We let t denote the size of the top part. The size t of the top part should be chosen "fairly randomly" from the set $[n] = \{0, 1, 2, \ldots, n-1\}$[1]. In practice, cut sizes are probably not chosen so uniformly; so in this paper we study ways to compensate for non-uniformity. We can also view the cut operation as one that "rotates" the stack of ballots by t positions.

An example of a single cut. As a simple example, if the given stack has $n = 5$ ballots:

$$\boxed{\text{A B C D E}},$$

where ballot A is on top and ballot E is at the bottom, then a cut of size $t = 2$ separates the stack into a top part of size 2 and a bottom part of size 3:

$$\boxed{\text{A B}} \ \boxed{\text{C D E}}$$

whose order is then switched:

$$\boxed{\text{C D E}} \ \boxed{\text{A B}}\,.$$

Finally, the two parts are then placed together to form the final stack:

$$\boxed{\text{C D E A B}}.$$

having ballot C on top.

Relative sizes We also think of cut sizes in relative manner, as a fraction of n. We let $\tau = t/n$ denote a cut size t viewed as a fraction of the stack size n. Thus $0 \leq \tau < 1$.

Choosing a cut size. We note that our work focuses on a simple version of the procedure, where the size of the cut is left solely to the judgment of the person making it. This leads to significant non-uniformity in the single-cut distribution.

However, we have also experimented with providing hints for the size of the cut. That is, before any cut is made, we use a random number generator to generate a

[1]A cut of size n is excluded, as it is equivalent to a cut of size 0.

number from 1 to 99. If the random number generator returns a value r, then the person making the cut tries to make the cut so the "top" part is about $r\%$ of the total stack.

We believe that these heuristics will provide a more uniformly random distribution, which could be used to find tighter bounds on the number of cuts required. However, this area still requires some more exploration. Thus, our data and proofs are based on the original version of k-cut without additional hints.

Iteration for k cuts. The k-cut procedure makes k successive cuts then picks the ballot at the top of the stack.

If we let t_i denote the size of the i-th cut, then the net rotation amount after k cuts is

$$r_k = t_1 + t_2 + \cdots + t_k \pmod{n} \ . \tag{6.1}$$

The ballot originally in position r_k (where the top ballot position is position 0) is now at the top of the stack. We show that even for small values of k (like $k = 6$) the distribution of r_k is close to $\mathcal{U}$.

In relative terms, if we define

$$\tau_i = t_i/n$$

and

$$\rho_k = r_k/n \ ,$$

we have that

$$\rho_k = r_k/n = \tau_1 + \tau_2 + \cdots + \tau_k \pmod{1} \ . \tag{6.2}$$

Drawing a sample of multiple ballots. To draw a sample of s ballots, our k-cut procedure repeats s times the operation of drawing without replacement a single ballot "at random." The s ballots so drawn form the desired sample.

Efficiency. Suppose a person can make six ("fairly random") cuts in approximately 15 seconds, and can count 2.5 ballots per second[2]. Then k-cut (with $k = 6$) is more efficient when the number of ballots that needs to be counted is 37.5 or more. Since batch sizes in audits are often large, k-cut has the potential to increase sampling speed.

For instance, assume that ballots are organized into boxes, each of which contains at least 500 ballots. Then, when the counting method is used, 85% of the time a ballot between ballot #38 and ballot #462 will be chosen. In such cases, one must count at least 38 ballots from the bottom or from the top to retrieve a single ballot. This implies that k-cut is more efficient 85% of the time.

This analysis assumes that each time we retrieve a ballot, we start from the top of the stack and count downwards. In fact, if we have to retrieve a single ballot from each box, this is the best technique that we know of. However, let us instead assume that we would like to retrieve t ballots in each box of n ballots These ballots are chosen uniformly at random from the box; thus, in expectation, the largest ballot position (the ballot closest to the bottom of the stack) will be $\frac{nt}{t+1}$. One possible way to retrieve these t ballots is to sort the required ballot IDs by position and retrieve them in order by making a single pass through the stack. This requires only counting $\frac{nt}{t+1}$ ballots in total to find all t ballots. Using our estimate that a person can count 2.5 ballots per second, this implies that each box will require $\frac{nt}{2.5(t+1)}$ seconds. Using k-cut, we will require 15 seconds per draw, and thus, $15t$ seconds in total.

This implies that k-cut is more efficient when

$$\frac{nt}{2.5(t+1)} > 15t$$
$$n > 37.5(t+1)\,.$$

Thus, if we require 2 ballots per box ($t = 2$), k-cut is more efficient in expectation when there are at least 113 ballots per box. When $t = 3$, then k-cut is more efficient in expectation when there are at least 150 ballots per box. Since the batch sizes

[2]These assumptions are based on empirical observations during the Indiana pilot audits.

in audits are large and the number of ballots sampled per box is typically quite small, we expect k-cut to show an increase in efficiency in practice. Moreover, as the number of ballots per box increases, the expected time taken by standard methods to retrieve a single ballot increases. With k-cut, the time it takes to select a ballot is *constant*, independent of the number of ballots in the box, assuming that each cut takes constant time.

Security We assume that the value of k is **fixed** in advance; you can not allow the cutter to stop cutting once a "ballot they like" is sitting on top.

Ballot Polling vs. Ballot Comparison Audits We suggest using k-cut primarily for ballot polling audits. Ballot comparison audits require comparing the paper ballot to its electronic interpretation. Thus, the order of the paper ballots usually needs to be maintained when performing a ballot comparison audit which makes the k-cut procedure trickier to implement and likely less efficient.

Chapter 7

Single-Cut Empirical Data and Analysis

In this chapter, we will explore the usability of our approximate sampling scheme by analyzing empirical data. Using this data, we will show that a single "cut" is noticeably non-uniform; in particular, we explore a few different models for our empirical distribution. Furthermore, we discuss metrics to measure how quickly the k-cut procedure converges to the uniform distribution, primarily focusing on the variation distance.

7.1 Empirical Results

We begin by observing that if an auditor could perform "perfect" cuts, we would be done. That is, if the auditor could pick the size t of a cut in a perfectly uniform manner from $[n]$, then one cut would suffice to provide a perfectly uniform distribution of the ballot selected from the stack of size n. However, there is no *a priori* reason to believe that, even with sincere effort, an auditor could pick t in a perfectly uniform manner.

As we previously suggested, we could use randomly generated "hints" to suggest approximately how large t should be. Another approach involves generating a random number r and weighing the ballots to remove approximately r ballots from the top.

In a similar flavor, we could generate r randomly and remove approximately r ballots from the top using a ruler and measuring the change in height of the stack. This procedure is commonly used in the finance industry[1].

In our experiments, we study the properties of the k-cut procedure for single-ballot selection, beginning with a study of the non-uniformity of selection for the case $k = 1$ and extending our analysis to multiple cuts. In our data set, the people making the cuts simply chose t as randomly as they could with no hints or other heuristics.

This section presents our experimental data on single-cut sizes. We find that in practice, single cut sizes (that is, for $k = 1$) are "somewhat uniform." We then show that the approximation to uniformity improves dramatically as k increases.

We had two subjects (Mayuri Sridhar and Ronald L. Rivest). Each author had a stack of 150 sequentially numbered ballots to cut. Marion County, Indiana kindly provided surplus ballots for us to work with. The authors made 1680 cuts in total. Table 7.1 shows the observed cut size frequency distribution.

If the cuts were truly random, we would expect a uniform distribution of the number of cuts observed as a function of cut size. In practice, the frequency of cuts was not evenly distributed; there were few or no very large or very small cuts, and smaller cuts were more common than larger cuts.

7.2 Model Fitting

Given the evident non-uniformity of the single-cut sizes in our experimental data, it is of interest to model their distribution. Such models allow generalization to other stack sizes, and support the study of convergence to uniformity with iterated cuts. In Figure 7-1, we can observe the probability density of the empirical distribution, compared to different models.

We let $\mathcal{E}$ denote the observed empirical distribution on $[n]$ of single-cut sizes, and let $\overline{\mathcal{E}}$ denote the corresponding induced continuous density function on $(0, 1)$, of

[1]Thank you to William Kresse for suggesting this technique during a conversation at the MIT Election Audit Summit

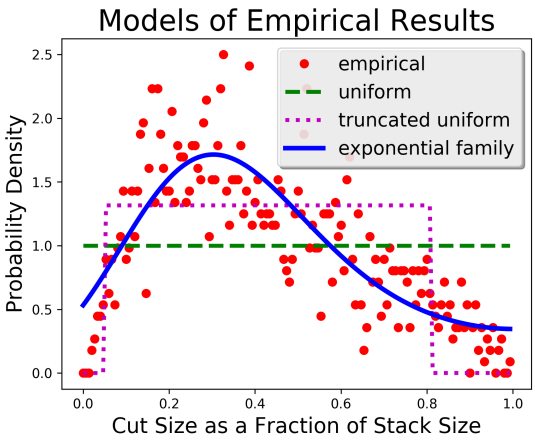


Figure 7-1: Models for Single Cut Sizes

Two models for cut sizes for a single cut, based on the data of Table 7.1. The horizontal axis is the size of the cut τ as a fraction of the size of the stack of ballots. The vertical axis is the probability density at that point. For reference, the uniform density $\overline{\mathcal{U}}$ (shown in green) has a constant value of 1. The red dots show the empirical distribution being modeled, which is clearly not uniform. The purple line shows our first model: the truncated uniform density $\overline{\mathcal{U}}(0.533, 0.813)$ on the interval $8/150 \leq \tau < 122/150$. This density has a mean absolute error of 0.384 compared to the empirical density, and a mean squared error of 0.224. The blue line shows our second model: the density function from the model $\overline{\mathcal{F}}$ of equation (7.2), which fits a bit better, giving a mean absolute error of 0.265 and a mean squared error of 0.114.

	0	1	2	3	4	5	6	7	8	9	Row Sum
0	0	0	0	2	3	5	5	6	10	7	38
10	10	6	11	12	16	10	11	16	12	16	120
20	21	22	7	18	25	15	25	21	18	16	188
30	16	23	15	20	19	19	15	16	20	20	183
40	18	17	22	24	12	17	17	20	25	28	200
50	16	13	17	17	17	20	14	16	27	13	170
60	15	17	14	13	14	14	13	13	17	16	146
70	10	9	8	10	14	16	14	21	25	11	138
80	13	11	11	5	14	14	14	8	15	12	117
90	13	9	17	19	10	6	14	6	2	4	100
100	12	8	10	8	5	10	6	11	9	9	88
110	4	9	9	8	4	9	6	9	7	9	74
120	10	7	6	5	4	6	8	5	6	3	60
130	4	4	8	4	6	0	4	6	2	4	42
140	1	3	2	4	0	2	3	0	0	1	16

Table 7.1: Empirical Single Cut Distribution
Empirical distribution of sizes of single cuts, using combined data from Mayuri Sridhar and Ronald L. Rivest, with 1680 cuts total. For example, ballot 3 was on top twice after one cut. Note that the initial top ballot is ballot 0.

relative cut sizes.

We consider two models of the probability distribution of cut sizes for a single cut.

The reference case (the ideal case) is **the uniform model**, where t is chosen uniformly at random from $[n]$ for the discrete case, or when τ is chosen uniformly at random from the real interval $(0, 1)$ for the continuous case. We denote these cases as $t \sim \mathcal{U}[n]$ or $\tau \sim \overline{\mathcal{U}}(0, 1)$, respectively.

We can define two different non-uniform models to reflect the observed data.

- **The truncated uniform model.** This model has two parameters: w (the least cut size possible in the model) and b (the number of different possible cut sizes). The cut size t is chosen uniformly at random from the set $[w, w + b] = \{w, w + 1, \ldots, w + b - 1\}$. We denote this case as $t \sim \mathcal{U}[w, w + b]$ (for the discrete version) or $\tau \sim \overline{\mathcal{U}}(w/n, (w + b)/n)$ (for the continuous version).

- An **exponential family** model. Here the density of relative cut sizes is modeled

as $\mathcal{F}(\tau) = \exp(f(\tau))$, where τ is the relative cut size and f is a polynomial (of degree three in our case).

Fitting models to data. We used standard methods to find least-squares best-fits for the experimental data of Table 7.1 to models from the truncated uniform family and from the exponential family based on cubic polynomials.

Fitted model - truncated uniform distribution. We find that choosing $w = 8$ and $b = 114$ provides the best least-squares fit to our data. This corresponds to a uniform distribution $t \sim \mathcal{U}[8, 122]$ or $\tau \sim \overline{\mathcal{U}}(0.00667, 0.813)$.

Fitted model - exponential family. Using least-squares methods, we found a model from the exponential family for the probability density of relative cut sizes for a single cut, based on an exponential of a cubic polynomial of the relative cut size τ.

The model defines

$$f(\tau) = -0.631 + 8.587\tau - 18.446\tau^2 + 9.428\tau^3 \tag{7.1}$$

and then uses

$$\overline{\mathcal{F}}(\tau) = \exp(f(\tau)) \tag{7.2}$$

as the density function of the exponential family function defined by f. We can see in Figure 7-1 that this seems to fit our empirical observations quite well.

7.3 Metrics for Non-Uniformity

As noted, the distribution of cut sizes for a single cut is noticeably non-uniform. Our proposed k-cut procedure addresses this by iterating the single-cut operation k times, for some small fixed integer k.

We assume for now that successive cuts are independent. Moreover, we assume that sampling is done with replacement, for simplicity. Using these assumptions, we provide computational results showing that as the number of cuts increases, the k-cut

procedure selects ballots with a distribution that approaches the uniform distribution for our empirical data, as well as for our fitted models. We compare by computing the variation distance of the k-cut distribution from $\mathcal{U}$ for various k. We also computed ϵ, the maximum ratio of the probability of any single ballot under the empirical distribution to the probability of that ballot under the uniform distribution minus one[2]. Our results are summarized in Table 7.2.

We can see that after six cuts, we get a variation distance of about 7.19×10^{-4}, for the empirical distribution, which is often small enough to justify our recommendation that six cuts being “close enough" for any RLA.

[2]In Section 10.3.1, we discuss why this value of ϵ is relevant

	Variation Distance			Max Ratio minus one		
k	$\mathcal{E}_k$	$\mathcal{U}_k[w, w+b]$	$\mathcal{F}_k$	$\mathcal{E}_k$	$\mathcal{U}_k[w, w+b]$	$\mathcal{F}_k$
1	0.247	0.24	0.212	1.5	0.316	0.707
2	0.0669	0.0576	0.0688	0.206	0.316	0.212
3	0.0215	0.0158	0.0226	0.0687	0.0315	0.0706
4	0.0069	0.00444	0.00743	0.0224	0.0177	0.0233
5	0.00223	0.00126	0.00244	0.00699	0.00311	0.00767
6	**0.000719**	**0.000357**	**0.000802**	**0.00225**	**0.00128**	**0.00252**
7	0.000232	0.000102	0.000264	0.000729	0.000284	0.000828
8	7.49×10^{-5}	2.92×10^{-5}	8.67×10^{-5}	0.000235	9.87×10^{-5}	0.000272
9	2.42×10^{-5}	8.35×10^{-6}	2.85×10^{-5}	7.59×10^{-5}	2.47×10^{-05}	8.95×10^{-5}
10	7.79×10^{-6}	2.39×10^{-6}	9.36×10^{-6}	2.45×10^{-5}	7.83×10^{-6}	2.94×10^{-5}
11	2.52×10^{-6}	6.86×10^{-7}	3.08×10^{-6}	7.9×10^{-6}	2.09×10^{-6}	9.67×10^{-6}
12	8.12×10^{-7}	1.97×10^{-7}	1.01×10^{-6}	2.55×10^{-6}	6.32×10^{-7}	3.18×10^{-6}
13	2.62×10^{-7}	5.64×10^{-8}	3.32×10^{-7}	8.23×10^{-7}	1.74×10^{-7}	1.04×10^{-6}
14	8.45×10^{-8}	1.62×10^{-8}	1.09×10^{-7}	2.66×10^{-7}	5.14×10^{-8}	3.43×10^{-7}
15	2.73×10^{-8}	4.63×10^{-9}	3.59×10^{-8}	8.57×10^{-8}	1.44×10^{-8}	1.13×10^{-7}
16	8.8×10^{-9}	1.33×10^{-9}	1.18×10^{-8}	2.77×10^{-8}	4.2×10^{-9}	3.71×10^{-8}

Table 7.2: k-Cut Convergence Rate

Convergence of k-cut to uniform with increasing k. Variation distance from uniform and ϵ-values for k cuts, as a function of k, for $n = 150$, where ϵ is one less than the maximum ratio of the probability of selecting a ballot under the assumed distribution to the probability of selecting that ballot under the uniform distribution. The second through seventh column headings describe probability distribution of single-cut sizes convolved with themselves k times to obtain the k-th row. Columns two and five give results for the distribution $\mathcal{E}_k$ equal to the k-fold iteration of single cuts that have the distribution of the empirical data of Table 7.1. Columns three and six gives results for the distribution $\mathcal{U}_k[w, b]$ equal to the k-fold iteration of single cuts that have the distribution $\mathcal{U}[8, 122]$ that is the best fit of this class to the empirical distribution $\mathcal{E}$. Columns four and seven gives results for the distribution $\mathcal{F}_k$ equal to the k-fold iteration of single cuts that have the distribution described in equations (7.1) and (7.2). The row for $k = 6$ is bolded, since we will show that with our mitigation procedures, 6 cuts is "close enough" to random.

Chapter 8

Convergence of k-Cut

In this chapter, we show that as the number of cuts gets very large, the k-cut procedure converges to the uniform distribution. That is, we prove this for the truncated uniform model of our single-cut data and argue that our proof will generalize for other models with minimal additional constraints. From here, we note that the rate of convergence is exponential which makes k-cut a promising candidate for an approximate sampling procedure.

8.1 Asymptotic Convergence

This claim is plausible, given the analysis of similar situations for continuous random variables. For example, Miller and Nigrini [15] have analyzed the summation of independent random variables modulo 1, and given necessary and sufficient conditions for this sum to converge to the uniform distribution.

For the discrete case, one can show that if once k is large enough that every ballot is selected by k-cut with some positive probability, then as k increases the distribution of cut sizes for k-cut approaches $\mathcal{U}$. We will prove this claim for the truncated uniform model for k-cut.

In this section, we assume that each cut follows the truncated uniform model $t \sim \mathcal{U}[8, 122]$, discussed in the previous chapter. Under this assumption, we can show that the k-cut procedure tends to the uniform distribution as k goes to infinity.

Theorem 2 *We assume that we have a stack of n ballots, where each "cut" in the stack of ballots is independent and follows the $\mathcal{U}[w, w+b-1]$ model, with $w = 8$ and $b = 115$. If this holds, as k goes to infinity, the probability of any ballot i being on the top of the stack approaches $\frac{1}{n}$, where n is the number of ballots in the stack.*

Proof: To see this, we can model the k-cut procedure as a random walk on a graph. In particular, we can construct a graph $G = (V, E)$, where the vertices correspond to a specific ordering of the deck of ballots. We note that there are exactly n possible orderings of the deck that can be reached through iterations on the k-cut procedure. That is, we never change the circular order of the ballots but instead just choose a subset of ballots to place on top. The deck A, B, C can only be arranged as one of $[[A, B, C], [B, C, A], [C, A, B]\}$. An edge (u, v) in the graph exists if we can move from arrangement u to arrangement v in a single cut.

We can assume that the person choosing "approximately at random" will always choose at least w ballots to remove from the top, and at most $w + b - 1$, but chooses uniformly within this interval per the $\mathcal{U}[w, w + b - 1]$ model. Thus, the graph is not fully connected. Each edge (u, v) in the graph has weight $\frac{1}{b}$, since we have b possible cuts we can make from any state u, and we choose a cut uniformly at random.

We note that each possible ordering in our deck is a state in a Markov chain, where the probability of arriving at some state i in the next step depends only on our current state. Thus, making k cuts in the deck corresponds to taking a length-k random walk on this Markov chain. We would like to show that the probability of being at any state u after k steps, as k tends to infinity, approaches $\frac{1}{n}$. To do this, we can show that the uniform distribution is a stationary distribution on this Markov Chain.

Without loss of generality, we will prove that the uniform distribution is stationary for $w = 0$. That is, this proof assumes a model of $\mathcal{U}[0, b-1]$. However, since changing the value of w adds a fixed bias to each cut, if the uniform distribution is stationary with $w = 0$, we know that it will be stationary for any model of the form $\mathcal{U}[w, w+b-1]$.

First, we note that the Markov Chain is strongly connected, for any $b \geq 1$, since we can reach any state v from a state u by making $|v - u|$ cuts of length 1.

Moreover, we note that the graph is aperiodic, since there are self-loops. That is, in our model of $\mathcal{U}[0, b-1]$, the cut removes 0 ballots from the top with some probability. This implies that the Markov chain has a unique stationary distribution.

To show that the uniform distribution is stationary, we assume that we have reached the uniform distribution at some timestep t. Then, at time $t+1$, we want to show that the distribution over the vertices is still uniform. To see this, we note that the probability of being at some state u at time $t+1$ can be represented as

$$Pr_{t+1}(u) = \sum_{v,(v,u)\in E} Pr_t(v) \cdot f(v,u)\,,$$

where $f(v,u)$ is the probability of transitioning from v to u. We note that, by hypothesis, $Pr_t(v) = \frac{1}{n}$ for all v. Since there are b possible neighbors for u, and $f(v,u) = \frac{1}{b}$ for all v,

$$Pr_{t+1}(u) = \frac{1}{n} \sum_{v,(v,u)\in E} f(v,u)\,,$$
$$Pr_{t+1}(u) = \frac{b}{nb} = \frac{1}{n}\,.$$

Thus, we have shown that the uniform distribution is stationary. Since the Markov Chain is ergodic, we know as k tends to infinity, the vector of state probabilities converges to the uniform distribution.

■

We note that, regardless of the model we use, if we assume each cut is made independently, we can prove that the uniform distribution is stationary for our Markov Chain. In particular, we note any row i of the transition matrix is a simple rotation of row $i-1$. This implies that any column must sum to 1, since the entries in the column are simply a permutation of the entries in a row. This shows that the matrix is doubly stochastic and the uniform distribution is stationary.

We note that this is not quite enough to prove that k-cut will converge for any distribution - for instance, if we only made cuts of size 0, then although the uniform distribution is stationary, we will not converge to it. However, a simple and sufficient

condition for our distribution to converge to the uniform distribution is that there exists some i such that we can transition from state 0 to state i with non-zero probability and transition from state 0 to state $i+1$ with non-zero probability. We claim that this is plausible for our k-cut procedure.

Furthermore, assuming that these conditions hold, the Markov chain will converge to its stationary distribution at an exponential rate from Theorem 4 of Arora's lecture[1].

8.2 Key Takeaways

We have shown that for large values of k, the k-cut procedure will converge to the uniform distribution. Intuitively, this shows that the k-cut procedure is a good substitute for uniformly random sampling. Moreover, theoretically, we have argued that this convergence happens at an exponential rate. Empirically, we have shown that, even for small values of k, convolutions based on our empirical data also get very close to the uniform distribution in terms of variation distance. Thus, we know that our approximate sampling distribution is quite uniform. However, we still need to design mitigation procedures to account for the residual non-uniformity to make k-cut compatible with RLAs.

Chapter 9

Sample Tally Mitigation

This chapter discusses a simple mitigation technique for dealing with residual non-uniformity by sample tally adjustment. As we have shown, for small values of k, k-cut is quite close to the uniform distribution. We will show how to compensate for the risk associated with the left-over non-uniformity by adjusting the sample tallies for the winner and all possible losers. However, we also discuss the drawbacks of using this technique which is very specific to RLAs.

9.1 Sample Tally Mitigation Overview

In this section, we will describe the set up for introducing approximate sampling into an RLA for a plurality election.

In particular, as described by Lindeman et al. [11], an RLA with a risk limit of α guarantees that with probability at least $(1 - \alpha)$ the audit will correct the reported outcome if it is incorrect. We want to show that we can maintain this risk-limiting property of RLAs, while introducing approximate sampling techniques.

There are two main assumptions that we use throughout this section. First of all, we assume that the audit stopping condition for an RLA is based only on the latest sample tally. Moreover, we assume that the audit stopping condition is "monotonic." In particular, we assume that moving ballots in the sample tally from the reported winner to some other candidate makes the audit more conservative. That is, we

assume that the sample tally (s_i, s_j), where Candidate i is the reported winner, is always more likely to satisfy the audit stopping condition than a sample tally $(s_i - c, s_j + c)$ for any $c > 0$. In Chapter 10, we will discuss mitigation strategies that do not require these assumptions.

In particular, the formal problem setup can be defined as follows:

- There are m candidates and n ballots.
- We represent the tally of the collection of ballots in the contest as an m-dimensional vector $X = (x_1, x_2, \ldots, x_m)$, where

$$\sum_{i=1}^{m} x_i = n \,,$$

  and X_i denotes the total number of votes cast for candidate i in the contest.

- Under uniform sampling, we expect that a ballot for candidate i will be chosen with probability

$$\frac{x_i}{n}.$$

  With our k-cut technique, a ballot for any candidate i is chosen with probability in the interval

$$[\frac{x_i}{n} - \delta, \frac{x_i}{n} + \delta],$$

  where δ depends on the number of cuts we choose. In practice, we use the variation distance between the uniform distribution and our empirical distribution after k cuts as our value for δ.

- We represent the collection of ballots in our sample as an m-dimensional vector s, where s_i is the number of votes in the sample for candidate i and

$$\sum_{i=1}^{m} s_i = z \,.$$

- For simplicity, we analyze the case where sampling is done with replacement.

A risk-limiting audit takes as input a "risk-limit" α (like 0.05), and ensures that if a reported contest outcome is incorrect, then this error will be detected and corrected with probability at least $1-\alpha$ [11]. To guarantee that a sample tally does not satisfy the audit stopping condition only because of approximate sampling procedures, we can adjust all the sample margins in a conservative manner, in order to maintain the risk limiting properties of an RLA.

To do this, we can calculate an upper bound on the number of extra ballots that are chosen for the reported winner, only due to approximate sampling, with high probability. We denote this value as d. Then, when trying to compute whether the risk limit between a pair of candidates, 1 and j is satisfied, we can "adjust" the sample tally. That is, we assume that the sample tally is actually $s_1 - d$ and $s_j + d$. With this sample tally, if the stopping condition between candidates 1 and j is still satisfied, then we know it would have been satisfied under uniformly random sampling as well.

We note that this sample tally adjustment procedure is very specific to the RLA procedure, defined by Lindeman et al. [11]. In this procedure, the stopping condition depends on the margin between the winner and the runner-up and satisfies the "monotonic" condition we defined above. Thus, adjusting the sample tally this way is guaranteed to be safe for this procedure. There may be other risk-limiting audit procedures which do not necessarily satisfy this condition. In these procedures, the sample tally mitigation procedure will not necessarily work. In Chapter 10, we discuss more general mitigation procedures for approximate sampling.

We let $\mathcal{G}$ denote the actual (approximate) probability distribution over $[n]$ from the sampling method chosen for the audit. In particular, this is the distribution over $[n]$ for the chosen ballot after k cuts are made. As before, we let $\mathcal{U}$ denote the uniform probability distribution over $[n]$. We can show that if $\mathcal{G}$ and $\mathcal{U}$ are quite close, then the value of d is likely to be small and the sample tallies do not require too much adjustment. In fact, we can use Theorem 3 to calculate the maximum required adjustment, with high probability.

Theorem 3 *If a ballot for any candidate i is chosen with probability in the real interval $[\frac{x_i}{n} - \delta, \frac{x_i}{n} + \delta]$, for any $\epsilon > 0$, the sample margin between any two candidates i*

and j requires a tally adjustment of at most d ballots, with probability $1-\epsilon$ for

$$d \geq \sqrt{-0.5z \ln(\epsilon/2)} + z\delta \,.$$

We can define a tally adjustment as moving d ballots from candidate i's sample tally to candidate j's sample tally. After this adjustment, with probability $1-\epsilon$, the resulting sample tally will satisfy the audit stopping condition between candidates i and j if and only if the audit stopping condition would have been satisfied under uniformly random sampling.

Proof: Without loss of generality, we consider candidates 1 and 2, where the reported winner is candidate 1. In reality, candidate 1 has x_1 ballots in the pool of all cast votes and candidate 2 has x_2 ballots, where $x_1 + x_2 \leq n$. However, in the worst-case situation, a ballot for candidate 1 is chosen with probability $\frac{x_1}{n} + \delta$ and a ballot for candidate 2 is chosen with probability $\frac{x_2}{n} - \delta$.

If we are sampling uniformly at random, we are drawing a ballot uniformly at random from a box containing x_1 ballots for candidate 1 and x_2 ballots for candidate 2.

We can model the use of approximately random sampling as drawing a ballot uniformly at random from a box containing $x_1 + n\delta$ ballots for candidate 1 and $x_2 - n\delta$ ballots for candidate 2. In this case, since we have a higher probability of drawing ballots for candidate 1, compared to uniformly random sampling, we have a higher probability of generating a sample which has more ballots for candidate 1. Since candidate 1 is the reported winner, these samples are more likely to satisfy the audit stopping condition early, which could violate the risk-limiting properties of the RLA. We would like to fix this by adjusting our model to have at least as many ballots in the pool for candidate 2 as the candidate would have under uniformly random sampling.

We can compensate for the approximate sampling by modeling some of the ballots for candidate 1 as actually being votes for candidate 2. In particular, we want to create a model population of cast votes with x_1 ballots for candidate 1, $x_2 - n\delta$ ballots for candidate 2 and $n\delta$ blank ballots. Every time we draw a blank ballot, we interpret it

as a vote for candidate 2. This model is equivalent to having x_2 ballots for candidate 2 and x_1 ballots for candidate 1 in the pool. That is, procedurally, each time we draw a ballot for candidate 1, with probability $\frac{n\delta}{x_i+n\delta}$, we ignore the vote written on the ballot and interpret it as a vote for candidate 2. This guarantees that we draw a ballot for candidate 1 and interpret it for candidate 1 with probability $\frac{x_i}{n}$ and we draw a ballot for candidate 1 and interpret it as a blank ballot with probability δ. (See Banuelos et al. [3] for a related approach to a similar problem.)

If we follow this procedure for every ballot in the sample, we are making the margin between candidates 1 and 2 at least as safe as it would have been under a uniformly random sampling procedure. That is, our model population has at least as many ballots for the runner-up as the uniform model has.

Given a sample size of z, we want to bound the number of blank ballots we will see under the approximate sampling scheme. Each draw from the collection cast votes follows a Bernoulli distribution, where each ballot has probability δ of being a blank ballot.

Thus, after z draws, we can define a random variable β as the number of blank ballots we see. In expectation, we will see $z\delta$ blank ballots. Following the binomial distribution formulas, β have a variance of $z\delta(1-\delta)$. We can apply Hoeffding's formula [8] on β to bound the maximum number of ballots we will see, with probability at least $1-\epsilon$. This becomes

$$\begin{aligned}\Pr[\beta > d] &\leq Pr[|\beta - z\delta| > d - z\delta] \\ &< 2\exp\left(\frac{-2(d-z\delta)^2}{z}\right).\end{aligned}$$

For this to be less than ϵ, we require

$$d \geq \sqrt{-0.5z\ln(\epsilon/2)} + z\delta\,.$$

■

Thus, we can formalize the mitigation process for plurality elections by adjusting

all pairs of sample tallies.

We assume that the auditing procedure draws a sample s of size z. Suppose that candidate 1 is the reported winner. The auditing procedure will make a worst-case assumption about the behavior of the probability distribution and "correct" the tallies before computing any statistics from the sample tally.

To "correct" the sample tally, we can move d ballots from s_1 (the sample tally for the reported winner) to some other candidate, Candidate j. We know, with probability $1 - \epsilon$, this is the most the sample tallies have been changed, and we change them in the most adversarial way possible - all samples are moved from the candidate 1 to candidate j. After this adjustment, we can again calculate the risk limit between candidate 1 and candidate j.

Thus, in Stark's RLA procedure, if the risk limit between these two candidates is satisfied, we know that it would have been satisfied under uniform sampling with probability at least $1 - \epsilon$.

After we make a decision about the risk limit between candidates 1 and j, we re-adjust the sample tallies back to the original values. We repeat this process for every possible pair of candidates, of the form $(1, j)$ where candidate 1 is the reported winner and $j \in [2, m]$.

Moreover, to compensate for the ϵ probability that the sample tallies are changed by more than d ballots, we change the risk limit of the audit. In particular, if the original RLA under uniform random sampling had a risk limit of α, then an RLA with approximate sampling should have a risk limit of $\alpha - \epsilon$.

9.2 Sample Tally Mitigation Empirical Analysis

The key intuition for k-cut is the notion that if the ballots are sampled in a nearly-uniform manner, then the audit will not be affected much—tallies and margins in the sample are only slightly different than what they would have been if sampling had been performed under a uniform distribution.

This section provides computational results supporting this intuition. We show

that a tight upper bound on the variation distance (compared to uniform) implies that sample tallies will not change much, using Theorem 3.

9.2.1 Case Study: Truncated Uniform Model

This section discusses the maximum change in sample tallies, assuming that each cut follows the truncated uniform model. That is, we will assume that each cut is made using the distribution of $\mathcal{U}[8, 122]$, with $n = 150$, as we had discussed in Chapter 7.

Then, using this model, we can calculate the maximum change in sample tallies given the corresponding variation distances, as seen in Table 9.1 below. Again, we note that this represents the maximum adjustment required between the reported winner and any other runner-up. In our mitigation procedure, we would need to adjust the sample tallies for every pair of the form $(1, j)$, where candidate 1 is the reported winner and j ranges from 2 to m. First, we calculate the maximum change in sample tally, with 99% probability for a fixed sample size of 100 ballots. This assumes that the stack contains 150 ballots, and each cut chooses a ballot uniformly at random from the 8th to the 121st ballot in the stack (inclusive). All draws are made with replacement.

Numberof Cuts	Max Change in Sample Tally
1	34
2	12
3	5
4	3
5	1
6	1
7	1
8	0
9	0

Table 9.1: Max Change in Sample Tally (Truncated Uniform Model) for Varying k
If each cut follows distribution $\mathcal{U}[w, w+b]$ with $w = 8$, $b = 114$, $\epsilon = 0.01$, and $n = 150$, the maximum change in sample tally between any runner-up and the reported winner due to approximate sampling. The sample size is fixed at 100 ballots.

Thus, we can see that when we use five cuts the maximum change in sample tally

due to approximate sampling is 1 ballot with 99% probability. Thus, we know that the margin between the reported winner and any runner-up is increased by at most 2 ballots due to approximate sampling.

Furthermore, as we increase the sample sizes, the changes in sample tallies remains quite small. For instance, if we choose $k = 5$, we can analyze the maximum change in sample tally, with 99% probability, as the sample sizes increases, as seen in Table 9.2.

We note that, here, the sample sizes that we choose are much larger than n, which we chose to be 150. The value of n here is the number of ballots in each *batch* that we are sampling from. However, often, in RLAs, there are ballots sampled from many different batches. That is, it is quite common to require 1-2 ballots from each stack of 150 ballots. However, the entire population being audited can consist of many such stacks. Thus, our required sample size might be much greater than 150. We discuss this process in more detail in Chapter 11.

Sample Size	Max Change in Sample Tally
100	1
250	2
500	3
1000	4
2000	7
5000	13

Table 9.2: Max Change in Sample Tally (Truncated Uniform Model) for Varying Sample Size
If each cut follows distribution $\mathcal{U}[w, w+b]$ with $w = 8$, $b = 114$, $\epsilon = 0.01$, and $n = 150$, the maximum change in sample tally for any candidate. This assumes that we use five cuts and varying sample sizes.

The maximum change in sample tally stays quite small even for large sample sizes, like 1,000 ballots. This implies that if each cut independently follows the $\mathcal{U}[w, w + b]$ distribution, then five cuts will be enough to provide a distribution that is close enough to the uniform distribution.

9.2.2 Case Study: Empirical Distribution

In practice, we know that the actual single-cut distribution is not quite as uniform as the truncated model. Thus, we can also calculate the maximum required change in sample tally, with high probability, if our cuts follow the empirical distribution.

First, we choose a fixed sample size of 100. We want to see the maximum change in sample tally, with 99% probability, using the empirical single-cut distribution $\mathcal{E}$, as described in Table 9.3.

Number of Cuts	Max Change in Sample Tally
1	35
2	13
3	6
4	3
5	2
6	1
7	1
8	0
9	0

Table 9.3: Empirical Max Change in Sample Tally for Varying k
Based on the empirical single-cut distribution $\mathcal{E}$, the maximum change in sample tally between any two candidates, with $\epsilon = 0.01$, a sample size of 100, and $n = 150$, for varying number of cuts.

Based on the empirical distribution, if we choose a fixed number of cuts, we can see how the maximum sample tally adjustment increases with the size of the sample. From the above data, we choose $k = 6$, since it has a small change in sample tally. This is described in Table 9.4.

Again, we can see that for reasonably small sample sizes, up to 1,000 ballots, the required sample tally adjustments remain quite small.

9.3 Sample Tally Mitigation Drawbacks

The sample tally mitigation procedure requires small adjustments to the sample tallies in order to make approximate sampling work with RLAs. However, there are a few

Sample Size	Max Change in Sample Tally
100	1
250	2
500	2
1000	3
2000	5
5000	9

Table 9.4: Empirical Max Change in Sample Tally for Varying Sample Size
Based on the empirical distribution, with $n = 150$, $k = 6$, $\epsilon = 0.01$, and varying sample sizes, the maximum change in sample tally between any two candidates.

drawbacks to using this procedure.

First, we note that this procedure makes a few assumptions about the nature of the audit itself. For instance, we rely on the fact that the audit stopping condition relies on the margins between candidates and design a specific mitigation procedure that assumes the audit is monotonic.

Moreover, we note that the mitigation procedure relies on a plurality election. It is not straightforward to modify this to voting methods like ranked-choice voting. We want to design a simpler mitigation procedure, which does not rely on the internals of the voting technique used.

Chapter 10

General Mitigation Procedures

In this chapter, we outline general mitigation procedures to use with approximate sampling for any risk limiting audit. In particular, we define a simple procedure which involves decreasing the risk limit of the RLA to account for mistakes due to approximate sampling. We prove a simple bound on how much risk limit adjustment is required and make a recommendation of $k = 10$ cuts for normal sample sizes. Then, we prove a tighter bound for the required adjustment and make a recommendation of $k = 6$ cuts for use in practice.

10.1 Overview of Risk Limit Adjustment

This section proves a very general result: for auditing an arbitrary contest (not necessarily a plurality contest), we show that *any* audit system can be adapted to work correctly with approximate sampling, specifically with the k-cut method, if k is large enough. This applies in particular to risk-limiting audits.

We let $\mathcal{G}$ denote the sampling distribution used for the audit; that is, $\mathcal{G}$ denotes the single-ballot sampling distribution. In our analysis, we will define $\mathcal{G}$ as the distribution produced by k-cut.

We expect that if k is sufficiently large, the resulting distribution of k-cut sizes will be so close to uniform that any statistical procedure cannot efficiently distinguish between the two distributions. That is, we want to choose k to guarantee that $\mathcal{U}$

and $\mathcal{G}$ are close enough so that any statistical procedure will behave very similarly on samples from each.

Previous work done by Baignères et al. [2] shows that there is an optimal distinguisher between two finite probability distributions, which depends on the KL-Divergence between the two distributions. We follow a similar model to this work; however, we develop a bound based on the variation distance between the two distributions.

10.1.1 General Statistical Audit Model

We construct the following model, summarized in Figure 10-1.

We define δ to be the variation distance between $\mathcal{G}$ and $\mathcal{U}$. We can find an upper bound for δ empirically, as seen in Table 7.2. If $\mathcal{G}$ is the distribution of k-cut, then by increasing k we can make δ arbitrarily small.

The audit procedure may require a sample of some given size z. We assume that all audits behave deterministically. That is, given the same sample of ballots, the audit procedure returns the same outcome every time.

When we are sampling each ballot from the uniform distribution, we denote the probability distribution over all possible size-z samples as $\mathcal{U}^z$. When we are sampling from $\mathcal{G}$, we denote the probability distribution over all possible size-z samples as $\mathcal{G}^z$. We do not assume that successive draws are independent.

Given the size z sample, the audit procedure can make a decision on whether to accept the reported contest result, escalate the audit, or declare an upset.

Without loss of generality, we will focus on the probability that the audit decides to accept the reported contest result, since it is the case where approximate sampling may affect the risk-limiting properties of an audit. Given a variation distance of δ, we show that if $\mathcal{G}$ and $\mathcal{U}$ are sufficiently close (that is, if k is large enough when using k-cut), then the difference between p and p' is extremely small.

For RLAs, we can simply decrease the risk limit α by $|p' - p|$ (or an upper bound on this value) to account for the difference. We would like to find a tight upper bound for $|p' - p|$.

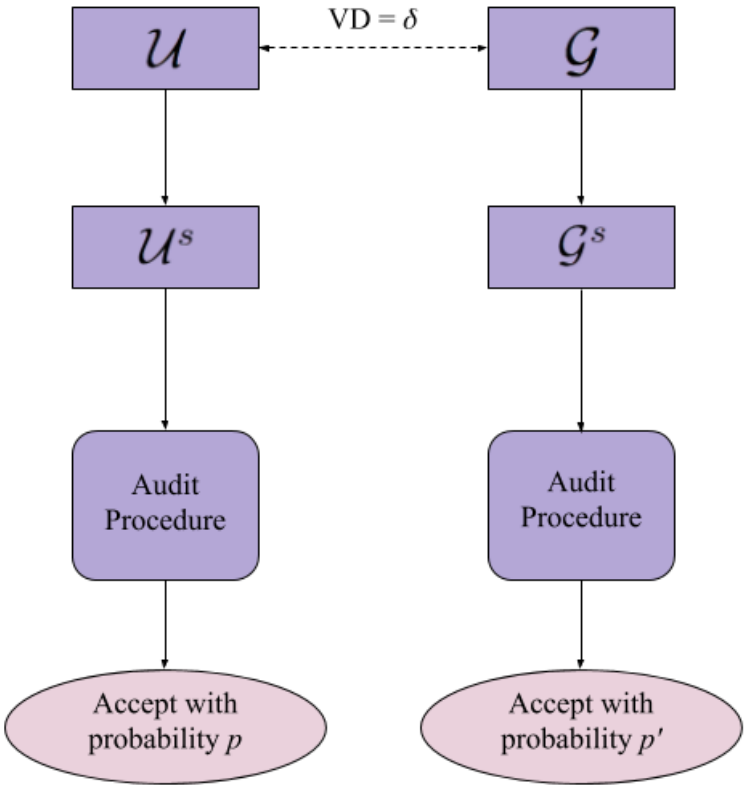


Figure 10-1: General Statistical Audit Model Overview

Overview of uniform vs. approximate sampling effects, for any statistical auditing procedure. The audit procedure can be viewed as a distinguisher between the two underlying distributions. If it gives significantly different results for the two distributions, it can thereby distinguish between them. That is, the audit is a statistical procedure that takes in input from two underlying distributions. If the distributions are close enough, we could use the audit to determine whether the original distribution was uniform or not. However, if p and p' are extremely close, then the audit cannot be used as a distinguisher.

10.2 A Loose Bound for Risk Limit Adjustment

This section provides a simple upper bound on the change in acceptance probability due to approximate sampling and provide empirical support for recommending $k = 10$ cuts. This section is primarily provided for intuition and can be skipped; the next section will provide a tighter bound to recommend $k = 6$ cuts for use in practice.

Lemma 2 *We denote the uniform distribution over ballots as $\mathcal{U}$, the approximate sampling distribution over ballots as $\mathcal{G}$, and the variation distance between $\mathcal{U}$ and $\mathcal{G}$ as δ. Then, for any ballot B in a stack of n ballots,*

$$\frac{\Pr[\textit{ballot } B \textit{ selected} \mid \mathcal{G}]}{\Pr[\textit{ballot } B \textit{ selected} \mid \mathcal{U}]} \leq (1 + \epsilon)\,,$$

for $\epsilon = n\delta$.

Proof: We know, from the definition of variation distance that

$$\Pr[\text{ballot } B \text{ selected} \mid \mathcal{G}] - \Pr[\text{ballot } B \text{ selected} \mid \mathcal{U}] \leq \delta\,.$$

Simple algebra yields that

$$\begin{aligned}\frac{\Pr[\text{ballot } B \text{ selected} \mid \mathcal{G}]}{\Pr[\text{ballot } B \text{ selected} \mid \mathcal{U}]} &\leq \frac{\delta}{\Pr[\text{ballot } B \text{ selected} \mid \mathcal{U}]} + 1\,,\\ &= 1 + n\delta\end{aligned}$$

We can then define

$$\epsilon = n\delta,$$

to guarantee

$$\frac{\Pr[\text{ballot } B \text{ selected} \mid \mathcal{G}]}{\Pr[\text{ballot } B \text{ selected} \mid \mathcal{U}]} \leq 1 + \epsilon\,.$$

■

Now that we have shown that the probability of choosing any specific ballot B is affected by a multiplicative factor of at most $(1 + \epsilon)$, we can calculate the change in probability of drawing a particular size-z sample.

Theorem 4 *If the probability of choosing any sample of size* $= 1$ *is affected by a multiplicative factor of at most* $(1+\epsilon)$*, then the max multiplicative factor for samples of size* $z > 1$ *is at most*

$$(1+\epsilon)^z \ .$$

Proof: The sample is selected sequentially, ballot by ballot. Each ballot drawn is selected with probability not more than $(1+\epsilon)$ greater than its probability of being selected under the uniform distribution.

The possible lack of independence between successive draws does not affect this argument.

■

We note that the $(1+n\delta)^z$ bound is not always tight. In Table 7.2, we find tighter multiplicative bounds for ϵ, rather than $n\delta$. However, assuming that we have some bound on the ratio of probabilities for a single ballot due to approximate sampling, we can calculate an upper bound on the probability that an audit accepts the reported result due to approximate sampling.

Theorem 5 *Denote the uniform distribution over ballots by* $\mathcal{U}$ *and the approximate sampling distribution over ballots by* $\mathcal{G}$*. Denote the multiplicative probability increase for a single ballot between* $\mathcal{U}$ *and* $\mathcal{G}$ *as* $(1+\epsilon)$*. This implies for any possible audit outcome* O*, after the audit procedure sees a sample of* s *ballots,*

$$\Pr[O|\mathcal{G}, z] - \Pr[O|\mathcal{U}, z] \leq (1+\epsilon)^z - 1 \, .$$

Proof: First, we note that under our assumption that the audit behaves deterministically, it returns the same outcome every time for a given set of inputs. Thus, the probability of seeing an outcome O is equivalent to the probability of seeing any input $\pi \in \Pi$ where Π is defined as the set of inputs that produce outcome O.

This implies that

$$\frac{\Pr[O \mid \mathcal{G}, z]}{\Pr[O \mid \mathcal{U}, z]} = \frac{\Pr[\pi \in \Pi \mid \mathcal{G}, z]}{\Pr[\pi \in \Pi \mid \mathcal{U}, z]} = \frac{\sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{G}, z]}{\sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{U}, z]}.$$

However, we note that any given input π to the audit procedure is a length-z sequence. If we define the probability of generating a particular sequence π under the uniform distribution as p_π, then we know that the probability of generating the sequence under the approximate distribution is at most $(1+\epsilon)^z p_\pi$, by Theorem 4.

This implies that

$$\begin{aligned}\frac{\Pr[O \mid \mathcal{G}, z]}{\Pr[O \mid \mathcal{U}, z]} &= \frac{\sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{G}, z]}{\sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{U}, z]} \\ &\leq \frac{(1+\epsilon)^z \sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{U}, z]}{\sum_{\pi \in \Pi} \Pr[\pi \mid \mathcal{U}, z]} \\ &= (1+\epsilon)^z .\end{aligned}$$

Thus, we can conclude that

$$\begin{aligned}\Pr[O \mid \mathcal{G}, z] - \Pr[O \mid \mathcal{U}, z] &\leq ((1+\epsilon)^z - 1) \Pr[O \mid \mathcal{U}, z] \\ &\leq (1+\epsilon)^z - 1 ,\end{aligned}$$

as required. ∎

Using our empirical data, from Table 7.2, after $k = 10$ cuts, we have $\epsilon = 2.45 \times 10^{-5}$. If we choose a sample size of 500 ballots, we can calculate the maximum change in probability of a given outcome, using the previous theorem, to get

$$\Pr[O|\mathcal{U}, z] - \Pr[O|\mathcal{U}_k[w, b], z] \leq 0.01233 .$$

This analysis shows that, regardless of the details of the auditing procedure, the chances of accepting an audit increases by at most 1.2% due to approximate sampling. For an risk-limiting audit procedure, we can compensate for this by reducing the

risk limit by 1.2%. We note that this generalizes easily to other statistical audit procedures. For instance, for Bayesian audits, we can similarly reduce the upset probability limit by 1.2%. That is, since we are bounding the change in probability of any outcome, we know that the probability of a deterministic Bayesian audit accepting an outcome is increased by at most 1.2% due to the approximate sampling. Thus, we decrease the upset probability by 1.2% to account for these "extra" successes which are simply due to sampling.

10.2.1 Empirical Mitigation by Adjusting Risk Limit

As described above, approximate sampling based on k-cut is compatible with deterministic risk-limiting auditing procedure if we choose k large enough.

We showed that, based on our empirical data, $k = 10$ gave a reasonably small change in the probability of accepting. This change can be compensated for by reducing the risk limit by the same amount.

To analyze how much risk limit adjustment is required, we first calculated the exact value of ϵ for $\mathcal{E}_k$ and the uniform distribution.

For simplicity, we chose $z = 500$ here. With these parameters, we calculated the maximum change in probability of picking a certain outcome. The results are shown in Table 10.1.

In practice, from the empirical data, if nine cuts are done, then we can compensate by decreasing the risk limit by 3.9%. If we do ten cuts, we can compensate by decreasing the risk limit by 1.2%. Once we do more than ten cuts, the change in risk limit is pretty negligible. However, this is for a fixed sample size of at most 500 ballots.

Based on these values, we can recommend doing $k = 10$ cuts. If we choose ten cuts, we can also look at how the probability of any outcome is changed for varying sample sizes. These results are shown in Table 10.2.

From this data, we can see that for sample sizes up to 1000 ballots, the maximum change in probability remains quite small. In these cases, we can compensate by appropriately adjusting the risk limit.

Number of Cuts	Max Change in Probability of an event O
1	1
2	1
3	1
4	1
5	32.6
6	2.12
7	0.440
8	0.125
9	0.039
10	0.012
11	0.0040
12	0.0012
13	4.11×10^{-4}
14	1.32×10^{-4}
15	4.28×10^{-5}

Table 10.1: Max Change in Probability (Loose Bound) for Varying k
Maximum change in probability of any outcome due to approximate sampling, in any risk-limiting audit procedure, based on the model $\mathcal{E}$ from the empirical data from Table 7.1. This assumes a maximum sample size of 500, with $n = 150$, and varies the number of cuts.

Max Sample Size	Max Change in Probability
100	0.00245
200	0.0049
500	0.0123
1000	0.0248
2500	0.0631
5000	0.130

Table 10.2: Max Change in Probability (Loose Bound) for Varying Sample Size
Maximum change in probability, of any outcome, due to approximate sampling, in any election procedure, based on the empirical data from Table 7.1. This assumes $n = 150$, $k = 10$ and varies the maximum sample sizes.

10.3 Tighter Bounds for Risk Limit Adjustment

This section provides a more complex bound on the change in probability of an audit outcome due to approximate sampling and provides empirical support for recommending $k = 6$ cuts.

We assume an auditing procedure $\mathbb{A}$ that accepts samples and outputs "accept" or "reject." We assume that each sample is a sequence of ballots. Each ballot is represented by a unique ballot ID. We model approximate sampling as providing $\mathbb{A}$ samples from a distribution $\mathcal{G}$. For our analysis, we use the empirical distribution of cuts given in Table 10-1. For uniform sampling we provide $\mathbb{A}$ samples from $\mathcal{U}$.

We show that the probability that $\mathbb{A}$ accepts an outcome incorrectly given samples from $\mathcal{G}$ is not much higher than the probability that $\mathbb{A}$ accepts that outcome given samples from $\mathcal{U}$. We let $\mathbb{B}$ denote the set of ballots that we are sampling from.

Theorem 6 *Given a fixed sample size z and the variation distance δ between the actual approximate sampling distribution $\mathcal{G}$ and the uniform distribution $\mathcal{U}$, the maximum change in probability that $\mathbb{A}$ returns "accept" due to the use of approximate sampling is at most*

$$\epsilon_1 + (1 + n\delta)^{z'} - 1\,,$$

where z' is the maximum number of "successes" seen in z Bernoulli trials with probability at least $1 - \epsilon_1$, where each trial has a success probability of δ, .

Proof: We define z as the number of ballots that we pull from the set of cast ballots before deciding whether or not to accept the outcome of the election. Based on our sampling technique, we draw z ballots, one at a time, from $\mathcal{G}$ or from $\mathcal{U}$.

We model drawing a ballot from $\mathcal{G}$ as first drawing a ballot from $\mathcal{U}$; however, with probability δ, we replace the ballot we draw from $\mathcal{U}$ with a new ballot from $\mathbb{B}$ following a distribution $\mathbb{F}$. We make no further assumptions about the distribution $\mathbb{F}$ which aligns with our definition of variation distance. When drawing from $\mathcal{G}$, we have probability at most $\frac{1}{n} + \delta$ of drawing b for any ballot $b \in \mathbb{B}$.

When we sample sequentially, we get a length-z sequence S of ballot IDs for each

of $\mathcal{G}$ and $\mathcal{U}$. We define $S[i]$ as the ballot ID at index i in the sequence S. Throughout this model, we assume that we sample with replacement, although similar bounds should hold for sampling without replacement. We define w as the ordered list of indices in the sequence S where both $\mathcal{G}$ and $\mathcal{U}$ draw the same ballot. We note that the list of indices is 0-indexed. Furthermore, for any list of indices w, we define s_w as the list of ballots IDs at those indices. That is, for a fixed draw, $\mathcal{U}$ might produce the sample sequence [1, 5, 29]. Meanwhile, $\mathcal{G}$ might produce the sample sequence sequence [1, 5, 30]. For this example, $w = [0, 1]$ and $s_w = [1, 5]$.

We define the set of possible size-z samples as the set D. We choose z' such that for any given value ϵ_1, the probability that w is smaller than $z - z'$ is at most ϵ_1. Using this set up, we can calculate an upper bound on the probability that $\mathbb{A}$ returns "accept." In particular, given the empirical distribution, the probability that $\mathbb{A}$ returns "accept" for a deterministic auditing procedure becomes

$$\Pr[\mathbb{A} \textit{ accepts} \mid \mathcal{G}] = \sum_{S \in D} \Pr[\mathbb{A} \textit{ accepts} \mid S] * \Pr[\textit{draw } S \mid \mathcal{G}] \,.$$

Now, we note that we can split up the probability that we can draw a specific sample S from the distribution $\mathcal{G}$. We know that with high probability, there are at most z' ballots being "switched." Thus,

$$\Pr[\mathbb{A} \textit{ accepts} \mid \mathcal{G}]$$

$$= \sum_{S \in D} \Pr[\mathbb{A} \textit{ accepts} \mid S] * \Pr[\textit{draw } S \mid \mathcal{G}, \leq z' \text{ ``switched'' ballots}] * \Pr[\text{S has } \leq z' \text{ ``switched'' ballots}] \,,$$

$$+ \sum_{S \in D} \Pr[\mathbb{A} \textit{ accepts} \mid S] * \Pr[\textit{draw } S \mid \mathcal{G}, > z' \text{ ``switched'' ballots}] * \Pr[\text{S has } > z' \text{ ``switched'' ballots}] \,.$$

Now, we note that the second term is upper bounded by

$$\Pr[\text{any size-z sample has more than } z' \text{ switched ballots}] \,.$$

We define the probability that any size-z sample contains more than z' switched

ballots as ϵ_1. Note that ϵ_1 is a function of δ, z, and z'.

We note that although the draws are not independent the probability of a size-z sample having more than z' switched ballots is dominated by a binomial distribution, with z draws and δ probability of success per draw.

Now, we focus on bounding the first term. We know that

$$\Pr[\mathbb{A}\ accepts \mid \mathcal{G}, \text{any sample has at most } z' \text{ switched ballots}]$$

$$= \sum_{S \in D} \Pr[\mathbb{A}\ accepts \mid S] * \Pr[draw\ S \mid \mathcal{G}, S\ has\ \leq z' \text{ ``switched'' ballots}]$$

Meanwhile, for the uniform distribution, we know that the probability of accepting becomes

$$\Pr[\mathbb{A}\ accepts \mid \mathcal{U}] = \sum_{S \in D} \Pr[\mathbb{A}\ accepts \mid S] * \Pr[draw\ S \mid \mathcal{U}]\,.$$

Thus, we know that the change in probability of $\mathbb{A}$ returning "accept" becomes

$$\Pr[\mathbb{A}\ accepts \mid \mathcal{G}] - \Pr[\mathbb{A}\ accepts \mid \mathcal{U}]$$

$$\leq \epsilon_1 + \sum_{S \in D} \Pr[\mathbb{A}\, accepts \mid S](\Pr[draw\ S \mid \mathcal{G}, S\ has\ \leq z' \text{ ``switched'' ballots}] - \Pr[draw\ S \mid \mathcal{U}])\,.$$

However, for any fixed sample S, we know that we can produce S from $\mathcal{G}$ in many possible ways. That is, we know that we have to draw at least $z - z'$ ballots that are from $\mathcal{U}$, which defines our list w. Then, we have to draw the remaining z' ballots from $\mathcal{G}$. For a sample S, we define all possible lists w which are length $z - z'$ as the set $\mathbb{W}$. This set contains all possible subsets of the list $[0...z-1]$ which are length $z - z'$. We condition on specific values of w in $\mathbb{W}$ to define the exact indices in the sample tally where the uniform and empirical sampling can differ. We note that $|\mathbb{W}| = \binom{z}{z'}$ and any possible value of w happens with equal probability. Then, for any specific $w \in \mathbb{W}$, we can define r as the remaining indices which are allowed to differ from uniform and approximate sampling.

For an example, we can consider the sequence of ballots S=[1, 5, 29]. For sim-

plicity, we assume that $z' = 1$. Now, we would like to bound the probability that $\mathcal{G}$ draws S. We can split this up into cases:

1. $\mathcal{G}$ draws [1, 5, 29] by drawing 1 and 5 from the uniform distribution, then drawing a "switched ballot" at slot 2.

2. $\mathcal{G}$ draws [1, 5, 29] by drawing 1 and 29 from the uniform distribution, then drawing a "switched ballot" at slot 1.

3. $\mathcal{G}$ draws [1, 5, 29] by drawing 5 and 29 from the uniform distribution, then drawing a "switched ballot" at slot 0.

Thus, we define the possible compatible shared list of indices $\mathbb{W}$ as

$$\mathbb{W} = \{[0, 1], [1, 2], [0, 2]\}.$$

For each possible list $w \in \mathbb{W}$, we can define r as the remaining possible positions where we sample from $\mathcal{G}$ instead of $\mathcal{U}$. That is, if $w = [0, 1]$, then $r = [2]$. In this case, we must first draw ballots 1 and 5 from the uniform distribution and ballot 29 from the actual distribution.

We can now calculate the probability that we draw some specific size-z sample S, given the empirical distribution and a fixed value of z'.

$$\Pr[\textit{draw } S \mid \mathcal{G}] = \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * \Pr[\textit{draw } s_r \mid \mathcal{G}] * \Pr[\text{switched ballots are at indices in } r]$$

However, we know that for each ballot b in s_r, we draw ballot b with probability at most $\frac{1}{n} + \delta$ or $\frac{1+n\delta}{n}$. That is, for an index i in r, we know that this ballot that may have been "switched." In particular, we draw the correct ballot from $\mathcal{U}$ with probability $\frac{1}{n}$. However, in addition to this, we replace it with a new ballot with probability δ - we assume that we replace it with the ballot $S[i]$ with probability 1. Thus, we draw the ballot $S[i]$ for this particular slot with probability at most $\frac{1}{n} + \delta$. Thus, we get

$$
\begin{aligned}
&\Pr[\textit{draw } S \mid \mathcal{G}] \\
&= \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * \Pr[\textit{draw } s_r \mid \mathcal{G}] * \Pr[\text{switched ballots are at indices in } r] \\
&\leq \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * (\frac{1+n\delta}{n})^{z'} * \Pr[\text{switched ballots are at indices in } r] \\
&\leq (1+n\delta)^{z'} \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * \Pr[\textit{draw } s_r \mid \mathcal{U}] * \Pr[\text{switched ballots are at indices in } r] \,.
\end{aligned}
$$

Now, we note that there are $\binom{z}{z'}$ possible sequences $w \in \mathbb{W}$ which determine where the "switched" ballots could be. Since each of these possible sequences occurs with equal probability, this becomes

$$
\begin{aligned}
&\Pr[\textit{draw } S \mid \mathcal{G}] \\
&\leq (1+n\delta)^{z'} \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * \Pr[\textit{draw } s_r \mid \mathcal{U}] * \Pr[\text{switched ballots are at indices in } r] \,. \\
&= (1+n\delta)^{z'} \sum_{w \in \mathbb{W}} \Pr[\textit{draw } s_w \mid \mathcal{U}] * \Pr[\textit{draw } s_r \mid \mathcal{U}] * \frac{1}{\binom{z}{z'}} \\
&= (1+n\delta)^{z'} \Pr[\textit{draw } S \mid \mathcal{U}] \,.
\end{aligned}
$$

Using this bound we can calculate our total change in acceptance probability. This becomes:

$$
\Pr[\mathbb{A} \textit{ accepts} \mid \mathcal{G}] - \Pr[\mathbb{A} \textit{ accepts} \mid \mathcal{U}]
$$

$$
\begin{aligned}
&\leq \epsilon_1 + \sum_{S \in D} \Pr[\mathbb{A} \textit{ accepts} \mid S](\Pr[\textit{draw } S \mid \mathcal{G}, S \textit{ has } \leq z' \text{ ``switched'' ballots}] - \Pr[\textit{draw } S \mid \mathcal{U}]) \\
&\leq \epsilon_1 + ((1+n\delta)^{z'} - 1) \sum_{S \in D} \Pr[\mathbb{A} \textit{ accepts} \mid S] \Pr[\textit{draw } S \mid \mathcal{U}] \\
&\leq \epsilon_1 + (1+n\delta)^{z'} - 1 \,,
\end{aligned}
$$

which provides us the required upper bound. ∎

10.3.1 Empirical Support

Our previous theorem gives us a total bound of our change in risk limit, which depends on our value of z' and δ. We note that we provide a general bound on the multiplicative increase in probability compared to the uniform distribution of $(1 + n\delta)$ for a single ballot, which is based on the variation distance δ. However, we note that this bound is often quite loose. The exact multiplicative increase in probability for a single ballot can also be calculated for our distribution $\mathcal{G}$ easily. Thus, if a ballot is chosen with probability at most $\frac{(1+\epsilon_2)}{n}$, then our bound on the change in acceptance probability for the audit becomes

$$\epsilon_1 + (1 + \epsilon_2)^{z'} - 1.$$

The values of ϵ_2 are recorded for varying number of cuts in Table 7.2.

We can calculate the maximum change in probability for a varying number of cuts using this bound. Here, we analyze the case of 6 cuts. To get a bound on z', we can model how often we switch ballots. In particular, this follows a binomial distribution, with z independent trials, where each trial has a δ_6 probability of success. Using the binomial survival function, we see at most 4 "switched ballots" in 1,000 draws with probability at least (1- 8.78×10^{-4}). From our previous argument, we know that our change in acceptance probability is at most $(1 + \epsilon_2)^4 - 1$. Using our value of ϵ_2 for $k = 6$, this causes a change in probability of at most 0.0090.

Thus, the maximum possible change in probability of incorrectly accepting this outcome is $0.0090 + 8.78 \times 10^{-4}$, which is approximately 9.88×10^{-3}. We can compensate for this by decreasing our risk limit by less than 1%.

10.4 Key Takeaways

We have outlined a simple mitigation procedure for RLAs which relies on risk limit adjustment. In particular, if we require a 5% risk limit for our audit, then we recommend setting a 4% risk limit in the software. Then, any residual non-uniformity from approximate sampling techniques like k-cut will be accounted for in the additional 1%

of risk. In practice, for sample sizes up to 1,000 ballots, we recommend $k = 6$ cuts. We note that after the first 1,000 ballots in the sample, using more uniformly random techniques (such as careful hand-counting) for additional ballots is theoretically safer.

Chapter 11

Approximate Sampling in Practice

In this chapter, we will discuss techniques for using approximate sampling in practice. In particular, we describe how to integrate our k-cut technique with the sampling plan provided by RLA software. Furthermore, we discuss how to choose values for k, heuristics, and usage guidelines to make k-cut simple and efficient in practice. Finally, we discuss the usage of k-cut in Indiana and Michigan and discuss empirical evidence that shows that k-cut provides an increase in efficiency.

11.1 Multi-Stack Sampling

Our discussion so far presumes that all cast paper ballots constitute a single "stack" and suggest using our proposed k-cut procedure is used to sample ballots from that stack. In practice, however, stacks have limited size, since large stacks are physically awkward to deal with. The collection of cast paper ballots is therefore often arranged into multiple stacks of some limited size.

The *ballot manifest* describes this arrangement of ballots into stacks, giving the number of such stacks and the number of ballots contained in each one. We assume that the ballot manifest is accurate. A tool like Stark's Tools for Risk-Limiting Audits [1] takes the ballot manifest (together with a random seed and the desired sample size) as input and produces a sampling plan. We note that k-cut is primarily

[1] `https://www.stat.berkeley.edu/~stark/Vote/auditTools.htm`

intended for use with ballot-polling audits, where the order of the ballots does not need to be maintained.

A sampling plan describes exactly which ballots to pick from which stacks. That is, the sampling plan consists of a sequence of pairs, each of the form: (stack-number, ballot-id), where ballot-id may be either an id imprinted on the ballot or the position of the ballot in the stack (if imprinting was not done).

Modifying the sampling procedure to use k-cut is straightforward. We ignore the ballot-ids, and note only how many ballots are to be sampled from each stack. That number of ballots are then selected using k-cut rather than using the provided ballot-ids.

For example, if the sampling plan says that 2 ballots are to be drawn from stack 5, then we ignore the ballot-ids for those specific 2 ballots, and return 2 ballots drawn approximately uniformly at random using k-cut.

Thus, the fact that cast paper ballots may be arranged into multiple stacks (or boxes) does not affect the usability of k-cut for performing audits.

11.2 Choosing Values for k

The major question when using the approximate sampling procedure is how to choose k. Choosing a small value of k makes the overall auditing procedure more efficient, since you save more time in each sample you choose. However, a small k requires more risk limit adjustment.

The risk limit mitigation procedure requires knowledge of the maximum sample size, which we denote as z^*, beforehand. We assume that the auditors have a reasonable procedure for estimating z^* for a given contest. One simple procedure to estimate z^* is to draw an initial small sample size, z, using uniform random sampling. Then, we can use a statistical procedure to approximate how many ballots we would need to finish the audit, assuming the rest of the ballots in the pool are similar to the sample. Possible statistical procedures which can be used here include:

- Replicate the votes on the ballots,

- Sample from the multinomial distribution using the sample voteshares as hyperparameters,
- Use the Polya's Urn model to extend the sample,
- Use the workload estimate as defined by Lindeman et al. [12], for a contest with risk limit α and margin m to predict the number of samples required.

Let us assume that we use one of these techniques and calculate that the audit is likely to be complete after an extension of size d. To be safe, we suggest assuming that the required additional sample size for the audit is at most $2d$ or $3d$, to choose the value of k. Thus, our final bound on z^* would be $z + 3d$.

Given this upper bound, we can perform our approximate sampling procedures and mitigation procedures, assuming that we are drawing a sample of size z^*. If the sample size required is greater than z^*, then the ballots which are sampled after the first z^* ballots should be sampled as uniformly at random as possible. This guarantees that the maximum change in probability of the audit incorrectly accepting the outcome due to approximate sampling will be the change in probability due to the first z^* ballots being sampled approximately at random. We recommend choosing k to guarantee that this is small, based on the value of z^*. In general, we recommend $k = 6$ cuts with a 1% risk limit adjustment.

We note that it is implausible to expect counting to be a "perfect" technique, as discussed by Goggin et al. [6]. Often, audit teams which have to count hundreds of ballots per draw will make some mistakes along the way. In the typical framework for an RLA, we assume that the counting is perfect, implying the sampling is exactly uniformly random, and calculate the risk. However, in k-cut, we can increase the uniformity of our sample simply by increasing k. Thus, in some cases like using $k = 10$, k-cut might prove to be more uniform than counting. Perhaps, in these cases, it is reasonable to assume that k-cut and counting are indistinguishable and not require any risk limit adjustment.

We would like to conduct additional studies to see the rate and bias of counting mistakes to make better recommendations in this area.

11.3 Heuristics for k-Cut

In usage of k-cut during the Michigan pilot audits, we added "hints," as described in Section 6.2. In particular, before asking the auditor to make a cut, we used Google's random number generator to generate a random number between 1 and 99. If the random number generator returned r, we asked the auditor to estimate $r\%$ of the ballots and remove them off the top of the stack.

11.4 Usage Guidelines for k-Cut

When training election officials to use k-cut during an election audit, we suggest providing the following set of guidelines, provided in Appendix C.

11.5 Usage in Indiana Pilots

On May 29–30, 2018, Marion County, Indiana held a pilot audit of contest results from the November 2016 general election[2]. This audit was held by the Marion County Election Board with assistance from the Voting System Technology Oversight Project (VSTOP) Ball State University, the Election Assistance Commission, and the current authors.

For some of the sampling performed in this audit, the "Three-Cut" sampling method of this paper was used instead of the "counting to a given position" method. The Three-Cut method was modified so that three different people made each of the three cuts; the stack of ballots was passed from one person to the next after a cut.

Although the experimentation was informal and timings were not measured, the Three-Cut method did provide a speed-up in the sampling process.

[2]Further notes on this pilot audit can be found at `http://bowencenterforpublicaffairs.org/wp-content/uploads/2018/06/VSTOP-Raleigh-Presentation-June-2018.pdf`.

11.6 Usage in Michigan Pilots

In usage of k-cut, we added "hints" to increase uniformity. In particular, before asking the auditor to make a cut, we used Google's random number generator to generate a random number between 1 and 99. If the generator returned r, we asked the auditor to estimate $r\%$ of the ballots and remove them off the top of the stack for the cut.

11.6.1 Rochester Hills Pilot

For the pilot in Rochester Hills, Michigan, we used a ballot-polling audit, which required a sample of 76 ballots to audit a proposition. We found that 50 ballots in our sample were for "Yes" and 26 ballots were for "No". The selected precincts had a total of 36666 votes. We used a uniform prior of one vote per candidate for the audit. Running this through the Bayesian audit tool, provided a 0.27% Bayesian upset probability.

We also experimented with k-cut for election-day ballots which required counting more than 200 ballots at a time. We chose $k = 6$ to use in practice and the new technique with "hints" as described above. We found that making 6 cuts took the Rochester Hill's audit teams approximately 60 seconds; counting a single ballot took the teams approximately 1 second.

Moreover, we noticed that the auditors always started counting at the top of the stack, rather than the bottom, which makes counting more inefficient than we had originally estimated.

11.6.2 Lansing Pilot

For the pilot in Lansing, Michigan, we used a hybrid audit to audit a judgeship. We drew 258 ballots since the race had quite a small margin. However, we found that there were many discrepancies in our sample, so the ballot-polling audit proved to have a slightly lower Bayesian upset probability than the Bayesian comparison audit. As before, we used a uniform prior of one vote per candidate for the ballot-polling audit. The Bayesian ballot-polling audit provided an upset probability of 9.95%.

To save time, we used k-cut for all election-day ballots and the counting technique for all the absentee ballots. As before, we chose $k = 6$ and provided hints. In Lansing, we timed 12 iterations of k-cut on varying batch sizes to find an average time for 6 cuts to be 65 seconds. However, we note that on batch sizes of approximately 1,000 ballots or larger, k-cut took closer to 80-85 seconds.

In Lansing, we also tested how much the hints helped by running an experiment. In particular, we took an ordered stack of ballots, which were numbered from 0 to 199. Then, we used Google's random number generator to generate a random number r between 0 and 199 and asked the subject to try to choose the ballot with r on top. Then, we reset the deck and repeated the process.

We analyzed the distribution of actually chosen numbers to calculate if the hints helped to provide a more uniform distribution. We note that our empirical data and thus, our analysis was based on a single-cut distribution that was quite non-uniform. That is, the Kolmogorov-Smirnov statistic for our single-cut distribution (without hints) compared to the uniform distribution was 294.0. Initial results with using these hints show a significant increase in uniformity. That is, based on 120 actual cut sizes made using the test described above, we calculated a Kolmogorov-Smirnov statistic of 11.4.

11.6.3 Kalamazoo Pilot

For the pilot in Kalamazoo, Michigan, we used a hybrid audit to audit the governor's race. We drew 40 ballots, 32 from the ballot-polling stratum and 8 from the ballot-comparison stratum. For the ballot-comparison audit, we had a prior of 50 for each (actual_vote, reported_vote) pair of the form (i, i) for any candidate i. We had a uniform prior of 0.5 for each (actual_vote, reported_vote) pair of the form (i, j) where $i \neq j$. The Bayesian ballot-comparison audit provided an upset probability of 0.03%.

In Kalamazoo, we used k-cut for most of the election-day ballots. However, we requested some teams count large numbers to provide a more fair comparison of how much time k-cut would save in practice.

We note that in Kalamazoo, counting ballots took an average of 1.7 seconds per ballot over 5 measurements. In each measurement, the county clerks had to count at least 100 ballots and we timed how long it took them to reach the appropriate ballot. From there, we computed their pace per ballot and averaged these measurements over 5 instances. We note that when approximately 100 ballots need to be counted, the average time was little more than 1 second per ballot. However, as the numbers increased, the pace slowed down. When the clerks had to count at least 300 ballots, the average time was closer to 1.8 seconds per ballot.

We also note that the batch sizes were quite large in Kalamazoo, so we also measured the k-cut timings for large batch sizes. We note that for large batch sizes (around 1600 ballots), making 6 cuts took about 2 minutes and 17 seconds on average. This is almost twice as long as the average time it took in other cities. For batch sizes around 1,000 ballots, the same group took about 1 minute and 34 seconds. Thus, while the timing of k-cut is not directly a function of batch size, it might not be accurate to say that the timing is independent of the size of the batch.

We also timed the audit team's entire process from start to finish. That is, we measured the time it took for a team to open the bag containing all the ballots, obtain a ballot using k-cut, and close the bag up. This overall process took slightly over 4 minutes in total[3].

[3]Some further videos and slides are posted at `http://people.csail.mit.edu/rivest/pubs.html#Riv18g`.

Part V

Conclusions

Chapter 12

Contributions and Future Work

In this chapter, we provide an overview of future work and extensions to the work that was explored in this thesis. We primarily focus on extensions for the approximate sampling project, although we discuss future avenues to explore in workload optimization and Bayesian audit analysis as well. Finally, we provide a list of our conclusions and contributions to the field of election tabulation auditing.

12.1 Future Work

12.1.1 Approximate Sampling

We would like to do more experimentation on the variation between individuals on their cut-size distributions. The current empirical results in this paper are based on the cut distributions of just the two authors in the paper. We would like to test a larger group of people to better understand what distributions should be used in practice.

After investigating the empirical distributions of cuts, we would like to develop "best practices" for using the k-cut procedure. That is, we would like to develop a set of techniques that auditors can use to produce nearly-uniform single-cut-size distributions; we started this work in Michigan, however, we would like to run more pilot experiments to make using the k-cut procedure much more efficient.

Moreover, we note that most of our analysis is not specific to k-cut. In fact, there are several different techniques for approximate sampling including:

- Using a random number generator to choose a ballot by position, then using a weighing machine to find the ballot in that location.
- Using a random number generator to choose a ballot position, then using a ruler to find the location of the ballot[1].
- k-cut or other variants on shuffling techniques.

We note that counting is often not perfect and is also a technique for "approximate" sampling. We would like to study the accuracy of counting in audit settings to see how many mistakes are made in practice.

Finally, we note that our analysis makes some assumptions about how k-cut is run in real life. For instance, we assume that each cut is made independently. We would like to run some empirical experiments to test our assumptions.

12.1.2 Other Mini-Projects

I had the chance to explore a few other projects in the course of my Master's, including workload estimation, workload optimization, and Bayesian audit analysis. Although my preliminary results in these projects were promising, I would love to explore different opportunities in each of these projects.

With regards to workload estimation, I would like to develop a better understanding of what operations during an audit are costly. We chose to analyze one – escalating to multiple rounds – however, we would like to explore more complex cost functions, especially in the multi-county multi-contest setting.

Similarly, in workload optimization, we primarily focused on Bayesian audits, rather than RLAs. I would like to explore similar techniques for RLAs and develop theoretical bounds on how much time can be saved by using these techniques. I also

[1]Thank you to Professor Fraud (William Kresse) for this suggestion, commonly used in the finance industry.

note that, even empirically, there is still work to do in finding the right hyperparameters and exploring how these optimization techniques generalize.

Finally, in the analysis of Bayesian audits, we found some initial results showing that Bayesian audits satisfy some statistical properties that we would expect from a good audit procedure. We note that additional work is being done in this area by Professor Vora at George Washington University, on relating Bayesian audits and RLAs.

12.2 Contributions

In this thesis, we have explored several problems in election auditing. We defined a few general areas of research in the field, and focused on increasing the efficiency of audits and on understanding Bayesian audits.

We proved that Bayesian audits have some good statistical properties; in particular, we showed that the probability of generating the exactly correct "non-sample" tally increases monotonically with the sample size until the last few ballots. This provides further evidence for Bayesian audits being a good candidate for a statistical audit procedure.

We then focused on understanding the efficiency of audits. In this theme, we provided tools using Jupyter notebooks, which implement workload estimations for RLAs to finish in a single round with high probability. We also designed optimization techniques for Bayesian audits, to distribute workload among different stratum at different rates to minimize the total number of sample ballots required We implemented these techniques in Rivest's Bayesian audit tool kit [16] to show a decrease in the number of required samples for a synthetic election.

Finally, we designed a simple approximately-uniform sampling scheme, k-cut, for choosing random ballots to sample in post-election audits. We analyzed the effects of approximate sampling, and designed a simple mitigation procedure to make the approximation work with the risk-limiting audit. We provided empirical support for this technique through pilot audits in Indiana and Michigan, where k-cut proved to

save significant amounts of time for the audit teams.

Part VI

Appendices

Appendix A

Ballot Polling Work Estimation

The Jupyter notebook for estimating high probability workloads for ballot-polling RLAs is provided in the next few pages.

Ballot Polling Work Estimation

December 31, 2018

1 Work Estimation for RLAs

These simulations follow the structure outlined in *A Gentle Introduction to Risk-Limiting Audits* and *BRAVO: Ballot-polling Risk-limiting Audits to Verify Outcomes*. They focus on estimating the required sample size for a ballot-polling audit.

We assume there are k candidates in a race, who each have a reported vote share s_t. There are n ballots cast. We ignore t, a tolerance factor for RLAs, for simplicity.

Thus, the RLA procedure, for a reported winner w, is as follows: - Initialize T = 1 - If the ballot is for the winner, multiply T by $2s_w$ - Else, if it is valid for anyone else, multiply T by $2(1 - s_w)$ - Stop when T is greater than $\frac{1}{\alpha}$

Thus, we want to choose a sample size m, so that after m ballots, we satisfy the stopping condition. Assuming there are m_w votes for the reported winner, the value of T will become:

$$T = (2s_w)^{m_w}(2(1 - s_w))^{m-m_w}$$

Solving this for m, where we assume that $m_w = c * m$ tells us:

$$\begin{aligned}
&T > \frac{1}{\alpha} \\
&\iff (2s_w)^{m_w}(2(1 - s_w))^{m-m_w} > \frac{1}{\alpha} \\
&\iff m_w \ln(2s_w) + (m - m_w)\ln(2(1 - s_w)) > \ln(\frac{1}{\alpha}) \\
&\iff m(c\ln(2s_w) + (1 - c)\ln(2(1 - s_w))) > \ln(\frac{1}{\alpha}) \\
&\iff m > \frac{\ln(\frac{1}{\alpha})}{c\ln(2s_w) + (1 - c)\ln(2(1 - s_w))}
\end{aligned}$$

In expectation, we expect $c = s_w$. This makes the expected sample size

$$m > \frac{\ln(\frac{1}{\alpha})}{s_w\ln(2s_w) + (1 - s_w)\ln(2(1 - s_w))}$$

This provides the bound shown in the BRAVO paper.

However, we would like to prove a high probability bound on finishing the audit within a single round.

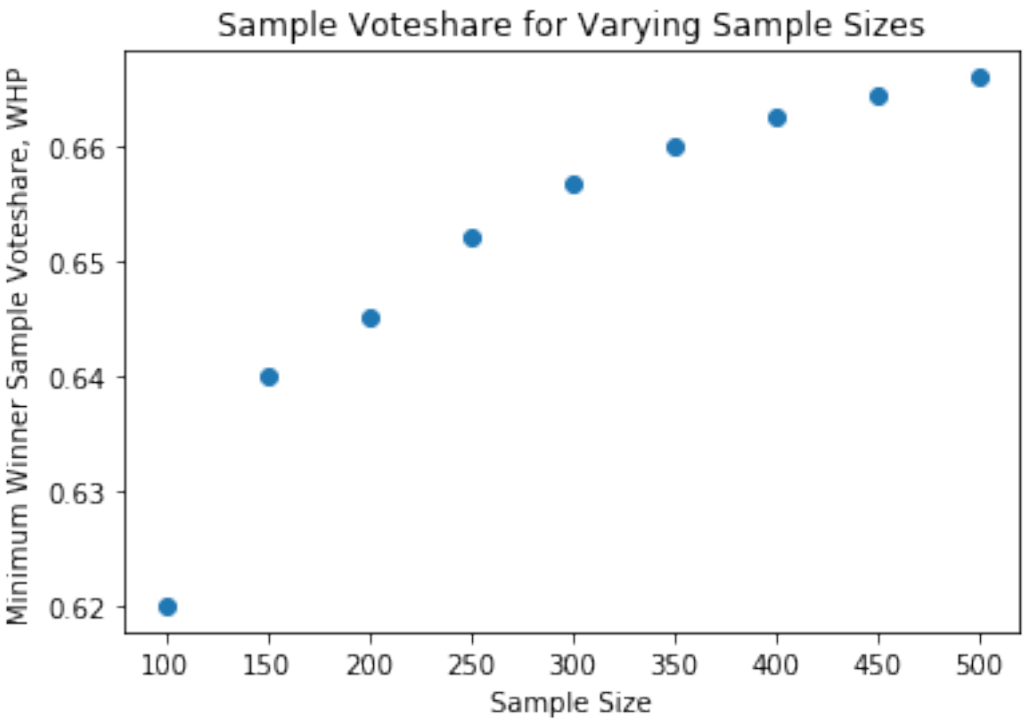
Thus, first, for an actual underlying voteshare, we calculate (with high probability - 95%) the sample size for finishing in one round. To do this, first we estimate values of c, for voteshares.

```
In [2]: import math
        from scipy import stats
        import matplotlib.pyplot as plt

In [3]: def predict_num_votes(winner_voteshare, sample_size, epsilon=0.05):
            """Returns a value of min_votes, where for the given sample size,
            the probability that we see at least min_votes samples for the winner
            is at least (1-epsilon)
            """
            bound_reached = False
            min_votes = 0
            while not bound_reached:
                if stats.binom.cdf(min_votes, sample_size, winner_voteshare) < (epsilon):
                    min_votes += 1
                    continue
                break
            return min_votes
```

Now, for a fixed value of ϵ, and varying the voteshare of the winner, we can predict the proportion c, where c is the percentage of the sample size that we will see for the reported winner. We focus on analyzing fixed voteshares, with varying sample sizes.

```
In [6]: for voteshare in [0.7]:
            epsilon = 0.05
            xs, ys = [], []
            for ss in [100, 150, 200, 250, 300, 350, 400, 450, 500]:
                xs.append(ss)
                ys.append(predict_num_votes(voteshare, ss, epsilon) / ss)
            plt.xlabel("Sample Size")
            plt.ylabel("Minimum Winner Sample Voteshare, WHP")
            plt.title("Sample Voteshare for Varying Sample Sizes")
            plt.plot(xs, ys, 'o')
            plt.savefig("sample_from_pop_voteshare.png")
            plt.show()
            plt.close()
```



Now, using these values of c, we can directly predict the number of samples required to finish the audit in a single round. In particular, we want to choose m and c to guarantee that:

$$m(c\ln(2s_w) + (1-c)\ln(2(1-s_w))) > \ln(\frac{1}{\alpha})$$

```
In [91]: def predict_t_value(c, ss, reported_voteshare):
             t_value = (2*(1-reported_voteshare))**(ss - c*ss)
             t_value *= (2*reported_voteshare)**(c*ss)
             return t_value
```

```
In [102]: def predict_workload(alpha, reported_voteshare, actual_voteshare, epsilon):
              """Estimate number of ballots required, based on risk limit, reported
              voteshare for the winner, a minimum bound on the actual voteshare for
              the winner, and epsilon (where (1-epsilon) is the min probability that
              after this many ballots the audit will finish)
              """
              ss = 0
              t_value = 0
              while t_value < 1/alpha:
                  ss += 1
                  c = predict_num_votes(actual_voteshare, ss, epsilon) / ss
                  t_value = predict_t_value(c, ss, reported_voteshare)
              return ss
```

Thus, to get an accurate work estimate, let us assume the winner has a reported voteshare of 70%. We are pretty confident that they have an actual voteshare of at least 65%. We want to finish the audit within the first round with probability at least 80% and our audit has a risk limit of 5%. We can estimate this by using the above functionality:

```
In [114]: predict_workload(0.05, 0.7, 0.65, 0.2)

Out[114]: 183
```

Thus, our workload estimation tool suggests sampling 183 ballots in the first round.

Note: Due to the way BRAVO works, if our actual voteshares and reported voteshares are not that similar, the number of ballots increases very quickly. However, this is not due to the workload estimation tool, but rather the nature of the probability test.

Appendix B

Ballot Comparison Work Estimation

The Jupyter notebook for estimating high probability workloads for ballot-comparison RLAs is provided in the next few pages.

Ballot Comparison Work Estimation

December 31, 2018

1 Work Estimation for Comparison RLAs

These simulations follow the structure outlined in *Super-Simple Simultaneous Single-Ballot Risk-Limiting Audits* They focus on estimating the required sample size for a ballot-comparison audit.

We assume there are k candidates in a race, who each have a reported vote share s_t. There are n ballots cast. We can define V as the smallest reported margin (ex. 0.1) between the reported winner and the runner-up. We can denote γ as the inflation factor, in the audit. We note that we require $\gamma > 1$. From the paper, we know that larger values of γ increase the initial sample size, but require less expansion if there are more overstatements than expected. For our simulations, we choose $\gamma = 1.01$.

Thus, the P-value for the ballot comparison audit is

$$P = (1 - 1/U)^s * (1 - 1/(2\gamma))^{-s_1} * (1 - 1/\gamma)^{-s_2}$$

where $U = \frac{2\gamma}{V}$ and there are s_1 single-vote overstatements and s_2 two-vote overstatements. Finally, s is the sample size drawn for the audit.

```
In [1]: import math
        import matplotlib.pyplot as plt
        from scipy import stats
```

```
In [2]: def estimate_p(gamma, sample_size, min_margin, max_so, max_do):
            """Estimates p value based on values for the inflation,
            the sample size, the minimum margin,
            the maximum number of single vote overstatements and
            the maximum number of double vote overstatements.
            """
            u = 2*gamma / min_margin
            p = (1-1/u)**sample_size
            p *= (1-1/(2*gamma))**(-max_so)
            p *= (1-1/gamma)**(-max_do)
            return p
```

```
In [3]: def calculate_max_overstatement(sample_size, o_rate, epsilon):
            """
            Calculates maximum number of overstatements seen, with probability (1-epsilon),
            assuming each ballot contains an overstatement with probability o_rate.
            """
```

```
    num_o = 0
    while stats.binom.sf(num_o, sample_size, o_rate) > epsilon:
        num_o += 1
    return num_o
```

Now, for a fixed election with $\gamma = 1.01$, and a smallest margin of 10%, we can compute the number of ballots to draw so the audit will complete in the first round with probability at least 90%.

To do this, first we estimate a 1-vote error rate of 0.5% and a 2-vote error rate of 0.1%. Using these numbers, we can calculate, for a given size, the maximum number of 1 and 2 vote overstatements we will see, with probability at least 90%. Then, using this, we can estimate the required sample size for an audit with a 5% risk limit to complete.

```
In [4]: gamma = 1.01
        min_margin = 0.1
        so_rate = 0.005 # single overstatement rate
        do_rate = 0.001 # double overstatement rate
        epsilon = 0.1
        alpha = 0.05

        sample_size = 1
        p_val = 1
        while p_val > 0.05:
            max_so = calculate_max_overstatement(sample_size, so_rate, epsilon)
            max_do = calculate_max_overstatement(sample_size, do_rate, epsilon)
            max_so = so_rate * sample_size
            max_do = do_rate * sample_size
            p_val = estimate_p(gamma, sample_size, min_margin, max_so, max_do)
            sample_size += 1
        print(sample_size)
```

```
72
```

Thus, we can see that we require a sample size of 72 ballots, for the comparison audit to complete for these settings.

Assuming that the overstatement rates are constant, we can see how this changes with the minimum margin in the election. In this case, we can vary the "high probability" bound. That is, we choose different values of ϵ, where the audit will complete in a single round with probability at least $(1 - \epsilon)$. Then, for each value of ϵ, we vary the minimum margin and calculate the required sample sizes.

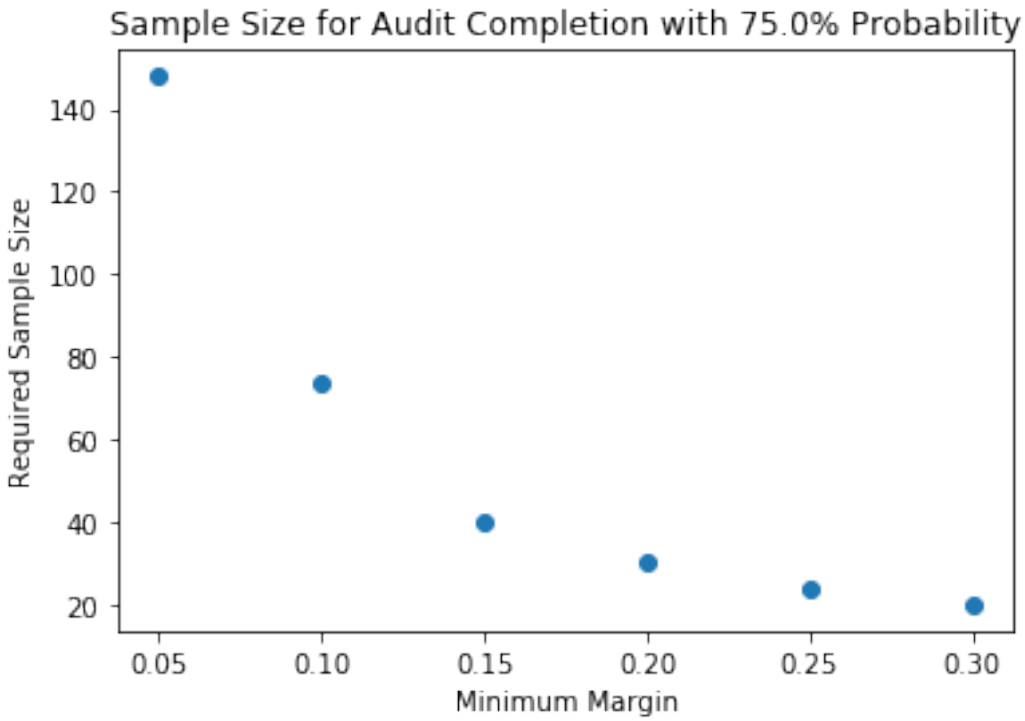
```
In [5]: gamma = 1.01
        so_rate = 0.005 # single overstatement rate
        do_rate = 0.001 # double overstatement rate
        alpha = 0.05

        for epsilon in [0.25]:
            xs, ys = [], []
```

```
min_margins= [0.05, 0.1, 0.15, 0.2, 0.25, 0.3]
for min_margin in min_margins:
    xs.append(min_margin)
    sample_size = 1
    p_val = 1
    while p_val > 0.05:
        max_so = calculate_max_overstatement(sample_size, so_rate, epsilon)
        max_do = calculate_max_overstatement(sample_size, do_rate, epsilon)
        p_val = estimate_p(gamma, sample_size, min_margin, max_so, max_do)
        sample_size += 1
    ys.append(sample_size)

plt.xlabel("Minimum Margin")
plt.ylabel("Required Sample Size")
plt.title("Sample Size for Audit Completion with {}% Probability".format(
    (1-epsilon)*100))
plt.plot(xs, ys, 'o')
plt.savefig("margin_vs_sample.png")

plt.show()
plt.close()
```



Appendix C

k-Cut Usage Guidelines

C.1 Procedure Overview

The k-cut procedure is used to simplify the process of choosing a random ballot from a stack of ballots. It involves making k cuts in the stack of ballots and then choosing the ballot on top as our random ballot. Each "cut" is the same as a cut in a deck of cards, where you choose a random portion of ballots from the top of the stack and move them to the bottom.

As a simple example of a single cut, if the given stack has $n = 5$ ballots:

$$\boxed{\text{A B C D E}},$$

where ballot A is on top and ballot E is at the bottom, then a cut of size 2 separates the stack into a top part of size 2 and a bottom part of size 3:

$$\boxed{\text{A B}}\ \boxed{\text{C D E}}$$

whose order is then switched:

$$\boxed{\text{C D E}}\ \boxed{\text{A B}}\ .$$

Finally, the two parts are then placed together to form the final stack:

$$\boxed{\text{C D E A B}}.$$

having ballot C on top. We repeat this process k times, and choose the ballot that ends up on top as our randomly sampled ballot.

C.2 Recommendations

Values for k In practice, we recommend using $k = 6$ cuts, before choosing the top ballot as our random sample. Recommending 6 cuts is based on some assumptions on the maximum possible sample size we would need for the audit – if more than 1,000 ballots need to be drawn, we would recommend drawing ballots after the first 1,000 using a counting technique, with pseudo-random number generators or other software tools.

Single Cut Strategies To make k-cut as efficient and correct as possible, we recommend choosing each cut size *as uniformly randomly as possible*. We note that, often, when people are asked to make "random" cuts, they tend to choose a cut size somewhere in the middle.

For instance, when making a cut in a stack of 100 ballots, people often choose approximately 50 ballots off the top and move them to the bottom. Please keep in mind that you should make a cut of size 1 (just removing the top ballot) as often as you make a cut of size 50. From our experience, we noticed that we made smaller cuts (less than half the stack) more often than larger cuts. Ideally, each size cut should happen with equal probability.

For help with this randomness, we suggest the following procedure. Each time you make a cut, use Google's random number generator to generate a random number between 1 and 99. Let us assume that Google generates the number 5. Then, try to estimate and remove 5% of the ballots from the top of the stack, then complete the cut. This procedure can be repeated for every cut.

Security To prevent any biases during the sampling process, we do not allow cuts of size 0. Every person making a cut has to remove at least one ballot from the top of the stack.

We have seen that k-cut can be used by multiple people. For instance, one person can make the first cut, another can make the second, and they can switch until 6 cuts have been made in total and draw the top ballot. This is allowed.

Multiple Ballot Selection If we need to choose multiple ballots from a single stack, we can perform 6 cuts and draw the ballot on top as our first random sample. Then, we can record the vote on it and replace it on the top of the stack (or not, depending on the audit procedure). From here, we can repeat the entire 6-cut process on the stack to get our next random sample. We note that we do not need to "reset" the deck in any way. That is, let's say our first ballot, ballot C, like the example above. Then, to draw our next sample, we can start making cuts with ballot C on top, without resetting the deck to have ballot A on top.

Bibliography

[1] Sanjeev Arora. Lecture 12: Random walks, markov chains, and how to analyse them; cos 521, 2013. https://www.cs.princeton.edu/courses/archive/fall13/cos521/lecnotes/lec12.pdf.

[2] Thomas Baignères and Serge Vaudenay. The complexity of distinguishing distributions. https://infoscience.epfl.ch/record/126225/files/BV08.pdf, 2008.

[3] Jorge H. Banuelos and Philip B. Stark. Limiting risk by turning manifest phantoms into evil zombies. https://arxiv.org/abs/1207.3413, 2012.

[4] Michelle Blom, Peter J. Stuckey, and Vanessa J. Teague. Ballot-polling risk limiting audits for irv elections, Oct. 2018. https://people.eng.unimelb.edu.au/michelleb/IRV-auditing.pdf.

[5] J. Bretschneider, S. Flaherty, S. Goodman, M. Halvorson, R. Johnston, M. Lindeman, R.L. Rivest, P. Smith, and P.B. Stark. Risk-limiting post-election audits: Why and how?, Oct. 2012. (ver. 1.1) http://people.csail.mit.edu/rivest/pubs.html#RLAWG12.

[6] Stephen N. Goggin, Michael D. Byrne, and Juan E. Gilbert. Post-election auditing effects of procedure and ballot type on manual counting accuracy, efficiency, and auditor satisfaction and confidence. *Election Law Journal*, 2012.

[7] Stacy D. Hill, László Gerencsér, and Zsuzsanna Vágó. *Stochastic Approximation on Discrete Sets Using Simultaneous Difference Approximations*. In *Proceeding*

of the 2004 American Control Conference, pages 2795–2798, June 2004. https://www.jhuapl.edu/spsa/PDF-SPSA/Hill_ACC04_discrete.pdf.

[8] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[9] K. Johnson. Election verification by statistical audit of voter-verified paper ballots. http://ssrn.com/abstract=640943, Oct. 31 2004.

[10] M. Lindeman, M. Halvorseon, P. Smith, L. Garland, V. Addona, and D. McCrea. Principle and best practices for post-election audits. www.electionaudits.org/files/best%20practices%20final_0.pdf, 2008.

[11] Mark Lindeman and Philip B. Stark. A gentle introduction to risk-limiting audits. *IEEE Security and Privacy*, 10:42–49, 2012. https://www.stat.berkeley.edu/~stark/Preprints/gentle12.pdf.

[12] Mark Lindeman, Philip B. Stark, and Vincent S. Yates. BRAVO: Ballot-polling risk-limiting audits to verify outcomes. In *Presented as part of the 2012 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*, Bellevue, WA, 2012. USENIX.

[13] Audrey Malagon and Ronald L. Rivest. Bayesian election audits in one page. http://people.csail.mit.edu/rivest/pubs/MR18.pdf.

[14] Albert W. Marshall and Ingram Olkin. Bivariate distributions generated from polya-eggenberger urn models. *Journal of Multivariate Analysis*, pages 48–65, 1990.

[15] Steven J. Miller and Mark J. Nigrini. The modulo 1 Central Limit Theorem and Benford's law for products. *International Journal of Algebra* **2**, no. 3:119–130, 2008.

[16] Ronald L. Rivest. Bayesian audit support program. https://github.com/ron-rivest/audit-lab/.

[17] Ronald L. Rivest. Bayesian tabulation audits: Explained and extended. https://arxiv.org/abs/1801.00528, January 1, 2018.

[18] Ronald L. Rivest and Emily Shen. A bayesian method for auditing elections. In J. Alex Halderman and Olivier Pereira, editors, *Proceedings 2012 EVT/WOTE Conference*, 2012.

[19] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[20] Philip B. Stark. Papers, talks, video, legislation, software, and other documents on voting and election auditing. https://www.stat.berkeley.edu/~stark/Vote/index.htm.

[21] Philip B. Stark. Super-simple simultaneous single-ballot risk-limiting audits. In *Proc. 2010 EVT/WOTE Workshop*, 2010. http://www.usenix.org/events/evtwote10/tech/full_papers/Stark.pdf.

[22] Philip B. Stark. Tools for ballot-polling risk-limiting election audits. https://www.stat.berkeley.edu/~stark/Vote/ballotPollTools.htm, 2017.

[23] Stephen Tu. The dirichlet-multinomial and dirichlet-categorical models for bayesian inference, 2014. https://people.eecs.berkeley.edu/~stephentu/writeups/dirichlet-conjugate-prior.pdf.