

Improving Decision-Making from Distributed Human Evaluations

Alexander Koujianos Goldberg

CMU-CS-26-117

June 2026

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Giulia Fanti (Co-Chair)

Nihar B. Shah (Co-Chair)

Tom Mitchell

John Ioannidis (Stanford University)

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2026 **Alexander Koujianos Goldberg**

This research was supported in part by the National Science Foundation under grant numbers 1763734, 1705007, 1942124, 2200410, and DGE2140739; the Office of Naval Research under grant number N000142212181; and the Bill & Melinda Gates Foundation.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: human evaluation, peer review, randomized selection, decision-making under uncertainty, privacy

To my parents.

Abstract

Many consequential decisions rely on aggregating noisy judgments from distributed human evaluators, often without access to an objective ground truth. Scientific peer review and grant funding are canonical examples, but similar challenges arise in hiring, admissions, and evaluation of AI models. The goal of this thesis is to understand and mitigate errors in distributed human evaluation to make better decisions. Toward this end, we conduct controlled experiments in real review processes and develop principled algorithms with provable guarantees.

In the first part of the thesis, we present experimental evidence on sources of error and the effectiveness of interventions in scientific peer review. We present results from two human-subjects experiments that we conducted at large-scale ML/AI conferences. The first studies whether meta-evaluation can improve review quality; the second studies a live deployment of a large language model assistant for authors. Together, these experiments demonstrate both the promise and limits of interventions aimed at improving review quality.

In the second part of the thesis, we develop algorithms for making selection decisions from error-prone evaluations. Randomized selection has been gaining adoption in scientific funding. However, existing designs of these peer review “lotteries” are often ad hoc. We formalize the desiderata motivating the use of randomized decisions in peer review and show that prior methods fail to meet their intended goals. We then develop efficient algorithms with provable guarantees that better address the motivations for randomization.

Finally, in the third part of the thesis, we study how to release data about evaluation processes without compromising participant privacy. Outside scrutiny of high-stakes evaluation systems requires data about reviews and outcomes, but releasing such data can re-identify participants or leak sensitive personal information. We examine this tension in pseudonymous time-series data and fraud-detection graph data, demonstrating practical privacy attacks and developing provably sound mechanisms for sharing useful data while protecting anonymity.

Acknowledgments

I would first like to sincerely thank my advisors, Giulia Fanti and Nihar B. Shah, for all of the advice, support, and generosity that they have provided throughout my PhD. The work in this thesis would not have been possible without their guidance. Through working with them, I became a much better researcher, and I found our collaboration not only intellectually rewarding, but also genuinely fun and often surprisingly funny. They modeled the kind of scientific rigor and intellectual honesty that the research in this thesis ultimately seeks to support and improve.

I would also like to thank the other members of my thesis committee, Tom Mitchell and John Ioannidis, for taking the time to review this thesis and provide feedback. I have greatly benefited from their insight, experience, and scientific leadership.

In addition, I am grateful to many wonderful collaborators I have worked with during my research career so far: Alekh Agarwal, Kyunghyun Cho, Danielle Belgrave, Isabelle Guyon, Thanh Gia Hieu Khuong, Alice Oh, Benedictus Kent Rachmat, Ivan Stelmakh, Ihsan Ullah, Zhiwei Steven Wu, and Zhen Xu.

Finally, I want to thank my friends and family for their constant and invaluable support.

Contents

1	Introduction	1
1.1	Contributions and Organization	2
2	Experimental Evidence from Peer Review	5
2.1	Measuring the Reliability of Review-Quality Evaluations	5
2.1.1	Motivation	6
2.1.2	Background	7
2.1.3	Study Design	8
2.1.4	Sources of Bias and Disagreement	11
2.1.5	Discussion	19
2.2	LLM Checklist Assistance Before Peer Review	20
2.2.1	Motivation	21
2.2.2	Background	21
2.2.3	Assistant Design and Deployment	22
2.2.4	Evaluation of Author Use	23
2.2.5	Robustness and Deployment Constraints	26
2.2.6	Discussion	27
3	Randomized Selection from Imperfect Evaluations	29
3.1	Robust Randomization Under Interval Uncertainty	30
3.1.1	Motivation	31
3.1.2	Lottery Deployments and Modeling Approach	33
3.1.3	Problem Formulation	36
3.1.4	Efficient MERIT Algorithm	38
3.1.5	Axiomatic Comparison of Lottery Rules	42
3.1.6	Empirical Evaluation	46
3.1.7	Discussion	51
3.1.8	Deferred Proofs	53
3.2	Smooth Lotteries for Stable Selection	54
3.2.1	Motivation	54
3.2.2	Background	56
3.2.3	Problem Formulation	58
3.2.4	Designing an Optimal Smooth Lottery	59
3.2.5	Comparisons to Other Stability Notions	63

3.2.6	Instability of Existing Partial Lottery Designs	66
3.2.7	Empirical Smoothness–Regret Comparison	67
3.2.8	Discussion	69
3.2.9	Deferred Proofs	71
4	Privacy-Preserving Release of Evaluation Data	73
4.1	Timing Privacy for Batched Activity	74
4.1.1	Motivation	74
4.1.2	Background	77
4.1.3	Problem Formulation	79
4.1.4	One-Sided Privacy Guarantees	83
4.1.5	Practical Considerations for Implementation	91
4.1.6	Empirical Linkage Attacks	94
4.1.7	Discussion	98
4.1.8	Deferred Proofs	100
4.2	Private Benchmarking for Fraud Detection	101
4.2.1	Motivation	101
4.2.2	Benchmark Privacy Risks	104
4.2.3	Problem Formulation	107
4.2.4	Background	109
4.2.5	Private Evaluation by Subsample-and-Aggregate	111
4.2.6	Private Evaluation by Synthetic Data Generation	112
4.2.7	Benchmarking Experimental Setup	115
4.2.8	Empirical Comparison of Private Benchmarking Methods	117
4.2.9	Discussion	120
4.2.10	Deferred Proofs	121
5	Conclusions	123
	Bibliography	125

List of Figures

2.1	Distribution of review-quality evaluation scores	10
2.2	Generating uselessly elongated reviews	10
2.3	Word counts of original and elongated reviews	11
2.4	Author outcome bias in review-quality evaluations	14
2.5	Inter-evaluator disagreement in review-quality scores	16
2.6	Example Checklist Assistant feedback	22
2.7	Author survey responses before and after Checklist Assistant use	24
2.8	Reported issues with Checklist Assistant feedback	24
2.9	Common categories of Checklist Assistant feedback	25
2.10	Changes in checklist response length after resubmission	26
2.11	Checklist Assistant score changes after resubmission	26
2.12	Checklist Assistant scores under adversarial revisions	27
3.1	Monotonicity failure under changing selection budgets	45
3.2	Maximum instability in threshold-based lotteries	45
3.3	Expected and worst-case utility of randomized selection methods	49
3.4	Selection utility under varying miscalibration levels	50
3.5	Worst-case utility under increasing missing review scores	50
3.6	Smooth lottery response to a review-score perturbation	55
3.7	Clipped Linear Lottery example	60
3.8	Regret and smoothness of existing partial lottery mechanisms	68
3.9	Regret versus smoothness for smooth lotteries and softmax	69
3.10	Empirical tightness of smoothness guarantees	69
4.1	Cartoon of reviewer deanonymization from batching	74
4.2	Reviewer deanonymization risk from batched timing	82
4.3	Expected delay under one-sided private delay distributions	86
4.4	Pareto frontier for delay added to batched and unbatched comments	89
4.5	Linking Wikipedia revisions using batched timing	95
4.6	Linkage attacks on Bitcoin transactions after privacy delay	97
4.7	Delay added to batched Bitcoin transactions	98
4.8	DP benchmarking error for fraud detectors on graph data	102
4.9	Privacy attack on a face-recognition benchmark	106
4.10	Bias from subsample-and-aggregate for fraud detector AUC	118
4.11	Error in DP synthetic graph sufficient statistics	119
4.12	AUC error for GNN fraud detectors on synthetic Yelp graphs	119

List of Tables

2.1	Effect of uselessly elongated reviews on overall quality scores	13
2.2	Effect of uselessly elongated reviews on criteria scores	13
2.3	Authors' bias toward positive review recommendations	15
2.4	Linear calibration model for review-quality scores	17
3.1	Marginal acceptance probabilities under MERIT and Swiss NSF	51
3.2	Stability definitions for randomized selection	63
4.1	Delay added to Wikipedia revisions by the privacy mechanism	94
4.2	Graph datasets for private fraud benchmarking	115

Chapter 1

Introduction

Many consequential decisions depend on judgments from distributed human evaluators. Scientific conferences ask reviewers to assess papers, funding agencies ask experts to judge proposals, and institutions in hiring, admissions, medicine, law, online governance, and AI evaluation all rely on human assessments to compare complex alternatives. These evaluations are often partial, subjective, and inconsistent, and decision makers usually lack an objective ground truth. Scientific peer review is one of the clearest examples. It is central to research, but it is also known to suffer from bias, subjectivity, fraud, miscalibration, inconsistency, and low effort [186].

This thesis studies how to make better decisions from distributed human evaluations under uncertainty. We focus on three sources of uncertainty. First, the evaluations themselves are error-prone. Reviewers may use scales differently, disagree about what quality means, or respond to irrelevant features of a submission. Second, the target of evaluation is often contested or unobserved. There is no definitive answer to which proposal would have produced the best science or which applicant would have succeeded in a position. Moreover, in many of these domains there is no clear single unit of measure with which to define the success or utility of a selection. Third, the data needed to study these systems are often private or incomplete, making it hard to know which statistical model of the evaluation process should be trusted in any given deployment.

The thesis develops methods for addressing these sources of uncertainty. We ask how to reliably measure sources of error in human reviews, how to make decisions in light of known issues with human evaluations, and what data can be released to support external scrutiny. This motivates three complementary approaches. We use experiments to measure when reviewer judgments and interventions succeed or fail; randomized selection mechanisms to represent uncertainty instead of hiding it behind arbitrary thresholds; and privacy-preserving release mechanisms to make evaluation systems more legible without making participation unsafe. Across these settings, the goal is not to replace human evaluation with a single automated rule. It is to build methods that respect the limits of human judgment while improving the decisions, evidence, and institutions built from it.

1.1 Contributions and Organization

First, Chapter 2 begins by addressing uncertainty through improved empirical understanding of real-world peer review processes. It presents two large-scale human-subjects experiments conducted inside live peer review processes at ML/AI conferences. The first asks whether people can reliably evaluate the quality of peer reviews. We study review-quality evaluations from authors, reviewers, and meta-reviewers at NeurIPS 2022, including a randomized controlled trial that isolates whether uselessly increasing the length of a review changes its perceived quality. The results show that evaluations of reviews inherit many of the same pathologies as reviews themselves: disagreement, miscalibration, author-outcome bias, and a preference for needlessly long reviews. This limits how much weight human review-quality scores can carry as measurements of true review quality. This study is based on joint work with Ivan Stelmakh, Kyunghyun Cho, Alice Oh, Alekh Agarwal, Danielle Belgrave, and Nihar B. Shah [71].

The second study in Chapter 2 evaluates a live deployment of an “LLM Checklist Assistant” at NeurIPS 2024. This is a deliberately narrow use of LLMs in peer review: the assistant gives authors feedback on whether they met concrete reporting guidelines in their papers before submission. We use it to characterize what scoped automated feedback can and cannot do for reviewing pipelines and find that the assistant surfaced concrete issues that many authors then fixed. This study is based on joint work with Ihsan Ullah, Thanh Gia Hieu Khuong, Benedictus Kent Rachmat, Zhen Xu, Isabelle Guyon, and Nihar B. Shah [68]. Together, the two studies show that interventions in peer review should be evaluated in the systems where they are deployed, because the same uncertainty that affects ordinary reviewing can also affect the signals used to improve it.

Second, Chapter 3 turns from measurement to algorithmic decision-making. Randomized selection, or explicit lotteries over qualified candidates, has been gaining adoption in scientific funding as a response to uncertainty in evaluation. Proponents have argued that randomization may reduce wasted reviewer effort on arbitrary tie-breaks, encourage high-risk proposals, counteract “rich-get-richer” effects, and reduce reviewer partiality [59, 60, 73, 79, 80]. Surveys also find substantial support among scientists for introducing randomized selection into grant funding [131, 169]. In practice, partial lotteries have been adopted by funding agencies including the New Zealand Health Research Council, the Swiss National Science Foundation, the European Research Council, Science Foundation Ireland, and the Volkswagen Foundation [1, 56, 131, 184, 218].

Despite this growing adoption, existing lottery designs are often ad hoc. Institutions may identify a set of “similarly qualified” candidates and sample uniformly from that set, without a formal account of why that distribution is appropriate. The central question in Chapter 3 is therefore: *given imperfect evaluations of candidate quality, what probability distribution should the decision maker use?* This question is especially important when decision makers do not trust a fully specified probabilistic model of review error. Probabilistic models of reviewer miscalibration have often performed poorly in real peer-review deployments [117, 187], and human miscalibration can be more complex than simple models assume [21]. Moreover, the future success of scientific work is inherently difficult to predict; peer-review scores are poor predictors of future citations and other long-run measures of impact [33, 183, 226]. Similar instability appears in other selection settings, including college admissions, where predictive models can yield

arbitrary rankings under small changes to the training data [119].

We study two algorithmic designs that target different, sometimes complementary desiderata. First, MERIT is a robust lottery for selecting from *interval-valued quality estimates* when the decision maker has uncertainty modeled by “Knightian uncertainty”: the decision maker may be able to describe a set of plausible orderings, but not assign a reliable probability measure over them [105, 196]. This mechanism is based on joint work with Giulia Fanti and Nihar B. Shah [69]. Second, “smooth lotteries” address a different motivation for randomization: stability. They guarantee that small changes in review scores produce only small changes in selection probabilities. This mechanism is based on joint work with Giulia Fanti and Nihar B. Shah [72]. MERIT targets robustness when the noise model itself is uncertain; smooth lotteries target stability when threshold effects would otherwise let tiny perturbations swing outcomes.

Third, Chapter 4 studies a data-release problem that sits behind both measurement and decision-making. Outside scrutiny of evaluation systems requires data about participation, reviews, outcomes, and algorithmic performance. But the same releases can re-identify participants or leak sensitive structure. This is not an abstract concern in peer review: reviewer identities, assignment patterns, bidding behavior, author outcomes, and discussion traces can all be sensitive. The chapter studies this tension in two settings. The first concerns pseudonymous time-series data, where batching behavior can deanonymize users even when their actions are recorded under different pseudonyms. We formalize this risk and develop one-sided differentially private delay mechanisms for sharing useful timing information while preserving anonymity. This study is based on joint work with Giulia Fanti and Nihar B. Shah [67].

The second privacy study concerns fraud-detection benchmarks on graph data. A data holder may want to evaluate submitted detection algorithms without releasing the underlying graph, but the benchmark score itself can leak information. We show that submitted algorithms can encode private graph queries and that off-the-shelf differentially private tools struggle to deliver useful accuracy on graph-structured data. This study is based on joint work with Giulia Fanti, Nihar B. Shah, and Zhiwei Steven Wu [70]. This setting illustrates a broader obstacle to transparency: the data needed to audit evaluation systems are often exactly the data that create privacy risk.

The three themes of the thesis are complementary in targeting the broader goal of improving review systems. The experiments in Chapter 2 identify which errors are worth designing around. The mechanisms in Chapter 3 take uncertainty as an input to decision-making rather than treating it as a nuisance to ignore. The privacy tools in Chapter 4 aim to keep these systems legible to outside scrutiny as they scale. The central claim is that better distributed evaluation requires joint attention to measurement, decision-making, and privacy: we need to know what evaluations can support, choose mechanisms that match uncertainty arising from error-prone evaluations, and release data to evaluate the system without making participation unsafe.

Chapter 2

Experimental Evidence from Peer Review

This chapter studies distributed human evaluation through two deployments inside large computer science peer-review processes. Both studies start from the same practical concern: peer review systems increasingly rely on secondary evaluations to improve the primary evaluation process, but these secondary signals are themselves produced by humans or by imperfect automated assistants. If such signals are noisy, biased, or strategically manipulable, then using them to reward reviewers, evaluate interventions, or certify submissions can reproduce the same weaknesses the system was meant to fix.

The first subchapter asks whether people can reliably evaluate the quality of peer reviews. We study review-quality evaluations from authors, reviewers, and meta-reviewers at NeurIPS 2022, including a randomized experiment that isolates whether uselessly increasing review length changes perceived quality. This setting gives a concrete way to measure whether evaluations of reviews capture stable review quality or instead reflect length, author outcomes, evaluator disagreement, and calibration differences.

The second subchapter asks whether a narrow LLM assistant can help authors satisfy checklist requirements before review. The deployment at NeurIPS 2024 is deliberately scoped: the assistant gives feedback to authors, not reviewers, and targets compliance with explicit checklist criteria rather than holistic judgments of paper quality. The study therefore examines a lower-risk use of LLMs in peer review while still surfacing questions about accuracy, usability, user expectations, and gaming.

Together, the two studies show both the promise and the fragility of using secondary evaluations to improve peer review. Evaluations can reveal useful signals and can prompt substantive changes, but they also inherit biases, inconsistency, and strategic risks. This motivates the later chapters, which ask how decision rules and data-release mechanisms can be designed once evaluation signals are known to be imperfect.

2.1 Measuring the Reliability of Review-Quality Evaluations

This subchapter begins the experimental part of the thesis by examining whether peer-review systems can reliably evaluate their own evaluation process. Before review-quality scores are used to reward reviewers or evaluate interventions, it is necessary to understand whether those scores be-

have like stable measurements or whether they reproduce the same biases and disagreement that affect paper review. The next subchapter turns from measuring human secondary evaluations to deploying an LLM assistant for authors; together, the two studies ask when auxiliary evaluation signals can be trusted in peer review.

This subchapter is based on joint work with Ivan Stelmakh, Kyunghyun Cho, Alice Oh, Alekh Agarwal, Danielle Belgrave, and Nihar B. Shah [71].

2.1.1 Motivation

Scientific peer review is a ubiquitous process used across many fields to evaluate research quality. While the peer review of papers is widespread, it is plagued with well-documented problems like bias, subjectivity, fraud, miscalibration, and low effort, among others (see [186] for a survey). Some of these problems may be mitigated via design of better incentives for high-quality reviewing, or via evidence-based policy design evaluated through controlled experiments. Both of these approaches depend on reliable evaluations of the quality of reviews. Therefore, this subchapter studies the research question: can different parties involved in the peer-review process (meta-reviewers, reviewers, authors) reliably evaluate the quality of reviews? We are driven by the two primary motivations:

1. *Designing incentive mechanisms for high-quality reviewing.* A number of past works propose incentive mechanisms for the peer-review process to motivate better reviewing [120, 193, 206, 233, 234]. For example, reviewers may earn credit towards future peer review of their own work when they complete high-quality peer reviews of others’ work. Already, at a number of journals and conferences, reviewers can be recognized for excellence in reviewing (e.g., NeurIPS “Top Reviewers”) where this recognition is generally given out on the basis of evaluations of review quality completed by editors or meta-reviewers. The European Science Foundation reports that many grant organizations evaluate quality of reviews and a substantial fraction of these organizations store the evaluations linked to the reviewers’ identities in their databases [64]. These mechanisms generally require reliable evaluation of review quality in order for incentives to be fair and useful. For example, [233] and [234] assume that authors will accurately provide a report of true review quality that can be used to incentivize effort on the part of reviewers.
2. *Experiments measuring efficacy of interventions in the peer-review process.* In numerous studies examining scientific peer review, the efficacy of changes to the peer-review process is assessed based on evaluating the quality of reviews under certain policy interventions (e.g., [22, 23, 25, 29, 36, 57, 111, 123, 168, 182, 195, 212, 213, 219]). These studies treat evaluations of review quality by fellow reviewers or editors/meta-reviewers as “gold standard” to measure the efficacy of the policies under consideration. In our research, we delve into the validity of using these scores by examining their reliability as true indicators of review quality.

Motivated by the need for evaluations of review quality, we conducted a quantitative study into the reliability of evaluating review quality at the Neural Information Processing Systems (NeurIPS) 2022 conference, a top-tier conference in the field of machine learning.¹ We recruited

¹In computer science, unlike many other research fields, conferences typically review full papers, are frequently

participants who served in different roles in the conference — paper authors, paper reviewers, and meta-reviewers who handle many papers at the conference. We then asked these participants to evaluate the quality of paper reviews and analyzed the reliability of their scores in several ways. Additionally, we conducted a randomized controlled trial to examine potential bias in scores of perceived quality towards longer reviews.

Using the data collected we assess the reliability of evaluating review quality along five dimensions: (i) uselessly elongated review bias, (ii) author-outcome bias, (iii) inter-evaluator agreement, (iv) miscalibration, and (v) subjectivity. Overall, our findings suggest that the evaluation of paper reviews faces many of the same issues as the reviewing of paper quality, like inconsistency, miscalibration, subjectivity, and biases with respect to irrelevant information. Therefore, care must be taken in relying on evaluation scores to either incentivize quality peer review or to experimentally measure changes in the quality of review due to these observed effects in evaluating review quality.

2.1.2 Background

Prior work on review-quality evaluation provides important context for our experimental design. The most relevant studies have surveyed authors, reviewers, journal editors, or meta-reviewers to understand how review quality is perceived and whether these perceptions are reliable.

At the computer vision conference CVPR 2012, a study [102] asked paper authors to evaluate reviewer quality. They found that length had a weak positive correlation with authors’ ratings of “helpfulness.” However, importantly, it is not possible to distinguish how much of the correlation was due to longer reviews having truly higher quality content versus longer reviews being spuriously perceived as higher quality. This subchapter addresses the issue of confounding by rigorously measuring the causal effect of length on perceived review quality through a randomized controlled trial where the treatment increases the length of the review without adding useful information.

The papers [102, 116, 165, 170, 224] all find that in authors’ evaluations of reviews on their own papers, the decision of accept or reject given by the reviewer is highly correlated with evaluation rating given by the authors. However, these prior works do not control for potential confounders. For instance, there may be systematic differences in the true review quality of accept and reject decisions. Here, we also collect evaluations of reviews by non-authors, which we use to control for these confounders. A related paper is [221] which develops an algorithm to de-bias such author-provided evaluations.

At NeurIPS 2020, the program chairs asked meta-reviewers to rate whether paper reviewers met their expectations [126]. They found that invited reviewers to the conference were not rated any higher than reviewers recruited from among the author pool. Additionally, they found that less experienced reviewers were actually rated slightly higher on review quality than more experienced reviewers. Similarly, a study at the ICML 2020 conference [195] designed a special process to recruit new paper reviewers and asked meta-reviewers to evaluate the review quality from this group and from the standard group of reviewers. They found that their newly recruited and trained reviewers were evaluated as higher quality than reviewers in the standard reviewer

a terminal publication venue and are ranked higher than journals.

pool according to a number of metrics which also included meta-reviewers’ evaluations of reviews. This subchapter does not focus on which reviewers are considered higher quality by meta-reviewers, but rather focuses on the reliability of these evaluations of reviews.

A number of scientific funding agencies collect assessments of peer review quality in the assessment of grant proposals. At Canada’s national health research funding agency, committee chairs were asked to evaluate the review quality of grant peer reviewers from 2019 to 2022 [5]. A report from the European Science Foundation on the evaluation of reviews found that such evaluations of review quality were quite common in grant funding agencies—in a survey of 30 funding organizations, they found that over 60% evaluate the quality of all reviews as standard practice [64]. These organizations then use review quality in a number of concrete ways, including to discard reviews deemed low quality and tagging the reviewer with qualifying information for future reference. These policies speak to the importance of assessments of review quality in having real consequences in existing peer review systems of funding agencies. The goal here is to systematically assess the reliability of evaluations of review quality.

In medical journals, there is literature going back over two decades on assessing review quality. The study [61] asked editors to evaluate the quality of peer reviews in medical journals and concluded that editors show strong agreement in their evaluations as measured by the intraclass correlation coefficient. Subsequent work [24] tested the efficacy of evaluating reviews by generating a fictitious manuscript with known flaws, obtaining peer reviews of the manuscript and then asking editors to evaluate quality of the peer reviews. They found that evaluation of review quality is somewhat correlated with number of flaws reported by the reviewers, indicating that assessment of review quality may in fact capture some objective qualities that make a review useful. In a cross-sectional study of journals in multiple disciplines, the study [170] analyzed authors’ and editors’ evaluations of review quality in Elsevier journal reviews from 2014 across medicine, science, and computer science. They found correlation between author satisfaction with the review and whether the review recommended acceptance. This subchapter studies similar questions on the reliability of evaluating peer review, but in the context of a large Computer Science conference.

A recent paper [135] analyzed whether length of reviews seems to capture review quality. They found a correlation between the length of reviews given to accepted journal articles and the future citations received by these articles, suggesting that review length may be associated with review quality. While it may be the case that longer reviews are sometimes of higher quality than shorter reviews, this subchapter asks whether uselessly elongating reviews can lead to spurious perceptions of higher quality.

Recent work [44, 45, 146, 192] has found that large language model (LLM)-based evaluations of LLM response quality tend to display a strong bias towards longer responses. This can in turn bias models that are fine-tuned using reinforcement learning from human feedback (RLHF) based on LLM evaluations towards generating long responses. The results here show that this length bias exists even in human evaluations in the context of scientific peer review.

2.1.3 Study Design

Throughout this subchapter, we use “evaluator/evaluation” to refer to the evaluation of reviews and “review/reviewer” to refer to reviews of papers.

We asked participants at NeurIPS 2022 to evaluate the quality of reviews given on papers at the conference. We recruited four types of evaluators:

1. *Meta-Reviewers*: Asked to evaluate reviews on one paper from their own pool of papers.
2. *Paper Reviewers*: Asked to evaluate other reviewers’ reviews on one paper that the participant reviewed for during the conference.
3. *Paper Authors*: Asked to evaluate all reviews on at most 2 of their own submitted papers.
4. *External Reviewers*: Reviewers and meta-reviewers from NeurIPS 2022 with relevant expertise who were asked to evaluate all reviews on one paper that they did not handle as part of the conference.

We recruited evaluators on an opt-in basis. First, a notification was sent to all reviewers, meta-reviewers, and authors asking if they were interested in participating. Those who said yes were included. Given the set of opt-in evaluators, we next chose papers and reviews for them to evaluate in a manner that maximized the amount of overlap in which reviews are evaluated. This was to enable us to then compare the evaluations from multiple evaluators on the same set of reviews. Additionally, in order to ensure that the external reviewers evaluated reviews on relevant papers, we chose papers so that “similarity” between the external reviewers and papers was high — here, similarity is defined as the similarity between the text of the paper and the text of the reviewers’ profile (past papers), which is used in NeurIPS 2022 and various other conferences to assign reviewers to papers in the peer review process. Overall, we recruited 7,740 evaluators across these 4 types of reviewers who rated 9,870 paper reviews, with a total of 24,638 evaluations completed. Among the participants, there were 493 meta-reviewers, 2,395 paper reviewers, 3,429 paper authors, and 1,423 external reviewers.

Evaluators were provided the review, along with the paper for which the review was written. Evaluators were asked to rate the overall quality of paper reviews on a 7 point scale. Higher ratings correspond to higher rated quality. Additionally, evaluators were asked to evaluate the reviews on the following four criteria:

1. *Understanding*: “The review demonstrates an adequate understanding of the paper.”
2. *Coverage*: “The review covers all the required aspects.”
3. *Substantiation*: “Evaluations made in the review are well supported.”
4. *Constructiveness*: “The review provides constructive feedback to authors.”

Evaluators rated each of these criteria on a 5-point Likert scale ranging from -2 (Strongly Disagree) to 2 (Strongly Agree). The evaluation form also contained additional explanation of each of the items: see Supplemental Material for the full questionnaire. We chose these criteria for the questionnaire based on proposed Review Quality Indicators (RQIs) for peer reviews [197, 211], additionally tailoring the questions to suit our needs of being concise and relevant to papers in the domain of machine learning.

We describe some basic statistics pertaining to the evaluations. In Figure 2.1, we show the overall distribution of scores for each type and the distribution of criteria scores. The overall score distribution is symmetric around the median score of 4. The distribution of scores for the criteria are all left-skewed, as evaluators were more likely to give positive scores on these criteria. We further analyze the mapping from criteria scores to overall scores in Section 2.1.4.

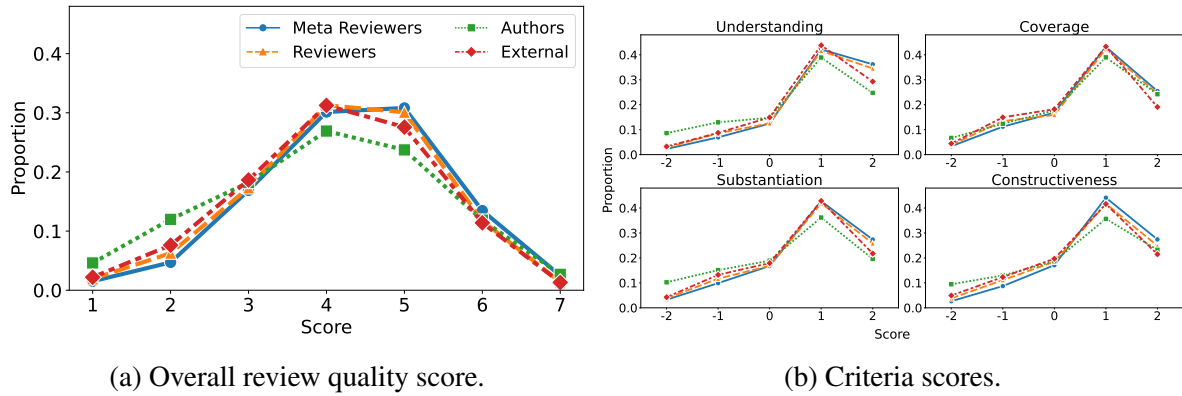


Figure 2.1: Marginal distribution of scores given to reviews by meta-reviewers, paper reviewers, authors, and external reviewers.

A Randomized Control Trial

We also conducted a randomized controlled trial where we manipulated the length of reviews in order to study the impact of review length on perceptions of quality. Specifically, we conducted an experiment where we selected 10 papers such that as many participants as possible had high textual similarity scores (indicating familiarity in the area of the paper) with at least one of the papers. The participants with high similarity scores were drawn from among the external reviewers, giving 458 total evaluators, 334 who served as reviewers and 124 who served as meta-reviewers on other papers at the conference. Importantly, unlike (meta)-reviewers and authors, the participants from this group of external reviewers had not seen the original reviews on these papers, allowing us to manipulate the reviews without their knowledge of the treatment.

For each of the selected papers, we chose one review at random and then manually created a longer version of this review, carefully ensuring that the underlying quality of the review did not improve as we increased the length. We adopted a combination of the following strategies to

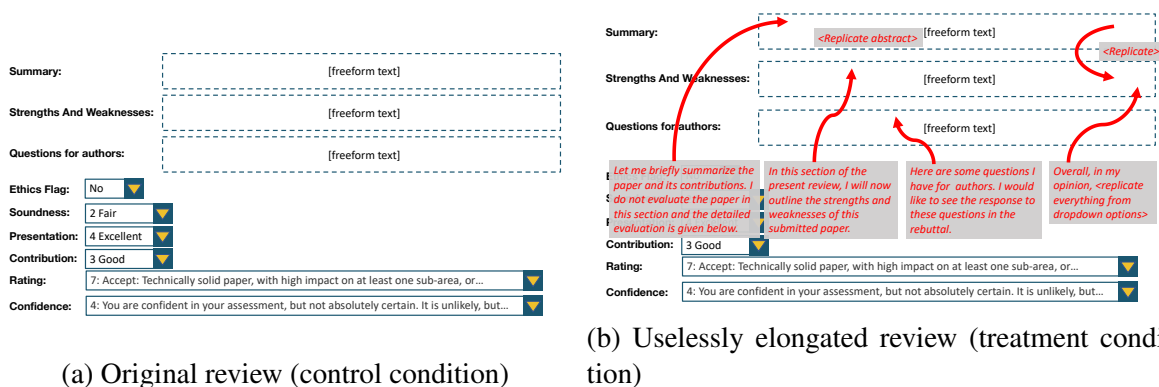


Figure 2.2: Generation of “uselessly elongated” reviews by adding unnecessary explanatory text (in red).

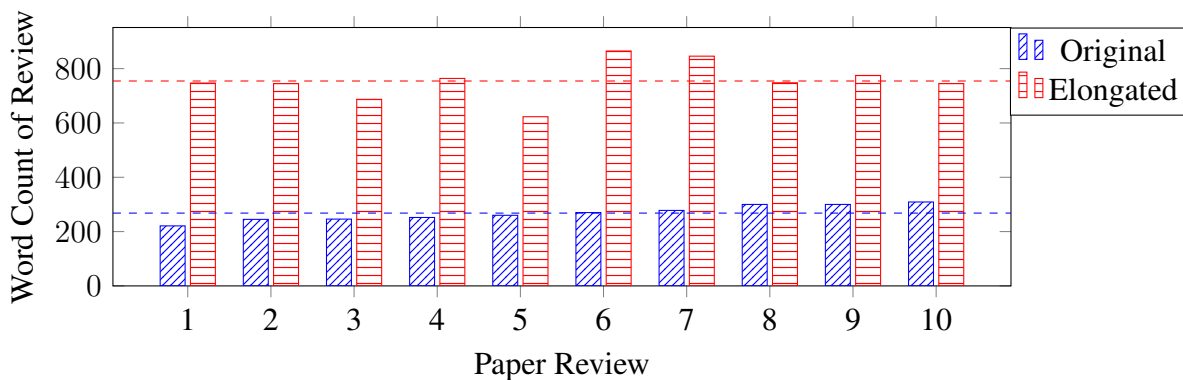


Figure 2.3: Word count of the ten original and uselessly elongated reviews. The word counts include text from the summary, strengths and weaknesses and questions boxes of the paper reviews and exclude the quantitative scores. The mean lengths of original and elongated reviews are shown as dashed lines.

do so: adding filler text at the beginning of each text box by repeating the text box header as an introductory sentence, repeating the summary in other sections like strengths and weaknesses, writing out the text from multiple-choice questions (Rating, Ethics Flag, Soundness, Presentation, etc.) in the text boxes, replicating the abstract of the paper in the summary box or in the body text of the review. See Figure 2.2 for an illustration of such an elongation. In Supplemental Material, we give examples of original and elongated reviews used in our experiment that pertain to accepted papers at NeurIPS 2022 which have publicly viewable reviews on OpenReview. As shown in Figure 2.3, across the 10 reviews the original reviews were roughly 200-300 words long, while the elongated reviews were roughly 600-850 words long. The mean word count of the original reviews was 268 words, compared to a mean of 755 for elongated reviews.

Then, each eligible participant was assigned to exactly one of the experiment papers. Additionally, each participant was assigned uniformly at random to either a “long” or “short” condition. When asked to evaluate a review for the assigned paper, participants in the *long* group were given the uselessly elongated version of the selected review while participants in the *short* condition were given the original version of the review. Participants were not informed about the specific goal of this additional experiment: we only notified them that the data they contributed would be used to gain insights about the review quality evaluation practice, but did not specifically mention the length confounder. We further discuss the setup of this experiment and our analysis in Section 2.1.4. We note that the experimental data from the RCT is not used in the rest of our analysis.

2.1.4 Sources of Bias and Disagreement

We now present the main results of our analyses on uselessly elongated review bias (Section 2.1.4), authors’ outcome-induced bias (Section 2.1.4), inter-evaluator (dis)agreement (Section 2.1.4), miscalibration (Section 2.1.4), and subjectivity (Section 2.1.4).

Bias Toward Uselessly Elongated Reviews

One concern in evaluations of reviews is that the evaluations may be biased by spurious factors that are not actually indicative of underlying quality, like review length. We hypothesize that evaluators may perceive longer reviews as better even if they are not of higher quality. In order to rigorously test this hypothesis, we conduct a carefully designed randomized controlled trial for the effect of “uselessly elongated review bias.”

Methods In our experiment, we used 10 reviews written on 10 different papers. For these 10 reviews, we received evaluations from 458 participants, who were either reviewers or meta-reviewers of some other papers at NeurIPS 2022. Each of the 10 reviews had two versions — the original *short* version and a *long* version, which was a uselessly elongated version of the same review containing more words but the same underlying content. Then, each of the participants was randomly assigned to either a *short* or *long* condition, meaning they reviewed either the short or long version of a review respectively. We then employed the Mann-Whitney U test to evaluate whether the perceived quality of the 10 selected reviews differs systematically between the *short* and *long* conditions. We compute a Mann-Whitney U-statistic as follows. We take all pairs of evaluations where the two evaluations are of a review on the same paper but one evaluates the short version and the other the long version. There are on average 23 evaluations per paper of the short version of the review and 23 evaluations per paper of the long version giving over 500 pairs of evaluations per paper. For each paper $p \in [10]$ we denote S_p as the set of evaluation scores of the *short* review on the paper and L_p the set of scores of the *long* review on the paper. Then, the test statistic $\tau \in [0, 1]$ is defined as:

$$\tau = \frac{1}{\sum_{p=1}^{10} |L_p||S_p|} \sum_{p=1}^{10} \sum_{x^s \in S_p} \sum_{x^\ell \in L_p} (\mathbb{I}(x^\ell > x^s) + 0.5 \mathbb{I}(x^\ell = x^s)).$$

One can interpret τ as the probability that a *long* review is scored higher than a *short* review by evaluators, breaking ties in scores at random. Note that under a null hypothesis of no effect, $\tau = 0.5$, so $\tau > 0.5$ indicates a positive bias of review length on quality score and $\tau < 0.5$ indicates negative bias.

To compute confidence intervals for the test statistic τ , we bootstrap reviewers in the *long* and *short* conditions within each review. Specifically, for 5,000 iterations, we independently bootstrap L_p and S_p for each review on each paper $p \in [10]$ and compute the test statistics on the bootstrapped set of reviewers. We then use 2.5 and 97.5 percentiles to construct a 95% confidence interval.

To test whether reviewers in the *long* and *short* conditions systematically differ in their scores, we apply a two-sided Fisher permutation test. For this, we permute evaluators within each review between the *long* and *short* conditions uniformly at random, ensuring that the number of reviewers in each condition remains the same. We then recompute the value of the test statistic for 20,000 permutations and compare these values with the original value of the test statistic to obtain p -values.

ROLE	SAMPLE SIZE	τ	95% CI	P VALUE	DIFFERENCE IN MEANS
REVIEWERS + META-REVIEWERS	458	0.64	[0.60, 0.69]	< 0.0001	0.56
REVIEWERS	334	0.65	[0.59, 0.71]	< 0.0001	0.58
META-REVIEWERS	124	0.61	[0.52, 0.71]	0.04	0.39

Table 2.1: Summary of results for the randomized controlled trial testing the effect of uselessly elongated review bias on overall quality score, separated according to the role of the evaluator in the conference.

CRITERIA	τ	95% CI	P VALUE	DIFFERENCE IN MEANS
OVERALL	0.64	[0.60, 0.69]	< 0.0001	0.56
UNDERSTANDING	0.57	[0.53, 0.62]	0.04	0.25
COVERAGE	0.71	[0.66, 0.76]	< 0.0001	0.83
SUBSTANTIATION	0.59	[0.54, 0.64]	0.001	0.31
CONSTRUCTIVENESS	0.6	[0.55, 0.64]	0.001	0.37

Table 2.2: Summary of results for the randomized controlled trial testing the effect of uselessly elongated review bias on criteria scores. Sample size is 458 for all statistics. Recall that the overall score is on a 7-point scale, while criteria scores are on a 5-point scale.

Results As shown in Table 2.1, we find a statistically significant positive impact of length on evaluations of review quality. For both reviewers and meta-reviewers, the uselessly elongated reviews receive higher scores than the original shorter reviews. The effect size for reviewers is similar to the effect size for meta-reviewers. Overall, the mean score for the *long* condition group was 4.29 compared to 3.73 for the *short* condition. As shown in Table 2.2, we also find a positive effect of length on the criteria scores. In particular, after Holm–Bonferroni correction, results are significant at level 0.05 for all the criteria, with the strongest effect on Coverage. These results suggest that it is possible for a reviewer to spuriously improve perceived quality of their review by adding to their review, even if the additions add no real value.

Author Outcome Bias

One potential source of bias in evaluating review quality that is distinct to authors is bias arising due to the positivity or negativity of a review. A number of past works have documented correlation between author’s satisfaction with paper reviews and whether the reviews recommended acceptance [102, 116, 165, 170, 224]. We find a similar correlation in our analysis. In Figure 2.4, we plot the review score given by a paper review against the mean evaluation of review quality given to that review for each type of evaluator. While meta-reviewers, reviewers and external reviewers do not show a strong trend in how the positivity of review score correlates with review quality assessments, for authors there is a clear positive trend with reviews recommending strong accepts receiving higher evaluations than reviews recommending strong rejects. This trend holds both for the overall review quality score and for assessments of specific criteria. While this visual

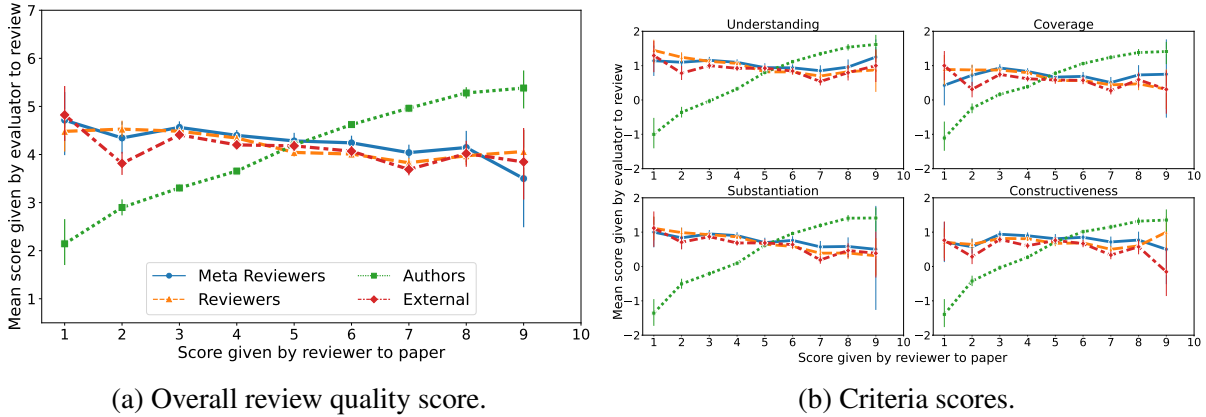


Figure 2.4: Review score given to a paper by a reviewer (x axis) plotted against the mean evaluation score of that review by evaluators (y axis), for each type of evaluator. Review scores range from 1 (strongest reject) to 10 (strongest accept.) Evaluations of reviews with a score of 10 are omitted from plot due to insufficient sample size ($n = 5$.)

suggests such a bias, it does not account for confounding factors, and hence we conduct a formal analysis in this section.

Methods In order to measure the presence of an outcome-induced bias in the evaluations of reviews provided by authors of respective papers, we estimate the effect of receiving a review with a “reject” recommendation versus an “accept” decision on authors’ evaluations of review quality. We conduct the following non-parametric analysis. We match pairs of evaluations where one evaluation is on a reject review (“weak reject” or below) and the other is on an accept review (“weak accept” or above) based on the following criteria:

1. Evaluation is done by the same author on the same paper.
2. The pair of reviews evaluated have similar length: the longer review is at most $1.5\times$ longer than the shorter.
3. Both reviews have at least 2 evaluations from non-authors and have received a mean overall evaluation score within 1 point of each other from non-authors.

Our matching criteria yields 418 pairs of evaluations. We then conduct a Mann-Whitney U test on the pairs of evaluations to determine whether accept reviews are likely to receive higher scores than reject reviews. In particular, given the $n = 418$ pairs of scores $\{(x_i^{\text{accept}}, x_i^{\text{reject}})\}_{i=1}^n$, the test statistic $\tau \in [0, 1]$ is computed as:

$$\tau = \frac{1}{n} \sum_{i=1}^n \left(\mathbb{I}(x_i^{\text{accept}} > x_i^{\text{reject}}) + 0.5\mathbb{I}(x_i^{\text{accept}} = x_i^{\text{reject}}) \right).$$

One can interpret the test statistic τ as the probability that an accept rating is scored higher than a reject rating by authors, breaking ties in scores at random. We run a two sided Fisher permutation test with 20,000 simulations to determine a p -value of the test statistic. The 95% confidence intervals are bootstrapped with 10,000 simulations.

CRITERIA	τ	95% CI	P VALUE	DIFFERENCE IN MEANS
OVERALL	0.82	[0.79, 0.85]	< 0.0001	1.41
UNDERSTANDING	0.78	[0.75, 0.81]	< 0.0001	1.12
COVERAGE	0.76	[0.72, 0.79]	< 0.0001	0.97
SUBSTANTIATION	0.80	[0.76, 0.83]	< 0.0001	1.28
CONSTRUCTIVENESS	0.77	[0.74, 0.80]	< 0.0001	1.15

Table 2.3: Summary of results for Mann-Whitney U test of authors’ bias towards reviews recommending accept compared to reviews recommending reject (on $n = 418$ pairs of reviews).

Results We find a treatment effect of $\tau = 0.82$ with a p -value of < 0.0001 in the overall quality scores, indicating that authors are positively biased towards reviews recommending accept over reviews recommending reject. Additionally, on average accept reviews received scores that were 1.406 points higher (on the 7 point evaluation scale) than reject reviews. We additionally test for differences in the criteria scores between the matched pairs of accept reviews and reject reviews. As shown in Table 2.3, we find a positive bias towards accept reviews on the Understanding, Coverage, Substantiation, and Constructiveness criteria respectively. These results are all statistically significant at a level of 0.05 after Holm-Bonferroni correction. The criteria scores were roughly 1 point higher on the 5-point review scale for accept reviews than Reject reviews. This indicates that authors’ positive bias towards reviews recommending accept manifests in criteria scores as well as overall scores. We note that authors did not have any explicit incentive in our experiment to rate accept reviews higher than reject reviews: there were no repercussions to paper reviewers for receiving positive or negative evaluation scores for their paper reviews nor for the acceptance decisions. Nonetheless, authors seemed to display an inherent bias towards reviews that were more positive towards their work. These results suggest that caution must be taken when asking authors to evaluate reviews on their own papers.

Inter-Evaluator Disagreement

One measure of the evaluation reliability is the consistency of scores. Consistency by itself is not sufficient for a useful evaluation process, for example, consistency is high if most evaluators simply give the median score out of laziness, but these evaluations are not useful. Nonetheless, consistency is one factor in evaluating reliability of evaluations, as we would generally like to obtain similar evaluations of review quality if we ask multiple people.

With this motivation, we follow the methods of [188] in their analysis of the reviews of papers (*not* evaluations of reviews) in the peer-review process of the NeurIPS 2016 conference. The NeurIPS 2016 conference asked reviewers to evaluate reviews on four criteria (but did not ask for an overall score). The analysis [188] computes the rate of agreement between reviews provided by a pair of reviewers on a pair of papers that they both review. In this manner, we compare the amount of agreement in reviews of papers (in NeurIPS 2016) with the amount of agreement in evaluations of reviews (in NeurIPS 2022).

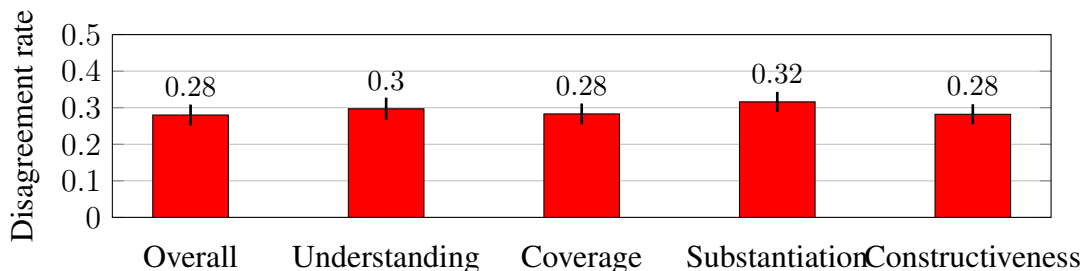


Figure 2.5: Inter-evaluator disagreement rates given to reviews.

Methods We compute the inter-evaluator (dis-)agreement following [188]. Consider any individual criterion or the overall score. Take any pair of evaluators and any pair of reviews that receives an evaluation from both evaluators. We say the pair of evaluators agrees on this pair of reviews if both score the same review higher than the other; we say that this pair disagrees if the review scored higher by one evaluator is scored lower by the other. Ties are discarded. We then compute the total number of agreements and disagreements. The total sample size (number of quadruples of paired review scores) in our calculations was $n_{\text{overall}} = 25,346$ for the overall score and $n_{\text{understanding}} = 18,658$, $n_{\text{coverage}} = 18,193$, $n_{\text{substantiation}} = 19,614$, $n_{\text{constructiveness}} = 19,870$ for each of the criteria scores. We show disagreement rates along with 95% confidence intervals in Figure 2.5. We note that a random baseline for the agreement rate if scores are drawn independently at random for each evaluation of a review (from any marginal distribution) is 0.5.

Results For the overall score, 29% of pairs of evaluations were ties, while for each of the criteria scores 37% to 40% of pairs were ties. In comparison, in the reviews of papers in NeurIPS 2016 [188], 35%-40% of pairs of criteria scores were tied. We now plot the rates of disagreements for the evaluations of NeurIPS 2022 reviews in Figure 2.5. The disagreement rates for both overall quality score and the criteria scores are approximately 0.3 on all criteria. In comparison, the same inter-evaluator disagreement statistic for reviews of papers in the NeurIPS 2016 [188], is in the range of 0.25 to 0.3. While the domains are different, these results suggest that evaluations of reviews and evaluations of papers have similar agreement rates.²

Miscalibration

Another issue in peer review of papers is evaluator miscalibration, that is the tendency for evaluators to exhibit idiosyncrasies such as giving especially lenient or harsh reviews [65, 178, 220]. In this section, we investigate whether the problem of miscalibration manifests itself in evaluating review quality.

Methods In order to estimate the degree of miscalibration, we fit a simple model that assume linear miscalibration in scores for each reviewer [34]. This allows for comparison to prior work

²In fact, an experiment at NeurIPS 2021 found that the rate of disagreement between co-authors of multiple jointly authored papers about the contribution of their own papers is 0.32, and that between authors of papers and the review process is 0.34 [173]. These disagreement rates are similar to what we found here for reviews of reviews.

α_f (Objective Quality Variance)	0.581
$\alpha_{b,1}/\alpha_f$ (Meta-Reviewer Offset Variance)	0.458
$\alpha_{b,2}/\alpha_f$ (Reviewer Offset Variance)	0.432
$\alpha_{b,3}/\alpha_f$ (Author Offset Variance)	0.780
$\alpha_{b,4}/\alpha_f$ (External Offset Variance)	0.441
σ^2/α_f (Subjective Score Variance)	1.467

Table 2.4: Fit parameters of linear calibration model.

in estimating miscalibration in paper review, where the same model of evaluation scores is employed. Specifically, we follow the methods of [34], modeling the evaluation scores as a linear combination of objective quality, evaluator bias and per-evaluation idiosyncrasy. The model assumes that the overall quality score given by evaluator j to review i , denoted as y_{ij} , is given by

$$y_{ij} = f_i + b_j + \epsilon_{i,j},$$

where

- $f_i \sim \mathcal{N}(\mu, \alpha_f)$ is an assumed “objective quality” of review i in the model, drawn from a normal distribution with mean μ and variance α_f ;
- $b_j \sim \mathcal{N}(0, \alpha_{b,g})$ is an “evaluator offset” capturing miscalibration of evaluator j . In order to capture differences in distributions of the four types of evaluators (meta-reviewers, reviewers, authors, and opt-in reviewers) we model the evaluator offset as a separate per-type normal distribution with mean 0 and variance $\alpha_{b,t}$ for $t \in \{1, 2, 3, 4\}$;
- $\epsilon_{i,j} \sim \mathcal{N}(0, \sigma^2)$ is the idiosyncrasy associated to this specific evaluation of review i by evaluator j .

This model is a Gaussian process with 6 variance hyperparameters to learn from evaluation data. We fit the parameters using maximum likelihood estimation. (We use Gaussian Process Regression maximum likelihood estimation implemented in the python package GPy.) We are particularly interested in the $\alpha_{b,g}$ parameter, the estimated variance of evaluator offsets for each reviewer type. Intuitively, $\alpha_{b,g}$ captures the degree of miscalibration for each type, with a larger value indicating that evaluators of that type are more likely to be miscalibrated.

Results In Table 2.4, we enumerate the values of the fit parameters, normalized by variance of objective quality scores α_f . First, observe that the (normalized) variance of author offset of 0.780 is much higher than the variance of evaluator offset for other types of approximately 0.45, suggesting that authors may be more likely to be miscalibrated than other types of reviewers. Second, let us compare with the miscalibration in the NeurIPS 2014 reviews of papers [34]. As mentioned earlier, we use the same model as that used in [34] to enable a direct comparison. The only difference is that the 2014 analysis had a single α_b term whereas we have a separate term for each evaluator type. For the NeurIPS 2014 reviews of papers, it was found that $\alpha_f = 1.28$, $\alpha_b/\alpha_f = 0.19$ and $\sigma^2/\alpha_f = 1.01$. This suggests that under the linear model of score generation, miscalibration in evaluating review quality may be at least as high as compared to evaluating paper quality.

Subjectivity

A frequent concern in peer review is subjectivity of reviewers. In the context of paper review, reviewers may have differing opinions about the relative importance of various criteria in determining overall quality of a paper, a phenomenon often referred to as “commensuration bias” [118]. For example, some reviewers may consider novelty of a paper more important towards overall quality whereas others may consider rigor more important. In our context of evaluating reviews, we asked evaluators to assess the quality of reviews on four specific criteria—understanding of the paper, coverage of required aspects of a review, substantiation with evidence, and constructiveness of the feedback. The overall score given by the evaluator then depends on how the evaluator maps these individual criteria to an overall quality score, and such a commensuration bias can result in arbitrariness in the evaluation process.

Methods Previous research has proposed learning a function that maps criteria scores to overall scores from the review data [160]. At a high level, this learned function is one that best fits the data while respecting monotonicity so that the function is consistent (that is, an improvement in any one criterion holding other criteria constant should not decrease the overall evaluation). We can obtain one measure of the degree of subjectivity in our evaluation process by computing the loss of this aggregate function learned from the evaluation data, where the loss is defined as the absolute difference between this aggregate function and the overall scores given by evaluators (averaged across all evaluations). Higher loss indicates that there is more variability in how evaluators map criteria to overall scores, suggesting higher subjectivity. Following the theory developed in [160], we choose the $L(1, 1)$ norm as our loss function.

In our approach, we learn a single function that is common to all the types of evaluators. An alternative approach would be to learn a separate function for each type. In order to evaluate the usefulness of this alternative approach, we randomly partition evaluation scores into a 75%–25% train-test split. We then fit a combined-evaluator type function on the training data and per-evaluator type functions on the training data to minimize $L(1, 1)$ loss. We evaluate the two approaches on the test data to obtain estimated test loss. To predict on criteria scores that were not present in the train data, we solve a convex optimization problem to minimize $L(1, 1)$ loss subject to monotonicity constraints with respect to the function learned on the train data and other points in the test data. Repeating this procedure 5 times, we find that the combined type function achieves a train loss of 0.456 and a test loss of 0.457, while the per-type functions achieves a train loss of 0.448 and a test loss of 0.465. This indicates that estimating different functions per-type does not improve model quality, so we continue to use the combined-evaluator type model.

Results Comparing overall scores given by evaluators to the scores assigned by the learned mapping from criteria scores to overall scores, we find evaluators had a mean loss of approximately 0.45. As a point of comparison, we also evaluate subjectivity in NeurIPS 2022 *paper* review data. We employ the same approach for the reviews on papers as we did for evaluations of reviews: we estimate a function mapping criteria scores to overall scores on the 33,371 reviews for papers in NeurIPS 2022 and compute the mean $L1$ loss. We note that the overall scores in the reviews of papers at NeurIPS 2022 used a 10 point review scale, whereas our evaluations of reviews used a 7 point review scale. We thus re-normalize the loss by $6/9$ (assuming a linear

mapping from the 10 point scale to the 7 point scale). We find that the loss on reviews of papers 0.402. While the criteria are different in the review of papers and evaluations of reviews, this result suggests that the degree of subjectivity is similar in paper review and in evaluating review quality at NeurIPS 2022.

2.1.5 Discussion

This subchapter analyzes the reliability of peer reviewing peer reviews. The central finding is that many problems that exist in reviews of papers—inconsistency, bias, miscalibration, and subjectivity—also arise when people evaluate reviews. Reviews of reviews may still be useful for designing incentives for high-quality reviewing and for measuring the effects of policy choices in peer review, but these results show that such evaluations should not be treated as direct measurements of review quality.

Several features of the study affect how the results should be interpreted. Participants knew that they were providing evaluations for an experiment, which may result in “Hawthorne” effects, and evaluators might behave differently if their evaluations had concrete consequences such as reviewer awards. Participation was also opt-in rather than compulsory. Although many analyses separate authors, reviewers, meta-reviewers, and non-author evaluators, there may still be selection effects within each group. The length experiment also required evaluators who had not themselves reviewed the paper, since original reviewers had seen the actual reviews; it would be useful to test whether length has the same effect for evaluators who are more familiar with the paper content. Finally, the review-quality scale used here differs from the 10-point scale used for NeurIPS paper reviews. The analyses rescale the two domains to make comparisons possible, but scale effects may remain.

There is one prominent problem which exists in reviews of papers which we are unable to study in the context of reviewing reviews—dishonest behavior. One form of dishonest behavior is that of “lone wolf” dishonesty in which reviewers, who are also authors of some submitted papers, deliberately manipulate the reviews they provide to increase the chances of their own papers being accepted [9, 39, 235]. A second form of dishonest behavior that has gained significant importance recently is that of collusion rings [85, 128, 217]. Here, a group of reviewers make a pact according to which they try to get assigned each others’ papers for review, and provide positive reviews to each other. In our study, the participants had no incentives for dishonesty since the review-quality evaluations had no downstream consequences in terms of paper acceptances. However, it is not hard to envisage that if the stakes of reviewing reviews become high (e.g., reviewer awards become important or even necessary for promotion) dishonest behavior may also be a problem in reviewing of reviews.

These caveats notwithstanding, the results have direct implications for the use of evaluations of reviews in improving the scientific peer-review process. In particular, they suggest that evaluations of review quality are themselves affected by biases, inconsistency, subjectivity, and miscalibration. This indicates that peer-review systems need more reliable approaches to evaluating review quality. For example, it may be helpful to consider some semi-automated or fully automated approaches to evaluation of review quality. In the applications of designing incentive mechanisms and measuring impacts of interventions in peer review, care needs to be taken in using human evaluations of review quality.

Some past works on incentivizing high quality paper review content ([233, 234]) have assumed that evaluators of review quality report “true quality.” Our results suggest instead that evaluators provide scores rife with biases and noise. Hence, incentive mechanisms need to account for these sources of noise and bias in order to fairly reward high quality review and penalize low quality review. In particular, the “uselessly elongated review bias” may create problems for the design of incentives for high quality review. On the one hand, the results suggest that reviewers who would like to be rewarded for higher quality review may be able to uselessly lengthen their reviews in order to be perceived as higher quality. On the other hand, longer reviews may genuinely be higher quality if a reviewer has completed a more detailed and thoughtful evaluation of a paper. Hence, an incentive designer needs to carefully account for review length, which may constitute a cheap (spurious) signal or a genuine signal of quality.

The issues in evaluating review quality also create issues when measuring the impact of an experiment in peer review. For instance, there is much recent interest in using large language models (LLMs) for reviewing papers [123, 132, 186, Section 9.6]. One recent study [123] generated reviews for a set of papers using the GPT-4 model and then asked authors of these papers to compare the quality of the model-generated reviews to human-written reviews. They found that LLM-generated reviews were rated as more helpful than some human-generated reviews. Our results indicate that these experiments, which use authors’ evaluations of reviews on their own papers, should take into account any bias stemming from the positivity or negativity of reviews given. Furthermore, if the LLM was writing uselessly longer reviews (e.g., the LLM adds more filler sentences), then uselessly elongated review bias could lead to false positive conclusions in this study. Thus, it is important to check for potential length bias when interpreting the effect of using an LLM to generate reviews.

Finally, methods to control for the effects of length bias in evaluations may be useful both in the setting we have considered of scientific peer review and in LLM-based evaluations, where recent works [44, 45, 146, 192] have demonstrated the existence of bias towards longer LLM responses when performing automated evaluations.

Overall, this subchapter pinpoints a number of specific pitfalls in evaluating review quality, which may negatively affect downstream applications that use these evaluations. A central direction for future work is to design better methods for evaluating review quality, or to explicitly account for sources of bias and inconsistency when review-quality scores are used in downstream applications.

2.2 LLM Checklist Assistance Before Peer Review

The preceding subchapter studied human evaluations of reviews after reviews have been written. This subchapter moves earlier in the peer-review pipeline and studies whether a narrowly scoped LLM assistant can help authors improve checklist compliance before reviewers see the paper. The connection is deliberate: both subchapters examine auxiliary signals around peer review, but this one asks whether an automated tool can support authors without becoming a substitute for reviewer judgment.

This subchapter is based on joint work with Ihsan Ullah, Thanh Gia Hieu Khuong, Benedictus Kent Rachmat, Zhen Xu, Isabelle Guyon, and Nihar B. Shah [68].

2.2.1 Motivation

Large language models (LLMs) are increasingly entering the scientific peer-review process. They are used by authors while preparing manuscripts [125], by systems that match reviewers to papers [26, 30, 227], and in proposals for generating or assisting with reviews [35, 36, 124, 134, 205]. This creates an immediate institutional question: can LLMs improve peer review in ways that are useful, measurable, and compatible with the incentives of the review process?

This subchapter studies a deliberately limited use case. Rather than asking an LLM to review papers, score papers, or advise reviewers, we evaluate a Checklist Assistant deployed to authors at NeurIPS 2024 before submission. The assistant reviewed authors’ responses to the NeurIPS checklist and returned feedback only to the authors. This design keeps the tool out of the accept/reject decision while still testing whether LLM feedback can help authors satisfy conference expectations around reproducibility, transparency, limitations, ethics, and documentation.

The deployment provides early evidence about both the promise and limits of LLM assistance in peer review. Authors were surveyed before and after use of the tool; 539 authors completed the pre-usage survey, 234 papers were submitted to the Checklist Assistant, and 78 post-usage survey responses were received. The main findings are:

1. Most surveyed authors reported that the assistant was useful: over 70% of post-usage respondents said that the feedback was useful and that it would lead them to modify their paper or checklist responses.
2. The assistant produced concrete feedback. Across checklist questions, its comments often identified specific missing details about claims, limitations, experiments, data, code, and compute resources.
3. The evidence of actual manuscript improvement is suggestive rather than causal. Some authors reported planned changes, and authors who resubmitted to the assistant often expanded or changed their checklist justifications, but the study cannot isolate the assistant’s causal effect on final paper quality.
4. The tool also exposed important risks. Authors reported inaccuracies and overly strict feedback, and an adversarial experiment showed that another LLM could rewrite checklist justifications to obtain higher scores without corresponding changes to the paper.

The contribution of this subchapter is therefore not a claim that LLMs should replace human judgment in peer review. Instead, it gives a concrete example of how LLMs can be evaluated in a real review-adjacent workflow, with attention to user perceptions, behavioral responses, and adversarial misuse.

2.2.2 Background

The NeurIPS checklist asks authors to document whether their submissions satisfy expectations around claims, limitations, theoretical assumptions, experimental reproducibility, code and data access, compute, ethics, broader impacts, safeguards, licenses, new assets, and human subjects research. Similar checklists are used in other scientific communities to promote transparent reporting and reproducibility [148, 149, 214]. In the context of this thesis, the checklist is useful

Question 3: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: There is no theory in this paper.

Review:

The answer provided by the authors is consistent with the content of the paper. Based on the detailed review of the paper, it is clear that the work primarily focuses on conducting an observational study to investigate the presence of citation bias in peer review processes. The study utilizes statistical modeling and analysis rather than theoretical developments that would involve theorems, lemmas, or formal proofs. Therefore, the classification of the answer as “NA” (Not Applicable) is accurate given the nature of the research presented. However, to provide constructive and actionable suggestions for improvement and to ensure thorough and critical assessment, it is beneficial to clearly outline the assumptions behind the statistical models and analyses employed. This would enhance the readability and understandability of the study’s methodology and its limitations. [...]

Question 12: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have used algorithms such as TPMS and PeerReview4All. We have cited them.

Review:

There does not appear to be any discrepancy between the authors’ justification for the cited question and the actual paper content in terms of providing proper credit and mentioning licenses for used assets, according to their statement. However, the detailed source citations such as the specific papers for TPMS (Toronto Paper Matching System) and PeerReview4All, including their licenses or terms of use, are not explicitly mentioned in the provided text. It’s essential to ensure that these aspects are clearly attributed and any terms of use are respected, as stated in the initial guidelines provided to authors. [...]

Figure 2.6: Example of checklist questions, answers, and LLM-provided review.

because it is both consequential and bounded: it covers aspects of submission quality that conferences care about, while giving an LLM a more specific task than open-ended paper review.

Prior work on LLMs in peer review has considered reviewer assignment, paper review generation, error detection, and checklist compliance [115, 132, 187, 230]. A key limitation of many evaluations is that they are retrospective or rely on outcome measures whose relationship to review quality is unclear. This subchapter complements that literature by studying a live deployment associated with a major conference and by measuring authors’ expectations, experiences, reported changes, and concerns.

2.2.3 Assistant Design and Deployment

The Checklist Assistant took as input a submitted paper, appendices, and checklist responses. For each checklist question, the assistant received the relevant question, the author’s answer and justification, the checklist guidelines, and the paper text. It returned written feedback and a score indicating either no significant concern or need for improvement. The assistant used GPT-4 through a third-party API. Authors used the tool voluntarily, outside the official NeurIPS submission system, and the outputs were not visible to reviewers, meta-reviewers, or program chairs. The study was approved by the Carnegie Mellon University Institutional Review Board.

Figure 2.6 illustrates examples of feedback provided by the Checklist Assistant for two dif-

ferent papers. In these examples, green indicates that the tool found “no significant concerns”, while orange signals “needs improvement” with the NeurIPS Paper Checklist standards. Authors are encouraged to carefully review any orange feedback, validate the identified issues, and make the necessary revisions to align with the checklist requirements.

The prompt instructed the LLM to assess whether each checklist response was accurate and sufficiently supported by the paper, to provide itemized feedback, and to assign a score. Scores of 0 and 0.5 were merged in the user interface into a single “needs improvement” category, reflecting a conservative design choice: the tool was meant to point authors toward issues, not to make fine-grained judgments about paper quality. The full prompt and additional implementation details are provided in the supplemental material.

2.2.4 Evaluation of Author Use

In our evaluations, we seek to address two main questions regarding the use of an LLM-automated Author Checklist Assistant:

1. Do authors perceive an LLM Author Checklist Assistant as a valuable enhancement to the paper submission process?
2. Does the use of an Author Checklist Assistant meaningfully help authors to improve their paper submissions?

To answer these questions, we combine three forms of evidence: pre- and post-usage surveys, qualitative freeform responses, and changes observed among authors who submitted the same paper to the assistant more than once. This design is intentionally observational. It can show whether authors perceived value and whether their behavior is consistent with using the feedback, but it cannot by itself prove that the assistant causally improved final paper quality.

Author Perceptions and Reported Issues

At registration, authors completed a pre-usage survey about their expectations of LLM feedback. After using the assistant, they were invited to complete a post-usage survey about whether the feedback was useful, accurate, and likely to change their submission. Responses used a four-point Likert scale. The pre-usage survey received 539 responses; the post-usage survey received 78 responses, representing 63 unique participants.

Figure 2.7 presents the survey responses collected before and after using the checklist verification tool. We include responses from authors who completed both surveys ($n=63$). In cases where authors submitted the survey multiple times for the same paper, we included only the earliest post-usage response. Including the duplicated responses made a negligible difference, with the proportion of positive responses changing by less than 0.02 across all questions.

Overall, the majority of authors responded positively regarding their experience with the Checklist Assistant. 70% of surveyed authors reported plans to make changes based on the feedback received, 70% reported that they found the assistant concretely useful, and 67% expressed excitement about using LLMs as Checklist Assistants in the future. Thus, a statistically significant majority of authors responded positively to “Will Modify”, “Useful” and “Excited to Use” after using the assistant based on a one-sided Binomial Test with Benjamini–Hochberg

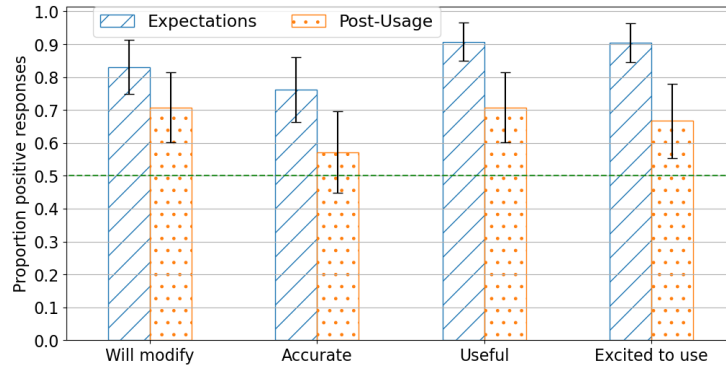


Figure 2.7: Responses to survey questions pre- and post-usage of the Checklist Assistant, from all authors who responded to both surveys ($n=63$). Error bars show 95% confidence intervals for the sample proportion. The majority of surveyed authors reported a positive experience using the Checklist Assistant.

Correction comparing the sample proportion to 0.5 (adjusted p -values of 0.001, 0.002, and 0.007 respectively).

It is notable that authors were even more positive before using the tool. Comparing pre- and post-usage responses, there was a statistically significant drop in positive feedback on the “Useful” and “Excited to Use” questions: a permutation test with 50,000 permutations gave Benjamini-Hochberg adjusted p -values of 0.007 and 0.013 for “Excited to Use” and “Useful” respectively, with effect sizes of -0.23 and -0.2 . This gap between expectations and experience is itself informative, since LLM tools may be adopted in peer review partly because users overestimate their reliability before seeing concrete outputs.

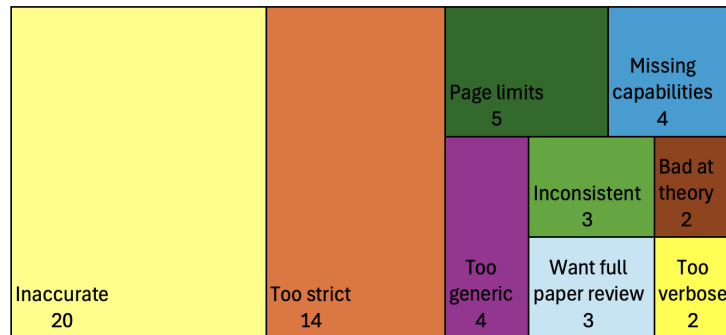


Figure 2.8: Summary of reported issues using checklist verification from freeform feedback on post-usage survey ($n=52$ out of 78 total survey responses). Numbers show the total number of authors who reported the issue.

The qualitative responses explain why enthusiasm declined after actual use. Among the 52 post-usage responses that described issues, the most common concerns were inaccuracy (20 authors) and excessive strictness (14 authors), followed by page-limit constraints, generic feedback, inability to process some content, inconsistency across submissions, and difficulty with theory papers. These responses are important for the thesis framing: an LLM assistant can be useful

even when it is imperfect, but its institutional role must be designed around those imperfections.

Changes to Submissions in Response to Feedback

The second question is whether the assistant plausibly changed author behavior. We use three signals: the content of the assistant’s feedback, authors’ self-reported planned changes, and changes between repeated submissions to the assistant.

To summarize the feedback, we used GPT-4 to extract key points from the assistant’s responses and cluster them into themes, manually checking a random sample of outputs for fidelity to the original feedback. Figure 2.9 shows common categories for four checklist questions. The assistant often identified concrete issues, such as missing experimental details, insufficient documentation of claims, unclear compute requirements, and incomplete discussion of limitations. At the same time, some feedback was generic or expanded the scope of the checklist question, which is consistent with authors’ reports that the assistant could be overly strict.

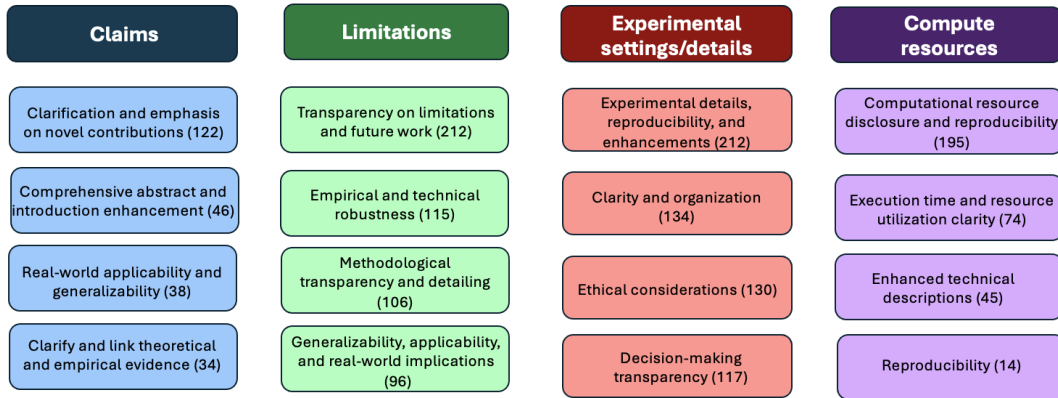


Figure 2.9: Most frequent categories of feedback given by the Author Checklist Assistant on four questions of the checklist. The types of feedback below each question were identified by using another LLM to summarize the main points in the feedback given by the checklist verifier on submitted papers. Frequency of each category of feedback is shown in parentheses (out of $n=234$ checklist submissions). Feedback categories for other checklist questions are provided in Supplemental Material.

Survey responses provide a second signal. Of the 78 post-usage survey responses, 45 included freeform feedback about changes the author made or planned to make; 35 of these described concrete changes. The most common planned change was to expand checklist justifications with more detail or references to paper sections. Other authors reported adding details about experiments, datasets, compute, limitations, or error bars, and a small number reported changing a checklist answer they had filled out incorrectly.

Repeated submissions provide a third signal. Among 40 pairs of repeated submissions, 22 changed at least one checklist answer and 39 changed at least one justification. Most changes were to justifications rather than yes/no/NA answers, and many were substantial: among changed justifications, more than half doubled in length. Figure 2.10 summarizes the changes in response length, while Figure 2.11 shows how the assistant’s scores changed after resubmission.

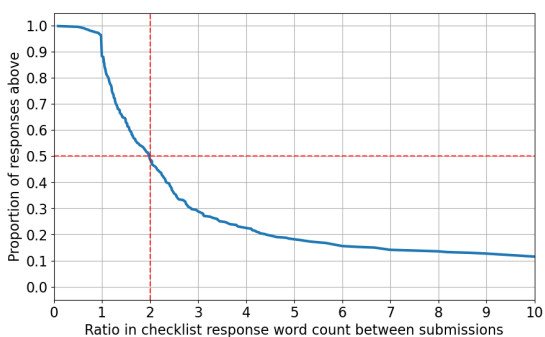


Figure 2.10: Ratio between the word count of checklist responses on the first and second submission on papers where the authors changed the justifications ($n = 362$ responses). The plot shows the proportion of checklist responses that increased by more than a given ratio—many authors increased the length of responses between re-submissions, with more than 50% of responses increasing by a factor of two or more in length.

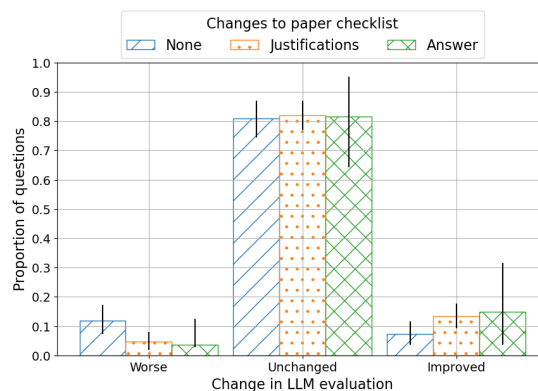


Figure 2.11: Changes in LLM evaluation of paper checklist questions between re-submissions ($n=40$ pairs) split by changes the author made to the checklist answer. Error bars show 95% confidence intervals obtained via bootstrapping by re-sampling the papers.

When authors changed neither answer nor justification, the assistant’s evaluation remained unchanged 81% of the time. When authors changed a justification, the evaluation improved 13% of the time and worsened 5% of the time; when authors changed an answer, it improved 15% of the time and worsened 2% of the time. These differences are noisy because the sample is small, but they are consistent with the qualitative evidence that authors used the assistant to make checklist responses more complete.

2.2.5 Robustness and Deployment Constraints

The assistant was designed for author-facing feedback, not for reviewer-facing verification. To test the risk of using it beyond that intended role, we simulated an adversarial author who uses another LLM to rewrite checklist justifications based on the assistant’s feedback without changing the underlying paper. The attack agent was given up to three attempts, matching the deployed submission limit, and then selected the highest-scoring rewritten response for each checklist question.

Figure 2.12 shows that this attack substantially increased Checklist Assistant scores: 14 out of 15 questions improved after three adversarial rewriting rounds. Manual inspection showed that some rewritten justifications introduced unsupported claims, such as hardware descriptions that were not in the paper or placeholder URLs for code repositories, while still receiving favorable scores. This result does not undermine the author-assistance use case, where the output is private

to authors and intended as advice. It does show that the same tool would be risky as an automated compliance verifier or as part of the formal review process without human oversight.

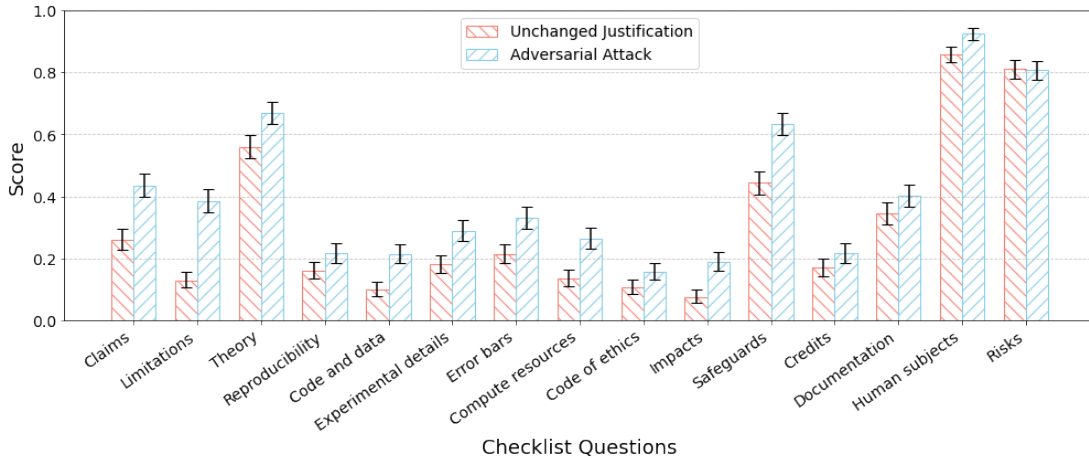


Figure 2.12: Evaluation of the Checklist Assistant scores on initial unmodified justifications and adversarially revised justifications for the ($n = 234$) submitted papers. 14 out of 15 questions show score increases for the answers refined through 3 adversarial attack rounds.

2.2.6 Discussion

The deployment of the Checklist Assistant gives early evidence that LLMs can support peer review when their role is narrow and their outputs are kept out of adjudication. Authors generally found the assistant useful, many reported that it would lead them to revise their submissions, and repeated submissions suggest that authors often expanded or corrected checklist justifications after receiving feedback. At the same time, the study shows why LLM adoption in peer review should be evaluated empirically rather than assumed to be beneficial. Authors reported inaccuracies and excessive strictness, and the adversarial rewriting experiment shows that higher LLM-generated scores need not correspond to real improvements in the paper.

For the thesis, the main lesson is methodological. Peer review is already incorporating LLMs through author writing tools, reviewer-assignment systems, and proposals for automated review assistance. This subchapter shows one way to assess such tools in the real world: deploy them in a bounded role, survey users before and after interaction, examine behavioral traces such as resubmissions, and test plausible misuse. The results support a cautious path for LLMs in peer review. They may help authors notice and repair omissions, but they should be treated as assistive signals whose value depends on scope, incentives, and measurement rather than as substitutes for human judgment.

Chapter 3

Randomized Selection from Imperfect Evaluations

The experiments in Chapter 2 show that human evaluations can be noisy, biased, and difficult to validate. This chapter asks what a decision maker should do after receiving such evaluations. In particular, we study randomized selection mechanisms for settings such as scientific funding, conference awards, admissions, and hiring, where a decision maker must choose a fixed number of candidates from imperfect evaluations.

Randomized selection has moved from a theoretical proposal to an active institutional design choice. In scientific funding, proponents argue that lotteries can reduce wasted reviewer effort on arbitrary tie-breaks, encourage high-risk proposals, counteract “rich-get-richer” effects, and reduce reviewer partiality [59, 60, 73, 79, 80]. Surveys find substantial support among scientists for introducing randomization into grant funding [131, 169]. Several funding agencies have adopted partial lotteries, beginning with the New Zealand Health Research Council [131], followed by the Swiss National Science Foundation [1, 79, 203], and more recently the European Research Council, Science Foundation Ireland, the Volkswagen Foundation, and others [56, 184, 200, 218]. Randomization has also been proposed or used in adjacent selection settings, including admissions, job screening, startup investment, and conference presentation decisions [1, 13, 104, 143, 164, 209].

Most existing deployments use a thresholded partial-lottery design. A decision maker first uses reviews to separate proposals into groups: proposals clearly above the funding line are funded, proposals clearly below it are rejected, and a middle set of proposals is entered into a uniform lottery. In the simplest randomize-above-threshold version, the decision maker chooses a minimum acceptable quality threshold and samples uniformly among all candidates above it. The Swiss NSF design is more explicitly uncertainty-aware: it estimates point values and confidence intervals for proposal quality, accepts proposals whose intervals lie above a provisional funding line, rejects proposals whose intervals lie below it, and allocates the remaining budget uniformly among proposals whose intervals overlap the funding line [79].

These designs are appealing because they are simple and communicable. They also capture a real institutional intuition: when evaluations cannot meaningfully distinguish among candidates near a boundary, it may be better to acknowledge that uncertainty through randomization than to pretend that a deterministic ranking is precise. But the existing designs leave important questions

unresolved. If the institution is randomizing because it is uncertain about the relative ordering of candidates, then the lottery should respect what the evaluations do and do not establish. If one candidate clearly dominates another, the weaker candidate should not be selected while the stronger one is rejected. If the institution is randomizing because small score differences near a threshold are arbitrary, then small changes in scores should not produce large jumps in selection probabilities. A uniform lottery over a thresholded middle group does not automatically satisfy either principle.

The central premise of this chapter is that randomization should not be treated as a last-resort tie-breaker invoked only after a deterministic ranking has failed. Instead, randomization can be part of the decision rule itself, with formal guarantees that match the reason randomization is being introduced. This chapter studies two core and complementary motivations. The first is robustness to unresolved uncertainty: when evaluations do not justify a complete ranking, the decision rule should respect the ambiguity that remains. The second is stability: when evaluations are numerical but noisy, small changes in scores should not cause abrupt changes in selection probabilities.

The two subchapters formalize these motivations and give algorithms that realize them. Section 3.1 develops MERIT, a robust lottery for selection from interval-valued quality estimates when the decision maker does not trust a full probabilistic model of evaluation error. It treats the set of rankings consistent with observed intervals as an ambiguity set and optimizes the worst-case expected utility of the selected set, while enforcing ex post validity constraints. Section 3.2 studies smooth lotteries, which are designed to make selection probabilities stable under small perturbations to review scores. It formalizes smoothness as a Lipschitz property of marginal selection probabilities and studies the tradeoff between stability and regret.

The goal of the chapter is not to argue that lotteries are always preferable to deterministic selection, or that these two approaches are in tension. Rather, the chapter shows that different institutional motivations for randomization can be made precise. Randomization can be used to respect uncertainty about candidate ordering, and it can be used to avoid arbitrary discontinuities near decision boundaries. In both cases, the design of the lottery should be transparent and aligned with the weakness in the underlying evaluation process that motivated randomization in the first place.

3.1 Robust Randomization Under Interval Uncertainty

The previous chapter showed that evaluation signals can be biased, noisy, and difficult to validate. This subchapter asks how a decision maker should act when those signals are still the only available basis for selection. We focus on settings where uncertainty is represented by intervals rather than a trusted probabilistic model, and develop a randomized rule that uses those intervals to make robust top- k decisions.

This subchapter is based on joint work with Giulia Fanti and Nihar B. Shah [69].

3.1.1 Motivation

As described at the start of this chapter, existing partial lotteries typically derive a probability distribution from peer-review assessments using a thresholded or interval-based heuristic. This subchapter initiates a principled approach to randomizing decisions from evaluations. We focus on the key question: *Given imperfect evaluations of candidate quality, what is a suitable probability distribution over applicants to make the random selection?*

We take the perspective of a funder who aims to select the highest quality grant proposals from a set of applicants. A key motivation for randomization in funding decisions is uncertainty about the relative quality of proposals. Indeed, many existing deployments describe peer review lotteries as a form of random “tie-breaking” between proposals of equal quality [154, 198]. Therefore, similar to previous work at the Swiss NSF [79], we assume that the funder estimates numeric quality *intervals* for each proposal; we are agnostic to the source of these intervals, which could be derived from a number of processes, described in Section 3.1.2. Concretely, one simple way a funder may generate intervals is by taking the minimum and maximum score given to each interval, instead of simply aggregating all scores to a single estimate like the mean or median. More generally, intervals may capture the funder’s uncertainty due to errors in the evaluation process, such as miscalibration and subjectivity, or underlying aleatoric uncertainty about the future success of proposals. Crucially, we capture settings with “*Knightian uncertainty*,” where a funder cannot assign a probability measure over possible outcomes [105]. Models that assume decision makers have Knightian uncertainty are widely applied in related applications such as policy-making [12, 179, 196], financial investment decisions [55, 138, 156], and R&D investment by businesses [4].

The assumption that a funder lacks a probabilistic model of their data is particularly apt when quality is determined by peer review. First, while there are many proposed probabilistic models of reviewer errors, these models have performed poorly in real deployments [117, 187, Section ‘Miscalibration’]. One possible reason for this poor performance is that in practice, human miscalibration is more complex than simple models [21]. Second, decision makers generally lack ground truth data with which to evaluate whether a given probabilistic model of review data is appropriate in a given setting, making it difficult to rely on a probabilistic model. Further, even if a decision maker had a reasonable model of review scores, these probabilities may not represent meaningful probabilities to the funder, since review scales are arbitrary and likely do not map linearly to the utility of selecting a candidate. Finally, the future success of candidates in the settings we consider may be inherently hard to predict. In the context of scientific peer review, many works find that it is difficult to predict future citations and peer review scores are poor predictors [33, 183, 226]. In the related area of college admissions, recent work demonstrated that predictive models used to rank candidates are highly unstable with respect to their training data, resulting in arbitrariness in the ranking from any single model [119].

In the absence of a reliable probabilistic model of proposal quality, we assume that a decision maker describes their uncertainty by estimating intervals of proposal quality rather than point estimates. Since we are agnostic to the source of these intervals and do not consider them to have a probabilistic interpretation, the funder draws conclusions from the intervals only about the relative ordering of proposal quality. If two proposals’ intervals overlap, then the funder does not have sufficient evidence to conclude that one proposal is higher quality than the other.

However, if one interval lies strictly above another, then the funder has sufficient evidence to conclude that one proposal dominates the other. We develop theory and a practical algorithm for making funding decisions on the basis of interval quality estimates, with the following **key contributions**:

1. *Modeling uncertainty as Knightian uncertainty intervals*: Prior work models a funder’s uncertainty by assuming a known probabilistic relationship between proposal quality and review scores [79], using Bayesian inference to construct confidence intervals. However, we show that in this fully Bayesian setting, a deterministic selection policy always maximizes expected utility, so randomization is unnecessary. In contrast, our model captures the motivation for randomizing without assuming a fully specified probabilistic model, using “Knightian” uncertainty intervals, that capture the funder’s estimates of the relative quality of proposals.
2. *A principled approach to randomization*: We formalize two key principles that guide the design of an algorithm for selection under interval uncertainty and show that randomization is necessary to meet these principles. First, given their uncertainty, the funder aims to solve a maxmin optimization problem, maximizing their worst-case utility over all rankings consistent with quality intervals. Hence, the funder randomizes in order to robustly optimize their worst-case utility. We call this principle *ex ante optimality*. Second, the funder makes a selection in a manner that respects *ex post validity*—if one proposal’s interval lies strictly above another proposal’s interval, then the higher quality proposal must be selected whenever the lower quality proposal is selected. Prior heuristic selection rules do not satisfy both of these principles.
3. *Efficient algorithm*: It is not obvious that it is possible to solve the maximin optimization problem efficiently, as it requires solving a linear program with number of constraints that grows exponentially in the number of selected proposals. In fact, many closely related problems from the literature on (partial order) graph algorithms are NP-hard [2, 58, 174, 229]. Nonetheless, we develop a polynomial time algorithm to solve the ex ante optimization problem and enforce ex post validity. We refer to this algorithm as **Maximin Efficient Randomized Interval Top- k (MERIT)**. We demonstrate the computational feasibility of MERIT on real-world peer review data from Swiss NSF grant reviews and NeurIPS 2024 and ICLR 2025 conferences. The algorithm runs on a standard personal laptop in under 5 minutes on inputs with over 10,000 candidates. We release an open source implementation of our MERIT algorithm at github.com/akgoldberg/lottery.
4. *Axiomatic comparison*: Since ground truth of candidate quality tends to be unavailable in the domains we consider, we initiate an axiomatic approach to comparing randomized mechanisms. We identify three desirable properties of “monotonicity in budget”, “stability”, and “reversal symmetry.” We prove that MERIT prevents instances of “maximal instability” and respects “reversal symmetry” while the existing randomization mechanisms studied do not. We show that all randomization mechanisms that we consider violate monotonicity and propose modifications to MERIT to enforce this axiom; however, these modifications cause us to violate ex ante optimality.
5. *Empirical comparison*: We evaluate MERIT against existing methods using synthetic data

based on real peer review data from major conferences (NeurIPS 2024, ICLR 2025) and grant agencies (Swiss NSF 2020). MERIT performs comparably to existing methods in expected utility under a linear reviewer error model used by the Swiss NSF [79] and many other prior works [6, 62, 177, 178]. However, under our worst-case objective—which assumes any ordering consistent with quality intervals could be true—our algorithm significantly outperforms deterministic selection and the Swiss NSF’s randomized approach.

The rest of the subchapter first gives the details of the two main existing partial-lottery designs and motivates our proposed approach. It then formalizes randomized selection from interval-valued estimates, develops an efficient algorithm for the resulting robust optimization problem, compares selection rules axiomatically, and evaluates MERIT on synthetic data and publicly available peer-review datasets. The most important formal proofs are collected at the end of the subchapter in Section 3.1.8.

3.1.2 Lottery Deployments and Modeling Approach

We now give the details of the existing partial-lottery designs summarized in the chapter introduction and highlight the drawbacks that motivate our proposed framework.

Existing Partial Lottery Deployments

In recent years, there have been many deployments of “peer review lotteries” in scientific funding decisions. Most deployments use an approach of “*randomize-above-threshold*.” As described in Algorithm 1, the funder chooses a minimum acceptable quality threshold and samples uniformly at random among all proposals that are above this threshold. This approach has been adopted to make funding decisions by the European Research Council [56], the British Academy [200], the Science Foundation of Ireland [184], the Volkswagen Foundation [218], and the Health Research Council of New Zealand [131] among others. Additionally, the USENIX Security Conference is randomly allocating long oral presentations among accepted papers to their conference in 2025 [209]. However, as we describe in Section 3.1.3, randomize-above-threshold may violate a desired principle of “ex post validity”, which says that if one proposal clearly dominates another, the stronger proposal should be funded if the weaker proposal is funded.

Algorithm 1 Randomize Above Threshold [56, 131, 184, 200, 209, 218]

Input: Set of proposals with point estimates and intervals; number accepted k .

Output: Set of proposals k .

Choose a threshold T (potentially based on the data).

Reject all intervals strictly below T and select uniformly at random among the remaining.

Taking a different approach, the *Swiss National Science Foundation (NSF)* [79, 203] pioneered a method that explicitly accounts for uncertainty about the quality of each proposal. They assume that each proposal has a latent true quality and review scores are generated based on these quality scores and reviewer-specific noise parameters. They assume priors on the model

parameters and develop methods to obtain point estimates and confidence intervals for the true quality of each proposal. As described in Algorithm 2, the Swiss NSF then samples k proposals by setting a “provisional funding line” as the k -th highest point estimate. All proposals with intervals strictly above the funding line are selected and all proposals strictly below the funding line are rejected. The remaining budget is allocated uniformly at random among proposals with intervals that overlap the funding line.

Algorithm 2 Swiss NSF Selection Algorithm [79]

Input: Set of proposals with point estimates and intervals of the quality of each proposal; number accepted k .

Output: Set of proposals k .

Rank proposals by decreasing point estimate and let $e_{(k)}$ be the point estimate of the k -th ranked proposal.

Let \mathcal{A} be the set of proposals with lower bound strictly above $e_{(k)}$, \mathcal{R} be the set of proposals with upper bound strictly below $e_{(k)}$, and \mathcal{P} the set of proposals with intervals that contain $e_{(k)}$.

Accept \mathcal{A} , reject \mathcal{R} , and accept $(k - |\mathcal{A}|)$ proposals chosen uniformly at random from \mathcal{P} .

The intuition provided by the Swiss NSF for their approach is that confidence intervals capture the funder’s uncertainty in estimating the proposal quality and randomizing decisions accounts for this uncertainty, thereby leading to better decisions. In contrast, as we prove in Theorem 3.1.1 below, when the funder assumes that review data is generated from a fully specified Bayesian model, there exists a deterministic selection of proposals that maximizes the funder’s expected utility for any utility function. Informally, *if the funder knows the model that generates their data, then they do not need to randomize in order to maximize their expected utility*. Below, we present the formal proposition and proof. We prove a general statement for a funder who estimates a total ranking of proposals and receives a pre-specified utility for any possible estimate of the ranking of proposals and true qualities of the proposals. The top- k selection problem is a case of this estimation problem where the funder’s utility specifies the utility of choosing k proposals based on the estimated ranking of proposals.

Proposition 3.1.1 (Optimality of deterministic selection in the fully Bayesian setting). *Consider a funder who estimates a ranking of proposals from review data in the following fully Bayesian setting. The funder observes review data $y \in Y$ for a set of proposals of true quality $\theta \in X$, for some measurable sets X and Y . The review data and true quality are generated jointly from a known probability distribution. Letting Π denote the set of all permutations of proposals, the funder estimates a ranking $\hat{\pi}$ of proposals and gains utility $u(\hat{\pi}, \theta)$ where $u : \Pi \times X \rightarrow \mathbb{R}$. The funder aims to choose a (potentially randomized) estimator of the true ranking $f : Y \rightarrow \Delta(\Pi)$ that maximizes their expected utility $\mathbb{E}_{y, \theta, \hat{\pi}}[u(\hat{\pi}, \theta)]$ where $\hat{\pi} \sim f(y)$. In this setting, there always exists a deterministic f that maximizes the funder’s expected utility.*

Theorem 3.1.1 suggests one drawback of the Swiss NSF’s method—in their model setting there may be a utility cost to randomization compared to choosing deterministically in an optimal manner. A second drawback is that the Swiss NSF’s algorithm for selecting proposals

from intervals violates natural axioms for a selection rule, like monotonicity in the budget k and stability, as we show in Section 3.1.5.

Interval-Based Robust Selection

This subchapter proposes a model that captures the motivation for randomizing due to uncertainty about the relative quality of the proposals. We show that if the funder cares about their *worst case utility* they must randomize decisions in order to robustly optimize their utility.

Specifically, we consider a funder who estimates intervals for each proposal based on data. These intervals need not come from any one particular model, but they should capture the funder’s inherent uncertainty about the relative quality of proposals.

Our interpretation of the quality intervals stems from the intuitive argument that if the intervals for two proposals overlap then the funder does not have enough evidence to distinguish between them. On the other hand, if the interval of proposal A dominates the interval of proposal B , then the funder has sufficient evidence to believe that A is better than B . Hence, the intervals define a partial ordering of proposals that represents a set of conclusions by the funder regarding the relative quality of different proposals. This ordering is the canonical “interval order” for a set of intervals. It captures the spirit of how the Swiss NSF interprets confidence intervals in their setup, as they describe that the intervals are “*used to identify proposals with similar quality and . . . these proposals are entered into a lottery to select those to be funded. The approach acknowledges that there are proposals of similar quality and merit, which cannot all be funded.*” [79]

Our approach only considers the relative ordering of the intervals and not additional information such as the amount of overlap between intervals, point estimates, or other information from a probabilistic model of the data. This makes our approach particularly applicable to settings where decision-makers have so-called “Knightian uncertainty”, that is, when they cannot assign a single probability measure over the set of possible outcomes [105, 196]. As further motivation, we describe a number of concrete settings in which a funder would construct intervals under this Knightian uncertainty:

1. *Imputation-based intervals*: in many grant funding panels most proposals have been reviewed by a large fraction of the reviewers. For example, in the Swiss NSF process [79], over 91% pairs of reviewer-proposal pairs received scores. If every reviewer scored every proposal, then the funder need not worry about reviewer miscalibration—the tendency for different reviewers to interpret the review scale differently[220]. Hence, the funder may first impute values for missing scores and then aggregate across reviewers. The funder can make minimal assumptions by adopting Manski bounds [139, 181]. Manski bounds generate *intervals* by imputing missing scores with a range of possible values from the minimum to maximum score. The aggregate scores are then given as an interval over the range of possible imputed values. Hence, the intervals represent the plausible set of values for each proposal’s aggregate review score, without a probabilistic interpretation.
2. *Intervals from model ensembling*: in many cases, a funder may have a number of plausible models of their data, each of which produces point estimates. Hence, the funder can estimate all plausible models of the data and ensemble into intervals by taking quantiles

of the point estimates. This type of ensembling has been applied to time-series forecasting problems where it is known as Quantile Prediction Averaging [161].

3. *Intervals based on expert input:* a frequently cited motivation for randomization is concerns about prejudice, for example, against highly original ideas or junior researchers. The funder may not be able to reliably estimate such sources of error from their observational, potentially sparse data, but can rely on prior controlled experiments that establish the rough magnitude of prejudices in the review process.
4. *Multi-criteria aggregation:* Reviewers are often given a number of criteria on which proposals are rated, for example “intellectual merit” and “broader impact.” The funder can consider multiple valid ways to aggregate criteria and generate intervals over the set of possible aggregations.
5. *Robustness to mis-specification of model used to generate intervals:* the funder may use a probabilistic model to estimate intervals and then draw conclusions about the ordering of the intervals, but may not trust other information about the distribution. This is a common assumption in the literature on distributionally robust optimization (see [172] for a survey), where the funder is said to be optimizing over a “support-only” ambiguity set.

3.1.3 Problem Formulation

Our methods apply to settings like admissions, scientific peer review, job screening, and financial investment, where decision makers estimate quality intervals and select top candidates based on these intervals. For concreteness, throughout our exposition, we will describe a *funder* choosing *proposals*.

Consider a funder who receives n proposals. From these, the funder wishes to select the k highest quality proposals. Note that n could be as large as thousands of proposals and k a fixed fraction of the total and can also be in the hundreds or thousands. For each proposal $i \in [n]$ ¹, the funder estimates an interval $[\ell_i, u_i] \subseteq \mathbb{R}$ representing a range of quality scores that the proposal could possibly take. A higher score indicates higher quality. The funder wishes to design a randomized selection mechanism to choose k proposals given the intervals. In order to design such a mechanism, we adopt two primary principles which we term as *ex ante* optimality and *ex post* validity, described below.

Ex ante optimality The funder’s utility is the expected number of the true top- k proposals that they select, ranked by their quality. Formally, let $\sigma : [n] \rightarrow [n]$ denote a ranking of the proposals where for each proposal $i \in [n]$, the rank of the proposal is denoted by $\sigma(i) \in [n]$. If the funder samples proposals with marginal probabilities $p \in [0, 1]^n$ and the true ranking of proposals is σ , their expected utility is $\sum_{i=1}^n p_i \mathbb{1}\{\sigma(i) \leq k\}$. Clearly, if the funder knew the true ranking σ , then they would optimize utility by choosing deterministically, i.e., by setting $p_i = 1$ for i with $\sigma(i) \leq k$, and 0 otherwise.

However, recall that the funder is uncertain about relative qualities of the proposals, as captured by overlaps in intervals. Any ordering that is consistent with overlapping intervals could

¹We use the standard notation $[\kappa]$ to denote set $\{1, \dots, \kappa\}$ for any positive integer κ .

be the true ranking. Hence, the intervals define a set of feasible rankings:

$$\Sigma_n = \{\sigma \text{ permutation of } [n] \mid \forall i, j \in [n], \ell_i > u_j \implies \sigma(i) < \sigma(j)\}$$

In other words, if proposal i has quality strictly above proposal j , then i is ranked higher than j in all $\sigma \in \Sigma_n$. As an example of an extreme case, if all n intervals overlap each other, then Σ_n consists of all possible permutations of the proposals.

For ex ante optimality, the funder optimizes their *worst case* utility over feasible rankings Σ_n :

$$\min_{\sigma \in \Sigma_n} \sum_{i=1}^n p_i \mathbb{1}\{\sigma(i) \leq k\} \quad (3.1)$$

The funder maximizes their worst-case expected utility by choosing the optimal marginal probabilities p solving the maximin optimization problem:

$$\max_{\substack{p \in [0,1]^n: \\ \|p\|_1 = k}} \min_{\sigma \in \Sigma_n} \sum_{i=1}^n p_i \mathbb{1}\{\sigma(i) \leq k\}. \quad (3.2)$$

Finally, the funder randomly chooses n proposals with marginal probabilities corresponding to p .

The ex ante optimization problem has a game theoretic interpretation that motivates the need for randomization. Our model corresponds to a zero-sum Stackelberg game where the funder is the “leader” who selects k proposals. The funder faces an adversarial “follower” who chooses a ranking of proposals. The leader’s utility is the number of top k proposals selected based on the adversary’s ranking, while the adversary’s utility is the negation of the leader’s. The Strong Stackelberg Equilibrium (SSE) is exactly the solution to Objective 3.2. It is well known that in an SSE, the leader may need to commit to a randomized (or mixed) strategy.

Ex post validity The ex post validity criterion requires that for any pair of proposals a and b , if b ’s quality interval lies strictly below a ’s interval and if b is selected, then a must also be selected.

Formally, a selection rule that takes as input a set of quality intervals I and outputs a set of selected proposals S satisfies *ex post validity*, if for all pairs of intervals $a, b \in I$ with $\ell_a > u_b$, and all outputs S selected with non-zero probability, $b \in S \implies a \in S$.

The ex post validity criteria ensures that the actual selected set of proposals is legitimate to stakeholders. In particular, if the funder rejects a proposal that dominates an accepted proposal, that would be unacceptable to the funder and to applicants.

While the ex post condition seems natural, the simple randomize-above-threshold mechanism can violate it: Suppose proposals a and b both lie above the threshold, but a dominates b . Because a and b are entered into a uniform lottery, a may be rejected at random, while b is accepted at random.

3.1.4 Efficient MERIT Algorithm

The ex ante optimization problem in (3.2) is equivalent to solving the following linear program (LP):

$$\begin{aligned}
 & \max_{p \in \mathbb{R}^n, v \in \mathbb{R}} v & (3.3) \\
 \text{subject to } & v \leq \sum_{i=1}^n p_i \mathbb{1}_{\{\sigma(i) \leq k\}}, \quad \forall \sigma \in \Sigma_n, \\
 & \sum_{i=1}^n p_i = k \text{ and } 0 \leq p_i \leq 1, \forall i \in [n]
 \end{aligned}$$

This LP can have on the order of $\binom{n}{k}$ constraints so its size is exponential in k . Recall that in practice, n can be thousands of proposals, out of which the funder selects k in the many hundreds or thousands, so this LP is intractably large. In fact, determining feasibility of a solution to the LP when Σ_n is defined by an arbitrary partial order, called the *minimum weight k -ideal problem*, is NP-hard [58]. Prior work has shown that the problem is solvable in polynomial time for a tree partial order, but our problem uses an interval order. Several algorithmic problems related to interval orders are also known to be NP hard, including maximum cut on interval graphs [2], interval graph completion [174] and minimizing weighted completion time of jobs subject to interval ordering precedence constraints [229].

In contrast, as we show in Section 3.1.4, the ex ante optimization problem is solvable in polynomial time using the ellipsoid method with a separation oracle. We present a practical cutting plane algorithm based on the theoretically polynomial time algorithm in Section 3.1.4. Finally, in Section 3.1.4, we describe how to ensure that random sampling with optimal marginal probabilities satisfies both ex ante and ex post conditions simultaneously, with an efficient algorithm. We call this end-to-end algorithm, summarized in Section 3.1.4, **Maximin Efficient Randomized Interval Top- k (MERIT)**.

Polynomial-Time Separation Oracle

We now develop a polynomial-time algorithm to solve linear program (3.3). Our approach solves the problem using a polynomial time “separation oracle” [74] with the ellipsoid algorithm. A separation oracle checks whether a proposed solution satisfies all the constraints. If the solution is feasible, the oracle confirms it. If not, it identifies (at least one) specific constraint that the solution violates. The separation oracle may be used to solve the LP without enumerating all (exponentially many) constraints by starting with a limited set of constraints and iteratively shrinking the possible feasible region of the LP through calls to the separation oracle. Our main result proves that this method yields a polynomial time algorithm:

Theorem 3.1.2 (Polynomial time solution). *The linear program (3.3) can be solved within accuracy ϵ of the optimal solution in polynomial time with respect to n and $\log(1/\epsilon)$ using the ellipsoid algorithm with Algorithm 3 as a separation oracle.*

The primary technical difficulty is the design of an efficient separation oracle. We present our separation oracle in Algorithm 3. Given a candidate solution (p, v) , the separation oracle

checks whether p achieves a worst-case objective value of at least v . If the worst-case objective value under p is greater than v , then the solution is feasible, if not the oracle returns a set of violated constraints. At a high level, the oracle works by constructing worst-case possible sets of top- k proposals. For each of the $k + 1$ intervals with the largest lower bounds, the algorithm constructs the worst-case set of top- k proposals that includes intervals 1 to $(i - 1)$ and excludes interval i in the top- k . Excluding an interval with a large lower bound constrains the set of intervals that must be in the top- k , since all intervals strictly below the interval with the large lower bound must be excluded from the top- k . In considering all such sets of intervals, the algorithm enumerates possible worst-case permutations with respect to p in time $O(nk)$. If the separation oracle finds a permutation that gives objective values smaller than v , it returns this permutation, which corresponds to a violated constraint in the LP. If it does not find any such permutation, then (p, v) is feasible. This separation oracle is used as a sub-routine in the ellipsoid algorithm to compute the optimal solution in polynomial time in Theorem 3.1.2.

Lemma 3.1.3 (Polynomial-time separation oracle). *For any candidate solution (p, v) to the linear program (3.3), Algorithm 3 returns \emptyset only if the candidate solution is feasible and returns a non-empty set of violated constraints otherwise. Moreover, Algorithm 3 runs in time $O(n \max\{k, \log n\})$.*

Algorithm 3 Polynomial-time Separation Oracle

Input: Candidate solution: $(p, v) \in [0, 1]^n \times \mathbb{R}$ with $\|p\|_1 = k$, number selected k , set of intervals $\{\ell_i, u_i\}_{i \in [n]}$ sorted in decreasing order of lower bound ℓ_i

Output: A set of violated constraints (\emptyset if (p, v) is feasible)

$Z \leftarrow \emptyset$

for $i = 1$ to $k + 1$ **do**

$S_i \leftarrow \{j \in (i, n] : \text{intervals } j \text{ and } i \text{ overlap}\}$

if $|S_i| \geq (k - (i - 1))$ **then**

Obtain \tilde{S}_i by sorting S_i by p and keeping only the $k - (i - 1)$ smallest values

if $v > \sum_{j=1}^{i-1} p_j + \sum_{j \in \tilde{S}_i} p_j$ [†] **then**

$Z \leftarrow Z \cup \left\{ \left\langle v \leq \sum_{j=1}^{i-1} p_j + \sum_{j \in \tilde{S}_i} p_j \right\rangle \right\}$

end if

end if

end for

return Z

[†]By convention, we take the empty sum from $j = 1$ to 0 to be 0.

Practical Cutting-Plane Algorithm

While theoretically polynomial time, in practice, the ellipsoid algorithm is impractical for problems with even a few hundred proposals. Hence, taking inspiration from the ellipsoid algorithm, we develop a practical “cutting plane” method to solve the LP (3.3).

The cutting plane algorithm is described in full in Algorithm 4. The algorithm starts by solving a relaxation of the LP without any of the worst-case value constraints to find an initial

(potentially infeasible) candidate solution (p, v) . Then, the algorithm repeatedly calls the separation oracle to check the feasibility of the current candidate solution. If the candidate solution is feasible, it is an optimal solution to the LP, since it is optimal for a relaxation of the full LP. If the candidate solution is infeasible, the algorithm adds the constraints returned by the separation oracle and re-solves the LP.

Algorithm 4 Cutting Plane Algorithm

Input: Number of proposals to select k , set of intervals $I = \{\ell_i, u_i\}_{i \in [n]}$ max iterations T

Output: Ex ante optimal vector of marginal probabilities p

Prune Intervals

For all intervals strictly below at least k other intervals, set $p_i = 0$ and remove.

For all intervals strictly above at least $n - k$ other intervals, set $p_i = 1$ and remove.

Let a be the number of intervals accepted. Update $k \leftarrow k - a$.

Initialize Linear Program

For each interval $i \in [n]$ compute $A(i)$, the number of proposals strictly above i and $B(i)$, the number of proposals strictly below i .

Using A and B , partition the intervals into w monotonically ordered subsets M_1, \dots, M_w .

Solve the following linear program to obtain initial p, v :

$$\begin{aligned}
 & \min_{v,p} && v \\
 & \text{s.t.} && \sum_{i=1}^n p_i = k \\
 & && p_i \in [0, 1] \quad \forall i \in [n] \\
 & && v \leq \sum_{j=1}^k p_j \\
 & && p_{M[i]} \geq p_{M[i+1]} \quad \forall i \in [|M| - 1], \forall M \in \{M_1, \dots, M_w\}
 \end{aligned}$$

Add Cuts

for T iterations **do**

$C \leftarrow \text{SeparationOracle}((p, v), k, I)$

if $C = \emptyset$ **then**

 ▷ Feasible solution

return p

else

 ▷ Infeasible solution

 Add constraints from C to the LP and solve the new problem to obtain new (p, v) .

end if

end for

return Failure

The cutting plane algorithm converges to a feasible optimal solution quickly in practice because it is initialized with a useful set of constraints on the feasible region of the problem.

These constraints prune the problem and impose monotonicity and symmetry constraints on the marginal probabilities p , based on the number of intervals above and below each proposal, which we define below.

Definition 3.1.4 (Number above (A) and number below (B)). *For each proposal $i \in [n]$, define:*

$$A(i) = |\{r \in [n] : \ell_r > u_i\}| \quad \text{and} \quad B(i) = |\{r \in [n] : \ell_i > u_r\}|,$$

that is, $A(i)$ is the number of intervals strictly above i and $B(i)$ is the number of intervals strictly below i .

The cutting plane algorithm first prunes all intervals guaranteed to always be in the top k or never be in the top k , which reduces the number of decision variables. Then, the cutting plane algorithm initializes the linear program with a set of constraints on the optimal marginal probabilities p . In particular, the algorithm partitions the intervals into “monotonically ordered subset” within which p can be assumed to be monotonically non-increasing without loss of optimality:

Definition 3.1.5 (Monotonically ordered subset). *For any subset of intervals $M \subseteq [n]$, we say that the subset of intervals is monotonically ordered if $\forall i \in [|M| - 1]$, $A(M[i]) \leq A(M[i + 1])$ and $B(M[i]) \geq B(M[i + 1])$, where $M[i]$ denotes the i -th element in M .*

The monotonicity constraints come from the observation that if proposal a has more proposals below it than proposal b and has fewer proposals above it than proposal b , then without loss of optimality, $p_a \geq p_b$. Our cutting plane algorithm therefore imposes constraints on these “monotonically ordered subsets” at the start, which greatly constrains the size of the feasible region, while still returning an optimal solution. We provide detailed analysis of the correctness and efficiency of this algorithm in Supplemental Material.

Enforcing Ex Post Validity

A solution to the ex ante optimality LP (3.3) returned by the Cutting Plane Algorithm (Algorithm 4) or the Ellipsoid Algorithm, is not guaranteed to output a vector of marginal probabilities, such that sampling proposals with these marginals always guarantees ex post validity. However, we prove that we can post-process any solution to the ex ante optimization problem, and then sample with marginal probabilities p to guarantee ex ante optimality and ex post validity simultaneously. This stands in contrast to the commonly used “randomize-above-threshold” approach to randomization, which does not guarantee ex post validity as described in Section 3.1.3.

Theorem 3.1.6 (Post-processing for ex post validity). *Given any ex ante optimal p , Algorithm 5 enables the funder to sample k proposals while satisfying both ex ante and ex post conditions and is computable in time $O(n^2)$.*

Theorem 3.1.6 applies the post-processing algorithm given in Algorithm 5 to a solution from the Cutting Plane Algorithm. For any $a, b \in [n]$ with $\ell_a > u_b$, Algorithm 5 terminates with $p_a = 1$ or $p_b = 0$. Moreover, Algorithm 5 never decreases the objective value of p . Hence, applying post-processing to an ex ante optimal solution is without loss of optimality and ensures that any sampling method that selects proposals with marginal probabilities p satisfies ex post validity.

Finally, there are many possible methods to implement this sampling (see [201] for a survey). We use the simple and classical “systematic sampling” [136] with runtime $O(n)$. This

method requires only one uniform random sample to implement, so the randomness could be easily documented for transparency, which is an important part of the current Swiss NSF procedure [203] where a lottery is conducted by publicly drawing papers from a glass jar. We provide pseudo-code in Supplemental Material.

Algorithm 5 Post-Processing of p for Ex Post Validity

Input: Vector of marginal probabilities p , sequence of intervals $\{[\ell_i, u_i]\}_{i \in [n]}$
Output: Vector of marginal probabilities p
Order the intervals by increasing u .
for $b \in [n]$ **do**
 if $p_b = 0$ **then**
 continue
 end if
 for a from n to $(b + 1)$ **do**
 if $\ell_a > u_b$ and $p_a < 1$ **then**
 $d \leftarrow \min\{p_b, 1 - p_a\}$
 $p_b \leftarrow p_b - d$
 $p_a \leftarrow p_a + d$
 end if
 end for
end for

Complete MERIT Procedure

The complete algorithm MERIT solves the ex ante optimization procedure with post-processing for ex post validity (Algorithm 5) followed by sampling. The algorithm is provably polynomial time if the ex ante optimization problem is solved using the ellipsoid method with a separation oracle as proved in Theorem 3.1.2. In practice, we solve the optimization problem using the cutting plane algorithm given in Algorithm 4, which yields a practically efficient algorithm, albeit with a non-polynomial time theoretical convergence guarantee, described in Supplemental Material.

Algorithm 6 MERIT Algorithm

Input: Number of proposals to select k , set of intervals $I = \{\ell_i, u_i\}_{i \in [n]}$
Output: Selection of k proposals
Compute an ex ante optimal vector of marginal probabilities p using Algorithm 4.
Apply ex post validity post-processing to p (Algorithm 5).
Sample k proposals from $[n]$ with marginal probabilities of inclusion given by p .

3.1.5 Axiomatic Comparison of Lottery Rules

In many of the applications we consider, like scientific funding or college admissions, there is no agreed upon ground-truth measurement of the quality of selections made by the decision maker.

Hence, it is unclear how to empirically measure whether one algorithm performs better than another algorithm in selecting “top” candidates. Therefore, we take inspiration from social choice theory [20] and initiate an axiomatic comparison of our MERIT method with the alternative methods discussed in Section 3.1.2—deterministic top- k selection, randomize above threshold, and the Swiss NSF method. We consider a number of natural properties, or “axioms”, that a selection algorithm should obey and theoretically analyze the behavior of MERIT and alternative algorithms with respect to these axioms.

Axioms for Randomized Selection

We propose three natural desiderata of algorithms for selecting proposals from quality assessments. We begin by defining a generic “randomized selection rule” which we will characterize axiomatically.

Definition 3.1.7 (Selection rule). *A selection rule receives as input a set of n quality estimates $I = \{(\ell_i, e_i, u_i)\}_{i \in [n]}$ where ℓ_i, u_i are lower and upper limits on the quality of the item i and $e_i \in [\ell_i, u_i]$ is a point estimate of the quality. Given a budget $k \in \{1, \dots, n\}$, the selection rule outputs a subset of $[n]$ of size k . We let $p(I, k) \in [0, 1]^n$ denote the vector of marginal probabilities of selecting each item:*

This definition captures methods that do not actually use intervals to make decisions like deterministic selection, methods that use only intervals like MERIT, and methods that use both intervals and point estimates like the Swiss NSF’s approach. Next, we define three axioms that a selection rule ought to obey.

First, we propose the axiom of “**monotonicity in budget**”. This axiom requires that increasing the number of proposals selected, k , should not decrease the probability of selection for any proposal. Formally:

Definition 3.1.8 (Monotonicity in budget). *A selection rule respects monotonicity in budget, if for any input I and for all budgets $k \in [n - 1]$, it must be that $p(I, k + 1)_i \geq p(I, k)_i \forall i \in [n]$.*

Second, an important property for a selection rule is “stability” — changing one interval should not lead to a large change in the behavior of the algorithm. We define an instance of instability that is undesirable for an algorithm. We consider two “extreme” behaviors for an algorithm. At one extreme, an algorithm behaves deterministically, selecting k proposals with probability 1, which is the minimum entropy distribution from which to sample. At the other extreme, the algorithm samples uniformly at random among all n proposals, which is the maximum entropy distribution. We say that an algorithm exhibits “**maximal instability**”, if changing a single interval by an arbitrarily small amount can change the algorithm’s behavior between these two extremes. Formally:

Definition 3.1.9 (Maximum instability). *A selection rule is maximally unstable, if there exist a pair of inputs I and J that differ by an arbitrarily small amount $\epsilon > 0$ in the quality estimate of a single proposal and a budget $k \in \{2, \dots, n - 2\}$ such that $p(I, k) = \frac{k}{n} \mathbf{1}_n$ (the mechanism samples uniformly at random among all proposals on input I) whereas $p(J, k) \in \{0, 1\}^n$ (the mechanism selects deterministically on input J).*

A selection algorithm should *avoid* maximum instability. We note that in the definition of instability, we restrict the budget to lie in $\{2, \dots, n - 2\}$, because stability with respect to chang-

ing a single proposal is not meaningful when choosing only one proposal to accept or reject. In particular, the top-1 or bottom-1 proposal can change entirely based on a change to a single proposal.

Finally, inspired by the “**reversal symmetry**” axiom from social choice theory [180], we define a natural notion of symmetry that says that when selecting 1 out of 2 proposals, if the scale of quality is reversed so that all intervals are flipped, then the selection rule should flip the probabilities of selection for the two proposals:

Definition 3.1.10 (Reversal symmetry). *For any input $I = \{(\ell_i, e_i, u_i)\}_{i \in [n]}$ where $\ell_i, e_i, u_i \in [0, 1] \forall i \in [n]$, let $I^{(R)} = \{(1 - u_i, 1 - e_i, 1 - \ell_i)\}_{i \in [n]}$ be the horizontally flipped (or reversed) input. Then, a selection rule that selects $k = 1$ out of $n = 2$ proposals respects reversal symmetry if for any pair of flipped inputs I and $I^{(R)}$, $p(I, 1) = (p_1, p_2)$ and $p(I^{(R)}, 1) = (p_2, p_1)$.*

Axiomatic Guarantees and Failures

We now present our theoretical results on which algorithms satisfy the three axioms defined. Deterministic top- k selection meets both criteria, but does not account for the funder’s uncertainty (our earlier ex ante requirement). We characterize randomized selection mechanisms with respect to these axioms in Theorem 3.1.11:

Theorem 3.1.11 (Axiomatic analysis). *Existing randomized algorithms have the following properties:*

1. *Swiss NSF and randomize-above-threshold both violate maximum instability, while MERIT is never maximally unstable.*
2. *Swiss NSF, randomize-above-threshold and MERIT all violate monotonicity in budget.*
3. *It is not possible to simultaneously satisfy ex ante optimality and monotonicity in budget.*
4. *Swiss NSF and randomize-above threshold violate reversal symmetry, while MERIT satisfies reversal symmetry.*

We give a formal proof of Theorem 3.1.11 in Supplemental Material. Here we provide examples where the three axioms fail. First, Figure 3.1 shows an example where none of the randomized selection mechanisms considered respect monotonicity in budget. In particular, when $k = 1$, both Swiss NSF and MERIT sample uniformly at random between 1 and 2, so proposal 2 is selected with probability $\frac{1}{2}$. However, when $k = 2$, proposal 1 is always in the top 2 and the other top-2 proposal could be any one of 2, 3, or 4, so both algorithms choose proposal 1 deterministically and sample uniformly among 2, 3, 4 meaning proposal 2 is selected with probability $\frac{1}{3}$. Hence, perhaps counterintuitively, proposal 2 is worse off when the budget k increases from 2 to 3. We note that randomize-above-threshold behaves identically to Swiss NSF here, if the threshold is chosen in a data-dependent manner as the k -th highest point estimate, so this same example shows that randomize-above-threshold violates monotonicity.

The example in Figure 3.1 also demonstrates claim (3) of Theorem 3.1.11 that it is not possible to simultaneously satisfy ex ante optimality and monotonicity in budget. In particular, the unique ex ante optimal p for this example are not monotonic in budget. As we discuss in Section 3.1.5, it is possible to modify MERIT to enforce monotonicity in the budget at the potential cost of ex ante optimality.

Interestingly, in reproducing the Swiss NSF’s selection process using real 2020 grant review

data, we find that both the Swiss NSF selection rule and MERIT exhibit non-monotonicity in budget in practice. For example, using the Swiss NSF method, there are proposals whose probability of selection *decreases* from 0.75 to 0.524 when increasing budget k from 89 to 90. This suggests that violations of the monotonicity in budget axiom can occur in real-world deployments.

Second, in Figure 3.2, we show an example that violates maximum instability for Swiss NSF and (data-dependent) randomize-above threshold. In this example, where $k = 3$, slightly decreasing the upper bound and point estimate of interval 3, changes the algorithm’s behavior from choosing the top-3 deterministically, to sampling uniformly at random among all proposals. In contrast, MERIT never exhibits maximal instability (for $k \in [2, n - 2]$). Our proof of this is in Theorem 3.1.11, follows by observing that MERIT only samples uniformly at random among all proposals if all intervals overlap, and will not select deterministically for any set of intervals where only a single interval does not overlap the remaining intervals.

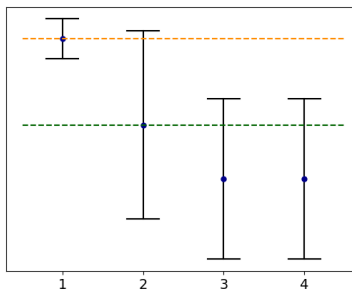


Figure 3.1: Example that violates *monotonicity* with respect to k for Swiss NSF and our MERIT algorithm. When $k = 1$, $p_2 = 1/2$ for both algorithms. However, when $k = 2$, $p_2 = 1/3$ for both algorithms.

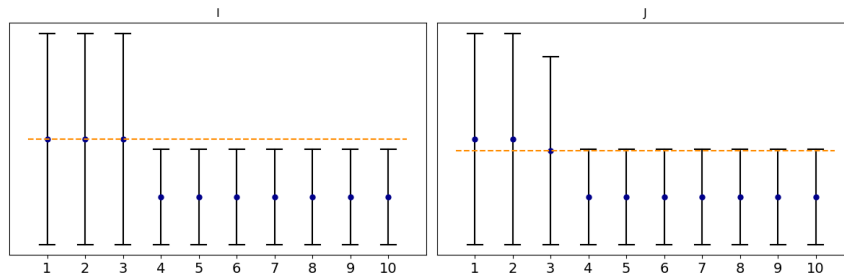


Figure 3.2: Example that violates *maximum instability* for Swiss NSF and randomize-above-threshold, with $k = 3$. Slightly decreasing the upper bound and point estimate of proposal 3, changes the algorithm’s behavior from selecting the top 3 deterministically (left) to sampling among all 10 proposals uniformly at random (right).

Finally, an example where reversal symmetry is violated by the Swiss NSF method is when selecting one proposal from two intervals with values $(0, 1)$ and $(0.1, 0.2)$ and point estimates at the midpoints of 0.5 and 0.15. Then, the Swiss NSF selection rule will accept interval 1 and reject interval 2. However, if quality is flipped, then proposal 1 remains the same, but proposal 2 has quality interval $(0.8, 0.9)$ with point estimate of 0.85. Then, for the reversed instance, Swiss

NSF samples uniformly at random between the two proposals, violating reversal symmetry. In contrast, MERIT samples uniformly at random between the two for both the original and reversed intervals.

Enforcing Monotonicity in Budget

In this section, we show that it is possible to modify MERIT to enforce monotonicity in budget, by solving the ex ante optimization problem sequentially for a budget ranging from 1 to the desired k with an additional constraint to the optimization problem that the sequence of p should be non-decreasing. At each step, we solve the optimization problem and then apply post-processing for ex post validity using Algorithm 5. This may come at the cost of ex ante optimality, but always enforces monotonicity in budget and ex post validity as we prove in Proposition 3.1.12.

Algorithm 7 Solving Sequence of Optimization Problems for Monotonicity

Input: Sequence of intervals $\{[\ell_i, u_i]\}_{i \in [n]}$, number selected k .

Output: Sequence of marginal selection probabilities $p^{(1)}, \dots, p^{(k)}$ for selecting 1 to k proposals.

Let $p^{(0)} = (0, \dots, 0)$

for $i \in [k]$ **do**

 Compute ex ante optimal p for selecting top- i proposals using Algorithm 4 with the additional constraint $p_j^{(i)} \geq p_j^{(i-1)} \forall j \in [n]$ to obtain optimal $p^{(i)}$.

 Post-process $p^{(i)}$ with Algorithm 5 for ex post validity.

end for

return $p^{(1)}, \dots, p^{(k)}$.

Proposition 3.1.12 (Monotonicity in budget k). *Algorithm 7 guarantees that $p^{(i)} \geq p^{(i-1)}$ for all $i \in [k]$ and any algorithm that samples i proposals with marginal probabilities $p^{(i)}$ respects ex post validity.*

3.1.6 Empirical Evaluation

In order to compare the performance of different selection algorithms, we use review data where we assume known proposal quality (or intervals containing the true quality). The funder observes error-prone estimates of quality—both interval estimates and point estimates—and then applies a selection rule. We present head-to-head comparisons of different selection algorithms under both the linear miscalibration model used by the Swiss NSF and other prior works, and under our worst-case model in Section 3.1.6. We then describe computational runtime experiments demonstrating the efficiency of our algorithm in MERIT in Section 3.1.6. Finally, we present qualitative insights into how different methods may lead to different outcomes in real peer review settings in Section 3.1.6.

Comparison of Methods in Expected Utility and Worst-Case Utility

We evaluate the performance of selection algorithms from two perspectives: (1) expected utility in a probabilistic model of reviewer behavior, and (2) worst-case utility over feasible rankings defined by score uncertainty intervals. While our algorithm, MERIT, is designed for worst-case optimality, it is informative to assess its expected performance under plausible generative models of the review process. Defining a realistic probability distribution over reviewer scores is challenging, but one model that has been widely adopted in the literature is a linear miscalibration model, which underlies the interval-generation method of the Swiss National Science Foundation (Swiss NSF) [79] and has been used in many other peer review settings [6, 62, 177, 178, 199].

Expected Utility under a Linear Miscalibration Model A popular probabilistic model used by the Swiss NSF and many others captures error in the review process due to *reviewer miscalibration*, the tendency for some reviewers to be stricter or more lenient than others. The model is linear, where each proposal has a numeric true quality and each reviewer has a miscalibration offset, both drawn from Gaussian distributions. The review score for a reviewer on a specific paper is a linear combination of true quality and reviewer offset plus Gaussian noise. This model has been applied widely to peer review data in prior work [6, 62, 177, 178]. The model is specified by an assignment of reviewers to proposals and three parameters for the standard deviation of the quality (σ_θ), reviewer miscalibration (σ_b), and per-review error (σ_ϵ).

We generate synthetic data from this model in two regimes based on real-world settings: (1) Swiss NSF’s grant review process with 350 proposals, 10 reviewers, and 80 proposals per reviewer and (2) Computer Science conference peer review with 1000 reviewers, 1000 papers, and 5 papers per reviewer. In both cases, we assign papers to reviewers uniformly at random and assume a score range of 1 to 10 with $\sigma_\theta = 2$, $\sigma_b = 1$ and $\sigma_\epsilon = 0.5$, matching parameters used in prior work [199]. In both settings, the funder aims to select the top one-third of proposals, reflecting realistic acceptance rates.

We generate 50 synthetic samples of review datasets. Then, following the methodology of the Swiss NSF, estimate model parameters and posterior expected rank of each proposal along with 50% confidence intervals for the expected rank given the observed review data. These point estimates and intervals are given as input to each selection rule. We test both a deterministic rule that selects the top- k by mean score, denoted *Deterministic (mean)*, and a deterministic rule that selects the top- k by estimated expected rank based on the model, denoted *Deterministic (model)*.

Our Worst-Case Objective Our worst-case objective (3.1), assumes that any ordering consistent with quality intervals could be the true ranking of proposals. The utility is then defined as the worst-case expected fraction of top- k proposals chosen over all possible orderings consistent with the intervals. We simulate generating such quality intervals in three scientific peer review scenarios and measure the performance of different selection methods with respect to our worst-case objective. First, we replicate the scientific grant funding process of the Swiss NSF. We use publicly released review data from their 2020 grant review process consisting of $n = 353$ proposals [79]. We generate intervals using their linear model of reviewer miscalibration. Additionally, we generate imputation-based intervals for the Swiss NSF data using Manski

bounds to impute missing reviewer-proposal scores and aggregating scores across reviewers using the median. Second, we use paper review data from the NeurIPS 2024 conference, available on OpenReview, which includes $n = 4035$ accepted papers. We simulate a process of allocating long talks (orals and spotlights) among accepted papers, inspired by the USENIX Security 2025 [209]. Third, we use paper review data from the ICLR 2025 conference, available on OpenReview, which includes all $n = 11520$ papers submitted to the conference. We simulate allocating paper acceptances at this conference. For NeurIPS and ICLR, there is no standard method to generate intervals. We therefore implement three different methods for generating intervals on NeurIPS and ICLR data:

1. *Leave-one-out intervals (LOO)*: taking inspiration from “jackknife” or leave-one-out intervals [10], we compute the range of possible mean review scores, leaving out one reviewer at a time for each paper.
2. *Gaussian error model credibility intervals*: similar to the Swiss NSF’s intervals, we assume a Gaussian model generates review scores based on underlying true quality scores and infer credibility intervals for the true quality. Specifically, for paper i , we assume paper has true quality $\theta_i \sim \mathcal{N}(0, 2)$, precision $\tau_i \sim \text{Gamma}(1, 1)$, and review scores on the paper are drawn i.i.d. from $\mathcal{N}(\theta_i, 1/\sqrt{\tau_i})$. The parameters of the priors are chosen to closely match those of the Swiss NSF model. We infer 50% credibility intervals for θ_i given the observed review scores using MCMC.
3. *Subjectivity intervals*: NeurIPS 2024 and ICLR 2025 both asked reviewers to provide numerical scores of papers’ soundness, presentation, and contribution in addition to overall scores. Previous works have observed that different reviewers may have different subjective views of which criteria matter to a paper’s quality introducing arbitrariness into overall review scores [118]. One proposed approach to mitigate this subjectivity is to learn a mapping from sub-criteria to an overall score based on peer review data [159] and then use this mapping to adjust the review score. We generate intervals by applying this method to adjust scores and taking the interval to be all values in between original scores and subjectivity adjusted scores.

We note that there are no confidence intervals for estimates generated in the worst-case setting as these are generated for a single dataset with a single set of intervals, where there is no randomness in the data generation process.

Results In Figure 3.3 we show results of our simulations. Under the Swiss NSF’s fully Bayesian linear model, MERIT performs similarly in expectation to the Swiss NSF’s randomized method and to deterministic selection using model-adjusted scores. Using the raw mean scores yields much lower performance, as this method does not account for the miscalibration error. In contrast, in the worst-case setting of our model, MERIT outperforms other methods on all baselines. This is expected as MERIT optimizes for this objective. Interestingly, our experiments demonstrate that the optimality gap can be quite large, with MERIT achieving utility of up to 0.19 more than Swiss NSF, the other randomized algorithm considered. Deterministic selection sometimes performs poorly, for instance on NeurIPS LOO and NeurIPS Gaussian, when intervals are relatively wide. Even though Swiss NSF’s selection rule randomizes based on the intervals, the optimality gap between Swiss NSF and our ex ante optimal MERIT algorithm can also be large,

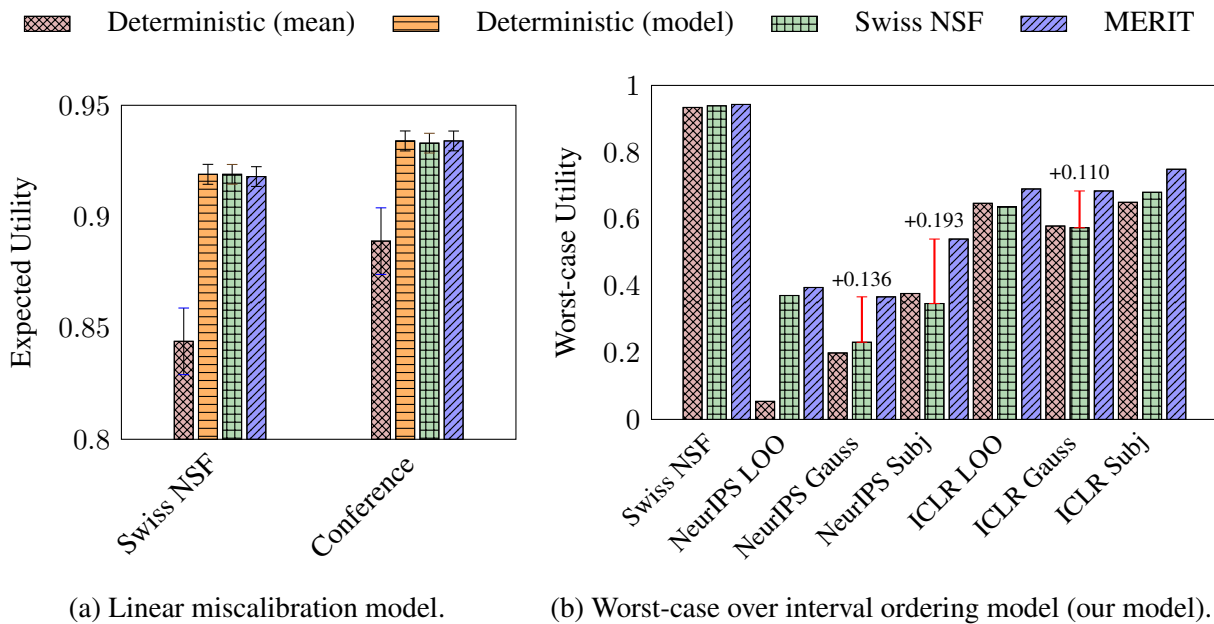
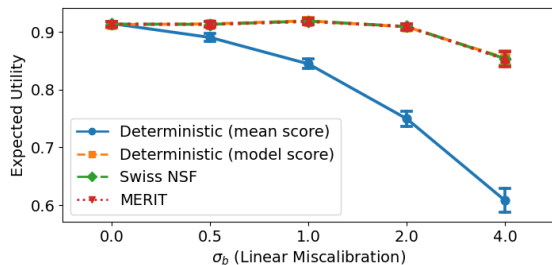


Figure 3.3: Proportion of top- k proposals selected by different methods with quality data generated under the Swiss NSF model of linear miscalibration and under our model of worst-case over feasible rankings. MERIT matches performance of algorithms designed for the Swiss NSF’s linear model, with expected utility averaged over 50 samples of synthetic data and error bars showing 95% CI for the sample mean. The gap between MERIT and other methods in the worst-case over intervals defined by our model can be substantial, as shown by the gaps in NeurIPS Gaussian, NeurIPS Subjectivity, and ICLR Subjectivity.

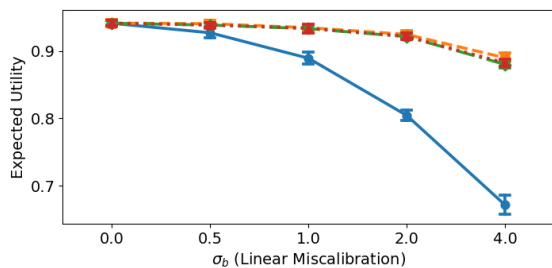
for example on NeurIPS Gaussian and NeurIPS Subjectivity data, where Swiss NSF does not perform much better than deterministic selection, while MERIT has higher utility. Taken together, these results suggest that MERIT does not degrade expected utility in the fully Bayesian setting, while offering additional robustness with respect to our worst-case objective.

We additionally conduct ablations on parameters of both models that provide insight into the relative performance of different methods under varying model settings. First, in Figure 3.4, we show utility of different methods in the Swiss NSF’s linear miscalibration model varying the degree of miscalibration. As the magnitude of miscalibration increases, deterministic selection using mean score degrades greatly because it does not account for error due to miscalibration at all. Meanwhile, the other methods perform similarly and maintain fairly high expected utility, even with a large degree of miscalibration. We observe similar results in additional ablations of the miscalibration model and when selecting the top one-tenth of proposals instead of top one-third, as shown in Supplemental Material.

In Figure 3.5, we test Manski bounds on the Swiss NSF dataset of grant reviews, where we impute missing values with the full range of scores. We artificially drop reviews to increase the sparsity of review scores, leading to worse utility. With 40% sparsity, deterministic achieves near zero utility as almost all intervals overlap. At all sparsity levels, MERIT outperforms both Swiss NSF and Deterministic selection. In fact, at sparsity of 0.15 to 0.25, the optimality gap between



(a) Swiss NSF simulated data.



(b) Conference simulated data.

Figure 3.4: Ablation study comparing methods under the Swiss NSF’s model of linear miscalibration with varying levels of miscalibration. Error bars show bootstrapped 95% CIs for the sample mean over 50 samples of randomly generated data from the model.

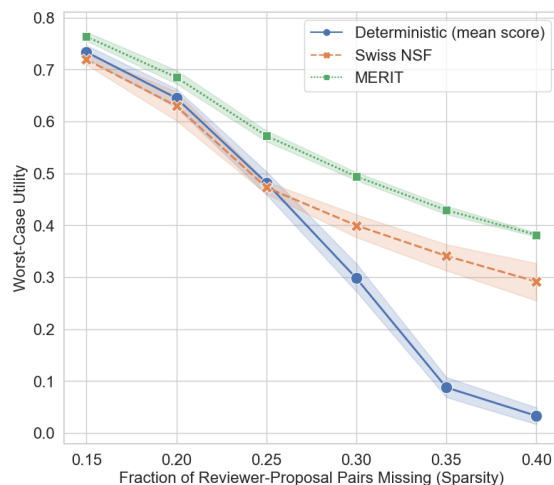


Figure 3.5: Worst-case utility over Manski bound intervals as a function of the fraction of reviewer-proposal pairs missing, with random dropping of review scores to increase sparsity for the Swiss NSF panel review dataset. Error bars show bootstrapped 95% CIs for the sample mean over 50 trials of randomly dropping review scores.

Swiss NSF and MERIT is the same as that of deterministic selection and MERIT.

Computational Efficiency

All experiments are run locally on a standard personal laptop (a 2019 MacBook Pro with a 2.6 GHz 6-core Intel Core i7 processor) without any specialized hardware. We solve linear programs using Gurobi 12.0.1 [77], which solves LPs using the simplex method. We find that for all methods MERIT runs in under five minutes, even with over 11,000 proposals. We provide further analysis on runtime, size of the LP solved, and convergence rate in Supplemental Material including details on variations in runtime as a function of different datasets and interval generation methods.

Comparison of Outcomes in Case Studies

In order to give insight into how differing approaches may lead to different types of lotteries in actual peer review settings, we present qualitative differences between MERIT and the Swiss NSF approach.

In Table 3.1, we provide a high-level comparison of the marginal selection probabilities under MERIT and the Swiss NSF algorithm. One simple comparison point is the number of proposals

that entered into a lottery (% Random) under each algorithm. We find that MERIT tends to randomize over an equal or greater number of candidates than the Swiss NSF algorithm—on Swiss NSF data, NeurIPS LOO, NeurIPS Gaussian, and ICLR Subjectivity the two algorithms randomize over a similar number of candidates, while on NeurIPS Subjectivity, ICLR LOO, and ICLR Gaussian, MERIT randomizes over more candidates than Swiss NSF. The biggest difference between the two approaches with respect to the outcome is that the Swiss NSF rule assigns the same probability to every proposal that enters its lottery tier, whereas MERIT allows a range of probabilities: the broadest span is on NeurIPS Gaussian where marginal probabilities range from 0.16 to 0.83. This reflects the additional flexibility of MERIT’s lottery. Finally, the relative weight MERIT puts in the “certain accept” group compared to Swiss NSF ($p = 1$) is not uniform—it is higher than Swiss NSF for NeurIPS Subjectivity, essentially tied on the original Swiss NSF set, and lower for the two ICLR variants, so the split between guaranteed and lottery funding depends on the structure of the intervals in each study rather than following a single trend.

Dataset	MERIT			Swiss NSF Algorithm		
	% Accept	% Random	Range of p	% Accept	% Random	Value of p
Swiss NSF	28.3	3.4	(0.5, 0.5)	28.0	3.7	0.54
Swiss NSF (Manski Bounds)	5.4	26.9	(0.75, 0.94)	5.4	26.9	0.92
NeurIPS LOO	3.4	16.5	(0.36, 0.94)	3.4	17.9	0.34
NeurIPS Gaussian	2.2	25.7	(0.16, 0.83)	4.0	27.5	0.20
NeurIPS Subjectivity	4.5	18.7	(0.14, 0.45)	1.6	10.4	0.78
ICLR LOO	11.1	32.4	(0.51, 0.88)	22.0	22.8	0.44
ICLR Gaussian	9.5	34.4	(0.45, 0.87)	21.2	27.7	0.4
ICLR Subjectivity	17.6	24.3	(0.50, 0.88)	16.2	25.2	0.63

Table 3.1: Comparison of marginal probabilities of acceptance by MERIT and Swiss NSF on each dataset. “Accept” = guaranteed to be selected ($p = 1$), while “Random” = entered into lottery ($0 < p < 1$).

3.1.7 Discussion

This subchapter introduced MERIT, a computationally efficient and principled lottery for top- k selection under uncertainty. By relying solely on interval information, rather than a fully specified generative model, MERIT respects the limits faced by real-world funders while providing a robust optimization justification for randomization. The case studies show that MERIT scales well to tens of thousands of candidates and can be applied to realistic peer-review datasets.

The framework is deliberately conservative. MERIT exploits only the ordering implied by quality intervals, which is appropriate when credible probabilistic models are unavailable but may be suboptimal in domains with well-calibrated predictive models. The current formulation also assumes that all selected proposals consume the same amount of budget. Many funding programs instead allow applicants to request different amounts of support, in which case the

funder must reason jointly about quality and cost. A natural extension is to allow partial funding by representing a request for c integral resources as c copies of the proposal's interval, but this raises incentive issues around truthful cost reporting and does not directly handle indivisible requests.

The model can also accommodate richer decision information. A funder may know, for example, that two proposals with overlapping intervals were evaluated by the same reviewer and that the reviewer clearly preferred one over the other. Adding such comparisons moves the problem from interval orders toward general partial orders, where the ex ante optimization problem is NP hard, so practical extensions would likely need heuristics that preserve the tractability of the interval-order case. Similarly, some funders may prefer a uniform lottery because equal probabilities are easier to explain to applicants and stakeholders. The optimization problem used in MERIT can be constrained to uniform sampling, producing an ex ante optimal uniform lottery subject to ex post validity; Supplemental Material reports simulations for this variant.

More general utility models and behavioral effects remain important for future deployments. The analysis here uses a 0-1 utility for selecting the top- k candidates, but funders may care about positional utility, cost-quality trade-offs, reviewer-time savings, or portfolio diversity. Randomization may also change how applicants and reviewers behave: applicants may alter effort or risk-taking, and reviewers may learn to act strategically around lottery rules. These questions connect the formal mechanism in this subchapter to the smoother lottery design studied next, where the focus shifts from interval uncertainty to the instability of selection probabilities under small score changes.

3.1.8 Deferred Proofs

This section gives proofs for the main structural results used in the MERIT subchapter.

Polynomial-Time Ex Ante Optimization

Proof of Lemma 3.1.3. Let v' be the smallest value found by Algorithm 3. We show that

$$v' = \min_{\sigma \in \Sigma_n} \sum_{i=1}^n p_i \mathbb{1}_{\{\sigma(i) \leq k\}}.$$

Equivalently, let K be the family of feasible top- k sets induced by rankings in Σ_n . The objective is $\min_{T \in K} \sum_{j \in T} p_j$.

Sort intervals by decreasing lower bound. For $i \in [k+1]$, let $K^{(i)}$ be the feasible top- k sets that include intervals $1, \dots, i-1$ and exclude interval i . These classes cover K . If interval i is excluded, then every additional member of the top- k set beyond $1, \dots, i-1$ must overlap interval i ; otherwise it would be strictly below i and could not be above i in an ex post valid ranking. Let

$$S_i = \{j \in (i, n] : \text{intervals } i \text{ and } j \text{ overlap}\}.$$

Every subset of S_i of size $k - (i - 1)$ is feasible together with $1, \dots, i - 1$, because all intervals in S_i overlap interval i and no lower-bound order prevents them from occupying the remaining top positions. Thus the worst set in $K^{(i)}$ is obtained by taking the $k - (i - 1)$ smallest p values in S_i . Algorithm 3 enumerates exactly these candidates over all i .

The returned cuts are therefore valid LP constraints, and the algorithm returns \emptyset exactly when no such constraint is violated. The running time is $O(n \max\{k, \log n\})$: sorting costs $O(n \log n)$, and the $k + 1$ scans can be implemented so that the relevant smallest p values are found in total polynomial time. \square

Proof of Theorem 3.1.2. Relax the equality $\sum_i p_i = k$ in LP (3.3) to $\sum_i p_i \leq k$. This does not change the optimum, since increasing any coordinate p_i up to 1 cannot decrease the worst-case objective. The feasible region then lies inside a box with side lengths polynomial in n : $0 \leq p_i \leq 1$ and $0 \leq v \leq k$.

The point $p_i = k/n$ and $v = 0$ is strictly feasible for the relaxed inequalities, so the feasible region is full-dimensional. By the standard ellipsoid method with a separation oracle, an ϵ -accurate optimum is found in time polynomial in the dimension, $\log(1/\epsilon)$, and the encoding length of the constraints, provided that separation is polynomial time. Lemma 3.1.3 supplies such an oracle. The constraints have integral coefficients of polynomial bit complexity, so the usual lower bound on the inscribed ball radius for rational polyhedra applies [76]. Hence the ellipsoid method solves the LP in time polynomial in n and $\log(1/\epsilon)$. \square

Ex Post Validity

Proof of Theorem 3.1.6. Fix intervals a, b with $\ell_a > u_b$, so a strictly dominates b . Algorithm 5 processes intervals in increasing order of upper bound. When it processes b , it transfers probability mass from b to any still-unsaturated interval above b , including a or intervals that dominate

a . After this step, either $p_b = 0$ or every interval that must precede b and can absorb mass has been saturated.

If $p_b = 0$, then b is never sampled and ex post validity is immediate for this pair. Otherwise, the mass transfer saturates a or all intervals above a in the relevant dominance chain; in particular the algorithm terminates with $p_a = 1$ for this pair. Subsequent iterations cannot increase p_b or decrease a saturated dominating coordinate in a way that violates the invariant.

It remains to show that the post-processing does not reduce ex ante objective value. For any feasible ranking $\sigma \in \Sigma_n$, if b is in the top k , then a is also in the top k . Moving probability mass from b to a therefore weakly increases $\sum_i p_i \mathbb{1}\{\sigma(i) \leq k\}$ for every feasible ranking in which b appears, and leaves the value unchanged otherwise. The worst-case value is thus not decreased. Applying the operation repeatedly preserves ex ante optimality and ensures that every dominating pair satisfies $p_a = 1$ or $p_b = 0$, which makes any sampling scheme with the final marginals ex post valid. \square

3.2 Smooth Lotteries for Stable Selection

The preceding subchapter used randomization to handle unresolved uncertainty about the ordering of candidates. This subchapter studies a different but complementary failure mode: even when scores are taken at face value, threshold-based lotteries can react discontinuously to small score changes. We therefore ask how to design a lottery whose marginal selection probabilities vary smoothly with the evaluations.

This subchapter is based on joint work with Giulia Fanti and Nihar B. Shah [72].

3.2.1 Motivation

As discussed in the chapter introduction, partial lotteries are increasingly used in scientific funding and related selection settings. A key practical motivation for randomization is to reduce deliberation over small quality differences:

[Randomization] reduces the time needed for an assessment or moderation panel to discuss applications. Panellists are less likely to spend time trying to arbitrarily differentiate between applications receiving similar quality-based scores that fall around the funding line, because they can “agree to disagree” by simply leaving it to chance [208].

Existing thresholded lottery designs sort proposals into groups: proposals above a high-quality threshold are funded automatically, proposals below a low-quality threshold are rejected, and proposals in the middle are entered into a uniform lottery with equal selection probabilities. This three-tier design is inherently unstable. As illustrated in Figure 3.6, a small increase in one review can move the highlighted candidate from rejection into the lottery group, causing a large jump in selection probability. More generally, any candidate near the boundary between reject, lottery, and accept groups may experience a discontinuous change in selection probability after an arbitrarily small score perturbation. In fact, Goldberg et al. [69] show that existing lottery designs can exhibit “maximum instability,” where an arbitrarily small change to a single review can shift the mechanism from deterministic selection to uniform random selection over all applicants. The empirical results in this subchapter show that this instability is not only theoretical. In

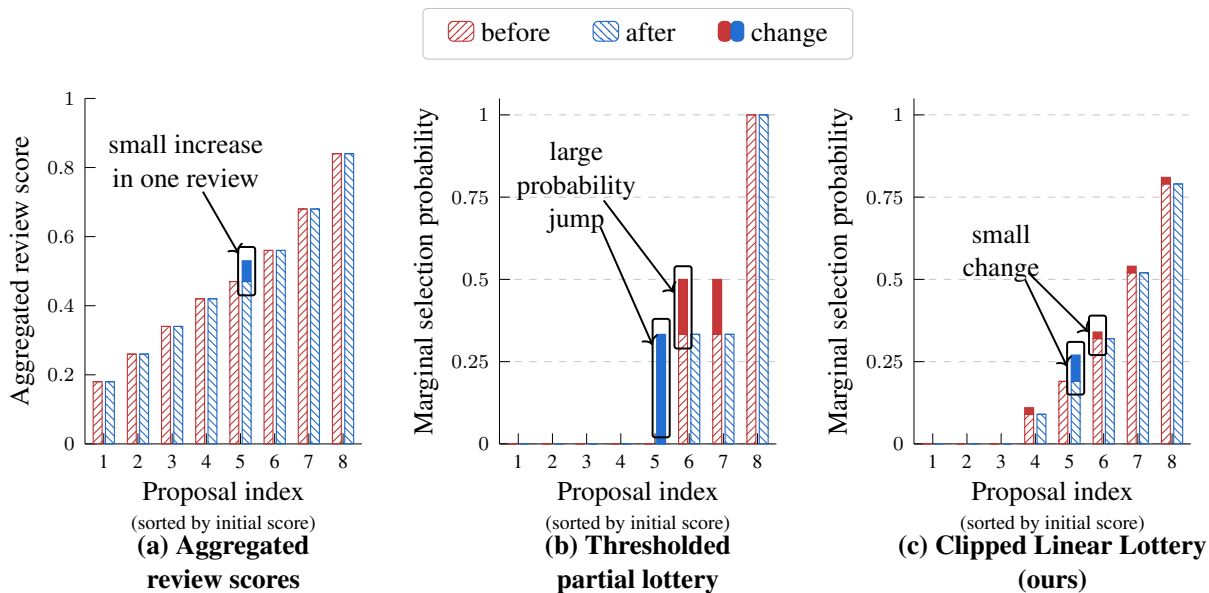


Figure 3.6: **Clipped Linear Lottery responds smoothly to score perturbations.** Marginal selection probabilities before and after a one-point increase in proposal 5’s review score, with 2 awards for 8 candidates. Under the thresholded partial lottery, proposal 8 is selected deterministically, while the remaining award is allocated with equal probability among proposals 6 and 7; the score increase lifts proposal 5, splitting the award three ways at $1/3$ each. A one-point change thus causes a discontinuous jump (proposal 5 from 0 to $1/3$) and abrupt drops for proposals 6 and 7. Under our Clipped Linear Lottery, the same perturbation induces only small, smoothly varying changes in every proposal’s selection probability.

Section 3.2.7, we find that in real peer review processes at ICLR, NeurIPS, and the Swiss NSF, changing a single review by one point can change an applicant’s selection probability by more than 0.3 under existing lottery designs.

Instability is problematic because it reintroduces the kind of arbitrariness from small changes to review scores that randomization is meant to prevent. If two candidates have nearly identical evaluations, a small change to one review should not sharply change chances of selection. We capture this requirement through *smoothness*: a selection rule is smooth if small changes in review scores can produce only bounded changes in selection probabilities.

In order to guarantee smoothness, we propose a *Clipped Linear Lottery*. Our Clipped Linear Lottery rule first aggregates reviews into a score for each candidate. It then assigns selection probabilities that increase linearly with this score, with probabilities clipped at zero and one when candidates are sufficiently weak or strong. As illustrated in Figure 3.6, a small review perturbation moves the highlighted candidate only slightly along the linear part of the probability curve. The result is a small change in selection probability, rather than a discontinuous jump across a decision boundary. The Clipped Linear Lottery has several practical benefits:

- *Stability.* We formally prove that small changes in individual reviews can only produce bounded changes in selection probabilities.
- *Quality awareness.* Higher-scoring candidates receive higher selection probabilities.

- *Transparency.* The rule is easy to explain, as selection probabilities increase linearly with estimated quality until they are clipped at zero or one.
- *Fairness.* The intuitive notion of Individual Fairness [54] requires that candidates with similar reviews receive similar selection probabilities. Bairaktari et al. [8] show that the clipped-linear form of the Clipped Linear Lottery is optimal for trading off Individual Fairness and selection quality. This gives an additional reason to adopt the Clipped Linear Lottery to ensure fairness in the treatment of similarly evaluated candidates.

These benefits require balancing smoothness against selection quality. A selection rule can be perfectly smooth by ignoring the reviews entirely, for example by sampling uniformly from all candidates. At the other extreme, a utility-maximizing rule deterministically selects the top- k candidates, but is not smooth. We formalize this tradeoff between smoothness and utility and show that the Clipped Linear Lottery is near-optimal.

Our **primary contributions** are:

1. **Formalizing smoothness for partial lotteries.** We define smooth selection rules using a Lipschitz condition on the mapping from review scores to marginal selection probabilities. This gives the interpretable guarantee that changing any single review score by one point shifts total selection probabilities by a bounded amount.
2. **Design and analysis of the Clipped Linear Lottery.** We introduce the Clipped Linear Lottery and prove that it is L -smooth. We also prove a worst-case regret bound matching a lower bound for any smooth selection rule within a factor of up to a factor of $(1 - k/n)$, where k/n is the acceptance rate.
3. **Comparison to alternative stability notions.** We show that standard Differential Privacy (DP) does not imply smoothness, while the generalization of Metric DP gives smoothness but at a factor- k regret penalty in the worst case. We then analyze top- k softmax sampling and prove that it achieves smooth marginal selection probabilities, but incurs higher regret than the Clipped Linear Lottery.
4. **Empirical validation on real peer review data.** On three real peer review datasets (ICLR 2025, NeurIPS 2024, Swiss NSF) we demonstrate that existing partial lottery designs are highly unstable under single-review perturbations and show that the Clipped Linear Lottery gives significantly better smoothness–utility tradeoffs than existing lottery designs in practice. We also demonstrate that the Clipped Linear Lottery dominates the softmax rule across smoothness levels and empirically confirm tightness of our theoretical bounds.

An open-source implementation of our algorithm and all experimental code are available at github.com/akgoldberg/smooth_lottery.

3.2.2 Background

Partial Lotteries for Scientific Funding and Peer Review. Recent works have designed algorithms for partial lotteries in scientific funding and peer review. MERIT [69] and the Swiss NSF Partial Lottery [79] estimate *intervals* for proposal quality, with overlapping intervals indicating uncertainty about the relative quality of proposals. These methods use intervals to decide when randomization is appropriate.

This subchapter targets a different design goal. Rather than explicitly modeling uncertainty about proposal quality, we require the mapping from review scores to marginal selection probabilities to be stable. Empirically, we show that existing interval-based lotteries are highly non-smooth in practice; see Section 3.2.7 for details. A key difference between this prior work and our formulation is what information each approach extracts from review scores. In particular, MERIT [69] emphasizes the use of ordinal information: intervals encode which proposals may or may not dominate others, making the approach natural when the numerical scale of review scores is not trusted or a funder wishes to avoid making assumptions about how reviewers interpret the scale. In contrast, our smoothness framework treats the review scale as meaningful enough to define small perturbations, and asks that these perturbations have only limited effects on marginal selection probabilities. In Section 3.2.6, we discuss how smoothness can be combined with interval-based validity constraints when a funder also wants to enforce them.

In practice, the choice depends on what concern is most important. If the funder wants the lottery to reflect uncertainty about which proposals are better, especially when numerical review scores are not fully trusted, then an interval-based method such as MERIT [69] is a natural fit. If the funder wants to limit how much any one review score can affect selection probabilities, then Clipped Linear Lottery is the more direct tool. These concerns can overlap, but the two mechanisms prioritize different failure modes.

Randomized Selection and the Clipped Linear Lottery. The Clipped Linear Lottery has appeared independently under different names in several contexts. In machine learning, Kong et al. [107] introduced it as the *RankMax* operator, an adaptive projection alternative to the softmax function, while Martins and Astudillo [141] proposed the closely related *sparsemax* (for the $k = 1$ case). Most relevant here, Bairaktari et al. [8] studied fair cohort selection and proved that the clipped-linear rule is exactly instance-optimal for maximizing linear utility subject to Individual Fairness constraints; we discuss this result in detail in Section 3.2.5. However, the Clipped Linear Lottery has not previously been proposed or analyzed as a mechanism for designing partial lotteries in evaluation and selection settings such as peer review, and it is not obvious *a priori* that it is a good mechanism for our problem of interest. We provide a novel smoothness analysis—formalizing a Lipschitz condition on the mapping from review scores to selection probabilities—together with near-matching upper and lower bounds on the worst-case regret of any smooth selection rule. These results establish that the Clipped Linear Lottery is near-optimal for a fundamentally different objective (the smoothness–regret tradeoff) than those considered in prior work, and that it achieves a more efficient smoothness–regret tradeoff than natural alternatives like the softmax.

Stable Selection. There is extensive literature on algorithmic stability notions for selection mechanisms, most prominently Differential Privacy (DP) [50, 52] and Individual Fairness (IF) [54]. We compare our smoothness definition and the Clipped Linear Lottery to these alternatives in Section 3.2.5, showing that standard DP does not imply smooth marginals, that Metric DP implies smooth marginals but at a factor- k regret penalty [7, 194], and that IF and smoothness are formally incomparable. As part of this comparison, we give a novel analysis of the Lipschitz constant of the top- k softmax mechanism [108, 144] (known as the Exponential Mechanism in

the DP literature), extending recent bounds for the $k = 1$ case [152] to the combinatorial setting of $k > 1$, which may be of independent theoretical interest.

Randomization in other aspects of peer review. There are a few other parts of the peer review process where explicit randomization is used in practice. Computer science conferences employ automated methods to assign reviewers to papers, and randomization [85] is used in this assignment to mitigate problems of fraud via collusion rings [128] and identity thefts [189]. The idea here is that no matter how a malicious reviewer and/or author games the system [81], there is a bounded probability of the target reviewer getting assigned target paper. Some computer science conferences use randomization to select which of the accepted papers will be presented as oral presentations [191, 209]. A consequence of interest is that randomization can allow for causal analysis of policies adopted in the review process [181].

3.2.3 Problem Formulation

Our methods apply to settings such as admissions, scientific peer review, and job screening, where decision makers estimate candidate quality and select the top candidates. For concreteness, throughout we will describe a *funder* evaluating *candidates*.

Randomized Selection Rule. The funder evaluates n candidates and seeks to select k candidates of highest quality. Each candidate receives r numeric reviews, with scores normalized to lie in the unit interval. The review data is therefore represented as a matrix $X \in [0, 1]^{n \times r}$, where $X_{i,j}$ denotes the score assigned to candidate $i \in [n]$ by reviewer $j \in [r]$.² For simplicity of exposition, we assume each candidate receives the same number of reviews; our formulation naturally extends to variable numbers of reviews (Supplemental Material).

The funder randomizes decisions by specifying a *randomized selection rule* $\mathcal{D} : [0, 1]^{n \times r} \rightarrow \Delta(\mathcal{K})$, where $\mathcal{K} = \{S \subseteq [n] : |S| = k\}$ is the set of all k -size subsets of candidates, and $\Delta(\mathcal{K})$ denotes the set of probability distributions over \mathcal{K} . Given review data X , the rule $\mathcal{D}(X)$ defines a distribution over size- k subsets of candidates. Since candidates experience decisions through individual acceptance probabilities, we focus on the induced *marginal selection probabilities* $p : [0, 1]^{n \times r} \rightarrow [0, 1]^n$, defined by $p_i(X) = \mathbb{P}_{S \sim \mathcal{D}(X)}[i \in S]$ for $i \in [n]$.

Smoothness. We formalize the requirement that the selection rule be stable to changes in review scores as a *smoothness* property: small changes in review scores should lead to small changes in candidates' chances of acceptance. Since these chances are determined by the marginal probabilities, we define smoothness as a Lipschitz condition on p :

Definition 3.2.1 (*L-smooth selection rule*). *For $L \geq 0$, the selection rule \mathcal{D} is L -smooth if its induced marginal probability function p satisfies*

$$\|p(X) - p(X')\|_1 \leq L \|X - X'\|_{1,1} \quad \text{for all } X, X' \in [0, 1]^{n \times r},$$

where $\|A\|_{1,1} = \sum_{i,j} |A_{i,j}|$.

²We use the standard notation $[\kappa]$ to denote the set $\{1, \dots, \kappa\}$ for any positive integer κ .

This condition ensures the stability of candidates' selection chances with respect to review scores: if a single review changes by δ , the total absolute change in selection probabilities is at most $L\delta$.

Utility. A selection rule should also select high-quality candidates. Hence, we introduce a utility function that depends on reviews. Given review data X , the funder has utility vector $u(X) \in \mathbb{R}^n$, where $u_i(X)$ represents the utility of selecting candidate i . The funder's utility when selecting a set of candidates is the sum of the selected candidates' utilities. Since \mathcal{D} induces marginal probabilities p , its expected utility is $p(X)^\top u(X)$. The *regret* of a selection rule is its utility loss relative to optimal deterministic selection, $R(\mathcal{D}; X) = \text{OPT}(X) - p(X)^\top u(X)$, where $\text{OPT}(X)$ is the utility obtained by selecting the k candidates with highest utility. Our goal is to design a L -smooth selection rule that minimizes the worst-case regret, giving the minimax design problem:

$$\begin{aligned} & \min_{\mathcal{D}: [0,1]^{n \times r} \rightarrow \Delta(\mathcal{K})} \max_{X \in [0,1]^{n \times r}} \{ \text{OPT}(X) - p(X)^\top u(X) \} \\ & \text{subject to} \quad \|p(X) - p(X')\|_1 \leq L \|X - X'\|_{1,1}, \quad \forall X, X' \in [0,1]^{n \times r}, \end{aligned}$$

where p denotes the marginal selection probabilities induced by \mathcal{D} .

Smooth Utility Function. We assume the funder evaluates candidates using a utility function that varies smoothly with the review scores. Specifically:

Definition 3.2.2 (D_u -Lipschitz utility function). *Utility function $u : [0,1]^{n \times r} \rightarrow \mathbb{R}^n$ is D_u -Lipschitz if for all review matrices $X, X' \in [0,1]^{n \times r}$, $\|u(X) - u(X')\|_1 \leq D_u \|X - X'\|_{1,1}$.*

For example, the mean score $u_i(X) = \frac{1}{r} \sum_{j=1}^r X_{i,j}$ is D_u -Lipschitz for $D_u = 1/r$. The median, min, and max score are all 1-Lipschitz. The Lipschitz utility function assumption is necessary. If the utility function were arbitrarily non-smooth, then a small change in review scores could produce a large change in utility, making it impossible for any smooth selection rule to achieve low worst-case regret with respect to this utility function.

3.2.4 Designing an Optimal Smooth Lottery

In Section 3.2.4, we introduce a simple, efficient, and interpretable *Clipped Linear Lottery* and analyze its smoothness and regret; in Section 3.2.4, we prove a lower bound establishing optimality of the Clipped Linear Lottery up to a factor of $\frac{1}{1-k/n}$. Since smoothness and regret depend only on the marginal probabilities, we design mappings from review data to selection probabilities; any sampling scheme implementing these marginals can be used for the final selection.

Clipped Linear Lottery

We now introduce the *Clipped Linear Lottery*, a clipped-linear partial lottery described in Algorithm 8. The selection rule assigns marginal selection probabilities by scaling candidate utilities, shifting them by a common intercept to satisfy the budget, and clipping the result to $[0, 1]$. This partitions candidates into three tiers: an auto-accept pool ($p_i = 1$), an auto-reject pool ($p_i = 0$),

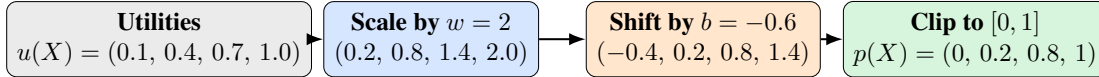


Figure 3.7: A concrete example of the Clipped Linear Lottery with $n = 4$ candidates and budget $k = 2$, following Algorithm 8. Utilities are scaled by $w = 2$, then shifted by an intercept $b = -0.6$ chosen so that the clipped probabilities sum to the budget. Clipping the shifted values to $[0, 1]$ gives marginals $p(X) = (0, 0.2, 0.8, 1)$, which sum to $k = 2$. Thus, the lowest-utility candidate is auto-rejected, the highest is auto-accepted, and the two middle candidates form the active lottery pool.

and a lottery pool where selection probabilities scale linearly with estimated quality. The slope w controls the smoothness–utility trade-off. Smaller L gives a flatter slope, expanding the lottery pool toward uniform selection. Larger L gives a steeper slope, shrinking the lottery pool toward deterministic top- k selection. Figure 3.7 illustrates the Clipped Linear Lottery on a concrete example.

Algorithm 8 Clipped Linear Lottery [8, 107]

Require: Reviews X , utility u with Lipschitz constant D_u , smoothness parameter $L > 0$, budget k
 Compute utilities $u_i \leftarrow u_i(X)$ for all $i \in [n]$
 Set the slope $w \leftarrow L/(2D_u)$
 Scale utilities $z_i \leftarrow w \cdot u_i$
 Compute intercept $b \in \mathbb{R}$ satisfying budget constraint,

$$\sum_{i=1}^n \text{clip}_{[0,1]}(z_i + b) = k$$

where $\text{clip}_{[0,1]}(x) = \min\{1, \max\{0, x\}\}$, using a capped-simplex projection.

Set marginal probabilities

$$p_i(X) \leftarrow \text{clip}_{[0,1]}(z_i + b), \quad i \in [n]$$

Sample size- k set $S \subset [n]$ with marginal probabilities $p(X)$, e.g., using systematic sampling [136]
return Marginal probabilities $p(X)$ and selected set S

The Clipped Linear Lottery has several useful properties.

Uniqueness. For every utility vector and budget, the Clipped Linear Lottery produces a unique probability vector [222].

Efficient computation. There are two computationally nontrivial steps in Algorithm 8. First, we must compute an intercept b satisfying the budget constraint (line 4). This can be done efficiently using existing algorithms for projection onto the capped simplex [8, 222]; for completeness, an exact capped-simplex projection implementation is provided in the supplemental material. Second, the Clipped Linear Lottery specifies marginal selection probabilities rather than a unique joint distribution over selected sets. Given marginals p , any sampling scheme that

returns size- k subsets with these marginals can be used for the final selection (line 6). One simple option is classical systematic sampling [136].

Monotonicity in budget. For fixed utilities and smoothness parameter L , increasing the budget k can never decrease any candidate’s selection probability. This contrasts with existing lottery designs, which can violate this natural monotonicity property [69].

Monotonicity of the lottery pool. For fixed utilities and budget, the size of the active lottery pool $\{i : 0 < p_i < 1\}$ is non-increasing in the smoothness parameter L . Equivalently, enforcing a smoother lottery by decreasing L can only increase, or leave unchanged, the number of candidates subject to randomization. Thus, a funder that wants to cap the lottery-pool size, for example for political or institutional reasons, can sweep over L and choose the smoothest lottery that satisfies this constraint.

The most important formal proofs are collected at the end of the subchapter in Section 3.2.9.

Smoothness We first show that the Clipped Linear Lottery satisfies the target smoothness guarantee. The following theorem shows that setting $w = L/(2D_u)$, as in Algorithm 8, is sufficient to guarantee L -smoothness.

Theorem 3.2.3 (Smoothness of the Clipped Linear Lottery). *If u is D_u -Lipschitz, then the Clipped Linear Lottery in Algorithm 8 guarantees L -smoothness.*

This smoothness bound is tight within a factor of $1 - \frac{1}{n}$; that is, there are instances where the Clipped Linear Lottery is at least $(1 - \frac{1}{n})L$ -smooth. The proof is given in Section 3.2.9.

The proof intuition is as follows. If b were fixed, then changing the reviews by magnitude δ could change the scaled utilities by at most $wD_u\delta$, since u is D_u -Lipschitz and clipping is 1-Lipschitz. The remaining factor of 2 comes from enforcing the budget. The intercept b must shift so that $\sum_i p_i(X) = k$, and this common shift can change many candidates’ probabilities at once. In the worst case, this doubles the total ℓ_1 change in marginal probabilities, giving at most $2wD_u\delta$. Thus choosing $w = L/(2D_u)$ guarantees L -smoothness.

Regret We next bound the worst-case regret. Before doing so, we establish a geometric characterization of the Clipped Linear Lottery that plays a central role in the analysis.

Proposition 3.2.4 (Projection characterization [107]). *For every X , the Clipped Linear Lottery outputs the Euclidean projection of $w \cdot u(X)$ onto the capped simplex $\mathcal{C}_{n,k} = \{p \in [0, 1]^n : \|p\|_1 = k\}$. That is, $p(X) = \arg \min_{p \in \mathcal{C}_{n,k}} \frac{1}{2} \|p - w \cdot u(X)\|_2^2$.*

Because the Clipped Linear Lottery finds the valid probability distribution that is geometrically closest to the scaled utilities, we can use the properties of this projection to bound regret. The proof in Section 3.2.9 leverages this interpretation, algebraically rewriting the regret as a simple quadratic function that depends only on the chosen probabilities:

Theorem 3.2.5 (Regret of the Clipped Linear Lottery). *The worst-case regret of the Clipped Linear Lottery satisfies*

$$\max_X R(\mathcal{D}; X) \leq \frac{k(1 - \frac{k}{n}) D_u}{2L}.$$

Corollary 3.2.6 (Mean utility). *If utilities are given by the mean review score for each candidate, then the worst-case regret of the Clipped Linear Lottery satisfies*

$$\max_X R(\mathcal{D}; X) \leq \frac{k(1 - \frac{k}{n})}{2 L r}.$$

To interpret the bound, note that regret always lies in $[0, k]$ because utilities lie in $[0, 1]$. Uniform random selection among all candidates is perfectly smooth and has worst-case regret $k(1 - k/n)$. Theorem 3.2.5 bounds the regret of the Clipped Linear Lottery by this natural baseline scaled by $D_u/(2L)$: larger L permits a steeper rule and therefore yields a smaller regret bound. Corollary 3.2.6 follows immediately because the mean utility has Lipschitz constant $D_u = 1/r$. Intuitively, as the number of reviews per candidate increases, each individual review has less influence on the mean, so the worst-case regret bound decreases for any fixed smoothness parameter.

Regret Lower Bound

We next show that the Clipped Linear Lottery achieves nearly the lowest regret attainable by any smooth mechanism. For simplicity, we state the result for mean utility in the main text; however, the proof technique and near-optimality guarantee extend to many other utility functions, including the median, minimum, and maximum review score (Supplemental Material).

Theorem 3.2.7 (Regret lower bound for L -smooth selection rules with mean utility). *Let utilities be the mean review scores $u_i(X) = \frac{1}{r} \sum_{j=1}^r X_{i,j}$, and let \mathcal{D} be any L -smooth selection rule. Then, the worst-case regret of \mathcal{D} is at least*

$$\max_X R(\mathcal{D}; X) \geq \begin{cases} \frac{k(1 - \frac{k}{n})^2}{2 r L}, & \text{if } L \geq \frac{1}{r} \left(1 - \frac{k}{n}\right), \\ k \left(1 - \frac{k}{n} - \frac{L r}{2}\right), & \text{if } L < \frac{1}{r} \left(1 - \frac{k}{n}\right). \end{cases}$$

Theorem 3.2.7 shows that the Clipped Linear Lottery is near-optimal for the smoothness–regret tradeoff. In the regime $L \geq \frac{1}{r}(1 - \frac{k}{n})$, the lower bound differs from the regret upper bound of the Clipped Linear Lottery (Corollary 3.2.6) only by a factor of $1 - \frac{k}{n}$. Thus, no smooth selection rule can substantially improve over the Clipped Linear Lottery in worst-case regret. The regime $L \geq 1/r$ is especially natural because the mean review score itself is $(1/r)$ -Lipschitz, so this corresponds to requiring selection probabilities to be roughly as smooth as the utility function. In the opposite limit, as $L \rightarrow 0$, the lower bound approaches $k(1 - k/n)$, matching the regret of uniform random selection, whose probabilities are completely independent of the reviews.

The proof, given in Section 3.2.9, uses an indistinguishability argument. Start from a baseline instance in which all candidates receive identical zero scores. Since the mechanism must select k candidates in expectation, some candidates must receive relatively low selection probability. We then perturb the scores of these least-likely candidates by a small amount δ , making them the uniquely best candidates. Smoothness limits how much their selection probabilities can

	Definition	Implication	Regret
Smooth selection (Def. 3.2.1)	Marginal selection probabilities vary smoothly with review scores.	No DP or IF guarantee.	Clipped Linear Lottery achieves near-optimal regret.
Individual fairness (IF)	Candidates with similar utilities receive similar selection probabilities.	No smooth marginals guarantee.	Clipped Linear Lottery is optimal.
Differential privacy (ϵ -DP)	Joint distribution over accepted candidates is stable under one-entry changes to the review matrix.	Does not imply $\ell_{1,1}$ -smooth marginals; arbitrarily small score changes may still cause discontinuous marginal changes.	Not a direct certification of smoothness.
Metric DP	Joint distribution changes with $\exp(\epsilon \ X - X'\ _{1,1})$.	Implies smooth marginals with $L \leq \epsilon k$.	Regret worse by a factor of $k \log(n/k)$.

Table 3.2: Comparison of algorithmic stability definitions for selecting k of n candidates.

increase in response to this perturbation. As a result, any smooth rule must leave substantial probability mass on lower-quality candidates, incurring regret that scales with the strictness of the smoothness constraint.

Our argument extends to many row-wise utility functions beyond the mean, including the minimum, maximum, and median review score. Supplemental Material gives the general version and shows that the Clipped Linear Lottery remains near-optimal, up to the same factor of $1 - \frac{k}{n}$, for these utility functions as well.

3.2.5 Comparisons to Other Stability Notions

We compare our smooth selection constraint against three prominent alternatives: Individual Fairness (IF), Differential Privacy (DP), and softmax-based randomized selection. As summarized in Table 3.2, these perspectives lead to sharply different guarantees. IF and smooth marginals are formally incomparable, although the Clipped Linear Lottery is also instance-optimal for IF (Section 3.2.5). Standard DP does not imply $\ell_{1,1}$ -smooth marginals because its neighboring relation treats arbitrarily small and large single review changes the same way (Section 3.2.5). Metric DP fixes this discontinuity issue and implies smooth marginals, but using it to certify L -smoothness incurs a $k \log(n/k)$ -fold regret penalty compared to the Clipped Linear Lottery. Finally, top- k softmax achieves smooth marginals but with higher regret than the Clipped Linear Lottery (Section 3.2.5).

Individual Fairness

Individual Fairness (IF) [54] requires that candidates with similar utilities receive similar selection probabilities (formally: $|p_i(X) - p_j(X)| \leq \alpha |u_i(X) - u_j(X)|$ for all i, j on every X). IF and our smooth-marginals condition are formally incomparable: smoothness bounds how the *same* candidates' probabilities change across review matrices, whereas IF bounds how *different* candidates' probabilities differ on the same matrix; neither implies the other (Supplemental Material). Remarkably, prior work shows that the Clipped Linear Lottery (with $w = \alpha$) is the

exact, instance-optimal solution to the IF-constrained utility-maximization LP [8, Theorem 3]. Thus, the Clipped Linear Lottery is not only near-optimal for smoothness across datasets, but also instance-optimal for individual fairness on any fixed review matrix.

Differential Privacy

Perhaps the most widely studied notion of algorithmic stability is differential privacy (DP) [52]. DP ensures that an algorithm’s output does not reveal too much about any single individual in the dataset. In our setting, DP constrains how much the *joint* distribution over selected subsets can change when a single review score is perturbed.

Definition 3.2.8 (ε -Differential Privacy). *A selection rule $\mathcal{D} : [0, 1]^{n \times r} \rightarrow \Delta(\mathcal{K})$ satisfies ε -differential privacy (ε -DP) if for all review matrices $X, X' \in [0, 1]^{n \times r}$ differing in at most one entry, and every subset $S \in \mathcal{K}$,*

$$\Pr[\mathcal{D}(X) = S] \leq e^\varepsilon \Pr[\mathcal{D}(X') = S].$$

Our L -smoothness condition is a statement about *marginal* selection probabilities, and does not imply DP. A perfectly smooth selection rule can output uniform marginals $(k/n, \dots, k/n)$ on all inputs, but implement those marginals using different joint distributions with disjoint support, which would not satisfy DP.

Conversely, standard DP does not imply our $\ell_{1,1}$ -smoothness condition. The issue is that standard DP uses a discrete neighboring relation: any change to one review score is treated as one neighboring change, regardless of whether that score changes by an arbitrarily small amount or by a large amount. Thus, standard DP bounds the size of a change in the marginal probabilities, but this bound is constant and does not scale linearly with the magnitude of the change in review scores. Hence, in general, there need not exist a finite L such that the smoothness condition holds for arbitrarily small changes in a single entry of the review matrix X , as we show in the following proposition.

Proposition 3.2.9 (Standard DP does not imply smooth marginals). *For every $\varepsilon > 0$, there exists an ε -DP selection rule that is not L -smooth with respect to $\|\cdot\|_{1,1}$ for any finite L .*

We prove Proposition 3.2.9 by counterexample in Supplemental Material. The mechanism randomizes between two selected sets, $S_1 = \{1, \dots, k\}$ and $S_2 = \{k + 1, \dots, 2k\}$, by looking at only a single review score, $X_{1,1}$, thresholding it at $1/2$, and using randomized response on the resulting bit to decide whether to favor S_1 or S_2 . Since the probability ratio is bounded by e^ε , the rule is ε -DP under the standard one-entry replacement definition. But the marginals are discontinuous at the threshold: an arbitrarily small perturbation of $X_{1,1}$ can change which set is favored, producing a constant-size change in the marginal probabilities. Hence standard DP does not imply smooth marginals. To obtain a DP-style condition that rules out such discontinuities, the privacy definition must scale with the distance between inputs. We use metric differential privacy, which generalizes DP by replacing the discrete neighboring relation with an arbitrary metric on the input space [27].

Definition 3.2.10 (ε -Metric Differential Privacy). *A selection rule $\mathcal{D} : [0, 1]^{n \times r} \rightarrow \Delta(\mathcal{K})$ satisfies ε -metric differential privacy with respect to $\|\cdot\|_{1,1}$ if for all review matrices $X, X' \in [0, 1]^{n \times r}$ and every subset $S \in \mathcal{K}$,*

$$\Pr[\mathcal{D}(X) = S] \leq \exp(\varepsilon \|X - X'\|_{1,1}) \Pr[\mathcal{D}(X') = S].$$

Metric DP controls the joint output distribution using the same distance on review matrices as our smoothness definition. This gives a bound on the smoothness of marginals:

Proposition 3.2.11 (Metric DP implies smoothness of marginals). *If a selection rule \mathcal{D} satisfies ε -metric differential privacy with respect to $\|\cdot\|_{1,1}$, then its induced marginals p are L -smooth for some*

$$L \leq \varepsilon k.$$

This bound is tight as $\varepsilon\|X - X'\|_{1,1} \rightarrow 0$; see Supplemental Material.

Metric DP is not regret-optimal for enforcing smooth marginals. Metric DP gives one way to certify L -smooth marginals: by Proposition 3.2.11, it suffices to take $\varepsilon \leq L/k$. Since metric DP implies standard DP for one-entry changes, known lower bounds for pure-DP top- k selection imply error $\Omega(kD_u \log(n/k)/\varepsilon)$ [7, 194]. Setting $\varepsilon \approx L/k$ gives regret $\Omega(k^2 D_u \log(n/k)/L)$. This is larger than the lower bound for L -smooth rules in Theorem 3.2.7 by an additional factor $k \log(n/k)$. Thus, metric DP does not by itself navigate the smoothness–regret tradeoff well.

Softmax (Exponential Mechanism)

Section 3.2.5 shows that metric differential privacy does not, in itself, yield a good smoothness–utility tradeoff; however, Proposition 3.2.11 is a worst-case guarantee and does not rule out that specific metric-differentially private mechanisms perform better than the generic smoothness bound in Proposition 3.2.11. We therefore study a well-known private selection mechanism: the *Exponential Mechanism* [144].

Given utilities $u \in \mathbb{R}^n$ and temperature $\tau > 0$, the softmax distribution is

$$\sigma_\tau(u)_i = \exp(u_i/\tau) / \sum_j \exp(u_j/\tau).$$

By the *top- k softmax rule* we mean the procedure that samples k items *without replacement* by repeatedly sampling from the re-normalized softmax distribution over the remaining items. Equivalently, this is the Gumbel-top- k procedure obtained by adding i.i.d. Gumbel noise to utilities and selecting the k largest perturbed utilities [108]. The exponential mechanism naturally extends to metric differential privacy: if the utility function is D_u -Lipschitz with respect to a given metric, then sampling one item with probability proportional to $\exp(u_i(X)/\tau)$ satisfies $(2D_u/\tau)$ -metric DP [96].

Theorem 3.2.12 (Smoothness and Regret of the Top- k Softmax Rule). *Assume u is D_u -Lipschitz. The top- k softmax rule with temperature τ induces L -smooth marginal selection probabilities with $L \leq \frac{2D_u}{e\tau}$, and the worst-case regret satisfies $\max_X R(\mathcal{D}; X) \leq k\tau \log n$.*

Theorem 3.2.12 shows that top- k softmax, like the Clipped Linear Lottery, can be tuned to satisfy any desired smoothness level. In particular, setting $\tau = \frac{2D_u}{eL}$ guarantees L -smooth marginals and gives regret at most $\frac{2D_u k \log n}{eL}$. This smoothness guarantee is stronger than what follows from the generic differential privacy argument above, since it does not lose an additional factor of k . However, it is still worse than the Clipped Linear Lottery’s regret bound by a logarithmic factor in n : the Clipped Linear Lottery scales as $O(k/L)$, while top- k softmax scales

as $O(k \log(n)/L)$. Thus, the Clipped Linear Lottery provides a better worst-case smoothness–regret tradeoff. This comparison is based on our theoretical upper bound on worst-case regret of the softmax. A sharper smoothness analysis for softmax could in principle narrow the gap, but our empirical results in Section 3.2.7 find instances where the bound is tight to within 5%.

As an additional practical advantage, the Clipped Linear Lottery has *sparse* support: it deterministically funds the strongest proposals and rejects the weakest, restricting randomness to borderline cases. By contrast, the softmax rule has full support and assigns strictly positive inclusion probability to every candidate.

The regret bound follows by extending standard guarantees for the Exponential Mechanism in the differential privacy literature [50]. However, the smoothness bound requires a novel analysis of the marginal inclusion probabilities induced by the top- k softmax rule, given in Supplemental Material.

3.2.6 Instability of Existing Partial Lottery Designs

The most common lottery design used in practice partitions candidates into three tiers based on a score or rank statistic: auto-accept above a high threshold, auto-reject below a low threshold, and uniform randomization among candidates in the middle tier [79, 131, 200, 218]. As discussed in the Introduction and illustrated in Figure 3.6, these designs share a structural source of instability. Since tier membership is determined by sharp thresholds, an arbitrarily small perturbation to a single review can move a candidate across a tier boundary, changing their selection probability by a constant amount. They are therefore inherently non-smooth.

Recent lottery designs, including the Swiss NSF procedure [1, 79] and MERIT [69], also take *interval estimates of candidate quality* as input. For example, an interval might be formed from the minimum and maximum review scores assigned to a proposal. These intervals capture uncertainty about the relative quality of proposals: if two candidates’ intervals overlap, the funder treats their ordering as uncertain; if one candidate’s interval lies strictly above another’s, the funder treats the higher-scoring candidate as clearly dominating the lower-scoring one. This idea is formalized through an *ex post validity* constraint [69]: if candidate i ’s lower bound exceeds candidate j ’s upper bound, then j should not be selected unless i is selected as well. This guarantee limits randomization to candidates who are plausibly comparable, preventing outcomes in which a clearly dominated candidate is selected while the candidate who dominates them is not.

Can we combine ex post validity with smoothness? In general, the two desiderata may conflict. Suppose all candidates have near-zero-width, non-overlapping intervals. Then ex post validity forces deterministic rank-based selection: the top k candidates must be selected, and the remaining candidates must be rejected. But an arbitrarily small perturbation could move the k -th candidate below the $(k + 1)$ -st candidate, causing their selection probability to jump from 1 to 0. This directly conflicts with smoothness, which requires selection probabilities to change gradually under small review perturbations.

Supplemental Material gives two ways to reconcile the Clipped Linear Lottery with ex post validity, depending on which guarantee the funder wants to enforce exactly. If the funder prioritizes exact smoothness, then it can relax the dominance relation by using intervals wide enough relative to the smoothness scale. Under this condition, whenever one candidate clearly dominates another, the Clipped Linear Lottery already assigns probability 1 to the dominant candidate or

probability 0 to the dominated candidate, so ex post validity holds automatically. If the funder instead prioritizes exact ex post validity, then it can project the Clipped Linear Lottery marginals onto the polytope of ex post valid marginal probabilities and sample only from valid sets. This preserves interval dominance in every realized outcome, but may weaken the global smoothness guarantee. Thus, smoothness and ex post validity are not diametrically opposed design goals, but combining them requires making explicit which guarantee is enforced exactly and which is relaxed.

3.2.7 Empirical Smoothness–Regret Comparison

We complement our theory with experiments on real peer review datasets and synthetic data.

Experimental setup. We evaluate the Clipped Linear Lottery and top- k softmax at acceptance rates $k/n \in \{10\%, 33\%, 50\%\}$, and compare against existing partial lottery designs. We use ICLR 2025 [163] ($n = 3,710, r_{\min} = 3$, scores 1–10), NeurIPS 2024 [162] ($n = 4,034, r_{\min} = 3$, scores 1–10), Swiss NSF [79] ($n = 353, r_{\min} = 5$, scores 1–6), and synthetic Beta reviews ($n = 200, r = 5$), drawn i.i.d. from $\text{Beta}(\alpha, \alpha)$ on $[0, 1]$ and discretized to 10 levels. For ICLR and NeurIPS, we restrict to accepted papers and simulate allocating oral presentations, a randomized-selection setting recently used at multiple computer science conferences [191, 209]. Since an allocation rate of roughly 10% matches historical oral-presentation rates at NeurIPS and ICLR, we present $k/n = 10\%$ results in the main text and defer $k/n \in \{33\%, 50\%\}$ results to Supplemental Material.

Smoothness of Existing Lotteries

We first test whether existing partial lottery designs are smooth in practice. We evaluate two interval-based mechanisms, MERIT [69] and the Swiss NSF procedure [79]. For each mechanism and dataset, we search over all single-review one-tick perturbations and identify the perturbation that maximizes the ℓ_1 change in marginal selection probabilities. We report the resulting empirical local smoothness, defined as $\|\Delta p\|_1 / \|\Delta X\|_{1,1}$.

Both mechanisms take point estimates and intervals of utility as input. For all datasets, we use the mean review score as the utility and construct intervals using “leave-one-out intervals” [69], computing the range of possible mean scores obtained by leaving out one reviewer at a time.

Figure 3.8 compares existing partial lottery designs with the Clipped Linear Lottery. Existing mechanisms can be highly unstable under small perturbations. For example, on both NeurIPS and ICLR, a single one-point review change can change a paper’s selection probability by more than 0.3 under either MERIT or the Swiss NSF mechanism; full results appear in Supplemental Material. By contrast, the Clipped Linear Lottery achieves much better smoothness at the same level of regret.

Regret–Smoothness Tradeoff

We next measure regret as a function of the smoothness parameter L . For each dataset and acceptance rate, we compute the regret of the Clipped Linear Lottery and top- k softmax over a

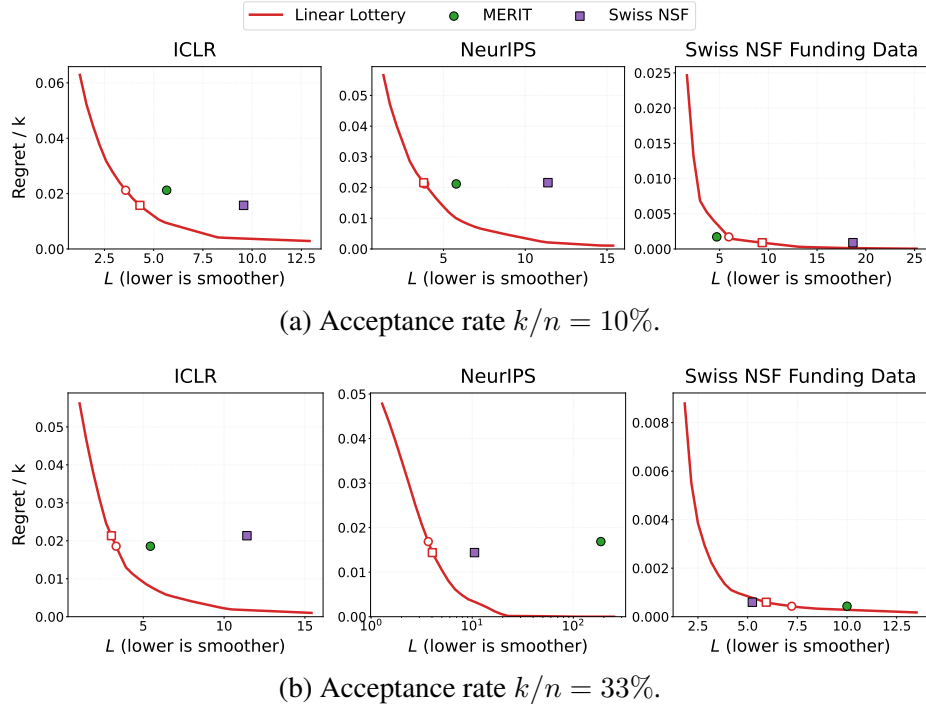


Figure 3.8: Regret–smoothness tradeoff for existing partial lottery mechanisms. Down and to the left indicates a better tradeoff. Points show MERIT and Swiss NSF at their empirical local smoothness under the worst one-review perturbation. Hollow red markers show worst-case smoothness guarantee of the Clipped Linear Lottery at the same regret level.

grid of target smoothness values. The Clipped Linear Lottery’s regret is computed exactly, while softmax regret is estimated using Monte Carlo sampling with 10,000 samples per estimate.

Figure 3.9 shows the regret–smoothness tradeoff at acceptance rate $k/n = 10\%$. We report regret normalized by the budget k ; under this normalization, the Clipped Linear Lottery bound depends on the acceptance rate only through the factor $(1 - k/n)$, so the curves have similar shape at other acceptance rates. Supplemental Material shows the same qualitative pattern for $k/n \in \{33\%, 50\%\}$. As predicted by theory, the Clipped Linear Lottery consistently achieves lower normalized regret than top- k softmax across datasets and smoothness levels.

Tightness of Smoothness Guarantees

Finally, we test whether our smoothness upper bounds are close to tight. For both the Clipped Linear Lottery and top- k softmax, we construct near-worst-case utility profiles with $u_i = 1$ for $i < k$, $u_k = B \in [0, 1]$, and $u_i = 0$ for $i > k$. We then perturb u_k to $u_k \pm \varepsilon$ and grid-search over B , ε , and the perturbation direction to maximize the empirical ratio $\|\Delta p\|_1 / \|\Delta X\|_{1,1}$.

Figure 3.10 compares the worst-case empirical smoothness to the target smoothness L . For the Clipped Linear Lottery, the empirical lower bound is within 99% of the theoretical upper bound at $n = 1000$, confirming that Theorem 3.2.3 is essentially tight. For top- k softmax, the empirical worst-case coefficient is within 95% of the bound from Theorem 3.2.12, showing that

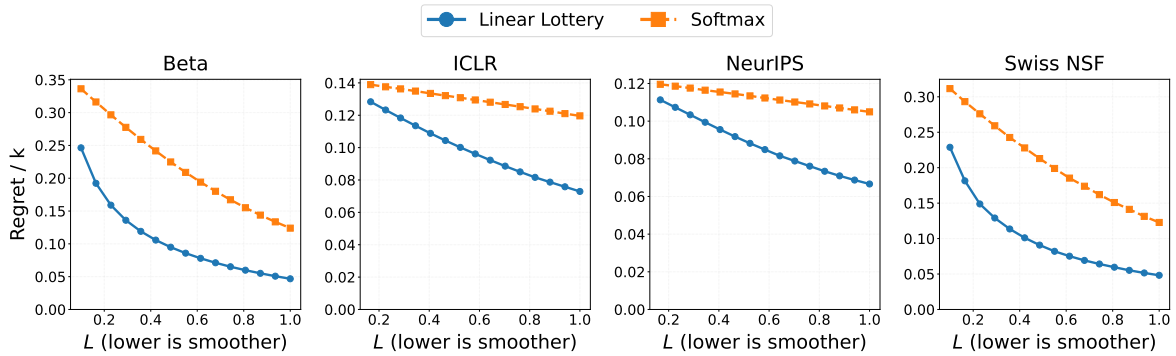


Figure 3.9: Regret vs. smoothness L at acceptance rate 10%. Down and to the left indicates a better tradeoff. The Clipped Linear Lottery achieves lower regret than softmax at every smoothness level across all datasets.

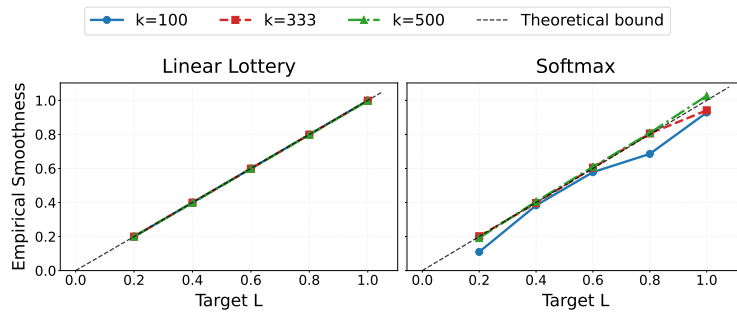


Figure 3.10: Worst-case empirical smoothness vs. target global smoothness L for $n = 1000$ and $k \in \{100, 333, 500\}$. The dashed line shows the theoretical upper bound. The bounds from Theorems 3.2.3 and 3.2.12 are essentially tight for both mechanisms.

our softmax analysis is also close to tight.

3.2.8 Discussion

This subchapter proposed smoothness as a design principle for partial lotteries: small changes in review scores should induce only small changes in selection probabilities. To satisfy this property, we introduced the Clipped Linear Lottery, analyzed its smoothness and regret, and showed that it is near-optimal for the resulting tradeoff up to a factor of $1 - \frac{k}{n}$. The comparison to Individual Fairness, Differential Privacy, and top- k softmax sampling shows that smoothness captures a distinct stability requirement. Empirically, the Clipped Linear Lottery achieves lower regret than softmax at matched smoothness on real peer-review datasets, while existing partial lottery designs can be highly unstable under small review perturbations.

The analysis relies on an exact, globally Lipschitz utility function, whereas practical utility estimates may be noisy, misspecified, learned from data, or only locally smooth. Extending the theory to these settings would make smooth lotteries more directly applicable to deployed systems. One possible route is to adapt ideas from differential privacy, where noise can sometimes be calibrated to local or smooth sensitivity rather than worst-case global sensitivity. An analo-

gous instance-adaptive lottery could preserve smoothness where it matters while reducing regret on benign instances.

The remaining questions are not only theoretical. Smooth lotteries are motivated by the institutional cost of sharp thresholds, but the analysis here does not measure how smoother mechanisms affect reviewer behavior, applicant trust, or deliberation in deployed settings. Future field studies could test whether smooth lotteries reduce fruitless discussion near decision boundaries and whether reviewers and applicants perceive them as more legitimate than threshold-based partial lotteries. Together with the interval-based rule in the previous subchapter, this provides two complementary ways to make randomized selection more principled: one addresses uncertainty about candidate ordering, and the other controls sensitivity to small score changes.

3.2.9 Deferred Proofs

This section gives the main proofs for the smooth lottery guarantees.

Smoothness of the Clipped Linear Lottery

Proof of Theorem 3.2.3. Let $u = u(X)$ and $u' = u(X')$, and let b, b' be the intercepts chosen by Algorithm 8. For each coordinate,

$$p_i = \text{clip}_{[0,1]}(wu_i + b).$$

By the triangle inequality,

$$|p_i - p'_i| \leq |\text{clip}(wu_i + b) - \text{clip}(wu'_i + b)| + |\text{clip}(wu'_i + b) - \text{clip}(wu'_i + b')|.$$

The first term is at most $w|u_i - u'_i|$ because clipping is 1-Lipschitz, so its sum is at most $w\|u - u'\|_1$.

For the second term, define $h_i(\beta) = \text{clip}_{[0,1]}(wu'_i + \beta)$. Since each h_i is monotone in β , the differences $h_i(b) - h_i(b')$ have a common sign. Therefore their absolute values sum to the absolute difference of their sums. Using the budget constraints $\sum_i p_i = \sum_i p'_i = k$, this second sum is also bounded by $w\|u - u'\|_1$. Hence

$$\|p - p'\|_1 \leq 2w\|u - u'\|_1.$$

Since u is D_u -Lipschitz and $w = L/(2D_u)$, the rule is L -smooth. \square

Regret of the Clipped Linear Lottery

Proof of Theorem 3.2.5. Let $u = u(X)$ and let p be the marginal vector produced by Algorithm 8. By Proposition 3.2.4, p is the Euclidean projection of wu onto the capped simplex $\mathcal{C}_{n,k} = \{q \in [0, 1]^n : \sum_i q_i = k\}$. Projection optimality gives

$$(p - wu)^\top (q - p) \geq 0 \quad \forall q \in \mathcal{C}_{n,k}.$$

Let q^* be the indicator vector of the top- k candidates by utility. Rearranging,

$$w u^\top (q^* - p) \leq p^\top (q^* - p).$$

The left side is w times the regret. If T is the top- k set, then

$$p^\top (q^* - p) = \sum_{i \in T} p_i - \sum_{i=1}^n p_i^2.$$

Maximizing this concave quadratic over the capped simplex gives value at most $k(1 - k/n)/4$. Thus

$$\mathbf{R}(\mathcal{D}; X) \leq \frac{k(1 - k/n)}{4w} = \frac{k(1 - k/n)D_u}{2L}.$$

\square

Regret Lower Bound

Proof of Theorem 3.2.7. Let X^0 be the all-zero matrix and $p^0 = p(X^0)$. Let T be the k candidates with smallest probabilities under p^0 . Since $\sum_i p_i^0 = k$, $\sum_{i \in T} p_i^0 \leq k^2/n$.

For $\delta \in [0, 1]$, define $X^{(T)}$ by setting all review scores of candidates in T to δ and leaving all other scores at zero. Then $\|X^{(T)} - X^0\|_{1,1} = kr\delta$. If $p = p(X^{(T)})$, smoothness implies

$$\|p - p^0\|_1 \leq Lkr\delta.$$

Because both marginal vectors sum to k , the total probability mass on T can increase by at most half the ℓ_1 change, so

$$\sum_{i \in T} p_i \leq \frac{k^2}{n} + \frac{Lkr\delta}{2}.$$

On $X^{(T)}$, the optimal utility is $k\delta$, while the rule obtains $\delta \sum_{i \in T} p_i$. Hence

$$\mathbf{R}(\mathcal{D}; X^{(T)}) \geq k\delta \left(1 - \frac{k}{n} - \frac{Lr\delta}{2}\right).$$

Optimizing this quadratic over $\delta \in [0, 1]$ gives the stated lower bound. □

Chapter 4

Privacy-Preserving Release of Evaluation Data

Transparent evaluation systems would benefit from more shared data: review discussions, assignment graphs, benchmark results, and other artifacts can help researchers diagnose bias, fraud, and instability. The difficulty is that these data often encode sensitive information about reviewers, authors, users, or fraud victims. This chapter studies two privacy problems that arise when data are released from evaluation systems.

The chapter takes a deliberately operational view of privacy. Rather than starting from a generic privacy definition and applying it unchanged, both subchapters begin by identifying the concrete leakage channel created by a plausible release. In the first setting, timing information can link actions made under different pseudonyms. In the second, benchmark scores can let an adversary encode private graph queries into submitted algorithms. In both cases, privacy risk comes not merely from the contents of the released data, but from the interaction between a release mechanism, an adversary's auxiliary information, and the constraints of the platform.

The first subchapter considers timestamped activity in pseudonymous forums, motivated by peer review comments and extended to Wikipedia and Bitcoin. It shows that batching tasks can compromise anonymity and develops delay mechanisms under a one-sided privacy definition tailored to the fact that fake events and indefinite withholding are not acceptable in these systems. The second subchapter considers private benchmarking of fraud detectors on graph-structured data, motivated in part by collusion detection in peer review. It shows that direct release of benchmark performance can leak membership or structural information, then evaluates differentially private benchmarking methods on graph data.

In both settings, the core theme is the same: useful release is possible only after identifying the exact leakage channel and designing a privacy definition that fits the operational constraints of the system. The results also show why privacy-preserving data release remains difficult in evaluation systems. The most useful data are often relational, temporal, or generated by strategic participants, and these features make simple anonymization or black-box differential privacy inadequate.

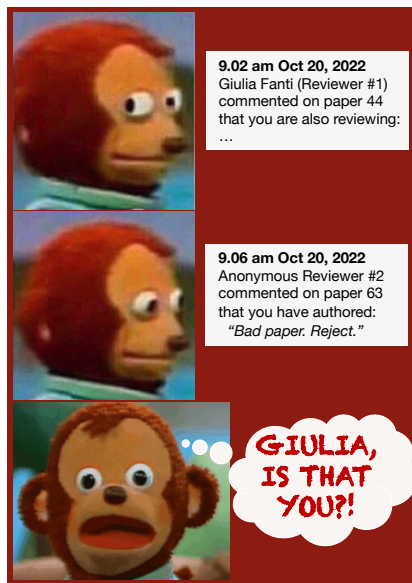


Figure 4.1: Cartoon illustration of reviewer de-anonymization due to batching.

4.1 Timing Privacy for Batched Activity

This subchapter begins the privacy part of the thesis by studying a leakage channel that arises before any substantive content is released: timing. In peer-review forums and other pseudonymous systems, users often complete related actions in batches, and the timing of those actions can link identities across contexts. The goal is to formalize this privacy risk and design delay mechanisms that reduce linkage without requiring fake activity or indefinite withholding.

This subchapter is based on joint work with Giulia Fanti and Nihar B. Shah [67].

4.1.1 Motivation

In a number of applications where anonymity is critical, users act under pseudonyms to preserve their privacy. For instance, in scientific peer review using online forums like OpenReview.net, reviewers make comments on papers that are publicly viewable. Reviewers (and meta-reviewers) who have been assigned multiple papers operate under different pseudonyms across their papers to remain anonymous. Other examples of publicly visible tasks where users operate under pseudonyms include Wikipedia editing and cryptocurrency transactions.

In many settings, it is common for users to engage in *batching* — the completion of several similar tasks at the same time. Batching occurs both due to natural bursts in activity (e.g., a person visits a website and makes many comments at once) or as a productivity strategy used to streamline work. Indeed, both academic studies [17, 113, 140] and popular media [114, 150, 151] recommend performing tasks like responding to emails in batches in order to improve efficiency and reduce work-related stress.

In peer-review forums such as computer science conferences, reviewers and meta-reviewers are often assigned multiple papers. We find empirically that reviewers and meta-reviewers are highly likely to batch their comments and/or reviews. Specifically, we analyze data from a top Computer Science conference¹ with thousands of papers, reviewers, and discussion com-

¹Name redacted for privacy.

ments. We find that when reviewers and meta-reviewers comment on multiple papers, they have a 30.10% chance of batching their comments within 5 minutes of one other. In comparison, any randomly chosen pair of reviewers and meta-reviewers had only a 0.66% chance of making comments on different papers within 5 minutes of each other.

While batching is normal human behavior, it introduces a risk of deanonymization in peer-review settings.² For example, in many open peer-review settings, comments are publicly posted. Furthermore, many conferences have policies that (meta-)reviewers for any paper know the identities of other (meta-)reviewers on that paper. Now, when a (meta-)reviewer batches their comments, an author may observe that two comments are generated at nearly the same time on their own paper and on another paper. The author can then link the identity of this anonymous (meta-)reviewer on their own paper to a (meta-)reviewer on the other paper. If the author knows the identity of the (meta-)reviewers on the other paper—for instance, if the author is the meta-reviewer or another reviewer for that paper—this can uncover the identity of the (meta-)reviewer of their own paper. See Figure 4.1 for a cartoon illustration.

A back-of-the-envelope calculation based on our aforementioned measurements in peer review suggests that if an author has a uniform prior over 10 possible (meta-)reviewers of their paper, then after observing a comment posted on their own paper within 5 minutes of another comment from one of these (meta-)reviewers on another paper, their posterior probability that this (meta-)reviewer made the comment increases to $\frac{0.301}{0.301+9(0.0066)} = 83.51\%$ as compared to the prior of 10%. Thus, the linking of (meta-)reviewers across papers using batched comments can undermine the anonymity of the peer review process.

Similar privacy risks due to batching arise in many systems where users generate publicly logged events under pseudonyms. For instance:

- *Inferring the identity of editors on Wikipedia articles.* Wikipedia provides public edit histories of articles. While edit history is public, Wikipedia users are known to maintain their anonymity for a variety of important reasons. For instance, one study of Wikipedia editors who use the anonymity network Tor found that editors are concerned about their privacy due to risks like “*threats of surveillance, violence, harassment, opportunity loss, reputation loss, and fear for loved ones.*” [63] These risks are especially acute for already marginalized groups like women and ethnic minorities. Thus, the study concludes that “*open collaboration communities must go beyond attracting participants, to develop social and technical arrangements that support contributors’ needs for privacy.*”

In order to address these privacy concerns, Wikipedia’s terms of service explicitly allow for the use of a pseudonymous alternate account: “*A person editing an article that is highly controversial within their family, social or professional circle, and whose Wikipedia identity is known within that circle, or traceable to their real-world identity, may wish to use an alternative account to avoid real-world consequences from their editing or other Wikipedia actions in that area.*” [228] However, as in the peer review example, batched

²This outcome is bad for a review system that needs a lot of interaction with the authors, but not for conferences where this is not expected nor allowed, like AAAI and IJCAI. The conference we analyzed was not on OpenReview.net but on a different conference management platform that does not make discussions public and has only a single-shot interaction between reviewers and authors (via a “rebuttal”). It is of interest to see an analogous analysis on conferences on OpenReview.net, but we do not have access to such data.

timing of article revisions can enable linkage of the second account to a known primary account. In practice, the batching of edits is ubiquitous on Wikipedia; our analysis of publicly logged Wikipedia article revisions shows that over 50% of all edits are made within 5 minutes of an edit from the same user on a different article. This common editing behavior may undermine the privacy of users employing a second account to preserve their anonymity.

- *Clustering crypto-currency transactions on a public blockchain.* In cryptocurrencies like Bitcoin, users' transaction histories are recorded on a public blockchain where a person can send or receive currency to an associated public key, which acts as a pseudonym. Users can have multiple addresses, each containing its own funds and identified by a different public key [15]. A transaction can (and often does) draw funds from multiple input addresses, particularly if no single address contains sufficient funds for a given transaction [14]. However, a common heuristic used in practice is to link multiple input addresses to a single transaction to the same user [16, 145]. Hence, users who wish to preserve their privacy can separate inputs from different addresses into different transactions to obfuscate the linkage between transactions from the same person [16].³ However, if a user batches these transactions in time across addresses, an adversary may use this timing (along with other signals) to still link together their multiple addresses. Linking pseudonyms together is a common first step in a full deanonymization attack. For instance, attacks on Bitcoin transactions begin by leveraging a user's "idioms of use" to cluster together addresses likely belonging to the same person [145]. The attacker then leverages a single known link to a real-world identity to de-anonymize the entire cluster.

These scenarios motivate the need for defenses against timing-based linkage attacks that exploit the batching of tasks by people. There is already extensive literature on privacy-preserving data release in various settings. However, there are a number of strict constraints in our setting that prevent these methods from being applicable. A common approach to preserving privacy is to introduce fake events to obscure patterns among the real events. However, in all three applications — peer review, Wikipedia, and cryptocurrency — generating fake events is highly undesirable or impractical, and withholding events indefinitely is also not possible. In the setting of commenting in peer review, it is undesirable to generate fake comments, as this would require giving made-up feedback to paper authors. Similarly, in Wikipedia, adding fake edits to articles can undermine the quality and legitimacy of the content. For cryptocurrencies, introducing dummy transactions would introduce additional financial cost, causing undesirable overhead. Furthermore, transactions include the amount of currency sent, so dummy transactions would require a sender to transfer actual funds just to preserve privacy. Instead, our approach is to design *delay mechanisms* that introduce random delays to the time at which events are posted on the platform (without the use of any dummy data) to preserve privacy. Thus, the mechanism will trade off privacy for additional delay in the system.

Contributions. This subchapter introduces the problem of anonymity compromise due to the batching of tasks in pseudonymous forums and then proposes defenses. The primary contributions are:

³There exist other cryptographic solutions (e.g., CoinJoin) that leak more information in exchange for cost benefits compared to generating multiple transactions [142].

- We identify the problem of deanonymization risk due to the batching of tasks by users of pseudonymous online forums. By analyzing data from an actual peer-review conference, we demonstrate that a simple attack using the timing of comments on an online forum can link anonymous (meta)-reviewer identities, increasing their certainty about a specific (meta)-reviewer’s identity to 83% from a prior of 10%. In analysis of Wikipedia article revisions, we show that batching of tasks on Wikipedia makes it possible to link editors across articles with an accuracy of 85% based only on the timing of their revisions.
- We formulate the problem of trading off privacy and delay in pseudonymous forums where users engage in batching. We show that standard notions of differential privacy (DP) [53] cannot be satisfied in our problem setting without introducing fake events or withholding events indefinitely. Therefore, we consider a “one-sided” relaxation of traditional DP [109]. Our formulation aims to prevent an adversary from inferring when batching happened, but allows an adversary to learn that batching did not happen.
- We propose a general framework for designing mechanisms that guarantee one-sided DP by adding independent random delay to batched and unbatched comments. We show that we can instantiate this framework with a number of different distributions and guarantee privacy. Notably, it is possible to guarantee privacy with non-negative versions of typical distributions used for differential privacy like the Laplace distribution and the Staircase distribution. It is also possible to guarantee privacy at any setting of the privacy parameters by adding delay drawn from a uniform distribution with inflated probability mass at 0, which we call the Zero-Inflated Uniform Mechanism.
- We establish the optimality of our Zero-Inflated Uniform Mechanism among mechanisms that add independent noise to each comment. In particular, we give a full characterization of the Pareto frontier of the expected delay added to batched and unbatched events by any mechanism that adds independent non-negative noise to comments, at any setting of privacy parameters, and show that our proposed mechanism achieves this frontier. This result may be of independent interest. While the uniform distribution is not typically used in the design of two-sided DP algorithms, our results show that for one-sided DP when only non-negative noise can be added (as is the case for streaming timing data) the Zero-Inflated Uniform Mechanism can optimally trade off privacy for utility.
- We conduct a series of experiments simulating linkage attacks using batched timing of tasks on Wikipedia article revision data and Bitcoin transaction data. These experiments reveal the applicability of our methods in preserving privacy in practice without exceedingly large delays.

Code for the experiments is available online at <https://github.com/akgoldberg/batching-privacy>.

4.1.2 Background

There is a substantial body of work on anonymity when sending packets over a network. However, as we discuss below, the techniques developed therein are inapplicable to our setting. Specifically, prior work has described deanonymization attacks which leverage correlated timing of packet arrivals. The work gives various defenses against such attacks [83, 93, 121, 190, 204,

210]. Anonymous networking seeks to prevent an adversary from inferring the sender and recipient of a given message. Packets are routed through a sequence of “mix nodes” to obscure the path taken. The highly correlated arrival times of packets on the first mix node and the last mix node in one path can enable inferences that a specific sender and recipient are communicating with one another. Prior work [121, 190] demonstrates the practical viability of deanonymization attacks that take advantage of batching in anonymous networks.

The defenses proposed in these papers rely on the introduction of *dummy packets* or “cover traffic” to a network, obscuring any instance of batching amidst many instances of spurious batching. In contrast, a critical constraint in the settings considered here is the *infeasibility of generating fake data* as a means of preserving privacy. Therefore, this subchapter considers mechanisms that delay batched arrivals in order to preserve anonymity, trading off delay for privacy, without introducing any synthetic data.

This subchapter defines privacy based on a “one-sided” relaxation of the popular notion of differential privacy [53]. The definition of one-sided DP was introduced in the paper [109] in a setting where contributors of individual data-points to a database have different privacy constraints and hence data-points can be classified as “sensitive” and “non-sensitive.” Here, this classification of sensitive and non-sensitive data-points is applied to batched and unbatched events. Interestingly, while the paper [109] shows that one-sided DP can improve utility compared to standard two-sided DP, we find that in our problem setting, one-sided DP admits useful privacy-preserving algorithms where two-sided DP does not admit any useful algorithms at all. We cannot readily apply algorithms from the paper [109] due to the constraint that we publish all data. Therefore, while they develop mechanisms that release a subset of non-sensitive data with no noise addition, while withholding all sensitive data entirely, we consider mechanisms that add noise to both sensitive and non-sensitive data-points and release all data-points.

Geng and Viswanath [66] address the question of optimal distributions for noise addition in standard differential privacy. They show that in order to minimize the magnitude of noise added to a query with known sensitivity, noise should be drawn from a “staircase” distribution, which has a probability density function that is roughly a piece-wise constant approximation of the Laplace distribution. The result in this subchapter can be seen as an analogous result in the one-sided DP regime. Specifically, we prove that for the one-sided relaxation of differential privacy, adding staircase noise is no longer optimal, but rather adding uniform noise with a possibly inflated probability of sampling 0 minimizes the magnitude of noise addition.

The running application in this subchapter is peer review. A few previous papers have considered certain issues of privacy in peer review, but with very different objectives and methods. The paper [41] considers the problem of miscalibration [62, 65, 178, 220] in peer review. They consider privacy leakage when correcting for such miscalibration and provide methods (for a simplified setting) to mitigate this leakage. The paper [40] provides privacy-preserving algorithms for releasing some peer-review data to allow researchers at large to analyze and address problems like subjectivity [118, 158] and miscalibration. The paper [85] considers the problem of coalition-based fraud [86, 128, 216, 231] in peer review, and provides a randomized algorithm to assign reviewers to papers to mitigate such fraud. They argue that such a randomized assignment algorithm has another benefit: it can allow for release of the data that underlies the automated assignment algorithm while still preserving some privacy about which paper was assigned to which reviewer. For a broader overview of research on peer review, see [185].

4.1.3 Problem Formulation

We now describe the problem formulation using the running example of peer-review.

Comment Arrivals. We call the event when a reviewer makes a comment on a paper a *comment arrival*. Each comment arrival consists of 4 elements: the text of the comment, a timestamp t when the comment arrived, a paper p to which it responds, and the reviewer r who made the comment. We assume that comments arrive in continuous time over an infinite time horizon, as this is the most general setup, although our analysis extends to any finite time horizon (for example, in the case where a conference has an end time after which comments can no longer be posted). We consider settings where the comments are publicly observable, as is the case for many conferences run on popular platforms like OpenReview.net.

Batching. In our initial model, we consider comments to be “batched” if they arrive simultaneously. Specifically, a set of 2 or more comment arrivals is *batched* if all comments in the set come from the same reviewer at the same time, and furthermore, the comments are all on different papers. In Section 4.1.5 we discuss how to extend the model to allow for a short gap between batched comments.

Comment Posting Mechanism. A comment posting mechanism M receives comments as they arrive and can choose to delay when they are posted, with the comments only becoming publicly visible at the time they are posted. The mechanism receives a streaming set of comment arrivals \mathcal{A} as input. It outputs a set of comments where each comment has identical content, paper, and reviewer to a comment in the input but with a potentially delayed timestamp. We place the following natural constraints on any *valid* comment posting mechanism:

1. (*Delay-Only*) If a comment arrives at time t it must be output at time t or later.
2. (*No Fake Data*) Any comment posted at time t must have arrived at or before time t .
3. (*Eventual Release of All Comments*) For any comment, letting D denote the potentially randomized delay introduced to the comment by the mechanism, it must be that $\lim_{T \rightarrow \infty} \Pr[D \leq T] = 1$.

Privacy. Our goal is to protect against an adversary who is trying to infer whether a specific pair of comment arrivals was batched. Following the widely-adopted framework of differential privacy, we consider a strong adversary who knows exactly when all comments arrived, except for one pair of comments that either arrived in a batch or at separate times. The adversary knows that comments arrive at the same time if batched and knows the exact inter-arrival time of the pair of comments if they arrive unbatched. In preserving privacy against such a strong adversary, we also provide privacy guarantees for general classes of weaker adversaries with less prior knowledge. For instance, in Section 4.1.5 we discuss an adversary who only has an estimate of the baseline distribution of inter-arrival times when comments are unbatched, rather than the exact inter-arrival time.

Ideally, we would like to provide a privacy guarantee with respect to the standard notion of differential privacy (DP). Such a DP guarantee would promise difficulty of distinguishing whether the mechanism was run on one of two neighboring inputs, where one input has an additional batched pair of comments compared to its neighbor. Unfortunately, as we prove in

Section 4.1.4, it is impossible to guarantee standard ε -DP in this setting. There are two main reasons for this impossibility. First, consider defining neighboring inputs to a DP mechanism where a pair of comments arrives simultaneously in one input when batched, but arbitrarily far apart when unbatched in the neighboring input. Then, to satisfy a traditional DP guarantee, batched comments must be delayed *indefinitely* to make these two inputs indistinguishable. Second, even with a bounded change in arrival time for any comment on neighboring inputs, we show that if the neighboring relation is symmetric (i.e., a pair of comments can be batched in one input and unbatched in the other, and it doesn't matter which input contains the batched comments), then to satisfy ε -DP the mechanism must delay a batched comment indefinitely.

In order to address the aforementioned roadblocks, we relax the definition of neighboring inputs in two ways. First, we introduce a real-valued parameter $\gamma > 0$ into our formulation of neighbors that bounds how far in time a batched comment can move in a neighboring input where it arrives unbatched. Second, we define neighbors in a one-sided manner: a set of comment arrivals neighbors another set only if it contains one *additional* pair of batched comments as compared to its neighbor. In contrast, a set of comment arrivals does not neighbor another set if it contains one *fewer* pair of batched comments than its potential neighbor. Formally, we define neighboring comment arrival sets as follows:

Definition 4.1.1 (γ -Neighboring Comment Arrival Sets). *A set of comment arrivals I^{batch} is γ -neighboring to set of comment arrivals I , if I^{batch} can be obtained from I by batching together one pair of comments that arrive separately in I . The comments must arrive within γ units of time of one another in I and the later comment moves to the earlier comment in I^{batch} to create a batch. Formally, $\exists(a, t, p, r), (a', t', p', r) \in I$ such that $p \neq p', 0 < t' - t \leq \gamma$ and $I^{\text{batch}} = (I \setminus \{c'\}) \cup \{(c', t, p', r)\}$.*

Note that this definition of adjacency is asymmetric as a set of comment arrivals with no pairs of batched comments is not γ -adjacent to any other sets of comment arrivals. As an example, consider the following pair of comment arrival sets I and I^{batch} :

$$I = \{(a_1, t = 1, p_1, r_1), (a_2, t = 2, p_1, r_2), (\mathbf{a_3}, \mathbf{t = 3}, \mathbf{p_2}, \mathbf{r_1})\}, \quad \text{and}$$

$$I^{\text{batch}} = \{(a_1, t = 1, p_1, r_1), (\mathbf{a_3}, \mathbf{t = 1}, \mathbf{p_2}, \mathbf{r_1}), (a_2, t = 2, p_1, r_2)\}.$$

Then under our definition above, I^{batch} is 2-neighboring to I . However, I is *not* 2-neighboring to I^{batch} .⁴

Now, we define privacy of a mechanism using a notion similar to the definition of one-sided differential privacy introduced in [109]. We note that apart from the one-sidedness of neighbors, our privacy formulation differs substantially from that of [109] as we focus on inputs differing in the timing of a pair of comments due to batching, while [109] considers databases where arbitrary entries are considered non-private.

For any finite time horizon T and set of comment arrivals \mathcal{A} , we will let $M_T(\mathcal{A})$ denote the output of the mechanism up to time T . Then, we define privacy as follows:

⁴The reader may have observed that the definition of neighboring comment arrival sets has a technical condition that a batched comment moves later in time in a neighboring input with one fewer instance of batching. It is possible to modify the formulation to let a batched pair of comments arrive at either one of the later or earlier arrival times of an unbatched pair in an adjacent input. This modified formulation would capture an even stronger adversary who knows the exact time frame in which a batched pair arrives. However, ensuring privacy against this adversary would require even more delay added to the system. Hence, we do not pursue this formulation.

Definition 4.1.2 ((ε, γ) -One-Sided Differential Privacy (OSDP)). For any $\varepsilon \geq 0$ and $\gamma > 0$, a comment posting mechanism M is (ε, γ) -one-sided differentially private if for any I, I^{batch} such that I^{batch} is γ -neighboring to I , for any time horizon T , and for any subset of possible outputs $O \subseteq \text{Range}(M_T)$ of the mechanism:

$$\Pr[M_T(I^{\text{batch}}) \in O] \leq e^\varepsilon \Pr[M_T(I) \in O].$$

This privacy definition guarantees that the likelihood of observing an outcome on an input with at least one instance of batching is never much larger than the likelihood of observing that outcome on an input with one *fewer* batched pair. Therefore, the mechanism obscures the fact that any pair of comments was batched. However, it is possible for the mechanism to reveal that a pair of comments was unbatched; we allow for outputs that occur with non-zero probability given input I but zero probability given input I^{batch} (unlike in standard two-sided DP). We argue that the one-sided definition effectively captures privacy risk due to batching, as the presence of a batched pair of comments is sensitive information, while the absence of batching is non-sensitive. We further discuss the motivation for only treating batching as sensitive via the concrete example of reviewer deanonymization by a meta-reviewer.

The privacy definition requires two parameters: ε and γ . The interpretation of ε is similar to two-sided DP as it quantifies the “level” of privacy: for smaller ε it is harder to distinguish neighboring inputs, whereas for larger ε it is easier to distinguish neighboring inputs. The γ parameter captures domain knowledge about what types of inputs can be neighbors, similar to restricting the domain of inputs in two-sided DP. Roughly, γ should capture how far apart consecutive comments would plausibly arrive if batching were not occurring. It is necessary for a practitioner to include this domain knowledge in the form of finite value γ as we prove that batched comments must be delayed by at least γ (in Section 4.1.4) and hence without this bound, comments must be withheld indefinitely. We give heuristics for how to set γ based on a hypothesis testing interpretation of the privacy definition in Section 4.1.5.

Utility. We measure the cost of our mechanism in terms of expected delay added to comments. Because the privacy guarantee is asymmetric, the mechanism can behave differently on batched and unbatched comments. Therefore, we will consider measuring utility in terms of expected delay to batched comments denoted $\mathbb{E}[B]$, expected delay to unbatched comments denoted $\mathbb{E}[U]$ or more generally any weighted sum of the two expectations.

Goal. Our goal is to design comment posting mechanisms that guarantee (ε, γ) -one-sided differential privacy for chosen privacy parameters ε and γ while minimizing the expected delay added to comments. We may add random delay to batched and unbatched comments drawn from different distributions B and U respectively. Therefore, we wish to design (ε, γ) -OSDP mechanisms that are Pareto optimal in trading off between $\mathbb{E}[B]$ and $\mathbb{E}[U]$ at any setting of ε and γ . Moreover, we want to allow practitioners to choose a mechanism on this Pareto frontier that minimizes an appropriate cost function suiting the requirements of their system. For instance, a system with a higher rate of batching may wish to weight delay to batched comments higher in their cost function than a system with a lower rate of batching. To this end, we consider minimizing any cost function that is a convex combination of expected delay to batched and unbatched comments. We

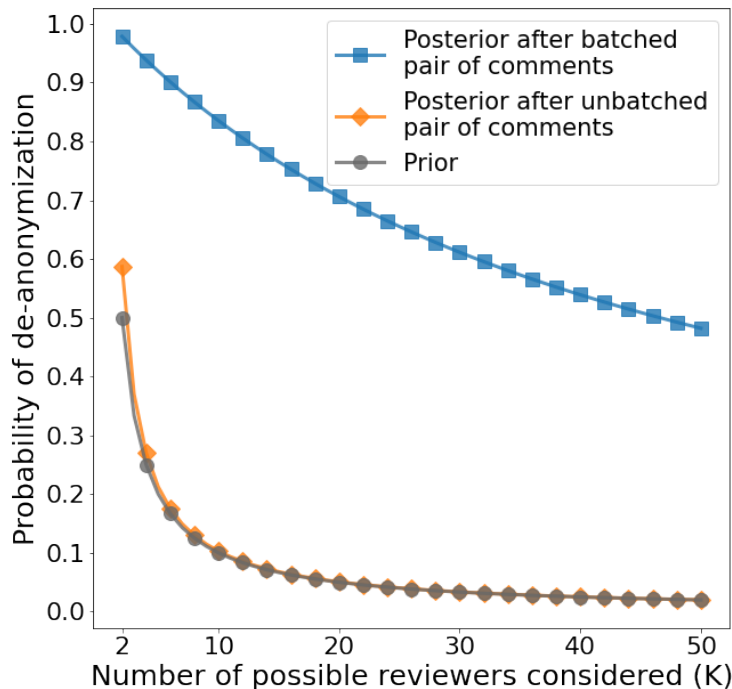


Figure 4.2: Success probability of de-anonymizing a (meta)-reviewer after learning that a pair of comments arrived together vs. learning that a pair of comments did not arrive together.

aim to provide the exact mechanism on the Pareto frontier that minimizes $w\mathbb{E}[B] + (1 - w)\mathbb{E}[U]$ for any choice of weighting parameter $w \in [0, 1]$ and any privacy parameters ε and γ . We note that this choice of utility function is without loss of generality. In particular, the feasible region of $\mathbb{E}[B]$ and $\mathbb{E}[U]$ is convex, as shown in the supplemental material. Therefore, any mechanism that is Pareto optimal in trading off $\mathbb{E}[B]$ and $\mathbb{E}[U]$ minimizes the weighted cost function for some choice of w (since any point on the Pareto frontier of a convex feasible region optimizes some weighted sum objective per Boyd [19, Chapter 4.7]).

Example: De-anonymizing reviewers. We now discuss the one-sided nature of privacy risk inherent to batching using the running example of a meta-reviewer de-anonymizing a reviewer or meta-reviewer of a paper they have authored. Recall the introductory scenario where a meta-reviewer observes two comments a and a' that arrive consecutively on different papers and are made by (meta)-reviewers r_1 and r' respectively (where it is possible that $r' = r_1$). The meta-reviewer knows that the first comment was made by r_1 and has a uniform prior over K possible reviewers who could have made a' (including r_1). They wish to de-anonymize r' based on whether or not a' arrived in a batch with a . From our aforementioned analysis of a conference peer review where we define two comments as “arriving together” if they arrive within 5 minutes of one another, we estimate that: $\Pr[a, a' \text{ arrive together} \mid r' = r_1] \approx 0.3$, while $\Pr[a, a' \text{ arrive together} \mid r' \neq r_1] \approx 0.0066$. Therefore, after learning that a and a' arrived together, the meta-reviewer’s posterior puts the most weight on $\Pr[r' = r_1 \mid a, a' \text{ arrived together}] = \frac{0.301}{0.301 + 0.0066(K-1)}$. On the other hand, after learning that a and a' did not arrive together, their pos-

terior puts the most weight on: $\Pr[r' = r_k \mid a_1, a_2 \text{ did not arrive together}] = \frac{0.9934}{0.699+0.9934(K-1)}$ for $K \neq 1$. We give further detail on how these statistics were estimated in Supplemental Material.

As shown in Figure 4.2, in learning that the pair of comments was batched, the meta-reviewer can identify the (meta)-reviewer of a paper they authored with much higher confidence than before observing the batched timing; on the other hand, by learning that the pair of comments was unbatched, the meta-reviewer’s posterior hardly changes from the prior. Our one-sided privacy definition captures this asymmetric privacy risk. It ensures that an adversary does not learn much about the sensitive information of whether two comments are likely to be batched after observing the time that comments get posted, while allowing the adversary to potentially learn the insensitive information that two comments were unbatched.

4.1.4 One-Sided Privacy Guarantees

In this section, we present our main theoretical results. First, in Section 4.1.4, we propose an algorithmic framework to design comment posting mechanisms that guarantee (ϵ, γ) -one-sided differential privacy under batching. In this framework, we add random noise to the timestamps of batched and unbatched comments, drawing the noise from a pair of distributions (B, U) that depend on parameters ϵ and γ .

Within this framework, there are many possible choices of the noise distributions (B, U) , and we investigate them in Section 4.1.4. For instance, one could use one-sided analogues of distributions commonly used for two-sided differential privacy, like exponential noise, which is the absolute value of the Laplace distribution [53], or one-sided staircase noise [66] (whose two-sided version is known to be optimal for two-sided DP [66]). However, we show that perhaps surprisingly, these distributions are all sub-optimal for the privacy-delay trade-off.

In Section 4.1.4 we provide another distribution – a zero-inflated uniform distribution with carefully chosen parameters – that we show guarantees one-sided differential privacy in our setting and also achieves a Pareto-optimal privacy-delay trade-off.

Finally, in Section 4.1.4, we motivate the usefulness of our one-sided DP formulation as a means of capturing the privacy-delay trade-off by showing that the popular two-sided definition of DP does not yield a useful privacy-delay trade-off for valid comment posting mechanisms.

Randomized Delay Framework

In Algorithm 9, we present a general recipe for designing randomized delay mechanisms. The meta-algorithm receives as input privacy parameters ϵ and γ as well as probability distributions B and U that depend on ϵ and γ . We will then prove that if pairs of distributions satisfy an “indistinguishability” property then Algorithm 9 yields a (ϵ, γ) -OSDP mechanism.

Algorithm 9 Framework for Designing a Randomized Delay Mechanism

Input: privacy parameter $\varepsilon > 0$, maximum time gap $\gamma > 0$, noise addition distributions B and U
for each comment arrival time t **do**
 if a set of batched comments arrives **then**
 For each comment, independently sample $D \sim B(\varepsilon/2, \gamma)$ and post the action at time $t + D$.
 else if if an unbatched comment arrives **then**
 Post the comment at time $t + D$ where $D \sim U(\varepsilon/2, \gamma)$
 end if
end for

Mechanisms within this framework satisfy two useful qualitative properties for deployment in real applications. First, because the noise is sampled at arrival time, we can tell each user the duration of the delay on their comment as soon as they create it. Second, because the noise is sampled independently for each comment, the algorithm does not require a centralized coordinator to determine post times. This enables privacy-sensitive individuals to implement the algorithm for themselves. This ability to be implemented locally is a necessary property for use in cryptocurrencies where there is no central trusted server.

Now, any choice of (B, U) can satisfy (ε, γ) -OSDP as long as B and U are indistinguishable in the following sense:

Definition 4.1.3 (One-Sided Indistinguishable Distributions). *Let B and U be non-negative random variables. We say that the ordered pair (B, U) is (ε, γ) -one-sided indistinguishable if, for any measurable set $S \subseteq \mathbb{R}$ and any $t_0 \in [0, \gamma]$, the distributions satisfy:*

$$\Pr[B \in S] \leq e^\varepsilon \Pr[U \in S - t_0],$$

where for any $S \subseteq \mathbb{R}$, $t \in \mathbb{R}$: $S - t = \{s - t | s \in S\}$.

The following theorem shows sufficiency of such one-sided indistinguishable distributions for guaranteeing privacy.

Theorem 4.1.4 (Privacy of Randomized Delay Mechanisms). *Let (B, U) be any pair of $(\varepsilon/2, \gamma)$ -one-sided indistinguishable distributions. Then, Algorithm 9 using B and U as noise-addition distributions guarantees (ε, γ) -one-sided differential privacy.*

The proof, given in Section 4.1.8, follows by observing that in neighboring inputs, a pair of comments that was batched becomes unbatched with one comment arrival moved forward by at most γ time units. Hence, if B and U have a likelihood ratio bounded by $e^{\varepsilon/2}$ for any values within γ time units of one another, it is hard to distinguish whether the mechanism was given an input with two unbatched comments arriving γ time units apart or two batched comments arriving at the same time (up to a multiplicative factor of e^ε).

Privacy-Preserving Delay Distributions

We now describe a number of possible choices for (ε, γ) -one-sided indistinguishable distributions (B, U) that can be used in our algorithmic framework. We show that we can use an exponential distribution, which is the one-sided version of the Laplace distribution. We can also add

noise from the absolute value of the staircase distribution, which was proven in [66] to be optimal for noise addition in two-sided DP, giving smaller delay than the exponential. Alternatively, we can add noise to unbatched comments drawn from a zero-inflated uniform distribution where we add 0 delay with probability $1 - p$ (for some parameter p) and delay drawn from a uniform distribution with probability p .

Theorem 4.1.5 (Choices of One-Sided Indistinguishable Distributions). *The following choices of B and U are (ε, γ) -one-sided indistinguishable:*

1. Exponential⁵: $B = \gamma + \text{Exponential}(\varepsilon/\gamma)$, $U = \text{Exponential}(\varepsilon/\gamma)$
2. Staircase [66]⁶: $B = \gamma + |\text{Staircase}(\varepsilon, \gamma)|$, $U = |\text{Staircase}(\varepsilon, \gamma)|$
3. Uniform: $B = \text{Unif}(\gamma, \frac{1}{1-e^{-\varepsilon}}\gamma)$, $U = \text{Unif}(0, \frac{1}{1-e^{-\varepsilon}}\gamma)$
4. Zero-inflated Uniform with parameter p . For $e^{-\varepsilon} < p \leq 1$:

$$B = \text{Uniform}\left(\gamma, \frac{p}{p-e^{-\varepsilon}}\gamma\right)$$

$$U = \begin{cases} 0 & \text{with probability } 1 - p \\ \text{Uniform}\left(0, \frac{p}{p-e^{-\varepsilon}}\gamma\right) & \text{with probability } p. \end{cases}$$

These choices of (B, U) incur the following expected delays:

1. Exponential: $\mathbb{E}[B] = \gamma(1 + \frac{1}{\varepsilon})$ and $\mathbb{E}[U] = \gamma\frac{1}{\varepsilon}$
2. Staircase: $\mathbb{E}[B] = \gamma(1 + \frac{e^{\varepsilon/2}}{e^{\varepsilon}-1})$, $\mathbb{E}[U] = \gamma\frac{e^{\varepsilon/2}}{e^{\varepsilon}-1}$
3. Uniform: $\mathbb{E}[B] = \frac{1}{2}\gamma(1 + \frac{e^{\varepsilon}}{e^{\varepsilon}-1})$ and $\mathbb{E}[U] = \frac{1}{2}\gamma(\frac{e^{\varepsilon}}{e^{\varepsilon}-1})$
4. Zero-inflated Uniform with parameter p : $\mathbb{E}[B] = \frac{1}{2}\gamma(p + \frac{pe^{\varepsilon}}{pe^{\varepsilon}-1})$ and $\mathbb{E}[U] = \frac{1}{2}\gamma(\frac{p^2e^{\varepsilon}}{pe^{\varepsilon}-1})$.

The proof of this theorem is given in Section 4.1.8. Note that the (uniform, uniform) noise additions are a special case of (uniform, zero-inflated uniform) taking $p = 1$. We highlight them separately in Section 4.1.5 as we introduce the zero-inflated uniform distribution for the first time here. In the next section, we show that a zero-inflated uniform distribution is Pareto optimal for appropriate choice of p .

Notably, the choice of parameters for the exponential and staircase distributions given in Theorem 4.1.5 are the optimal choice of parameters in the sense that they minimize expected delay at fixed values of privacy parameters ε and γ when adding i.i.d. exponential or staircase noise plus a constant offset to all comments:

Theorem 4.1.6 (Optimal Choice of Parameters for the Exponential and Staircase Distributions). *Let B, U be non-negative noise-addition distributions that guarantee (ε, γ) -OSDP when used in Algorithm 9 where $B = O_B + P_1$ and $U = O_U + P_1$ for constants $O_B, O_U > 0$ and non-negative random variable P_1 . Then, if P_1 is an exponential random variable or a staircase random variable, $\mathbb{E}[B]$ and $\mathbb{E}[U]$ are minimized at any values of ε, γ by the choice of parameters in Theorem 4.1.5 such that B, U are $(\varepsilon/2, \gamma)$ -one-sided indistinguishable.*

⁵In the notation to follow, we parameterize the exponential distribution by its rate.

⁶The staircase distribution is parameterized by 3 values $\varepsilon, \Delta, \gamma$ in [66]. Here, we take $\text{Staircase}(\varepsilon, \gamma)$ to mean the staircase distribution with $\varepsilon = \varepsilon$, $\Delta = \gamma$ and $\gamma = \frac{1}{1+e^{\varepsilon/2}}$, which is the optimal value of γ to minimize expectation per [66].

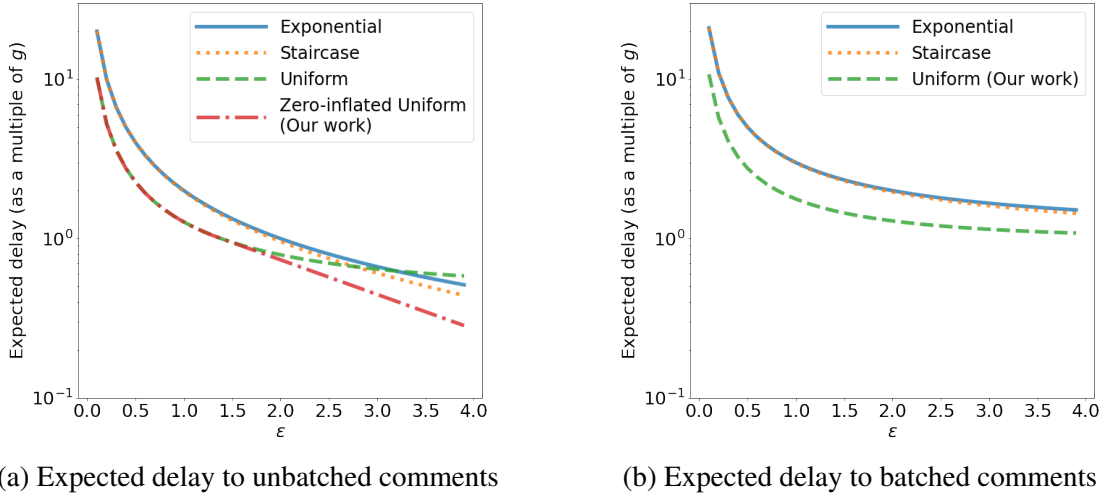


Figure 4.3: Expected delay (as a multiple of γ) for Algorithm 9 with the exponential, staircase, zero-inflated uniform and uniform distributions at varying values of ϵ . All distributions use the optimal setting of parameters at a given ϵ . The zero-inflated uniform parameter is chosen to minimize delay to unbatched comments. The delay (y axis) is plotted on a log scale.

The proof of the above theorem can be found in Supplemental Material. By Theorem 4.1.5 and Theorem 4.1.6, adding i.i.d. exponential or staircase noise plus a constant offset is strictly sub-optimal in minimizing expected delay as zero-inflated uniform noise can achieve lower delay at the same privacy level.

Corollary 4.1.7. *Among (ϵ, γ) -OSDP mechanisms following the framework of Algorithm 9, taking B and U to be i.i.d. exponential or staircase distributions (with constant offsets) is strictly sub-optimal in minimizing $\mathbb{E}[B]$ and $\mathbb{E}[U]$ for any values of ϵ and γ . In particular, using the zero-inflated uniform mechanism with appropriate choice of p can achieve lower expected delay for both $\mathbb{E}[B]$ and $\mathbb{E}[U]$ at any values of privacy parameters ϵ and γ .*

In this setting, the exponential and staircase distributions typically used in two-sided DP add significantly more delay than zero-inflated uniform noise, especially at small values of ϵ . In Figure 4.3, we show the expected delay for the optimal exponential, staircase, uniform, and zero-inflated uniform at each setting of ϵ . For both batched and unbatched comments, the uniform and zero-inflated uniform distributions add a factor of nearly two times less delay than the staircase and exponential at small values of ϵ . For larger values of ϵ , all of the aforementioned distributions add similar delay, with uniform adding the least delay to batched comments and the zero-inflated uniform adding the least delay to unbatched. In the next section, we formally prove that zero-inflated uniform noise is Pareto optimal and characterize the optimal choice of p for any objective function that is a weighted sum of $\mathbb{E}[U]$ and $\mathbb{E}[B]$ based on the setting of ϵ .

Pareto-Optimal Delay Mechanism

Algorithm 10 Zero-Inflated Uniform Mechanism

Input: privacy parameter $\varepsilon > 0$, maximum time gap $\gamma > 0$, weighting of expected delay to batched comments $w \in [0, 1]$

Set $p = \min \left\{ e^{-\varepsilon/2} \left(1 + \sqrt{1 + e^{\varepsilon/2} \frac{w}{1-w}} \right), 1 \right\}$

for each comment arrival time t **do**

if a set of batched comments arrives **then**

 For each comment, independently sample $D \sim \text{Unif} \left(\gamma, \frac{p}{p - e^{-\varepsilon/2}} \gamma \right)$ and post the comment at time $t + D$.

else if an unbatched comment arrives **then**

 Post the comment at time $t + D$ where $D = 0$ with probability $1 - p$ and $D \sim \text{Unif} \left(0, \frac{p}{p - e^{-\varepsilon/2}} \gamma \right)$ with probability p .

end if

end for

In this section, we derive the Pareto frontier (trading off the expected delay for batched and unbatched comments) of noise-addition distributions for a given (ε, γ) -one-sided indistinguishability constraint (Definition 4.1.3). We show that adding zero-inflated uniform noise with appropriate choice of parameter p achieves optimal expected delay among mechanisms that add independent noise to each comment. While the optimality result holds only within the class of mechanisms that adds *independent* noise to each comment, this constraint allows for an algorithm to be implemented locally without requiring coordination by a centralized server. This constraint is a common property of many deployed privacy-preserving algorithms. For instance, local differential privacy [99] requires that randomization needed for privacy is added locally by each holder of a data-point, and the Tor anonymous network [42] protocol requires that initiators of connections choose the (random) path on which to send a message themselves.

Given an (ε, γ) -one-sided privacy constraint, our algorithmic framework (Algorithm 9) has many choices of noise-addition distributions that can guarantee privacy. In terms of delay, there are two quantities to optimize – the delay incurred by batched comments and that incurred by unbatched comments. A natural utility objective to consider is a convex combination of the two expectations:

$$w\mathbb{E}[B] + (1 - w)\mathbb{E}[U], \quad \text{for a given parameter } w \in [0, 1].$$

The parameter $w \in [0, 1]$ determines how much weight is given to batched comments in the utility function. For example, a user of our algorithm may estimate the relative rate of batching in the system and set w to this value to optimize for the overall average expected delay across all comments.

We present our main algorithm as Algorithm 10. Our algorithm follows our previously introduced framework (Algorithm 9). It chooses U as a zero-inflated uniform distribution with

a carefully chosen value of parameter p (dependent on ε and w), and chooses B as a uniform distribution. The following theorem now proves that for any privacy parameters our algorithm is indeed Pareto optimal – it optimally trades off privacy and unbatched delay and batched delay.

Theorem 4.1.8 (Pareto optimality of the Zero-Inflated Uniform Mechanism). *Algorithm 10 is Pareto optimal between expected delay to batched and unbatched comments at a given setting of (ε, γ) among valid (ε, γ) -OSDP mechanisms that add independent noise to each comment. Further, given weight parameter $w \in [0, 1]$ and privacy parameters (ε, γ) as input, Algorithm 10 minimizes cost function $w\mathbb{E}[B] + (1-w)\mathbb{E}[U]$ at any given privacy level (ε, γ) among mechanisms adding independent noise drawn from distributions B and U to batched and unbatched comments respectively.*

We give a proof sketch below and include the main argument in Section 4.1.8.

Proof sketch. Roughly, the proof proceeds as follows:

- We consider any (ε, γ) -indistinguishable noise addition distributions (B, U) added to batched and unbatched comments respectively. Using results from [66], we argue that for large enough $i \in \mathbb{N}$, we can approximate B and U arbitrarily well with random variables that have piece-wise constant probability density functions and each constant interval has length γ/i .
- We establish properties of any Pareto optimal (B_i, U_i) by directly proving that we can decrease the expectation of both $\mathbb{E}[B_i]$ and $\mathbb{E}[U_i]$ for any pair of distributions that violates these properties. Taken together the properties yield the exact form of any Pareto optimal B_i and U_i . Taking limits as $i \rightarrow \infty$ gives that the Pareto frontier is realized by uniform and zero-inflated uniform distributions for some setting of η . The proof follows by directly proving that we can decrease the expectation of both $\mathbb{E}[B_i]$ and $\mathbb{E}[U_i]$ for any pair of distributions that violates these properties.
- Finally, we analytically solve for the value of parameter η in the zero-inflated uniform distribution that minimizes weighted objective $w\mathbb{E}[B] + (1-w)\mathbb{E}[U]$ for any $w \in [0, 1]$.

□

As shown in Figure 4.4, for smaller privacy budgets where $\varepsilon \leq 2 \ln(2)$, there is a single point on the Pareto frontier. Adding uniform noise with no inflated probability mass at 0 minimizes $\mathbb{E}[B]$ and $\mathbb{E}[U]$ simultaneously. For larger ε , it is possible to trade off between $\mathbb{E}[B]$ and $\mathbb{E}[U]$, achieving near-zero delay to unbatched comments. In practice, a user can decide what value of p to use based on their preferred convex combination of $\mathbb{E}[B]$ and $\mathbb{E}[U]$.

Note that our result holds for all mechanisms that add independent noise to each comment, as the delay added to comments must be $(\varepsilon/2, \gamma)$ -one-sided indistinguishable to preserve privacy. Therefore, the zero-inflated uniform mechanism (Algorithm 10) is the Pareto optimal mechanism among this class of algorithms. It may be possible to add even less delay with mechanisms that can coordinate across comments and correlate noise addition; this remains an important extension of the model.

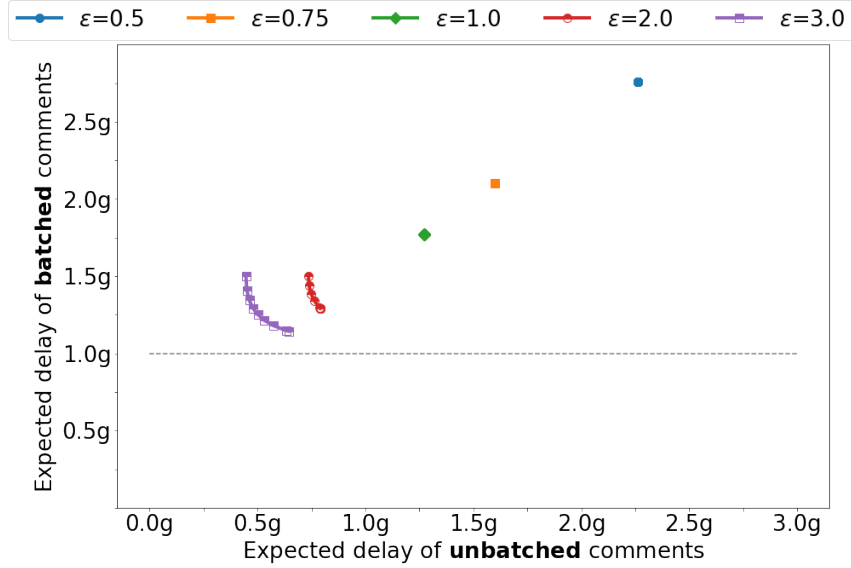


Figure 4.4: Pareto frontier for the expected delay added to batched and unbatched comments ($\mathbb{E}[B]$ and $\mathbb{E}[U]$) at different values of privacy parameter ϵ .

Impossibility of “Two-Sided” Differential Privacy

In the prior sections, we have characterized the privacy-utility trade-off for the one-sided relaxation of differential privacy. One might wish to obtain similar results for the standard two-sided definition of differential privacy, which would provide even stronger privacy guarantees. In this section, we prove the impossibility of guaranteeing two-sided differential privacy under the constraints of a valid comment posting mechanism. These results motivate the further modeling assumptions on the adversary’s prior knowledge about batching and attempted attacks that are used in the definition of (ϵ, γ) -OSDP.

First, we recall the standard definition of two-sided differential privacy. The key difference between this definition and our one-sided Definition 4.1.2 is in the formulation of “neighboring” inputs. In our one-sided definition, we use an asymmetric relation for neighboring inputs where one input with an additional batched pair of comments neighbors an input with one fewer pair. This captures the notion that batching is sensitive while the absence of batching is insensitive. For two-sided DP, we will give a definition with an abstract notion of neighbors and then concretely instantiate this definition with different possible notions of neighboring inputs. Critically, we will consider *symmetric* relations for neighboring inputs in the definition of two-sided DP. This corresponds to preventing an adversary from inferring both whether batching occurred and whether batching did not occur.

Recall that $M_T(\mathcal{A})$ denotes the output of the mechanism up to time T . Then:

Definition 4.1.9 (Two-Sided Differential Privacy for Batched Arrivals:). *For any $\epsilon \geq 0, \delta \in [0, 1]$, a comment posting mechanism M is (ϵ, δ) -differentially private if, for any time horizon T and for any subset $O \subseteq \text{Range}(M_T)$ of possible outputs of the mechanism:*

$$\Pr[M_T(I') \in O] \leq e^\epsilon \Pr[M_T(I) \in O],$$

where I and I' are two “neighboring” sets of comment arrivals.

Now, we state our main impossibility result. We consider three natural definitions of neighboring sets of comment arrivals. The first definition adds or removes a comment from the set of comment arrivals corresponding to the notion of “unbounded” differential privacy in the literature [46]. The second definition moves a comment from being batched to unbatched in neighboring inputs by changing its timestamp, corresponding to the notion of “bounded” differential privacy in the literature [53]. Finally, the third definition restricts the second definition of neighbors further by placing a bound on how far a comment can move (which we call γ), similar to the practice of constraining the domain of possible inputs to a differentially private mechanism. We show that it is not possible to guarantee privacy for any of these notions of neighbors:

Theorem 4.1.10 (Impossibility of Two-Sided Differential Privacy). *For any of the following natural definitions of “neighboring” sets of comment arrivals, there is no two-sided differentially private, valid comment posting mechanism with delay scaling as $o(1/\delta)$:*

<i>Definition of “Neighboring” Sets of Comment Arrivals</i>	<i>Impossibility Result</i>
(1) <i>Add or remove a batched comment</i>	<i>No valid (ε, δ)-DP posting mechanism for $\varepsilon < \infty, \delta < 1$</i>
(2) <i>Move a batched comment to another arrival time where it is no longer batched</i>	<i>No valid (ε, δ)-DP posting mechanism for $\varepsilon < \infty, \delta < 1$</i>
(3) <i>Move a batched comment by at most γ units of time to another arrival time where it is no longer batched</i>	<i>For any $T \geq 0$, any valid (ε, δ)-DP posting mechanism delays a comment by at least T with probability $\geq 1 - 2\delta \left(\frac{T}{\gamma} + 1\right)$</i>

The proof of the above theorem can be found in the supplemental material. Intuitively, we cannot guarantee privacy with definition (1) of neighbors because it would require creating a fake comment since a comment that exists in one input does not exist in the adjacent input. It is not possible to satisfy privacy with definition (2) of neighbors, as a comment could move arbitrarily far in time, requiring infinite delay to be added to comments. For definition (3) of neighbors, we show that we can define a sequence of neighboring inputs such that a comment is shifted γ units of time in the future on every other input in the sequence. Since the privacy guarantee must hold pairwise between each neighboring input in the sequence, the mechanism can only release comments within time T with probability of roughly $\delta T/\gamma$ in order to make inputs that are T/γ neighbors away from each other in the sequence sufficiently indistinguishable from one another.

Note that even if we considered mechanisms acting on a finite time horizon, the proof above suggests the only mechanism admitted under two-sided DP using definition (2) is the trivial mechanism that releases all comments at the end of the time period:

Corollary 4.1.11. *Suppose comments are known to arrive only during a finite time horizon T after which no more comments will arrive or be posted. Then, any valid posting mechanism that satisfies two-sided (ε, δ) -DP using Definition (2) of neighbors in Theorem 4.1.10 posts all comments at time T .*

This is both intuitively and formally sufficient for preserving privacy from timing attacks

since it eliminates all timing information, but is expensive in terms of delay incurred. In particular, in the peer review setting, releasing all comments simultaneously at the end of the review period eliminates potential for replies and ongoing discussion.

It follows from the impossibility of definition (3) of neighboring sets that there is no valid comment posting mechanism satisfying differential privacy with $\delta = 0$ for this notion of neighbors, since any differentially private mechanism would violate the property that valid comment posting mechanisms eventually release all comments. Additionally, even taking $\delta > 0$, the probability of experiencing a delay longer than T only decreases *linearly* in δ and T . Typically, δ is selected to be $o(1/n)$ [49], where n is the database size—in our case, the number of comments in the observed stream. For $\gamma = O(1)$, this implies that any mechanism satisfying a two-sided DP guarantee for γ -neighboring inputs (and choosing $\delta = o(1/n)$) has a non-negligible probability of delaying comments by $\Omega(n)$.

4.1.5 Practical Considerations for Implementation

In this section, we address two important practical considerations to putting into practice our privacy formulation and algorithm. First, we provide theoretically motivated heuristics for setting the parameter γ in the privacy definition. Second, we give simple extensions to the privacy model and algorithm that allow for handling the realistic setting where batched comments do not arrive all at the same exact time, but rather with a short duration in between.

Setting Privacy Parameters

Recall that our privacy definition includes a parameter γ that captures what types of inputs can be neighbors. In particular, γ bounds how far apart in time a pair of potentially-batched comments could arrive if batching had *not* taken place. In this section, we provide a heuristic for setting γ in practice. We will argue that a reasonable way to set γ for a given comment is as a percentile of an empirical distribution of comment inter-arrival times. For example, in a peer-review conference we might set γ to be the median inter-arrival time of comments at a similar prior conference. Alternatively, larger conferences commonly classify papers into tracks, so γ could be chosen for each track individually. We provide more examples of setting γ in practice in our experiments on Wikipedia and Bitcoin in Section 4.1.6.

First, we motivate this heuristic by modeling an adversary conducting a hypothesis test to determine if a comment was batched or not. The privacy parameters γ and ε can be chosen based on the desired (in)efficacy of this adversary’s test. A natural way to model a privacy attack is to consider an adversary—say, a meta-reviewer who submitted a paper to a conference—who suspects that a comment a made on their paper may share a reviewer with one of the papers in the set S of papers they are handling. The adversary conducts a hypothesis test to determine whether the comment they received arrived in a batch with any comment on papers in that set. Let t_1 denote the arrival time of a and let t_2 denote the arrival time of the comment in S that arrives closest in time to t_1 . The adversary knows that if the comments did not arrive in a batch, then they arrived with a gap $t_2 - t_1$ following some distribution D (for instance, this might be the empirical distribution of comment inter-arrival times on the previous day). If the pair of comments does arrive in a batch, the adversary assumes they arrived simultaneously. Thus, the

adversary wishes to distinguish between the following hypotheses:

$$\begin{aligned} H_0 : t_2 - t_1 &\sim D && (a \text{ is not batched with any comment in } S) \\ H_1 : t_1 &= t_2 && (a \text{ is batched with at least one comment in } S) \end{aligned}$$

The adversary will observe the output of the mechanism and decide to either accept or reject the null hypothesis. If they reject the null hypothesis, they conclude that the comment was batched with a comment in S . Their hypothesis test is defined by “rejection region” R , or the set of outputs on which the adversary concludes that batching occurred. The quality of a given test is determined by the trade-off between its “power” and “type I error”:

$$\begin{aligned} \text{Power} &= \Pr[M(O) \in R; H_1] \\ \text{Type I Error} &= \Pr[M(O) \in R; H_0] \end{aligned}$$

Similar to prior work on differential privacy [223], [95], we show that an adversary conducting a hypothesis test to determine if batching occurred will face a poor trade-off between power and type I error given an output of a mechanism that is OSDP with gap γ :

Proposition 4.1.12. *If a mechanism M satisfies (ϵ, γ) -OSDP, then for any comment a , set of comments S arriving with inter-arrival time distribution D , and any hypothesis test deciding if a was batched with a consecutively arriving comment in S :*

$$\text{Power} \leq \frac{e^\epsilon}{q(\gamma)} (\text{Type I Error})$$

where $q(\gamma) = \Pr[|x| \leq \gamma; x \sim D]$ is the CDF of inter-arrival times.

The proof of this proposition can be found in Supplemental Material. This interpretation of the (ϵ, γ) -OSDP guarantee in terms of error rates of an attacker’s hypothesis test motivates our heuristic to choose the parameter γ . Previous work on timing attacks [121, 190] measures the success of attacks in terms of the trade-off between power and type I error. In particular, these works report a single number “error crossover rate,” the point at which $\text{type I error} = 1 - \text{power}$. We envision the system operator (i.e., the entity adding the delay) first specifying a tolerable error crossover rate; for example, consistent with prior work on timing attacks [121, 190], the operator might choose to tolerate an error crossover rate of 0.25. Next, the system operator should choose a privacy parameter ϵ . Since the interpretation of ϵ is similar to traditional two-sided DP, operators may use common heuristics for selecting ϵ ; for instance, our operator might choose $\epsilon = 0.8$. Given these parameters, Proposition 4.1.12 shows how to select g to ensure that the desired error crossover rate is satisfied. In our running example, we would choose g to be the 75th percentile of the inter-arrival time distribution.

Handling Non-Simultaneous Batching

In our basic model of batching, we make the idealized assumption that all comments in a batch arrive at the same exact clock time. In practice, in many settings, batched actions will not be taken at the *exact* same time, but rather with some short delay between them. For example, it is natural for a Wikipedia editor to spend many minutes working on a revision, so revisions in

a single batch may arrive with a few minutes of delay in between. Likewise, reviewers in peer review may comment on papers one after the other, leading to a short delay despite batching.

In this section, we describe a simple extension to our model and algorithm that allows us to handle non-simultaneity in practice. We introduce a new threshold β , below which we consider two comments to have been batched — if two comments come from the same reviewer within time β we consider them to have arrived in a batch. We will assume that $\beta < \gamma$, as we wish to capture scenarios where batching leads a comment to arrive earlier than it would have without batching. We can capture this scenario by replacing the notion of neighbors in our model with the following:

Definition 4.1.13 (γ -Neighboring Comment Arrival Sets with β -batching). *For $\beta < \gamma$, a set of comment arrivals I^{batch} is γ -neighboring with β -batching to set I , if I^{batch} can be obtained from I by batching together a pair of comments that arrive separately within γ time units of one another in I , moving the later comment to within β of the earlier comment. Specifically, $\exists(a, t, p, r), (a', t', p', r) \in I$ such that $p \neq p', 0 \leq t' - t \leq \gamma$ and $I^{\text{batch}} = I \setminus \{a'\} \cup \{(a', t'', p', r)\}$ where $0 \leq |t'' - t| \leq \beta$.*

We define privacy the same as in Definition 4.1.2, but with this modified notion of γ -neighboring with β -batching. In what follows, we describe how we incorporate this relaxed notion of batching into our algorithm.

First, we propose a simple front-end change that can be employed in conjunction with any mechanism in our algorithmic framework of randomized delay mechanisms (Algorithm 9) if we trust users to accurately report when they will engage in batching. The solution is to ask users when they create a comment if they plan on creating more comments on their other papers within the next β units of time (and hence will generate batched comments). If the user answers affirmatively, then we treat their current comment as well as any subsequent comments they make within β time units as batched and add delay drawn from B to the batched comments. If not, we add delay from U to the unbatched comments. Here, we take (B, U) to be one-sided $(\varepsilon/2, \gamma + \beta)$ -indistinguishable. Since neighboring inputs can differ on two comments with arrival times at $(t, t - \beta)$ and $(t + \beta, t + \gamma)$ respectively, it is now necessary to add noise from $(\varepsilon/2, \gamma + \beta)$ -indistinguishable distributions to preserve privacy by the same reasoning as Theorem 4.1.4.

Algorithm 11 Framework for Handling Non-Simultaneous Batching

Input: privacy parameter $\varepsilon > 0$, maximum gap $\gamma > 0$, batching threshold $0 \leq \beta < \gamma$, noise addition distributions B and U

for comment arriving at time t **do**

Hold the comment until time $t + \beta$.

if the same reviewer batched another comment during time t to $t + \beta$ **then**

Sample $D \sim B(\varepsilon/2, \gamma + \beta)$ and post the action at time $t + \beta + D$.

else

Post the comment at time $t + \beta + D$ where $D \sim U(\varepsilon/2, \gamma + \beta)$

end if

end for

In settings where we do not expect users to reliably report that they will batch tasks, we can

	Mean Delay				Maximum Delay			
	$\varepsilon = 0.1$	$\varepsilon = 0.5$	$\varepsilon = 1.0$	$\varepsilon = 2.0$	$\varepsilon = 0.1$	$\varepsilon = 0.5$	$\varepsilon = 1.0$	$\varepsilon = 2.0$
(a) $\gamma = 11$	118	31	20	15	229	54	33	22
(b) $\gamma = 36$	343	83	50	35	672	152	88	56

Table 4.1: Mean and maximum delay (in minutes) added to Wikipedia article revisions within the category “21st-century American Politicians” for γ set to the (a) 25th and (b) 50th percentile of the historical inter-arrival distribution.

use a simple extension to our algorithmic framework, described in Algorithm 11, where we delay all comments by an additional β units of time, using that duration to determine whether or not the comment was batched. The algorithm pays an additional β in overhead to decide whether a comment was batched or not. Privacy follows by the same reasoning as in Theorem 4.1.4. In general, our initial problem formulation captures the most essential features of the problem of preserving privacy in the presence of batching. As we have shown in this section, it is straightforward to extend our model to better capture the properties specific to a given application.

4.1.6 Empirical Linkage Attacks

We conduct two sets of experiments using publicly available data on Wikipedia article revisions and Bitcoin transactions.

Wikipedia

In a dataset of revisions on all Wikipedia articles from January 1st to 31st, 2022 obtained from the Wikimedia API, we aggregate over 3.5 million article revisions (after filtering out bot accounts), averaging roughly 80 revisions per minute. Due to the high baseline rate of editing, it would be difficult for an adversary to identify that two revisions are batched without narrowing down the set of possible articles they consider. Therefore, we focus on a subset of Wikipedia revisions within which an adversary tries to link editors. One natural clustering of articles likely to contain batched revisions is by category: each article on Wikipedia is associated with a set of categories capturing the main topics covered. In the following experiments, we analyze articles belonging to the category “21-st century American Politicians.” We chose this category because it contains potentially controversial political topics so editors may have privacy concerns in editing these pages. For instance, one news report describes how editors of Donald Trump’s Wikipedia page (one of the pages captured in the category) “are fighting a brutal, petty battle over every word [137].” Additionally, this category receives a large number of revisions per month, yielding a sample size of 13,430 revisions. Among this set of revisions, roughly 20% were generated in a batch with another revision on a page in the same category (where we consider revisions to be batched if they arrive within 5 minutes of one another and are made by the same user). The threshold of 5 minutes captures 92% of pairs of immediately consecutive revisions by a single editor on different articles within this category.

While we do not have access to the true identities of editors who use multiple accounts, we can track all revisions made by the same account and identify when this account engages in

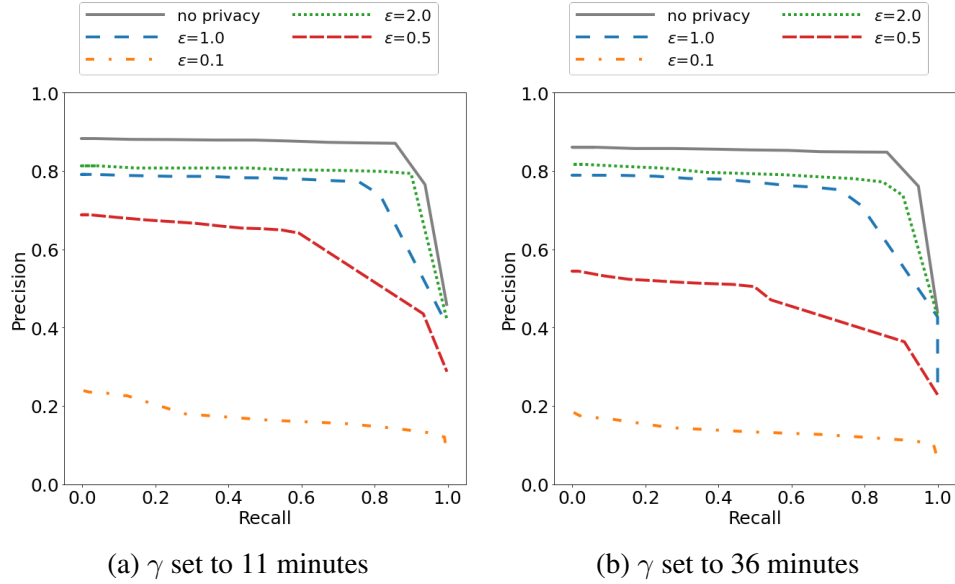


Figure 4.5: Accuracy in linking pairs of Wikipedia article revisions within the category “21st-century American Politicians” based on batched timing (averaged over 5 runs of the randomized privacy mechanism).

batching. We simulate an attack where an adversary tries to link revisions to their creator on the basis of timing, while ignoring the usernames of editors. We consider a simple attack model that proves to be quite effective in the absence of any privacy-preserving mechanism. In the attack, the adversary tries to classify each pair of revisions on two different articles as either batched or unbatched. The adversary chooses a cutoff $c \geq 0$: if a pair of revisions are posted within c minutes of one another, the adversary classifies the pair as batched and concludes that the comments were made by the same person, and if not, the adversary classifies the pair as unbatched (in which case, the adversary draws no conclusion). When the adversary correctly classifies a batched pair as batched, we call this a true positive, while if the adversary incorrectly classifies an unbatched pair as batched, we call this a false positive. The adversary can trade off between false positives and true positives by choosing the value of c accordingly, with higher values of c yielding more true positives, but also more false positives, than smaller values of c . The efficacy of the attack is measured in terms of its precision and recall where precision = $\frac{\text{number of pairs correctly classified as batched}}{\text{total number of pairs classified as batched}}$ and recall = $\frac{\text{number of pairs correctly classified as batched}}{\text{total number of pairs that were batched}}$. An effective attack has simultaneously high precision and recall.

In Figure 4.5, we show the precision and recall of this attack under various settings of privacy parameters ϵ and γ . We find that attack efficacy is quite high when no privacy mechanism is deployed — for instance, it is possible to obtain recall of 85% at a precision of 80%. We then apply the zero-inflated uniform mechanism (Algorithm 10) and measure the reduction in attack efficacy over the “no privacy” baseline. We run the mechanism with w set to 1, as this minimizes the worst-case expected delay added to any single comment in the system. Because batching is not perfectly simultaneous on Wikipedia — editors take time between making each revision — we simulate deployment of the user interface extension to Algorithm 10 described in

Section 4.1.5 where $\beta = 5$ minutes. We set γ based on the heuristic from Section 4.1.5 where γ is a percentile of the inter-arrival distribution of revisions made in the first week of the month. We then simulate deployment of the algorithm over the last three weeks of the month. Using this method, we can set $\gamma = 11$ minutes by choosing the 25th percentile or $\gamma = 36$ minutes at the 50th percentile. The experiment reveals that precision and recall are significantly improved by use of the mechanism as shown in Figure 4.5. In terms of delay, Table 4.1 shows the mean and maximum delay added to comments. Additional results are provided in the supplemental material.

Thus we find that Algorithm 10 renders the privacy attack much less effective while introducing reasonable delay. For instance, taking $\gamma = 11$ and $\varepsilon = 0.5$ corresponds to an average delay of roughly 1 hour 20 minutes and maximum delay of 2.5 hours, but makes the attack substantially less accurate: the attack now achieves around 65% recall at 60% precision compared to the non-private baseline which achieves 85% recall at 80% precision. The heuristic attack used in Figure 4.5 may not be optimal for an adversary who has knowledge of the zero-inflated uniform mechanism, but not access to the internal randomness of the mechanism. Identifying an optimal attack is beyond the scope of this work. However, since the same noise distribution B is added to all comments that arrive in a batch, we expect the heuristic attack to perform well in expectation.

Bitcoin

In Bitcoin, we wish to protect against linkage attacks on users of Bitcoin who use multiple addresses to transmit currency to the same recipient address at the same time. We aggregate data of all confirmed transactions broadcast to the Bitcoin peer-to-peer network in the week of August 1, 2022 to August 7, 2022, consisting of approximately 250,000 transactions per day. While we cannot tie different addresses to real-world identities, for the purposes of our experiments, we consider the following proxy: we define a “batch” to have occurred when two transactions from different input addresses are sent to the same output address within 1 minute of one other. This represents a key use-case of our algorithm, wherein a person holding Bitcoin in multiple addresses wishes to draw from these different sources to complete a transfer to a single output address. After filtering for transactions originating from addresses with unusually high volume of transactions that likely represent cryptocurrency exchanges, there are about 3,000 transactions per day arriving in a batch per our definition, representing 1.2% of all transactions.

We consider a privacy attack similar to the linkage attack described in the Wikipedia application. In the Bitcoin setting, an adversary tries to identify whether pairs of transactions arrived in a batch or not. The adversary observes the times at which transactions to the same output address are broadcast to the Bitcoin P2P network and applies a threshold to the time difference between the pair to decide whether the transactions arrived in a batch. In a “basic” attack, the adversary uses a single threshold for all transactions. In an “informed” attack, we assume the adversary knows the value of γ that was used by the privacy mechanism for each transaction (which the mechanism may vary by output address) and sets a per-address threshold as a linear function of the γ used for that address. In incorporating this additional information about the privacy mechanism, the adversary can obtain a better trade-off between false positives and true positives. We measure efficacy of the attack in terms of precision and recall. Since we define batching to occur when multiple inputs are sent to the same output address within 1 minute of each other,

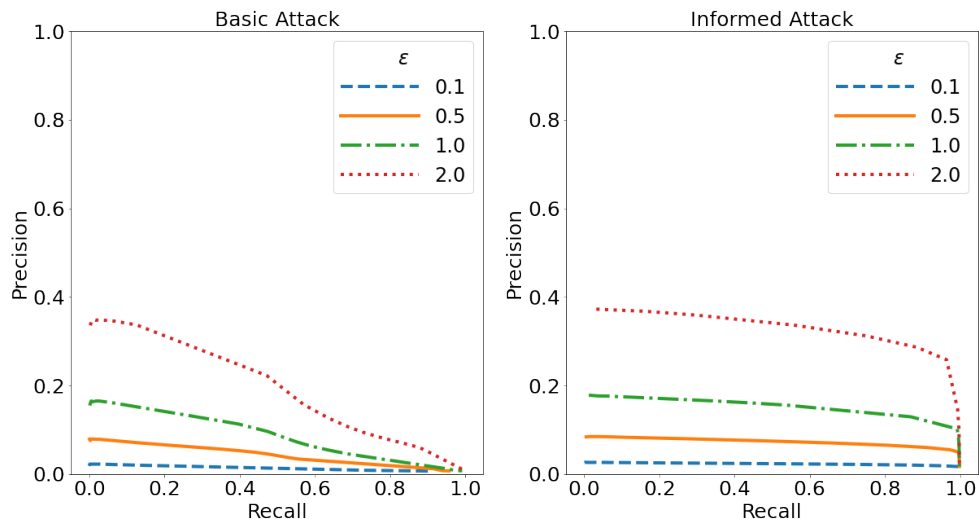


Figure 4.6: Performance of basic and informed linkage attacks on Bitcoin transactions when γ is set to the median historical inter-arrival time for an output address.

the adversary can observe exactly when batching occurred if no privacy mechanism is deployed and obtain a precision and recall of 100% in identifying whether transactions arrived at the same time or not (recall that in this experiment, we lack ground truth about batched transactions).

To obscure the timing of transactions, we simulate the zero-inflated uniform mechanism (Algorithm 10) to add delay to the time at which transactions are broadcast to the Bitcoin P2P network. In order to select the value of γ , we estimate the inter-arrival distribution of transactions to a given output address in the prior 7 days and set γ to a percentile of this distribution. In particular, in this section we use the median of the inter-arrival distribution. In Supplemental Material, we give additional results for experiments where γ is set to the 25th and 75th percentile of the inter-arrival distribution. If the output address of a transaction received no other transactions in the prior 7 days, we set γ to 10 minutes, as this is the baseline duration of time a Bitcoin user has to wait for a transaction to be confirmed on the blockchain. Most ($> 90\%$) unbatched transactions are sent to output addresses with no recent transaction history, so we use the value of $\gamma = 10$ for these transactions. However, roughly 80% of batched transactions are sent to output addresses with transaction history.

The use of Algorithm 10, with γ set per output address, makes it difficult to identify whether transactions to the same output address arrived at the same time. For $\varepsilon = 1$, even the informed attack has precision of only 20% at high recall. The basic attack performs much worse, indicating that an adversary needs to incorporate additional information about baseline inter-arrivals of transactions in order to accurately identify batching.

This improvement in privacy comes at the expense of added latency. In Figure 4.7, we show the cumulative density function of delay added to batched Bitcoin transactions averaged over 5 samples from the privacy-preserving uniform distribution. In general, we can add delay of less than 4 hours to most transactions. For the setting of $\varepsilon = 1$, the mechanism adds delay of under 2 hours to 70% of transactions. While this is slower than a Bitcoin transaction when no privacy mechanism is used, it is still substantially faster than many other means of transferring money,

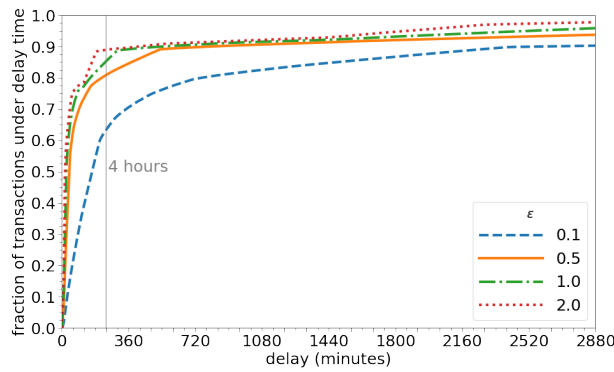


Figure 4.7: Cumulative distribution of delay added to batched Bitcoin transactions (averaged over 5 trials). Delay is drawn from a privacy-preserving uniform distribution with γ set to the median of the inter-arrival time of transactions to an output address within the past 7 days.

like wire transfers. As such, privacy-sensitive users could realistically deploy this algorithm in their Bitcoin wallets to protect the unlinkability of their transactions.

4.1.7 Discussion

This subchapter introduced the problem of anonymity compromise caused by task batching in pseudonymous forums. The empirical analyses show that batching is common in peer review, Wikipedia editing, and cryptocurrency transactions, and that simple timing attacks can use this behavior to link actions that users intended to keep separate. The theoretical results show that standard two-sided differential privacy is too strong for this streaming release setting without fake events or indefinite withholding, while one-sided differential privacy yields useful delay mechanisms.

The main practical tradeoff is that delay changes the user experience of the platform. In the Wikipedia experiments, the zero-inflated uniform mechanism sometimes releases article revisions in a different order than they arrived; at reasonable privacy settings, roughly 10% of revisions were reordered within an article. This can create confusion when there are dependencies between revisions. A similar issue arises in peer review, where comments may respond to one another. Supplemental Material discusses a privacy-preserving queue-based mechanism that outputs delayed comments in arrival order. That mechanism satisfies a different relaxation of differential privacy rather than the (ϵ, γ) -OSDP guarantee, leaving a useful design question: how much ordering structure can be preserved while maintaining a meaningful timing-privacy guarantee?

Deployment also depends on adoption. The privacy guarantee here holds for any pair of events where each event uses the delay mechanism independently of what other users choose to do, so a single privacy-sensitive user can still benefit by applying the mechanism to all of their events. At the same time, widespread adoption may provide additional amplification because more delayed events make timing patterns harder to interpret. Understanding how adoption rates interact with the choice of γ and ϵ is therefore important for translating the formal guarantee into operational guidance. The next subchapter turns from timing metadata to a different release

interface, algorithmic benchmark feedback, where privacy loss arises from reported performance scores rather than publication times.

4.1.8 Deferred Proofs

This section includes the central privacy and optimality proofs for the batching mechanism.

Privacy of Randomized Delay Mechanisms

Proof of Theorem 4.1.4. It is enough to analyze the pair of comments that differs between neighboring inputs. In the batched input, two comments arrive at the same time and both receive independent delays from B . In the neighboring unbatched input, one of the comments is shifted by some $t_0 \in [0, \gamma]$ and the relevant unbatched comment receives delay from U .

Let $S, S' \subseteq \mathbb{R}$ be measurable release-time events for the two comments. By $(\varepsilon/2, \gamma)$ -one-sided indistinguishability,

$$\Pr[B \in S] \leq e^{\varepsilon/2} \Pr[U \in S] \quad \text{and} \quad \Pr[B \in S'] \leq e^{\varepsilon/2} \Pr[U \in S' - t_0].$$

Multiplying the two inequalities gives

$$\Pr[B \in S] \Pr[B \in S'] \leq e^\varepsilon \Pr[U \in S] \Pr[U \in S' - t_0].$$

All other comments have identical arrival times in the two inputs and are randomized in the same way, so their probabilities cancel in the likelihood ratio. Thus every output event has probability under the batched input at most e^ε times its probability under the neighboring unbatched input, proving (ε, γ) -OSDP. \square

One-Sided Indistinguishable Distributions

Proof of Theorem 4.1.5. For distributions with densities b and u , one-sided indistinguishability requires $b(t) \leq e^\varepsilon u(t - t_0)$ for all t and all $t_0 \in [0, \gamma]$, with the convention that $u(s) = 0$ for $s < 0$.

For the shifted exponential pair, $b(t) = 0$ for $t < \gamma$, and for $t \geq \gamma$,

$$\frac{b(t)}{u(t - t_0)} = \frac{\exp\{-\varepsilon(t - \gamma)/\gamma\}}{\exp\{-\varepsilon(t - t_0)/\gamma\}} \leq e^\varepsilon.$$

The staircase case follows from the standard staircase indistinguishability guarantee [66], applied to the one-sided absolute-value distribution with shift γ .

For the uniform pair, b is supported on $[\gamma, \gamma/(1 - e^{-\varepsilon})]$ and u is supported on $[0, \gamma/(1 - e^{-\varepsilon})]$. On the support of b , the density ratio is exactly e^ε in the worst case and is zero off the support. For the zero-inflated uniform pair, the same argument applies to the continuous part: whenever b has positive density, $u(t - t_0)$ is at least the continuous density scaled by $e^{-\varepsilon}$; the point mass at zero only increases the probability assigned by U to sets that include zero. The expectation formulas follow by direct computation from the stated supports and mixture probabilities. \square

Pareto Optimality of the Zero-Inflated Uniform Mechanism

Proof sketch for Theorem 4.1.8. Consider any pair of independent delay distributions (B, U) that satisfies the one-sided indistinguishability constraints. First approximate both distributions by

piecewise-constant densities on intervals of length γ/i . This approximation preserves indistinguishability and converges in expected delay as $i \rightarrow \infty$.

For the piecewise-constant problem, any Pareto-optimal solution must have three structural properties. First, B places no mass below γ ; otherwise it would be distinguishable from an unbatched delay shifted by at most γ . Second, the density of B above γ can be taken to be non-increasing, since moving any higher-delay mass left weakly reduces expected delay without weakening privacy. Third, once B is fixed, U is determined by putting the smallest density required by the likelihood-ratio constraints at positive times and placing any remaining mass at zero, which minimizes unbatched delay.

These properties force the Pareto frontier to be uniform delay for batched comments and zero-inflated uniform delay for unbatched comments. The only remaining degree of freedom is the mixture parameter p . Substituting the resulting expectations into the weighted objective

$$w\mathbb{E}[B] + (1 - w)\mathbb{E}[U]$$

gives a one-dimensional convex optimization problem. Differentiating yields the value of p used in Algorithm 10, clipped at 1 when the unconstrained optimum lies outside the feasible interval. Thus the mechanism minimizes every supporting weighted objective and is Pareto optimal. \square

4.2 Private Benchmarking for Fraud Detection

The previous subchapter studied privacy leakage from timing metadata. This subchapter turns to a different release interface: benchmark feedback. When a data holder evaluates submitted algorithms on private graph data and reports performance, the reported score can itself leak information about the underlying dataset. We study this problem through fraud detection benchmarks, where graph structure is central and standard differentially private evaluation tools face severe utility challenges.

This subchapter is based on joint work with Giulia Fanti, Nihar B. Shah, and Zhiwei Steven Wu [70].

4.2.1 Motivation

Fraud constitutes a pernicious problem across numerous domains, manifesting as fake product reviews, fraudulent payments, and the resale of stolen goods, among other harms [32]. The scale of fraud losses is driven in part by the difficulty of detecting fraud: today, the problem is primarily handled by automated detectors with high false positive and false negative rates. Although many organizations dedicate entire teams to fraud detection, other organizations outsource the development of fraud detection mechanisms to third parties, such as vendors of fraud detection software and/or third-party researchers [37, 110, 176]. However, effective outsourcing requires enterprises to share internal fraud data, which can be challenging due to privacy regulations (e.g., GDPR) and/or the risk of leaking trade secrets through shared datasets. As a result, the lack of publicly shareable data has limited research progress on detection of fraudulent behaviors in privacy-sensitive domains. For example, in scientific peer review, there is a lack of real data on reviewer-paper assignments. This limits researchers' ability to evaluate solutions to the

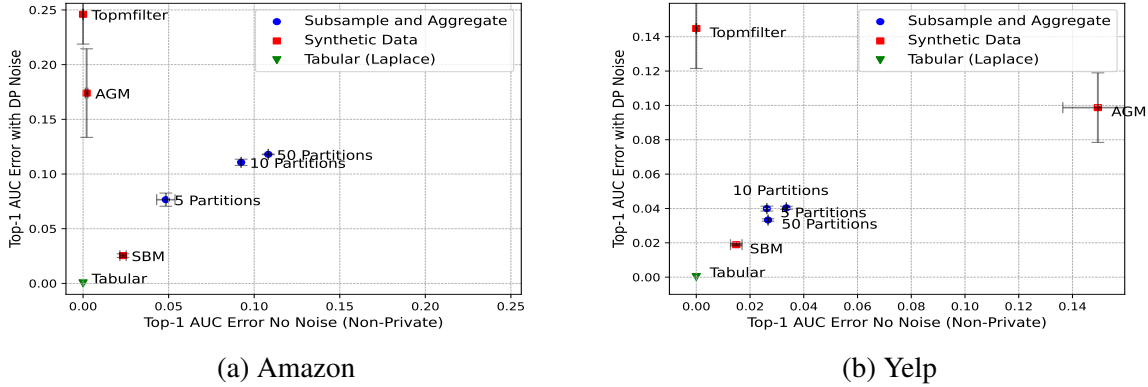


Figure 4.8: Comparison of DP benchmarking methods for releasing the best AUC score among 10 fraud detectors with privacy budget of $\epsilon = 5.0$. The horizontal axis captures error due to inductive bias (i.e., the underlying graph model, without DP noise); the vertical axis captures error including DP noise. More complex synthetic data methods (Topmfilter and AGM) can model the data without privacy, but suffer from high variance due to DP noise addition, undermining utility. Subsample-and-aggregate distorts graph structure extensively, even before adding random noise to outputs. All current methods incur large utility cost on graph data compared to tabular data. Error bars show the standard error of the mean across 10 simulations of each method.

problem of detecting rings of colluding reviewers [84, 128], necessitating reliance on laboratory-generated [88] or semi-synthetic [90] data.

Problem Statement. This subchapter explores a paradigm for outsourcing fraud detection in which sensitive data remains within an organization’s boundaries. Rather than sharing data externally, the organization invites third parties to submit fraud detection algorithms, leveraging domain knowledge and public data, which are then evaluated and ranked by the data holder. These third parties may be incentivized through financial or reputational rewards, such as leaderboard-based competitions [94]. This paradigm has gained traction as a practical solution for enabling public evaluation on privacy-sensitive data. In recent years, it has been adopted in several high-profile applications, including fraud detection for Amsterdam’s social welfare system [127], biometric identification benchmarking in India [207], and facial recognition evaluation by the U.S. National Institute of Standards and Technology (NIST) [153]. We study this setting under two key constraints that have not been jointly explored before:

1. *Private algorithm evaluation:* We observe that if the accuracy of a fraud detector is released directly, it can leak sensitive information about the underlying test data (Section 4.2.2). We therefore consider methods for evaluating algorithms, and releasing their results, under a differential privacy (DP) constraint [47].
2. *Graph-structured data:* Many prominent fraud domains, such as financial fraud or product review fraud, have graph-structured datasets. We focus on fraud detection algorithms (and privacy solutions) that can be applied to graph-structured data.

More precisely, we consider a *benchmarking server*, which has a private graph G consisting

of a set of known fraudulent vertices V_1 (of size n_1) and a set of benign vertices V_0 (of size n_0).

The benchmarking server’s goal is to *evaluate* one or more fraud detection algorithms and *communicate* the result back to the algorithm designers. The benchmarking server receives a fraud detection algorithm \mathcal{A} . The fraud detection algorithm takes as input a vertex v and the entire graph G and outputs $\mathcal{A}(G, v)$ which is a numerical score where a higher score indicates a higher likelihood of fraud. For example, the fraud detection algorithm could score a vertex by its degree. The benchmarking server returns an *accuracy statistic* for the fraud detection algorithm on graph G . Concretely, we consider the *AUC score*, which is defined as:

$$f_{\text{AUC}}(\mathcal{A}, G) = \frac{1}{n_1 n_0} \sum_{v_0 \in V_0} \sum_{v_1 \in V_1} \mathbb{1}[\mathcal{A}(G, v_1) > \mathcal{A}(G, v_0)].$$

The AUC score represents the probability that a randomly chosen fraudulent vertex is scored higher than a randomly chosen benign vertex. It is a commonly used accuracy statistic for class-imbalanced binary classification problems like fraud detection [75].

Challenges and Approach. Existing techniques for differentially private release of statistics cannot be easily applied to graph-structured data (Section 4.2.3). The main challenge is that benchmarking fraud detection algorithms on graph-structured data relies on high-sensitivity queries over the graph, meaning that the query result can change significantly if even a single node’s neighbors are altered in the graph (Definition 4.2.3). Making such algorithms DP requires large amounts of noise, undermining utility.

The goal of this subchapter is to instantiate and benchmark different classes of techniques for evaluating fraud detection algorithms over graph-structured data under a DP constraint. We evaluate two approaches for dealing with high-sensitivity queries: (1) *Subsample-and-aggregate* partitions the dataset into non-overlapping datasets, then evaluates the fraud detectors over each partition. The average accuracy over the partitions is low-sensitivity, and can be released with less noise than without partitioning. (2) *Synthetic graph data* generates a DP copy of the true graph; then, fraud detectors are evaluated on this synthetic graph.

Contributions. Our primary contributions are:

1. We formulate the problem of *differentially private benchmarking of fraud detectors on private graph data*. **We describe a simple privacy attack on a system that benchmarks user-submitted algorithms on private data.** Our attack applies broadly to benchmarking frameworks that evaluate algorithms on the entire dataset at once rather than per-user or instance-level evaluation—such as graph data or authentication systems where access to the full dataset is required for evaluation. **In simulations of a deployed facial recognition benchmarking system, we show that this attack is practical**—concretely, our attack achieves near perfect accuracy in identifying whether individuals’ data is present in a private dataset, with a True Positive Rate of 0.98 at a False Positive Rate of 0.00.
2. We then evaluate the potential of differential privacy as a solution concept for preserving privacy of graph data used in a benchmarking system. Across methods, **we observe a severe trade-off between bias introduced by distorting the graph and noise required to**

compensate for computing high sensitivity statistics on the graph. This result is captured in Figure 4.8, which shows the error in privately benchmarking the best AUC score among a set of 10 fraud detectors. We plot the error of each DP benchmarking method without noise added (inductive bias) against error after adding noise to ensure differential privacy. Among synthetic data methods, more complex methods (TopmFilter and AGM) have lower inductive bias, but much higher noise addition to preserve privacy than the simpler SBM. Subsample-and-aggregate tends to distort graph structure extensively, even before adding random noise to outputs, but then has low additional error from the random noise. All current methods to satisfy DP on graph data incur large utility cost compared to tabular data.

3. To explain these results, we conduct detailed ablations on both subsample-and-aggregate and synthetic data methods. While these methods introduce inductive bias in different ways, they exhibit a similar trade-off — the less we bias our graph representation, the more noise we must add to satisfy DP.

Our code is available at https://github.com/akgoldberg/private_fraud_benchmarking.

4.2.2 Benchmark Privacy Risks

We start by describing an attack that a malicious actor can use to compromise the privacy of an individual included in the dataset used for algorithmic benchmarking.

Encoding Private Queries in Benchmark Submissions

Consider a bad actor who wishes to answer a binary query, such as whether an edge exists between two vertices in the graph. The adversary needs three capabilities. (1) *An accurate fraud detector*: for example, a known algorithm from the literature which does better than random ($AUC > 0.5$). (2) *An inaccurate fraud detector*: for example, scoring vertices at random (expected $AUC = 0.5$). (3) *The ability to identify vertices in G* : this depends on what information the private server gives to the fraud detection algorithm.⁷ In many cases, G may include extensive metadata per vertex, which makes it easy to identify vertices. Even without metadata, there are many de-anonymization attacks leveraging only the graph structure (see [91] for a survey), which enable an adversary to identify vertices.

⁷For a general binary query, the attacker would need an accurate estimator for that query given the private graph dataset.

Algorithm 12 Attacker’s Submission to the Benchmarking Server

Require: Accurate fraud detector \mathcal{A} , pair of vertices v_1, v_2 .
Check if an edge exists between v_1 and v_2 in private graph G .
if edge (v_1, v_2) exists **then**
 Run accurate fraud detection algorithm \mathcal{A} on G .
else
 Return a random fraud label for each vertex in G .
end if

The adversary submits a “fraud detector” to the benchmark described in Algorithm 12. The adversary identifies the relevant pair of vertices, and runs the accurate fraud detector if an edge exists between the vertices or an inaccurate fraud detector otherwise. If the adversary observes a high AUC score, they learn that an edge exists, while if they observe a low AUC score they learn that the edge does not exist. In effect, the benchmark allows a malicious actor to answer any binary query on G by encoding the query as either a high-accuracy or low-accuracy fraud detector. The privacy attack we describe above applies to a range of benchmarking systems that evaluate user-submitted algorithms on private data, as demonstrated in the next section.

Attacking a Real-World Benchmark

To demonstrate the practical implications of the attack, we take an existing privacy-sensitive application as a case study: the National Institute of Standards and Technology (NIST) Face Recognition Technology Evaluation (FRTE). The FRTE benchmarks algorithms for facial recognition on sensitive private datasets of face images like mugshots, visa applicants, and border-crossing photos. The FRTE benchmarks the task of “1:N face identification”. A 1:N face identification algorithm matches a given “probe image” against a large “gallery dataset” of images, returning an image in the gallery of the same person. While the dataset consists of face images, not a graph, the same attack proposed for graph data can be applied to the face recognition benchmark system.

Specifically, an attacker with access to a reasonably accurate facial recognition algorithm can exploit the benchmarking system to determine whether one or more specific individuals’ faces are included in the gallery dataset; for example, suppose that an attacker wants to know if Bob is in the gallery dataset. The attacker obtains an image of Bob. When the attacker’s submission searches the gallery dataset for a given probe image, the attacker first uses the accurate facial recognition algorithm to check if Bob’s image matches any image in the gallery dataset. If yes, they use the accurate algorithm on the *actual probe image* (i.e., not Bob). If Bob is not in the gallery, they use the inaccurate algorithm on the probe image. A high accuracy score on the benchmark implies Bob’s presence, while a low score indicates his absence.⁸

⁸We have disclosed this vulnerability to NIST, which has implemented steps to reduce its exploitability. Moreover, note that this attack only reveals membership in the dataset—it does not reveal other information about the individuals in the gallery, such as date of the photo or biographic information.

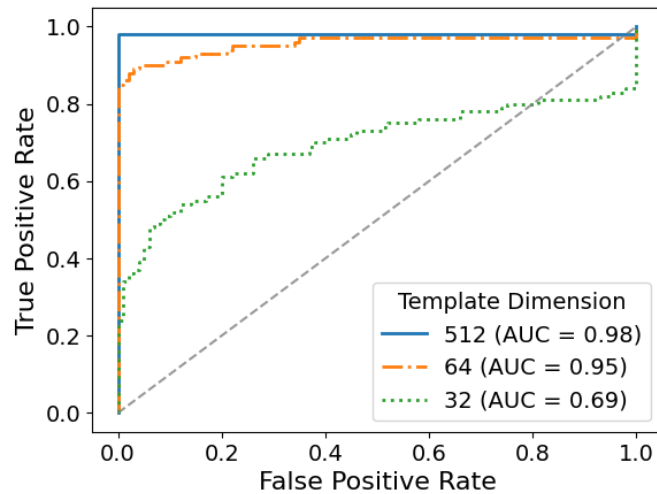


Figure 4.9: ROC curve of the privacy attack on NIST’s FRTE benchmark, simulated on the CelebA dataset.

Attack Effectiveness

To evaluate the practical viability of the attack, we simulate the 1:N face recognition benchmark using the publicly available CelebA dataset, which contains faces of over 10,000 celebrities [133]. We use an open-source, deep learning-based facial recognition model ArcFace [38] as the accurate model. The attacker uses the face recognition model to generate embeddings (templates) of images and then performs “identification” using cosine similarity between embeddings. We vary the adversary’s capabilities by reducing the dimension of the embeddings used by the model from 512 to 64 and 32. Then, true positive and false positive rates of the attack can be computed by varying the threshold at which the attack concludes that an attack image is present in the private data. In Figure 4.9, we show the ROC curve of the attack. **Using 512-dimensional embeddings, the attack achieves a TPR of 0.98 at an FPR of 0.00, successfully identifying 98 out of 100 gallery members while avoiding false matches.** Even with 64- or 32-dimensional embeddings, the attack remains effective, achieving high AUC scores. This result highlights that the attack is feasible even on the FRTE benchmark using simple, open-source facial recognition models.⁹

The FRTE privacy vulnerability demonstrates how publicly releasing algorithmic benchmarking results can compromise the anonymity of individuals in private datasets. In the remainder of this subchapter, we focus on benchmarks where the private data used for evaluation is a graph, as graph datasets are common in fraud detection tasks and pose significant technical challenges.

⁹Our code and a detailed FRTE benchmark description are available at <https://github.com/akgoldberg/face-recognition-privacy-attack>.

4.2.3 Problem Formulation

Based on the privacy risk posed by the attack in Section 4.2.2, we ask how to protect dataset privacy for the benchmarking server. We consider three different operating modes for the benchmarking server: (1) *one-shot*: the server releases the AUC score for a single submitted fraud detector, (2) *full leaderboard*: the server returns the AUC score for a set of submitted fraud detectors, and (3) *top-1 release*: the server releases the best-performing fraud detector among a set of submitted algorithms.

Incorporating Differential Privacy

We propose that the fraud benchmarking server satisfy a relaxation of differential privacy (DP) [47] that protects benign vertices:

Definition 4.2.1 (Protected differential privacy [101]). *Two graphs G, G' are neighboring if:*

1. *G and G' share the same partitions of fraudulent and non-fraudulent vertices V_1 and V_0 .*
2. *G can be obtained from G' by rewiring the edges of one benign vertex and/or changing that vertices' metadata.*

Let f denote the benchmarking server that given a graph and fraud detector outputs an estimate of the AUC score. The server f satisfies ϵ -protected differential privacy if for any two neighboring graphs G, G' , any fraud detection algorithm \mathcal{A} and any possible set of outputs \mathcal{O} :

$$\Pr[f(G, \mathcal{A}) \in \mathcal{O}] \leq e^\epsilon \Pr[f(G', \mathcal{A}) \in \mathcal{O}].$$

The definition of protected DP is identical to the standard definition of differential privacy, except in how “neighboring” graphs are defined. Specifically, Standard DP allows G and G' to differ in the data of *any* vertex in the graph, not just a benign vertex. Protected DP is a relaxation of standard DP in that any graphs that are neighbors per the definition of protected DP are also neighbors under standard DP. Any mechanism that satisfies standard DP also satisfies protected DP. We will drop “protected” and refer to protected differential privacy as DP for brevity throughout.

We primarily adopt this relaxed notion of privacy to improve utility. In many real-world graphs, the rate of fraud is low. Hence, requiring that the released accuracy statistic does not change much if we change the connections of these fraudulent vertices makes it difficult to release high-fidelity benchmarks. Still, we believe this relaxation is useful. In fraud detection, it is natural to hold different privacy expectations for fraudulent participants (many of which may even be fake [89]) compared to legitimate ones. For example, in online review platforms or social networks, fraudsters can correspond to non-human bots who do not have the same privacy status as humans. Second, in domains where ground truth for fraud does not currently exist, a common approach is to inject simulated fraudulent vertices into a real graph of benign vertices [89]. As these fraudulent vertices are simulated, they do not require privacy protection. Finally, we note that the benchmarking server can still provide baseline protections for fraudulent vertices, like not exposing personally identifiable information (PII), while guaranteeing stronger DP protection for benign vertices.

In the definition of neighboring graphs, we adopt the strong notion of *node* differential privacy, which protects all of the edges of any single benign vertex. Many prior works employ a

weaker notion of *edge* differential privacy [28, 92, 97, 155], which defines neighboring graphs as graphs that differ in a single edge. Node-DP gives a much stronger guarantee. In particular, the unit of participation in the dataset is at the vertex-level (i.e., whether an Amazon user’s reviews are included in the dataset or not) and node-DP promises that the inclusion of a given individual in the dataset does not reveal much information about them. We note that protected DP inherits the *composition property* of standard DP:

Theorem 4.2.2 (Composition [47]). *For any two fraud detectors \mathcal{A}_1 and \mathcal{A}_2 , if releasing $f(\mathcal{A}_1, G)$ satisfies ε_1 -protected DP and releasing $f(\mathcal{A}_2, G)$ satisfies ε_2 -protected DP, then releasing both results on graph G , $(f(\mathcal{A}_1, G), f(\mathcal{A}_2, G))$ satisfies $(\varepsilon_1 + \varepsilon_2)$ -DP.*

This property is helpful in moving from one-shot release of fraud detectors to releasing a leaderboard of many fraud detectors.

Challenges of Graph Data

Even evaluating a single fraud detector on graph data proves challenging under DP constraints. To understand why, let us compare our setting to evaluating a fraud detector on tabular data. Evaluating a single fraud detector can be seen as a problem of releasing a (noisy) query result. A simple mechanism that solves the query release problem adds random noise with variance scaled to the “sensitivity” of this query, which is defined as follows.

Definition 4.2.3 (Global Sensitivity). *For a query $f : \mathcal{X} \rightarrow \mathbb{R}^d$, define its global sensitivity*

$$\Delta_f = \max_{G, G' \text{ neighbors}} \|f(G) - f(G')\|_1$$

as the worst-case change in f across any two neighboring graphs.

Then, a canonical mechanism, termed the Laplace Mechanism, scales noise to the global sensitivity:

Definition 4.2.4 (Laplace Mechanism [47]). *On any input G the Laplace Mechanism with privacy parameter ε releases*

$$\tilde{f}(G) = f(G) + \text{Laplace}(\Delta_f/\varepsilon).$$

The Laplace Mechanism satisfies ε -DP.

In the tabular setting, model evaluation is a low sensitivity query and therefore can be released by directly applying the Laplace mechanism. Consider a simple case where fraud detector \mathcal{A} is a fitted logistic regression model (the weights of the model are fixed). Changing any row of a tabular dataset only changes the features of that row and hence changes at most a single fraud prediction score. Therefore, when evaluating the fixed model on tabular data, the AUC score can only change by $\frac{1}{n_0}$. The Laplace mechanism can then release the true AUC score of the fraud detector plus Laplace noise with variance $\frac{2}{(\varepsilon n_0)^2}$.

In contrast, consider evaluating the logistic regression model on a graph where features of each vertex include graph statistics like the degree of each vertex. Because features of each vertex depend on other vertices, changing any one vertex can change features of all other vertices in the graph. In the worst-case, changing a vertex changes fraud prediction scores for all other vertices in the graph, so the AUC score has a large global sensitivity of 1. As this is the largest possible value AUC can take, the Laplace mechanism must add so much noise that the entire signal is lost.

This subchapter focuses on addressing the high sensitivity of model evaluation on graph data. In cases where queries of a dataset have large worst-case sensitivity there are three classes of solutions in the DP literature:

1. (*Subsample-and-aggregate*) Force low sensitivity of the AUC score by applying “subsample-and-aggregate.” Partition the vertices of the graph into k equally sized subsets, compute AUC score on each partition and then, directly release the average AUC score across partitions plus noise.
2. (*Synthetic data*) Generate DP synthetic data that captures some structure of the private graph and run fraud benchmarking on this private graph data.
3. (*Calibrate noise to “local sensitivity.”*) Estimate (an upper bound) on how sensitive f_{AUC} is on the specific graph and fraud detection algorithm \mathcal{A} and calibrate noise to this sensitivity, which may be much lower than the worst case global sensitivity. This approach includes mechanisms like Propose-Test-Release, Smooth Sensitivity, and the Inverse Sensitivity Mechanism [48, 157] as well as recent work on privatizing black-box scripts run on private data [106].

We give instantiations of subsample-and-aggregate and synthetic data generation algorithms tailored to the benchmarking server setting and run extensive empirical evaluations to understand opportunities and shortcomings. We do not evaluate local sensitivity based methods, because these approaches are computationally infeasible in our setting as they would require enumerating every possible neighboring graph and evaluating fraud detectors on these graphs to estimate a bound on local sensitivity. The rest of the subchapter gives background on private model evaluation and arbitrary-code execution under DP constraints, instantiates the subsample-and-aggregate framework, describes the synthetic graph methods used as baselines, and then evaluates these methods empirically.

4.2.4 Background

To our knowledge, this work is the first to consider the problem of model evaluation on graph data under differential privacy constraints. The closest background comes from private model evaluation for tabular data, private model selection, arbitrary-query systems, and differentially private graph analysis. One line of work (starting with [175]) proposes a framework of “verification servers” wherein analysts fit a model of data (e.g., a linear regression model) on a synthetic dataset and then employ a “verification server” which holds non-synthetic data to perform quality checks that their model is useful like goodness-of-fit tests. In subsequent work, they show how to design such a verification server for the special case of assessing quality of fit of a linear regression model [237] and build a prototype for data on US government employees [11]. This work validates the usefulness of a “synthetic data plus verification server” model for allowing public use of private datasets. However, these works focus on tabular data rather than graph data, which poses specific challenges as we detail in Section 4.2.3.

A recent line of work in DP machine learning (starting with [130] and extended in [31, 166]), looks at a closely related problem of model selection under differential privacy constraints. This work assumes the ability to train and evaluate a single model with ϵ -DP, which is reasonable for the tabular-data setting where algorithms like DP-SGD are effective at model training and

the Laplace mechanism (or Gaussian mechanism) solves model evaluation. These works focus on choosing the (nearly) optimal model in minimizing loss among a large set of models without paying for privacy loss that grows with the number of models. Here, we observe that on graphs, even ignoring model training, the seemingly straightforward step of one-shot model evaluation is difficult under differential privacy constraints. The methods for private selection could be applied in conjunction with our proposals for one-shot DP release.

A number of works have considered the problem of running arbitrary queries on private data. Subsample-and-aggregate, first proposed in [157], is one popular method for reducing the sensitivity of a query. A practical instance of this framework was implemented in GUPT [147], which allows researchers to run arbitrary scripts on private data using subsample-and-aggregate. That paper shows that their system can enable researchers to fit models like k -means clustering and logistic regression on a (tabular) dataset of chemical compounds and achieve utility close to non-private for privacy budgets as small as $\epsilon = 2$. More recently, subsample-and-aggregate has been applied in the context of training machine learning models [167] via the “PATE” framework. PATE trains a “teacher” model on each partition of data, then labels an unlabeled public dataset using an aggregation of predictions from each teacher model, and finally trains a model on this dataset. This subchapter focuses on model evaluation rather than training, as the model evaluation task is already challenging in the graph setting. We observe a number of distinct difficulties in applying subsample-and-aggregate to benchmarking fraud detectors on graphs. First, subsampling a *graph* often introduces significant bias to the estimates of graph statistics on each subset, which changes assessments of the best choice of the number of partitions to use. Second, in fraud detection problems there are often very small numbers of fraudsters. Hence, subsampling these fraudsters can lead to very few per partition and poor utility. We propose upsampling fraudulent entities satisfying a relaxed notion of privacy to address this issue. We then perform extensive empirical evaluation to understand how the subsample-and-aggregate framework compares to synthetic data generation algorithms for this problem.

A recent work [106] also considers the problem of running arbitrary code on a private dataset and gives a new mechanism called TAHOE that finds a subset of the data on which the script is “stable.” They compare against subsample-and-aggregate on simple queries like producing a histogram of the data and find their method is competitive in accuracy. However, TAHOE is computationally expensive and can only be run efficiently when the dataset can be expressed as a histogram of finitely many values, which is inapplicable to graph data.

There is a long and rich line of work in differentially private analysis of graph data. We discuss the literature on generating synthetic graphs in more detail in Section 4.2.6 where we detail our choice of synthetic graph algorithms to benchmark. These synthetic graph algorithms require estimating statistics of the graph (like degree distribution or number of triangles) under node differential privacy. This introduces new challenges as the prior works on synthetic data generation use the weaker notion of edge-DP to estimate statistics. We generically transform edge-DP estimation into reasonably accurate node-DP estimation using the idea of smoothly projecting a graph to the space of limited-degree graphs from [18, 100]. There may be additional improvements in applying existing synthetic data generation methods under the node-DP privacy regime by applying more tailored estimation procedures for specific graph statistics.

4.2.5 Private Evaluation by Subsample-and-Aggregate

The first approach we consider in privatizing fraud benchmarking is the subsample-and-aggregate framework [157]. Recall from Section 4.2.3 that a key challenge of releasing a DP estimate of the AUC score of a fraud detector on a graph is that this query has global sensitivity of 1, equal to the range of the AUC score. Subsample-and-aggregate forces low sensitivity of the query by partitioning the dataset into k disjoint sets and estimating AUC on each partition.

Our algorithm follows the template described above for benign vertices, that is, we partition the benign vertices into k disjoint sets of equal size. However, in fraud graphs, there are often very few fraudulent vertices. For example, in the Elliptic Bitcoin financial fraud dataset [225] there are only 11 fraudsters out of over 6,000 vertices. Partitioning these fraud vertices into a reasonable number of partitions to achieve low sensitivity (say $k \geq 5$) would destroy any structure of the sub-graph of fraud vertices.

We therefore modify typical subsample-and-aggregate for the fraudulent vertices by allowing *duplication* of fraudsters across partitions. For each partition, we sample a subset of fraudulent vertices, where the rate of sub-sampling is controlled by a parameter ρ . We term this instance of subsample-and-aggregate as Partition, Duplicate, and Aggregate (PDA), described in Algorithm 13. Note that taking $\rho = 1$ results in duplicating all fraud vertices in each partition, while taking $\rho = \frac{1}{k}$ is similar to typical subsample-and-aggregate, but with the difference that fraudulent vertices may be sampled into multiple partitions.

Algorithm 13 Partition, Duplicate, and Aggregate

Parameters: privacy parameter $\varepsilon > 0$, number of partitions k , fraud sub-sampling rate ρ .
Input: fraud detector \mathcal{A} , accuracy statistic f with global sensitivity Δ , fraud vertices V_1 , benign vertices V_0 , graph G on vertex set $V_0 \cup V_1$.
 Randomly partition benign vertices V_0 into k equally sized sets $V_0^{(1)}, \dots, V_0^{(k)}$.
for $i \in [k]$ **do**
 Sample $V_1^{(i)}$ uniformly from all subsets of V_1 of size $\rho \cdot |V_1|$.
 Let G_i be the subgraph of G induced by $V_0^{(i)} \cup V_1^{(i)}$.
end for
 Sample $Z \sim \text{Laplace}(\Delta/(k\varepsilon))$.
 Release $Z + \frac{1}{k} \sum_{i=1}^k f(\mathcal{A}, G_i)$.

It is straightforward to prove that Algorithm 13 guarantees differential privacy:

Proposition 4.2.5. *For any choice of sub-sampling rate $\rho \in (0, 1)$, number of partitions $k > 1$ and privacy parameter $\varepsilon > 0$, Algorithm 13 guarantees ε -Protected Differential Privacy (Definition 4.2.1).*

The proof follows from a standard proof of privacy for subsample-and-aggregate: changing the data of any benign vertex impacts at most 1 of the k partitions between any two neighboring graphs, and the accuracy score on this partition can change by at most 1 since f has global sensitivity (Definition 4.2.3) of 1. Hence, the mean across partitions has global sensitivity of $\frac{1}{k}$ and privacy follows from the Laplace mechanism (Definition 4.2.4). We note that in practice, the subsample-and-aggregate framework does not introduce substantial computational overhead.

We further develop intuition around the error of partition, duplicate, and aggregate as a function of number of partitions and subsample-rate via a simple example in Supplemental Material.

Runtime

We observe in experiments that subsample-and-aggregate often significantly speeds up evaluation of fraud detectors. The reason is that Algorithm 13 requires running the same fraud detector on k partitions of the dataset, each of size roughly $\frac{1}{k}$ the original number of vertices in the dataset (in fact, slightly larger due to duplication of fraud vertices). In many cases, the runtime of a fraud detector actually decreases by a factor of more than $\frac{1}{k}$ per partition. This can happen for two reasons. First, many graph algorithms are polynomial in the number of vertices in the graph (for example, an algorithm that cubes the adjacency matrix to compute number of triangles per vertex). Hence, the partitioning gives a polynomial $\frac{1}{k}$ improvement in total computational cost. Second, partitioning can only decrease the total number of edges across all partitions since no edge can be duplicated in multiple partitions. Therefore, any algorithm with runtime dependent on number of edges is faster when run on all the partitioned graphs rather than the original graph. In addition, Algorithm 13 can be parallelized by running the fraud detector on each partition separately, which may make it easier for the benchmarking server to efficiently execute submitted fraud detectors in practice.

Running Multiple Private Benchmarks

Algorithm 13 provides a method for one-shot release of the AUC score of a single fraud detector. In order to apply it to full leaderboard release, we can invoke composition (Theorem 4.2.2) and subdivide the privacy budget among many fraud detectors. For example, if we have 10 algorithms to benchmark, we run each with privacy budget of $\varepsilon/10$ per fraud detector. As it is harder to provide good utility for smaller ε , we expect our accuracy of estimation to degrade in the number of detectors benchmarked.

In many real-world settings it is useful to release only the best or the top- m fraud detectors, for example when running a competition. In the case of top-1 release, we can use the *Report Noisy Arg Max* mechanism [51]. This mechanism adds Laplace noise to any (finite) number of queries as per the Laplace mechanism, but then only releases the name of the query with the largest (noisy) value. Rather than paying composition cost that grows in the number of queries, this procedure is ε -DP. In our case, then, we can apply Algorithm 13 to arbitrarily many fraud detectors and then at the end only publicly release the name of the detector with the highest noisy AUC score. This guarantees ε -DP when each run of Algorithm 13 is run using privacy parameter ε . While we do not experiment with releasing the top- m fraud detectors, recent work [171] shows that releasing the top- m fraud detectors ranked by noisy AUC (among a larger set of fraud detectors) only incurs total privacy loss of $m\varepsilon$.

4.2.6 Private Evaluation by Synthetic Data Generation

In this section, we describe our choice of synthetic graph data generation algorithms to benchmark; the surveys [82, 122] provide a useful overview of such algorithms. Many methods do

not handle *labeled* vertices. Such methods cannot be applied to our problem, as synthetic data for fraud detection benchmarking needs to differentiate between fraudulent and benign vertices. Additionally, most existing work focuses on satisfying the weaker notion of edge-level DP, while we wish to satisfy node-level DP. Therefore, we focus on the following 3 methods that all handle labelled vertices and are amenable to transformation into a node-level DP algorithm:

1. *Stochastic block model (SBM)*: Estimate a stochastic block model with two communities (fraud and non-fraud) and sample a graph based on the SBM parameters. For a fixed number of benign and fraud vertices, the stochastic block model has three parameters p_1, p_0, p_{01} . Each edge in the graph is sampled independently at random with probability p_1 if both of its endpoints are fraudulent, p_0 if both are benign, and p_{01} if one is fraudulent and the other is benign.
2. *Attributed social graph (ASG)*: [92]: Estimate the connection probabilities with and between fraud and non-fraud vertices (as in the SBM), but additionally estimate number of triangles in the graph and the degree sequence of the graph. Then, sample a graph that matches these noisy statistics. We run two versions of this method, with and without the triangle statistic.
3. *Top- m -filter* [155]: Directly perturb the adjacency matrix of the graph. In particular, flip each edge in the graph and then perform a filtering step to remove edges to match a noisy estimate of total number of edges.

In general, synthetic data methods first compute graph statistics under differential privacy, which provide a succinct representation of the graph, and then generate the synthetic graph based on these (noisy) statistics. More expressive graph models may better represent the graph structure, but tend to require the estimation of noisier sufficient statistics due to differential privacy. We choose methods that lie along this spectrum of model complexity. On one end, the *SBM* represents a simple model of the graph, with statistics that can be accurately estimated under differentially private. On the other end, *Top- m -filter* attempts to release the full adjacency matrix, which requires large relative noise addition to each entry to guarantee privacy, but best represents the graph structure if privacy were not a concern.

We briefly discuss other popular synthetic graph models considered in the literature. One approach uses exponential random graph models (ERGMs) to model vertex-labelled graph data [98, 129]. These methods are difficult to scale to graphs of more than a few hundred vertices, and prior empirical evaluations are limited to graphs of this size. Hence, they are not applicable to the types of fraud graphs we consider which are larger by an order of magnitude. Recent work [236, 239] has considered using Graph Neural Networks (GNNs) to generate synthetic graph data from graph statistics. They find that directly using DP-SGD (stochastic gradient descent) to train the GNN leads to poor utility. However, it is possible to obtain useful synthetic data by computing vectors of vertex-level graph statistics (like histograms of triangles and 2-paths) under edge-level DP. Estimating these sub-graph histogram statistics under *node-level DP* requires much larger noise addition. For instance, even assuming that a graph has no vertices of degree greater than T , the triangle histogram has sensitivity of greater than T^2 , while we would expect most vertices to participate in far fewer than T triangles. Because node-level DP would require much more noise than the experiments in those papers, which add noise scaled to $\frac{1}{\epsilon}$, we do not focus on these methods here. Finally, some methods incorporate a community detection step [28, 238] that first

clusters vertices of the graph and then estimates connection probability parameters between these clusters to incorporate into the graph generative model. It is unclear how to make this clustering step satisfy node-level DP with reasonable utility.

Guaranteeing Node-Level Differential Privacy. The algorithms we consider were designed to provide edge-level differential privacy. In privately computing sufficient statistics of the graph, these algorithms add Laplace noise proportional to the worst-case sensitivity of a statistic to the change of a single edge in a graph. In order to guarantee node-level privacy in this noise addition step, we use the idea of projecting the graph to the space of graphs with bounded maximum degree from [18, 100] and then adding noise proportional to this “restricted sensitivity.” For a given graph G , choice of truncation threshold T , and graph statistic g , the full workflow is:

1. (Naive truncation). Truncate graph G by removing all vertices with degree above D .
2. Estimate the “smooth sensitivity” S of the naive truncation operation per [100].¹⁰
3. Add Laplace noise with scale proportional to $S \cdot RS_T(g)$ where $RS_T(g)$ represents the “restricted sensitivity” of g on graphs of max degree D , that is the maximum change in g between any two node-adjacent graphs of max degree T .

We summarize the framework for node-private synthetic data release in Algorithm 14. Since the max degree and average degree of the fraud graphs used (see Table 4.2) tends to be much smaller than the number of vertices in the graph, the restricted sensitivity tends to be much lower than the global sensitivity. For example, the global sensitivity of the number of edges in the graph is $n_0 - 1$, while the restricted sensitivity at threshold T is only T . We test synthetic data generation algorithms for a variety of choices of threshold.

Note that using this method with Laplace noise actually guarantees the relaxation of (ϵ, δ) -differential privacy due to the use of “smooth sensitivity” [157]. We fix δ to 10^{-8} for all experiments on synthetic data methods. Additionally, to provide a fair comparison against our subsample-and-aggregate method which relaxes privacy for fraudulent vertices, we compute statistics that rely only on the fraudulent nodes without noise.

Algorithm 14 Framework for Node-Private Synthetic Data Release

Parameters: privacy parameters $\epsilon > 0, \delta \in (0, 1)$, degree threshold T

Input: fraud vertices V_1 , benign vertices V_0 , graph G on vertex set $V_0 \cup V_1$, vector of sufficient statistics $g(G)$ with restricted sensitivity Δ_T .

Remove all benign vertices from G with degree greater than T .

Compute the β -smooth sensitivity $S_{trunc}^\beta(G, T)$ of the truncation operation, where $\beta = \frac{2\epsilon}{\log(1/2\delta)}$.

Sample $Z \sim \text{Laplace}(2S_{trunc}^\beta(G, T) \cdot \Delta_T / \epsilon)$.

Release noisy statistics $\tilde{g}(G) = g(G) + Z$.

Sample synthetic graph \tilde{G} based on $\tilde{g}(G)$.

¹⁰From [100], Proposition 6.1 we can compute the smooth sensitivity $S_{trunc}^\beta(G, T)$ of the truncation operation as follows. Let $N_t(G, T)$ denote the number of benign vertices with degree in range $[T - t, T + t + 1]$ and $C_t(G, T) = 1 + t + N_t(G, T)$. Then, $S_{trunc}^\beta(G, T) = \max_{t \geq 0} e^{-\beta t} C_t(G, T)$.

4.2.7 Benchmarking Experimental Setup

In the following section, we describe our datasets, fraud detectors, and metrics used in empirical evaluations.

Datasets

	Yelp	Amazon	Peer Review	Elliptic
Vertices	11,473	11,944	2,483	6,621
Edge Density (%)	0.41	6.17	0.77	0.04
Num Fraud	1,657	821	22	11
Max Degree	236	6,991	255	47
Mean Degree	47.45	736.50	19.12	2.51

Table 4.2: Graph test datasets

We test methods for fraud benchmarking on 4 datasets representing a variety of domains and graph structures. All graphs are undirected unipartite graphs. In *Yelp* [43] and *Amazon* [43] each vertex represents a reviewer with edges denoting common reviews on the products/restaurants and fraudulent reviewers represent spammers and low-rated reviewers respectively. *Peer Review* consists of paper reviewers at a computer science conference with edges denoting mutual bids on each other’s papers [232]. Following [89] we inject a clique of 22 fraudulent reviewers with edge density of 0.8 among these reviewers into the graph, which corresponds to the smallest injected clique that was possible to detect in prior work. Finally, in *Elliptic* [225] each vertex in the graph represents a transaction from the Bitcoin blockchain, an edge represents a flow of Bitcoins between one transaction and the other, and fraudulent nodes are illicit transactions. We take a single time-step from the entire Elliptic graph (summary statistics in Table 4.2).

We run analyses of subsample-and-aggregate and synthetic data algorithms on validation datasets to understand settings of hyperparameters before comparing these methods against each other. We use four validation datasets. For Yelp, we use a random split of the vertices with 11k vertices in the test set and 11k in the validation set. For Elliptic, we use different disjoint time periods for validation and test. For Amazon and Peer Review, there are not standard train-test splits used in past work. We therefore use the entire graph for evaluation, and generate validation graphs to set hyperparameters by estimating parameters of a stochastic block model (SBM) and sampling from this model.

Fraud Detectors

We evaluate 10 simple fraud detectors that do not require learning, and 3 detectors that learn on a subset of the benchmark data. Specifically we evaluate the following fraud detectors:

- *(Negative) Degree [87]*: rank by the degree of each vertex.
- *(Negative) Clustering Coefficient*: rank by the clustering coefficient of each vertex, inspired by [3].
- *SVD Error [87]*: take the singular value decomposition of the adjacency matrix to obtain a low rank approximation (for specified rank r). Then, rank each vertex by reconstruction error (aggregating over edges by taking either the sum or the max over edges). We use $r = 10$ for the sum and $r = 50$ for the max, chosen to maximize average AUC across all datasets in a grid search.
- *Community Detection*: run Leiden community detection [202] to place cluster vertices in a cluster and rank by cluster size (with larger clusters less likely to be fraudulent).
- *Aggregations*: take weighted averages of the (normalized) scores or maximum scores obtained from subsets of the prior methods.
- *GraphSAGE (SAGE) [78]*: learns a function that aggregates embeddings from neighboring nodes to learn a node embedding for each vertex and uses these embeddings to predict fraud labels.
- *Graph Convolutional Network (GCN) [103]*: learns node representations by applying a layer-wise convolutional operation that aggregates and normalizes features from immediate neighbors.
- *Graph Attention Network (GAT) [215]*: employs attention mechanisms to weight the influence of neighboring nodes during aggregation.

The last three detectors involve a learning component; we trained each GNN on a random 80% of vertices and then assessed AUC score on the held-out 20%. These algorithms give a wide range of AUC scores on each dataset. For example, on Yelp, GCN performs the best with an AUC score of 0.73 and SVD Error (Sum) performs poorly with an AUC score of 0.34. In contrast, on Peer Review, SVD Error (Sum) performs the best with an AUC score of 0.88, while Neg Degree has very bad performance with AUC of 0.12.

Measuring Utility

We consider three metrics to compare utility across methods. Each metric corresponds to one of the release modes for the benchmarking server: one-shot, full leaderboard and top-1 release. Let $\{\mathcal{A}_i\}_{i=1}^m$ denote a set of m fraud detectors to benchmark on graph G , $f_{\text{AUC}}(\mathcal{A}_i, G)$ denote the true AUC score for fraud detector i on G and $\tilde{f}_{\text{AUC}}(\mathcal{A}_i, G)$ denote the noisy DP estimate of the AUC score. For the one-shot release, where we wish to release AUC for a single fraud detector, we calculate *L1 error*: $|f_{\text{AUC}}(\mathcal{A}_i, G) - \tilde{f}_{\text{AUC}}(\mathcal{A}_i, G)|$.

When evaluating top-1 release of the best fraud detector among a set of fraud detectors we measure utility by the distance between the true AUC of the true best fraud detector (computed without any privacy) and the true AUC of the released best fraud detector. That is, we define top-1 error as:

$$f_{\text{AUC}}(\mathcal{A}_{top}, G) - f_{\text{AUC}}(\mathcal{A}_{top'}, G) \text{ where } top = \sigma^{-1}(1), top' = \tilde{\sigma}^{-1}(1).$$

For the full leaderboard setting we use the weighted Kendall-Tau distance between rankings [112]. We discuss this metric and provide results in this setting in Supplemental Material.

In addition to using the AUC score as an accuracy metric, we evaluate the F1 score of fraud detectors, another popular measure of fraud detector accuracy. We find similar results to that of AUC score, but F1 score tends to be even more difficult to release accurately. These additional results are included in Supplemental Material.

4.2.8 Empirical Comparison of Private Benchmarking Methods

In this section we provide results of our comparison of Subsample-and-Aggregate and Synthetic Graph Generation (4.2.8), and experiments to understand trade-offs between distorting graph structure and adding noise to preserve privacy for Subsample-and-Aggregate (4.2.8) and Synthetic Graph Generation (4.2.8).

Comparison of Algorithms

We benchmark subsample-and-aggregate against synthetic data algorithms for the concrete task of releasing the best fraud detectors among a set of fraud detectors. We choose parameters of subsample-and-aggregate (number of partitions and sub-sampling rate) based on the best parameters for each dataset in releasing a ranking of all fraud detectors on the validation dataset. This follows prior work [147], which assumes access to public datasets from which one could estimate subsample-and-aggregate hyperparameters.

In Figure 4.8, we show results of the DP benchmarking methods for top-1 release. We decompose the error into *inductive bias* from how a method distorts graph structure, and error from the addition of *privacy-preserving random noise*. To visualize this, we plot each method without privacy-preserving noise on the x-axis and with noise needed to preserve privacy on the y-axis. Specifically, for Non-Private subsample-and-aggregate we only apply the graph partitioning and do not add Laplace noise to the AUC score. For non-private synthetic data methods we compute sufficient statistics for each method without any noise addition and then generate a graph using those sufficient statistics. Among synthetic data methods, Topmfilter has no error without privacy as it releases the full adjacency matrix, while SBM and AGM introduce error even without privacy. However, after adding noise needed for privacy, SBM performs the best among synthetic data methods. Perhaps surprisingly, this is true even for GNN-based fraud detectors as shown in Figure 4.12. Subsample-and-aggregate distorts graph structure extensively, even with only 5 partitions, resulting in high error without privacy. Additional results are provided in the supplemental material.

Subsample-and-Aggregate

In experiments on four validation datasets, we seek to understand how the parameters of the algorithm—number of partitions k and rate of sub-sampling fraud in each partition ρ —impact the bias, variance from Laplace noise addition and overall distortion of fraud detector rankings.

On each dataset we run Algorithm 13 for 10 trials for each choice of parameters k , ρ and ε . We show per-dataset results on the Yelp and Elliptic validation datasets in this section. Additional

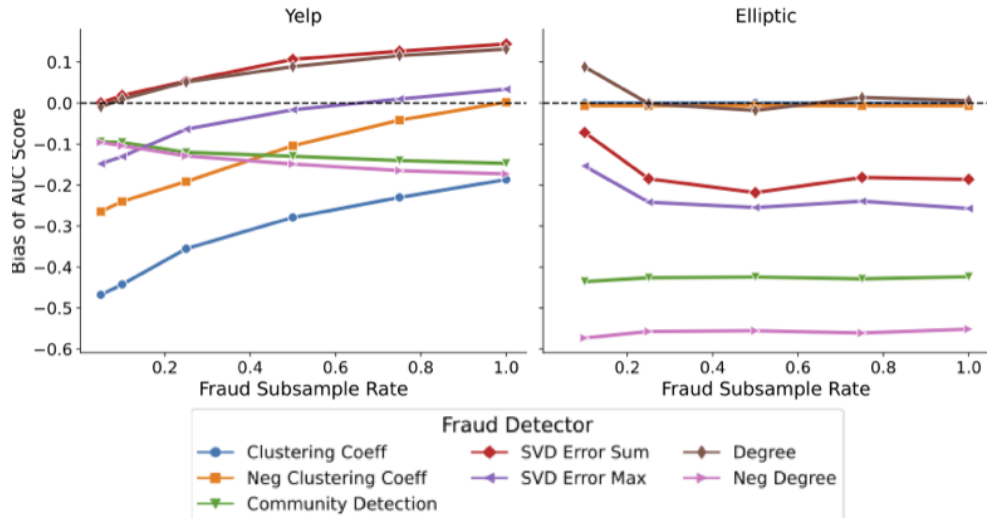


Figure 4.10: Bias to the AUC score introduced by subsample-and-aggregate for each fraud detector varying the fraud subsample rate (ρ) while fixing number of partitions $k = 20$. Subsample-and-aggregate introduces extensive bias to all fraud detectors, with the sign and magnitude of the bias varying widely across fraud detectors.

results are provided in the supplemental material.

In general, we find that partitioning the graph into random sub-graphs introduces significant bias to estimates of graph statistics. The sign and magnitude of this bias can differ widely across fraud detectors. In Figure 4.10, we show bias per fraud detector fixing the number of partitions at $k = 20$ and varying the fraud sub-sampling rate. We find that it is not always possible to achieve zero bias for a given fraud detector for a given number of partitions $k = 20$. For instance, the clustering coefficient detector has negative bias on the Yelp dataset at all values of ρ . This makes sense as removing benign vertices from the graph may change the distribution of fraud detection scores for benign vertices such that it is not possible to recover a similar distribution at any rate of sub-sampling fraudulent vertices. We additionally find, as expected, that the magnitude of bias increases with the number of partitions (k) although the sign of the bias differs across fraud detectors, with additional plots provided in the supplemental material. These results explain the poor performance of subsample-and-aggregate in Figure 4.8, as subsampling tends to distort graph structure extensively, biasing different fraud detectors in different ways thereby undermining the utility of the ranking of fraud detectors.

Synthetic Graph Generation

In our experiments we aim to isolate error introduced due to choice of graph model and noisy estimation of sufficient statistics. For each synthetic data generation algorithm, we generate 10 synthetic data sets. For each synthetic graph method, we subdivide the privacy budget evenly between the different parameters to estimate. We note that it may be possible to better distribute privacy budget between different statistics, which is an interesting area for future investigation. We test degree truncation thresholds as a function of the max degree of each graph, so 1.0 is a

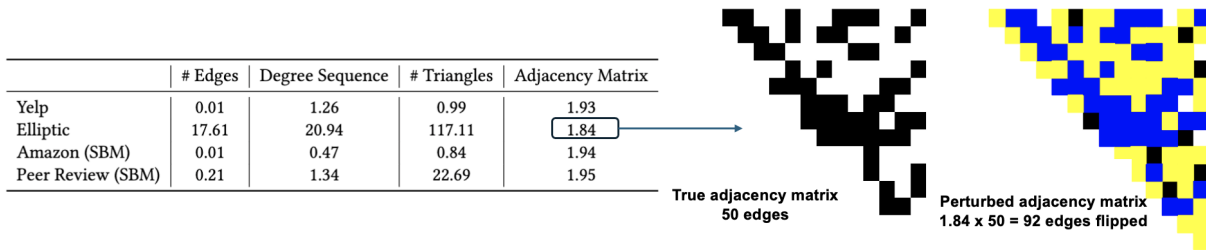


Figure 4.11: Normalized mean absolute error (MAE) introduced to each of the synthetic graph sufficient statistics. We fix $\varepsilon = 5.0$ and the degree cutoff to $1 \times$ the graph’s max degree. It is possible to estimate SBM parameters accurately, while other parameters have large noise addition. We give an example of what relative error of 1.84 means for the adjacency matrix on the right, where an adjacency matrix with 50 edges has 92 edges flipped: 42 removed (yellow) and 50 added (blue.)

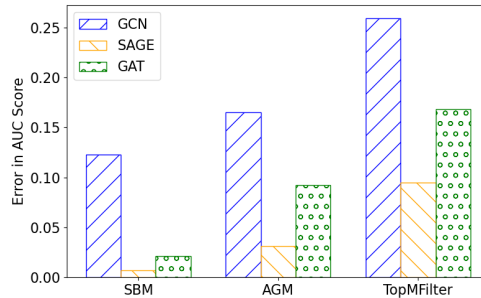


Figure 4.12: Error in AUC score for GNN-based fraud detectors on Yelp data using synthetic graphs with $\varepsilon = 5.0$.

threshold exactly equal to the maximum degree benign vertex in a graph while 0.5 removes all nodes with degree > 0.5 times the max degree. In this section, we report results with threshold of 1 and give additional results for 0.5 in Supplemental Material.

We find that outside of the SBM, it is necessary to introduce large distortion to the sufficient statistics of each graph model in order to preserve privacy, as shown in Figure 4.11. We show the proportional change in each noisy sufficient statistic compared to its true value, taking the mean over a vector-valued statistic. On Yelp, Amazon, and Peer Review it is possible to estimate the edge count for the SBM with high accuracy at $\varepsilon = 5.0$, perturbing the edge count by 1% of the total number on Yelp and Amazon. Elliptic is an extremely sparse graph (0.04%), so we introduce much larger relative error. For degree sequence and number of triangles, the amount of error is one to two orders of magnitude larger, with error generally at least 50% of the value of the original statistic. Unsurprisingly, the adjacency matrix cannot be accurately estimated under node-DP via direct noise addition. We highlight the amount of noise addition needed to preserve privacy in a simple example of relative error of 1.84 on a 15x15 adjacency matrix, shown in Figure 4.11. Even after aggressively truncating high-degree nodes, the addition of DP noise results in flipping the same number of edges as were originally in the adjacency matrix. This large distortion of sufficient statistics explains the poor accuracy of AGM and TopMfilter.

4.2.9 Discussion

This subchapter defined the problem of privately benchmarking fraud detectors on graph-structured data. The experiments benchmarked two popular frameworks from the DP literature, subsample-and-aggregate and synthetic data generation, and characterized a common tradeoff: both methods must balance error from distorting graph structure against error from privacy-preserving noise addition. The results show that this tradeoff is especially severe on graph data, where the statistics needed for fraud detection can be highly sensitive and where small groups of fraudulent vertices can be lost under naive subsampling.

Several design questions follow from these findings. First, the choice of method and hyperparameters, such as the number of partitions in subsample-and-aggregate, can have a large effect on utility. A practical benchmarking server therefore needs private procedures for model and hyperparameter selection, not only private procedures for releasing a fixed score. Second, existing DP synthetic graph algorithms are not tailored to fraud detection. The experiments show that they can introduce substantial bias even before adding privacy noise, suggesting that they do not capture the graph structures most relevant for fraudulent behavior. Third, synthetic graph generation may be more effective if it models graph metadata rather than graph structure directly. Metadata is attributable to individual vertices and can therefore be modeled more like tabular data, while connectivity between groups of vertices could be estimated through lower-sensitivity edge counts between feature clusters, as in a stochastic block model.

Taken together, these results highlight privacy vulnerabilities in benchmarking fraud detectors on private data and explain why utility is difficult to preserve on graph-structured benchmarks. This subchapter closes the privacy part of the thesis by showing that privacy-preserving data release is not a single technical problem: timing metadata, benchmark scores, graph structure, and the operational purpose of a release each create different constraints on what a useful privacy mechanism must do.

4.2.10 Deferred Proofs

This section records the main privacy proofs used in the private fraud benchmarking subchapter.

Composition for Protected Differential Privacy

Proof of Theorem 4.2.2. Let G and G' be protected-neighboring graphs that differ only in the protected data of one benign vertex. For each fraud detector \mathcal{A}_i , the theorem assumes that releasing $f(\mathcal{A}_i, G)$ is ε_i -protected differentially private. Therefore, for every measurable set S_i of possible outputs,

$$\Pr[f(\mathcal{A}_i, G) \in S_i] \leq e^{\varepsilon_i} \Pr[f(\mathcal{A}_i, G') \in S_i].$$

For a joint output event S over the vector of released scores, expand the probability by conditioning sequentially on the previous released scores. At step i , the conditional release is still an ε_i -protected DP computation on the same neighboring graphs, even when the submitted detector or later computation is chosen adaptively from previous outputs. Multiplying the likelihood-ratio bounds over $i = 1, \dots, m$ gives

$$\Pr[(f(\mathcal{A}_1, G), \dots, f(\mathcal{A}_m, G)) \in S] \leq e^{\sum_i \varepsilon_i} \Pr[(f(\mathcal{A}_1, G'), \dots, f(\mathcal{A}_m, G')) \in S].$$

Thus the vector release is $(\sum_i \varepsilon_i)$ -protected differentially private. \square

Privacy of Partition, Duplicate, and Aggregate

Proof of Proposition following Algorithm 13. Consider neighboring graphs that differ in the protected data of one benign vertex. Algorithm 13 randomly partitions benign vertices into k disjoint groups, while fraudulent vertices are sampled separately into each partition. The changed benign vertex can affect at most one partition. Since the accuracy score has global sensitivity at most 1 on a single partition, changing that vertex can change the average

$$\frac{1}{k} \sum_{i=1}^k f(\mathcal{A}, G_i)$$

by at most $1/k$.

The final release adds Laplace noise with scale calibrated to this sensitivity, namely $\Delta/(k\varepsilon)$ when the per-partition score has sensitivity Δ . By the Laplace mechanism, the released aggregate score is ε -protected differentially private. The duplicate-and-subsample step for fraudulent vertices does not weaken the guarantee because protected neighboring only varies benign vertices; fraudulent vertices are not protected by the privacy definition used in this benchmark. \square

Chapter 5

Conclusions

This thesis studied how to make better decisions from distributed human evaluations. Many institutions must rely on aggregated human judgments even though these judgments can be noisy, biased, and hard to validate. We developed three complementary responses. First, measure where auxiliary evaluation signals fail or help. Second, design decision rules that explicitly reflect uncertainty and instability in those review signals. Third, release evaluation data through mechanisms that account for the concrete privacy risks created by the release.

The experimental chapter showed that secondary evaluation signals in peer review can be useful but should be interpreted carefully. Evaluations of review quality are not clean measurements of true review quality. They are affected by review length, author outcomes, disagreement, miscalibration, and subjectivity. This matters because review-quality evaluations are increasingly tempting as inputs to incentives and policy evaluation. If these scores are used without correction, they can reward superficial signals or reproduce biases from the original review process. The LLM Checklist Assistant study showed a more constructive possibility: an automated tool can help authors identify issues before review.

The randomized selection chapter developed decision rules for settings where imperfect evaluations are nevertheless the basis for consequential choices. MERIT formalized one reason to randomize: when quality estimates are interval-valued and the decision maker does not trust a full probabilistic model, randomization can be the solution to a robust optimization problem. Smooth lotteries formalized a second reason: when hard thresholds make selection probabilities highly sensitive to small score changes, randomization can enforce stability. These mechanisms do not imply that lotteries are always the right policy. Rather, they show that if an institution chooses to randomize, the lottery should be tied to a clear model of what has gone wrong in the evaluation process. They give practical, efficient, and well-motivated methods to realize the motivations for using randomized mechanisms in the first place.

Finally, the privacy chapter showed that transparency in evaluation systems is itself a technical design problem. Timing metadata can deanonymize pseudonymous users when people batch related actions. Benchmark scores can leak information about private graph data when submitted algorithms are allowed to depend on the private instance. In both cases, useful release requires identifying the leakage channel and designing a privacy definition around the operational constraints of the platform. Generic anonymization or black-box application of standard privacy tools is not enough when the released data are temporal, relational, or produced by strategic

participants.

A theme across the three chapters is that measurement, decision-making, and transparency feed back into each other. The pathologies we found in review-quality scores are also what motivates lotteries that do not treat scores as ground truth. The lotteries themselves create new artifacts worth releasing and require evaluation on privacy-gated data. This feedback loop suggests a broader lesson: distributed evaluation systems should be designed around the uncertainty they actually produce. Better experiments can show which signals deserve trust, better mechanisms can make uncertainty explicit in consequential decisions, and better privacy tools can make the resulting systems auditable without exposing the people who participate in them.

Bibliography

- [1] David Adam. Science funders gamble on grant lotteries. *Nature*, 575(7784):574–575, 2019. doi: 10.1038/d41586-019-03572-7. URL <https://www.nature.com/articles/d41586-019-03572-7>. 1.1, 3, 3.2.6
- [2] Ranendu Adhikary, Kaustav Bose, Satwik Mukherjee, and Bodhayan Roy. Complexity of maximum cut on interval graphs. *Discrete & Computational Geometry*, 70(2):307–322, 2023. 3, 3.1.4
- [3] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Advances in Knowledge Discovery and Data Mining: 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21-24, 2010. Proceedings. Part II 14*, pages 410–421. Springer, 2010. 4.2.7
- [4] Sara Amoroso, Pietro Moncada-Paternò-Castello, and Antonio Vezzani. R&d profitability: the role of risk and knightian uncertainty. *Small Business Economics*, 48:331–343, 2017. 3.1.1
- [5] Clare Arden, Nadia Martino, Sammy Nag, Robyn Tamblyn, David Moher, Adrian Mota, and Karim Khan. Three years of quality assurance data assessing the performance of over 4000 grant peer review contributions to the Canadian institutes of health research project grant competition. *FACETS*, 2023. 2.1.2
- [6] Yukino Baba and Hisashi Kashima. Statistical quality estimation for general crowdsourcing tasks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013. 5, 3.1.6, 3.1.6
- [7] Mitali Bafna and Jonathan Ullman. The price of selection in differential privacy. In *Conference on Learning Theory*, pages 151–168. PMLR, 2017. 3.2.2, 3.2.5
- [8] Konstantina Bairaktari, Paul Tsela Langton, Huy Nguyen, Niklas Smedemark-Margulies, and Jonathan Ullman. Fair and useful cohort selection. *Transactions on Machine Learning Research*, 2023. 3.2.1, 3.2.2, 8, 3.2.4, 3.2.5
- [9] Stefano Balietti, Robert Goldstone, and Dirk Helbing. Peer review and competition in the art exhibition game. *Proceedings of the National Academy of Sciences*, 2016. 2.1.5
- [10] Rina Foygel Barber, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1):486–507, 2021. 1
- [11] Andrés F Barrientos, Aaron R Williams, Joshua Snoke, and CM Bowen. Differentially private methods for validation servers. Technical report, Urban Institute research report,

2021. 4.2.4

- [12] Yakov Ben-Haim and Maria Demertzis. Decision making in times of knightian uncertainty: An info-gap perspective. *Economics*, 10(1):20160001, 2016. 3.1.1
- [13] Joël Berger, Margit Osterloh, and Katja Rost. Focal random selection closes the gender gap in competitiveness. *Science Advances*, 6(47):eabb2142, 2020. doi: 10.1126/sciadv.aabb2142. URL <https://www.science.org/doi/10.1126/sciadv.aabb2142>. 3
- [14] Bitcoin Wiki. <https://en.bitcoin.it/wiki/Transaction#Input>. Accessed on November 3, 2022. 4.1.1
- [15] Bitcoin Wiki. Address reuse. https://en.bitcoin.it/wiki/Address_reuse, 2021. Accessed on April 21, 2021. 4.1.1
- [16] Bitcoin Wiki. https://en.bitcoin.it/wiki/Common-input-ownership_heuristic, 2022. Accessed on November 3, 2022. 4.1.1
- [17] Christopher Blank, Shaila Zaman, Amanveer Wesley, Panagiotis Tsiamyrtzis, Dennis R. Da Cunha Silva, Ricardo Gutierrez-Osuna, Gloria Mark, and Ioannis Pavlidis. *Emotional Footprints of Email Interruptions*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450367080. URL <https://doi.org/10.1145/3313831.3376282>. 4.1.1
- [18] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, page 87–96, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450318594. doi: 10.1145/2422436.2422449. 4.2.4, 4.2.6
- [19] Stephen Boyd and Lieven Vandenbergh. *Convex optimization*. Cambridge university press, 2004. 4.1.3
- [20] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of Computational Social Choice*. Cambridge University Press, 2016. 3.1.5
- [21] Lyle Brenner, Dale Griffin, and Derek J Koehler. Modeling patterns of probability calibration with random support theory: Diagnosing case-based judgment. *Organizational Behavior and Human Decision Processes*, 97(1):64–81, 2005. 1.1, 3.1.1
- [22] Michael Callaham and Charles McCulloch. Longitudinal trends in the performance of scientific peer reviewers. *Annals of emergency medicine*, 57(2):141–148, 2011. 2
- [23] Michael L Callaham and John Tercier. The relationship of previous training and experience of journal peer reviewers to subsequent review quality. *PLoS medicine*, 4(1):e40, 2007. 2
- [24] Michael L Callaham, William G Baxt, Joseph F Waeckerle, and Robert L Wears. Reliability of editors' subjective quality ratings of peer reviews of manuscripts. *Jama*, 280(3): 229–231, 1998. 2.1.2
- [25] Michael L. Callaham, Robert K. Knopp, and E. John Gallagher. Effect of Written Feedback by Editors on Quality of Reviews: Two Randomized Trials. *JAMA*, 287(21):

- 2781–2783, 06 2002. ISSN 0098-7484. doi: 10.1001/jama.287.21.2781. URL <https://doi.org/10.1001/jama.287.21.2781>. 2
- [26] L. Charlin and R. S. Zemel. The Toronto Paper Matching System: An automated paper-reviewer assignment system. In *ICML Workshop on Peer Reviewing and Publishing Models*, 2013. 2.2.1
- [27] Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In *International Symposium on Privacy Enhancing Technologies*, pages 82–102. Springer, 2013. 3.2.5
- [28] Xihui Chen, Sjouke Mauw, and Yunior Ramírez-Cruz. Publishing community-preserving attributed social graphs with a differential privacy guarantee. *Proceedings on Privacy Enhancing Technologies*, 2020:131 – 152, 2019. URL <https://api.semanticscholar.org/CorpusID:202540124>. 4.2.3, 4.2.6
- [29] Kevin C Chung, Melissa J Shauver, Sunitha Malay, Lin Zhong, Aaron Weinstein, and Rod J Rohrich. Is double-blinded peer review necessary? the effect of blinding on review quality. *Plastic and reconstructive surgery*, 136(6):1369–1377, 2015. 2
- [30] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. Specter: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, 2020. 2.2.1
- [31] Edith Cohen, Xin Lyu, Jelani Nelson, Tam’as Sarl’os, and Uri Stemmer. Generalized private selection and testing with high confidence. *ArXiv*, abs/2211.12063, 2022. URL <https://api.semanticscholar.org/CorpusID:253761282>. 4.2.4
- [32] Federal Trade Commission. As Nationwide Fraud Losses Top \$10 Billion in 2023, FTC Steps Up Efforts to Protect the Public. <https://www.ftc.gov/news-events/news/press-releases/2024/02/nationwide-fraud-losses-top-10-billion-2023-ftc-steps-efforts-protect-public>, February 2024. (Accessed on 02/29/2024). 4.2.1
- [33] Corinna Cortes and Neil D Lawrence. Inconsistency in conference peer review: Revisiting the 2014 neurips experiment. *arXiv preprint arXiv:2109.09774*, 2021. 1.1, 3.1.1
- [34] Corinna Cortes and Neil D. Lawrence. Inconsistency in conference peer review: Revisiting the 2014 NeurIPS experiment, 2021. 2.1.4, 2.1.4
- [35] Mike D’Arcy, Alexis Ross, Erin Bransom, Bailey Kuehl, Jonathan Bragg, Tom Hope, and Doug Downey. Aries: A corpus of scientific paper edits made in response to peer reviews. *ArXiv*, abs/2306.12587, 2023. URL <https://api.semanticscholar.org/CorpusID:259224834>. 2.2.1
- [36] Mike D’Arcy, Tom Hope, Larry Birnbaum, and Doug Downey. MARG: multi-agent review generation for scientific papers. *arXiv preprint arXiv:2401.04259*, 2024. 2, 2.2.1
- [37] Datavisor. Datavisor: AI Powered Fraud Platform for Enterprise, 2024. URL <https://www.datavisor.com/>. (Accessed on 02/29/2024). 4.2.1
- [38] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular

- margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019. 4.2.2
- [39] Komal Dhull, Steven Jecmen, Pravesh Kothari, and Nihar B Shah. Strategyproofing peer assessment via partitioning: The price in terms of evaluators’ expertise. In *HCOMP*, 2022. 2.1.5
- [40] Wenxin Ding, Nihar B. Shah, and Weina Wang. On the privacy-utility tradeoff in peer-review data analysis. In *AAAI Privacy-Preserving Artificial Intelligence (PPAI-21) workshop*, 2020. 4.1.2
- [41] Wenxin Ding, Gautam Kamath, Weina Wang, and Nihar B. Shah. Calibration with privacy in peer review. In *ISIT*, 2022. 4.1.2
- [42] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. <https://svn-archive.torproject.org/svn/projects/design-paper/tor-design.pdf>, 06 2004. 4.1.4
- [43] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management, CIKM ’20*, page 315–324, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3411903. 4.2.7
- [44] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024. 2.1.2, 2.1.5
- [45] Esin Durmus, Faisal Ladhak, and Tatsunori Hashimoto. Spurious correlations in reference-free evaluation of text generation. *arXiv preprint arXiv:2204.09890*, 2022. 2.1.2, 2.1.5
- [46] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-35908-1. 4.1.4
- [47] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-35908-1. 1, 4.2.3, 4.2.2, 4.2.4
- [48] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, STOC ’09*, page 371–380, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585062. doi: 10.1145/1536414.1536466. URL <https://doi.org/10.1145/1536414.1536466>. 3
- [49] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, aug 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL <https://doi.org/10.1561/0400000042>. 4.1.4
- [50] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy.

Foundations and trends® in theoretical computer science, 9(3-4):211–487, 2014. 3.2.2, 3.2.5

- [51] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, aug 2014. ISSN 1551-305X. doi: 10.1561/04000000042. URL <https://doi.org/10.1561/04000000042>. 4.2.5
- [52] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography, 3rd Theory of Cryptography Conference, TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. doi: 10.1007/11681878_14. 3.2.2, 3.2.5
- [53] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography, TCC’06*, page 265–284, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3540327312. doi: 10.1007/11681878_14. URL https://doi.org/10.1007/11681878_14. 4.1.1, 4.1.2, 4.1.4, 4.1.4
- [54] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012. 3.2.1, 3.2.2, 3.2.5
- [55] Larry G Epstein and Tan Wang. Intertemporal asset pricing under knightian uncertainty. In *Uncertainty in Economic Theory*, pages 429–471. Routledge, 2004. 3.1.1
- [56] European Research Council. Peer review – not perfect but still the best system that we have. <https://erc.europa.eu/news-events/magazine-article/peer-review-not-perfect-still-best-system-we-have>, November 2023. Accessed: 2025-05-07. 1.1, 3, 3.1.2, 1
- [57] Arthur T Evans, Robert A McNutt, Suzanne W Fletcher, and Robert H Fletcher. The characteristics of peer reviewers who produce good-quality reviews. *Journal of general internal medicine*, 8:422–428, 1993. 2
- [58] Ulrich Faigle and Walter Kern. Computational complexity of some maximum average weight problems with precedence constraints. *Operations Research*, 42(4):688–693, 1994. 3, 3.1.4
- [59] Ferric C Fang and Arturo Casadevall. Research funding: The case for a modified lottery, 2016. 1.1, 3
- [60] Thomas Feliciani, Junwen Luo, and Kalpana Shankar. Funding lotteries for research grant allocation: An extended taxonomy and evaluation of their fairness. *Research Evaluation*, 33:rvae025, 2024. 1.1, 3
- [61] Irene D Feurer, Gary J Becker, Daniel Picus, Estella Ramirez, Michael D Darcy, and Marshall E Hicks. Evaluating peer reviews: pilot testing of a grading instrument. *JAMA*, 272(2):98–100, 1994. 2.1.2
- [62] Peter A. Flach, Sebastian Spiegler, Bruno Golénia, Simon Price, John Guiver, Ralf Herbrich, Thore Graepel, and Mohammed J. Zaki. Novel tools to streamline the conference review process: Experiences from SIGKDD’09. *SIGKDD Explor. Newsl.*, 11(2):

- 63–67, 2010. ISSN 1931-0145. doi: 10.1145/1809400.1809413. URL <http://doi.acm.org/10.1145/1809400.1809413>. 5, 3.1.6, 3.1.6, 4.1.2
- [63] Andrea Forte, Nazanin Andalibi, and Rachel Greenstadt. Privacy, anonymity, and perceived risk in open collaboration: A study of tor users and wikipedians. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, CSCW '17*, page 1800–1811, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450343350. doi: 10.1145/2998181.2998273. URL <https://doi.org/10.1145/2998181.2998273>. 4.1.1
- [64] European Science Foundation. ESF survey analysis report on peer review practices, 2011. Available online https://www.esf.org/fileadmin/user_upload/esf/PeerReview-Practices_Survey2011.pdf. 1, 2.1.2
- [65] H. Ge, M. Welling, and Z. Ghahramani. A Bayesian model for calibrating conference review scores, 2013. Available online <http://mlg.eng.cam.ac.uk/hong/unpublished/nips-review-model.pdf> Last accessed: April 4, 2021. 2.1.4, 4.1.2
- [66] Quan Geng and Pramod Viswanath. The optimal mechanism in differential privacy. In *2014 IEEE International Symposium on Information Theory*, pages 2371–2375, 2014. doi: 10.1109/ISIT.2014.6875258. 4.1.2, 4.1.4, 4.1.4, 2, 6, 4.1.4, 4.1.8
- [67] Alexander Goldberg, Giulia Fanti, and Nihar B. Shah. Batching of tasks by users of pseudonymous forums: Anonymity compromise and protection. In *Abstract Proceedings of the 2023 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 89–90, 2023. doi: 10.1145/3578338.3593525. 1.1, 4.1
- [68] Alexander Goldberg, Ihsan Ullah, Thanh Gia Hieu Khuong, Benedictus Kent Rachmat, Zhen Xu, Isabelle Guyon, and Nihar B. Shah. Usefulness of LLMs as an author checklist assistant for scientific papers: NeurIPS’24 experiment. *arXiv preprint arXiv:2411.03417*, 2024. doi: 10.48550/arXiv.2411.03417. URL <https://arxiv.org/abs/2411.03417>. 1.1, 2.2
- [69] Alexander Goldberg, Giulia Fanti, and Nihar B. Shah. A principled approach to randomized selection under uncertainty: Applications to peer review and grant funding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 1.1, 3.1, 3.2.1, 3.2.2, 3.2.4, 3.2.6, 3.2.7
- [70] Alexander Goldberg, Giulia Fanti, Nihar B. Shah, and Zhiwei Steven Wu. Benchmarking fraud detectors on private graph data. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 378–389, 2025. doi: 10.1145/3690624.3709170. 1.1, 4.2
- [71] Alexander Goldberg, Ivan Stelmakh, Kyunghyun Cho, Alice Oh, Alekh Agarwal, Danielle Belgrave, and Nihar B. Shah. Peer reviews of peer reviews: A randomized controlled trial and other experiments. *PLOS ONE*, 20(3):e0320444, 2025. doi: 10.1371/journal.pone.0320444. 1.1, 2.1
- [72] Alexander Goldberg, Giulia Fanti, and Nihar B. Shah. Smooth partial lotteries for stable randomized selection. *arXiv preprint*, 2026. 1.1, 3.2

- [73] Joanie Sims Gould, Anne M Lasinsky, Adrian Mota, Karim M Khan, and Clare L Ardern. Threats to grant peer review: a qualitative study. *BMJ open*, 15(2):e091666, 2025. 1.1, 3
- [74] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981. 3.1.4
- [75] Prince Grover, Julia Xu, Justin Tittelfitz, Anqi Cheng, Zheng Li, Jakub Zablocki, Jianbo Liu, and Hao Zhou. Fraud dataset benchmark and applications, 2023. 4.2.1
- [76] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1988. 3.1.8
- [77] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2025. URL <https://www.gurobi.com>. 3.1.6
- [78] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 4.2.7
- [79] Rachel Heyard, Manuela Ott, Georgia Salanti, and Matthias Egger. Rethinking the funding line at the swiss national science foundation: Bayesian ranking and lottery. *Statistics and Public Policy*, 9(1):110–121, 2022. 1.1, 3, 3.1.1, 1, 5, 3.1.2, 2, 3.1.2, 1, 3.1.6, 3.1.6, 3.2.2, 3.2.6, 3.2.7, 3.2.7
- [80] Serge PJM Horbach, Joeri K Tijdink, and Lex M Bouter. Partial lottery can make grant allocation more fair, more efficient, and more diverse. *Science and Public Policy*, 49(4): 580–582, 2022. 1.1, 3
- [81] Jhih-Yi (Janet) Hsieh, Aditi Raghunathan, and Nihar Shah. Vulnerability of text-matching in ML/AI conference reviewer assignments to collusion rings. arXiv, 2024. 3.2.2
- [82] Y. Hu, F. Wu, Q. Li, Y. Long, G. Garrido, C. Ge, B. Ding, D. Forsyth, B. Li, and D. Song. Sok: Privacy-preserving data synthesis. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 2–2, Los Alamitos, CA, USA, may 2024. IEEE Computer Society. doi: 10.1109/SP54263.2024.00002. URL <https://doi.ieeecomputersociety.org/10.1109/SP54263.2024.00002>. 4.2.6
- [83] Omid Javidbakht and Parv Venkitasubramaniam. Delay anonymity tradeoff in mix networks: Optimal routing. *IEEE/ACM Transactions on Networking*, 25(2):1162–1175, 2017. doi: 10.1109/TNET.2016.2624023. 4.1.2
- [84] Steven Jecmen, Hanrui Zhang, Ryan Liu, Nihar Shah, Vincent Conitzer, and Fei Fang. Mitigating manipulation in peer review via randomized reviewer assignments. *Advances in Neural Information Processing Systems*, 33:12533–12545, 2020. 4.2.1
- [85] Steven Jecmen, Hanrui Zhang, Ryan Liu, Nihar B. Shah, Vincent Conitzer, and Fei Fang. Mitigating manipulation in peer review via randomized reviewer assignments. In *NeurIPS*, 2020. 2.1.5, 3.2.2, 4.1.2
- [86] Steven Jecmen, Nihar B Shah, Fei Fang, and Vincent Conitzer. Tradeoffs in preventing manipulation in paper bidding for reviewer assignment. In *ICLR workshop on ML Evaluation Standards*, 2022. 4.1.2
- [87] Steven Jecmen, Minji Yoon, Vincent Conitzer, Nihar B. Shah, and Fei Fang. A dataset on malicious paper bidding in peer review, 2022. URL <https://arxiv.org/abs/22>

- [88] Steven Jecmen, Minji Yoon, Vincent Conitzer, Nihar B Shah, and Fei Fang. A dataset on malicious paper bidding in peer review. In *Proceedings of the ACM Web Conference 2023*, pages 3816–3826, 2023. 4.2.1
- [89] Steven Jecmen, Nihar B. Shah, Fei Fang, and Leman Akoglu. On the detection of reviewer-author collusion rings from paper bidding, 2024. 4.2.3, 4.2.7
- [90] Steven Jecmen, Nihar B Shah, Fei Fang, and Leman Akoglu. On the detection of reviewer-author collusion rings from paper bidding. *Transactions on Machine Learning Research*, December 2024. 4.2.1
- [91] Shouling Ji, Prateek Mittal, and Raheem Beyah. Graph data anonymization, de-anonymization attacks, and de-anonymizability quantification: A survey. *IEEE Communications Surveys & Tutorials*, 19(2):1305–1326, 2017. doi: 10.1109/COMST.2016.2633620. 4.2.2
- [92] Zach Jorgensen, Ting Yu, and Graham Cormode. Publishing attributed social graphs with formal privacy guarantees. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD '16*, page 107–122, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450335317. doi: 10.1145/2882903.2915215. URL <https://doi.org/10.1145/2882903.2915215>. 4.2.3, 2
- [93] Sachin Kadloor, Parv Venkitasubramaniam, and Negar Kiyavash. Preventing timing analysis in networks: A statistical inference perspective. *IEEE Signal Processing Magazine*, 30(5):76–85, 2013. doi: 10.1109/MSP.2013.2261131. 4.1.2
- [94] Kaggle. Kaggle: Your machine learning and data science community, 2024. URL <https://www.kaggle.com/>. (Accessed on 02/29/2024). 4.2.1
- [95] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, 2017. 4.1.5
- [96] Parameswaran Kamalaruban, Victor Perrier, Hassan Jameel Asghar, and Mohamed Ali Kaafar. Not all attributes are created equal: dx-private mechanisms for linear queries. *Proceedings on Privacy Enhancing Technologies*, 2020. 3.2.5
- [97] Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *ACM Trans. Database Syst.*, 39(3), oct 2014. ISSN 0362-5915. doi: 10.1145/2611523. URL <https://doi.org/10.1145/2611523>. 4.2.3
- [98] Vishesh Karwa, Pavel N. Krivitsky, and Aleksandra B. Slavković. Sharing social network data: differentially private estimation of exponential family random-graph models. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 66(3):481–500, 2017. ISSN 00359254, 14679876. URL <http://www.jstor.org/stable/44682587>. 4.2.6
- [99] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, 2008. doi: 10.1109/FOCS.2008.27.

4.1.4

- [100] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In Amit Sahai, editor, *Theory of Cryptography*, pages 457–476, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-36594-2. 4.2.4, 4.2.6, 2, 10
- [101] Michael Kearns, Aaron Roth, Zhiwei Wu, and Grigory Yaroslavtsev. Private algorithms for the protected in social network search. *Proceedings of the National Academy of Sciences*, 113:201510612, 01 2016. doi: 10.1073/pnas.1510612113. 4.2.1
- [102] Aditya Khosla, Derek Hoiem, and Serge Belongie. Analysis of reviews for cvpr 2012, 2013. 2.1.2, 2.1.4
- [103] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 4.2.7
- [104] Jonathan Klick. More of a declaration than a constitution. *The Daily Pennsylvanian*, February 2024. URL <https://www.thedp.com/article/2024/02/penn-forward-vision-academic-ideological>. Guest Column. 3
- [105] Frank Hyneman Knight. *Risk, uncertainty and profit*, volume 31. Houghton Mifflin, 1921. 1.1, 3.1.1, 3.1.2
- [106] Nitin Kohli and Paul Laskowski. Differential privacy for black-box statistical analyses. *Proceedings on Privacy Enhancing Technologies*, 3:418–431, 2023. 3, 4.2.4
- [107] Weiwei Kong, Walid Krichene, Nicolas Mayoraz, Steffen Rendle, and Li Zhang. Rankmax: An adaptive projection alternative to the softmax function. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. 3.2.2, 8, 3.2.4
- [108] Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International conference on machine learning*, pages 3499–3508. PMLR, 2019. 3.2.2, 3.2.5
- [109] Ios Kotsogiannis, Stelios Doudalis, Sam Haney, Ashwin Machanavajjhala, and Sharad Mehrotra. One-sided differential privacy. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 493–504, 2020. doi: 10.1109/ICDE48307.2020.00049. 4.1.1, 4.1.2, 4.1.3
- [110] Kount. Kount: Fraud detection and chargeback management solutions, 2024. URL <https://kount.com/>. (Accessed on 02/29/2024). 4.2.1
- [111] Maria K Kowalczyk, Frank Dudbridge, Shreeya Nanda, Stephanie L Harriman, Jigisha Patel, and Elizabeth C Moylan. Retrospective analysis of the quality of reports by author-suggested and non-author-suggested reviewers in journals operating on open or single-blind peer review models. *Bmj Open*, 5(9):e008707, 2015. 2
- [112] Ravi Kumar and Sergei Vassilvitskii. Generalized distances between rankings. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, page 571–580, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605587998. doi: 10.1145/1772690.1772749. URL <https://doi.org/10.1145/1772690.1772749>. 4.2.7

- [113] Kostadin Kushlev and Elizabeth W. Dunn. Checking email less frequently reduces stress. *Computers in Human Behavior*, 43:220–228, 2015. ISSN 0747-5632. doi: <https://doi.org/10.1016/j.chb.2014.11.005>. URL <https://www.sciencedirect.com/science/article/pii/S0747563214005810>. 4.1.1
- [114] Kostadin Kushlev and Elizabeth W. Dunn. Stop checking email so often, Jan 2015. URL <https://www.nytimes.com/2015/01/09/opinion/sunday/stop-checking-email-so-often.html>. 4.1.1
- [115] Ilia Kuznetsov, Osama Mohammed Afzal, Koen Dercksen, Nils Dycke, Alexander Goldberg, Tom Hope, Dirk Hovy, Jonathan K. Kummerfeld, Anne Lauscher, Kevin Leyton-Brown, Sheng Lu, Mausam, Margot Mieskes, Aurélie Névéol, Danish Pruthi, Lizhen Qu, Roy Schwartz, Noah A. Smith, Tamar Solorio, Jingyan Wang, Xiaodan Zhu, Anna Rogers, Nihar B. Shah, and Iryna Gurevych. What can natural language processing do for peer review? *arXiv:2405.06563*, 2024. 2.2.2
- [116] Conny Kühne, Klemens Böhm, and Jing Zhi Yue. Reviewing the reviewers: A study of author perception on peer reviews in computer science. In *6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2010)*, pages 1–8, 2010. doi: 10.4108/icst.collaboratecom.2010.33. 2.1.2, 2.1.4
- [117] John Langford. ICML acceptance statistics, 2012. <http://hunch.net/?p=2517> [Online; accessed 14-May-2018]. 1.1, 3.1.1
- [118] Carole J Lee. Commensuration bias in peer review. *Philosophy of Science*, 82(5):1272–1283, 2015. 2.1.4, 3, 4.1.2
- [119] Jinsook Lee, Emma Harvey, Joyce Zhou, Nikhil Garg, Thorsten Joachims, and René F Kizilcec. Algorithms for college admissions decision support: Impacts of policy change and inherent variability. *arXiv preprint arXiv:2407.11199*, 2024. 1.1, 3.1.1
- [120] Minhyeok Lee. Game-theoretical analysis of reviewer rewards in peer-review journal systems: Analysis and experimental evaluation using deep reinforcement learning. *arXiv preprint arXiv:2305.12088*, 2023. 1
- [121] Brian N. Levine, Michael K. Reiter, Chenxi Wang, and Matthew Wright. Timing attacks in low-latency mix systems. In Ari Juels, editor, *Financial Cryptography*, pages 251–265, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-27809-2. 4.1.2, 4.1.5
- [122] Yang D. Li, Michaela F. Purcell, Thierry Rakotoarivelo, David B. Smith, Thilina Ranbaduge, and Kee Siong Ng. Private graph data release: A survey. *ACM Computing Surveys*, 55:1 – 39, 2021. 4.2.6
- [123] Weixin Liang, Yuhui Zhang, Hancheng Cao, Binglu Wang, Daisy Ding, Xinyu Yang, Kailas Vodrahalli, Siyu He, Daniel Smith, Yian Yin, Daniel McFarland, and James Zou. Can large language models provide useful feedback on research papers? a large-scale empirical analysis, 2023. 2, 2.1.5
- [124] Weixin Liang, Yuhui Zhang, Hancheng Cao, Binglu Wang, Daisy Ding, Xinyu Yang, Kailas Vodrahalli, Siyu He, Daniel Smith, Yian Yin, et al. Can large language models pro-

- vide useful feedback on research papers? a large-scale empirical analysis. *arXiv preprint arXiv:2310.01783*, 2023. 2.2.1
- [125] Weixin Liang, Yaohui Zhang, Zhengxuan Wu, Haley Lepp, Wenlong Ji, Xuandong Zhao, Hancheng Cao, Sheng Liu, Siyu He, Zhi Huang, et al. Mapping the increasing use of LLMs in scientific papers. *arXiv preprint arXiv:2404.01268*, 2024. 2.2.1
- [126] Hsuan-Tien Lin, Maria Florina Balcan, Raia Hadsell, and Marc’Aurelio Ranzato. What we learned from NeurIPS 2020 reviewing process, Oct 2020. URL <https://neuripsconf.medium.com/what-we-learned-from-neurips-2020-reviewing-process-e24549eea38f>. 2.1.2
- [127] Elliot Little. Amsterdam built the ‘perfect’ ethical ai system. it still failed. here’s why. *Medium*, June 17 2025. URL <https://medium.com/@elliottJL/amsterdam-built-the-perfect-ethical-ai-system-it-still-failed-here-s-why-8dc8072beea3>. 8 min read. 4.2.1
- [128] Michael L Littman. Collusion rings threaten the integrity of computer science research. *Communications of the ACM*, 64(6):43–44, 2021. 2.1.5, 3.2.2, 4.1.2, 4.2.1
- [129] Fang Liu, Evercita Eugenio, Ick Hoon Jin, and Claire Bowen. Differentially private generation of social networks via exponential random graph models. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 1695–1700, 2020. doi: 10.1109/COMPSAC48688.2020.00-11. 4.2.6
- [130] Jingcheng Liu and Kunal Talwar. Private selection from private candidates, 2018. 4.2.4
- [131] Mengyao Liu, Vernon Choy, Philip Clarke, Adrian Barnett, Tony Blakely, and Lucy Pomeroy. The acceptability of using a lottery to allocate research funding: a survey of applicants. *Research integrity and peer review*, 5:1–7, 2020. 1.1, 3, 3.1.2, 1, 3.2.6
- [132] Ryan Liu and Nihar B Shah. Reviewergpt? an exploratory study on using large language models for paper reviewing. *arXiv preprint arXiv:2306.00622*, 2023. 2.1.5, 2.2.2
- [133] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 4.2.2
- [134] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery, 2024. URL <https://arxiv.org/abs/2408.06292>. 2.2.1
- [135] Abdelghani Maddi and Egidio Luis Miotti. On the peer review reports: It’s not the size that matters... really? *arXiv preprint arXiv:2309.02000*, 2023. 2.1.2
- [136] William G Madow. On the theory of systematic sampling, ii. *The Annals of Mathematical Statistics*, 20(3):333–354, 1949. 3.1.4, 6, 3.2.4
- [137] Aaron Mak. Inside the brutal, petty war over donald trump’s wikipedia page, May 2019. URL <https://slate.com/technology/2019/05/donald-trump-wikipedia-page.html>. 4.1.6
- [138] Andrey Malenko and Anton Tsoy. Asymmetric information and security design under knightian uncertainty. *Available at SSRN*, 3100285, 2020. 3.1.1

- [139] Charles F. Manski. Nonparametric bounds on treatment effects. *The American Economic Review*, 80(2):319–323, 1990. 1
- [140] Gloria Mark, Shamsi T Iqbal, Mary Czerwinski, Paul Johns, Akane Sano, and Yuliya Lutchyn. Email duration, batching and self-interruption: Patterns of email use on productivity and stress. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 1717–1728, 2016. 4.1.1
- [141] Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, pages 1614–1623. PMLR, 2016. 3.2.2
- [142] Felix Konstantin Maurer, Till Neudecker, and Martin Florian. Anonymous coinjoin transactions with arbitrary values. In *2017 IEEE Trustcom/BigDataSE/ICSS*, pages 522–529. IEEE, 2017. 3
- [143] David McKenzie. Identifying and spurring high-growth entrepreneurship: Experimental evidence from a business plan competition. *American Economic Review*, 107(8):2278–2307, 2017. 3
- [144] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 94–103. IEEE, 2007. 3.2.2, 3.2.5
- [145] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. A fistful of bitcoins: Characterizing payments among men with no names. *Commun. ACM*, 59(4):86–93, mar 2016. ISSN 0001-0782. doi: 10.1145/2896384. URL <https://doi.org/10.1145/2896384>. 4.1.1
- [146] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024. 2.1.2, 2.1.5
- [147] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. Gupta: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 349–360, 2012. 4.2.4, 4.2.8
- [148] David Moher, Kenneth F Schulz, and Douglas G Altman. The consort statement: revised recommendations for improving the quality of reports of parallel-group randomised trials. *The lancet*, 357(9263):1191–1194, 2001. 2.2.2
- [149] David Moher, Alessandro Liberati, Jennifer Tetzlaff, Douglas G Altman, and the PRISMA Group*. Preferred reporting items for systematic reviews and meta-analyses: the prisma statement. *Annals of internal medicine*, 151(4):264–269, 2009. 2.2.2
- [150] Kaleigh Moore. How to improve productivity with time batching. Monday blog, Nov 2021. Accessed on April 25, 2022. 4.1.1
- [151] Mark Murphy. If you haven’t tried time batching, you’ll be shocked at how quickly it improves your productivity and happiness. Forbes Magazine, 2021. Accessed on April 25, 2022. 4.1.1
- [152] Pravin Nair. Softmax is $1/2$ -Lipschitz: A tight bound across all ℓ_p norms. *Transactions on Machine Learning Research*, 2025. 3.2.2

- [153] National Institute of Standards and Technology (NIST). Face recognition vendor test (frvt) — 1:n identification. <https://pages.nist.gov/frvt/html/frvt1N.html>, 2025. Accessed on July 3, 2025. 4.2.1
- [154] Nature Editorial. The case for lotteries as a tiebreaker of quality in research funding. *Nature*, 609(7928):653, September 2022. doi: 10.1038/d41586-022-02959-3. URL <https://www.nature.com/articles/d41586-022-02959-3>. Accessed 2025-05-14. 3.1.1
- [155] Hiep H. Nguyen, Abdessamad Imine, and Michaël Rusinowitch. Differentially private publication of social graphs at linear cost. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ASONAM '15*, page 596–599, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450338547. doi: 10.1145/2808797.2809385. URL <https://doi.org/10.1145/2808797.2809385>. 4.2.3, 3
- [156] Kiyohiko G Nishimura and Hiroyuki Ozaki. Irreversible investment and knightian uncertainty. *Journal of Economic Theory*, 136(1):668–694, 2007. 3.1.1
- [157] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, STOC '07*, page 75–84, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595936318. doi: 10.1145/1250790.1250803. URL <https://doi.org/10.1145/1250790.1250803>. 3, 4.2.4, 4.2.5, 4.2.6
- [158] Ritesh Noothigattu, Nihar Shah, and Ariel Procaccia. Loss functions, axioms, and peer review. *Journal of Artificial Intelligence Research*, 2021. 4.1.2
- [159] Ritesh Noothigattu, Nihar Shah, and Ariel Procaccia. Loss functions, axioms, and peer review. *Journal of Artificial Intelligence Research*, 70:1481–1515, 2021. 3
- [160] Ritesh Noothigattu, Nihar Shah, and Ariel Procaccia. Loss functions, axioms, and peer review. *J. Artif. Int. Res.*, 70:1481–1515, may 2021. ISSN 1076-9757. doi: 10.1613/jair.1.12554. URL <https://doi.org/10.1613/jair.1.12554>. 2.1.4
- [161] Jakub Nowotarski and Rafał Weron. Computing electricity spot price prediction intervals using quantile regression and forecast averaging. *Computational Statistics*, 30:791–803, 2015. 2
- [162] OpenReview. NeurIPS 2024 Conference Submissions. <https://openreview.net/submissions?venue=NeurIPS.cc/2024/Conference>, 2024. Accessed May 5, 2026. 3.2.7
- [163] OpenReview. ICLR 2025 Conference Submissions. <https://openreview.net/submissions?venue=ICLR.cc/2025/Conference>, 2025. Accessed May 5, 2026. 3.2.7
- [164] Margit Osterloh and Bruno S Frey. Dealing with randomness. *Management Revue*, 30(4): 331–345, 2019. 3
- [165] Konstantina Papagiannaki. Author feedback experiment at pam 2007. *ACM SIGCOMM Computer Communication Review*, 37(3):73–78, 2007. 2.1.2, 2.1.4

- [166] Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. *ArXiv*, abs/2110.03620, 2021. URL <https://api.semanticscholar.org/CorpusID:238419564>. 4.2.4
- [167] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate, 2018. 4.2.4
- [168] Piitu Parmanne, Joonas Laajava, Noora Järvinen, Terttu Harju, Mauri Marttunen, and Pertti Saloheimo. Peer reviewers’ willingness to review, their recommendations and quality of reviews after the finnish medical journal switched from single-blind to double-blind peer review. *Research Integrity and Peer Review*, 8(1):14, 2023. 2
- [169] Axel Philipps. Research funding randomly allocated? a survey of scientists’ views on peer review and lottery. *Science and Public Policy*, 49(3):365–377, 2022. 1.1, 3
- [170] Shelly M. Pranić, Mario Malički, Stjepan Ljudevit Marušić, Bahar Mehmani, and Ana Marušić. Is the quality of reviews reflected in editors’ and authors’ satisfaction with peer review? a cross-sectional study in 12 journals across four research fields. *Learned Publishing*, 34(2):187–197, 2021. 2.1.2, 2.1.4
- [171] Gang Qiao, Weijie Su, and Li Zhang. Oneshot differentially private top-k selection. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8672–8681. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/qiao21b.html>. 4.2.5
- [172] Hamed Rahimian and Sanjay Mehrotra. Frameworks and results in distributionally robust optimization. *Open Journal of Mathematical Optimization*, 3:1–85, July 2022. ISSN 2777-5860. doi: 10.5802/ojmo.15. URL <http://dx.doi.org/10.5802/ojmo.15>. 5
- [173] Charvi Rastogi, Ivan Stelmakh, Alina Beygelzimer, Yann N Dauphin, Percy Liang, Jennifer Wortman Vaughan, Zhenyu Xue, Hal Daumé III, Emma Pierson, and Nihar B Shah. How do authors’ perceptions of their papers compare with co-authors’ perceptions and peer-review decisions? *arXiv:2211.12966*. *Short blog: <https://blog.ml.cmu.edu/2022/11/22/neurips2021-author-perception-experiment/>*, 2022. 2
- [174] Ramamurthy Ravi, Ajit Agrawal, and Philip Klein. Ordering problems approximated: single-processor scheduling and interval graph completion. In *Automata, Languages and Programming: 18th International Colloquium Madrid, Spain, July 8–12, 1991 Proceedings 18*, pages 751–762. Springer, 1991. 3, 3.1.4
- [175] Jerome P Reiter, Anna Oganian, and Alan F Karr. Verification servers: Enabling analysts to assess the quality of inferences from public use data. *Computational Statistics & Data Analysis*, 53(4):1475–1482, 2009. 4.2.4
- [176] Riskified. Riskified: Fraud Prevention & Chargeback Fraud Protection, 2024. URL <https://www.riskified.com/>. (Accessed on 02/29/2024). 4.2.1
- [177] Magnus Roos, Jörg Rothe, and Björn Scheuermann. How to calibrate the scores of biased

- reviewers by quadratic programming. In *AAAI Conference on Artificial Intelligence*, 2011. 5, 3.1.6, 3.1.6
- [178] Magnus Roos, Jörg Rothe, Joachim Rudolph, Björn Scheuermann, and Dietrich Stoyan. A statistical approach to calibrating the scores of biased reviewers: The linear vs. the nonlinear model. In *Multidisciplinary Workshop on Advances in Preference Handling*, 2012. 2.1.4, 5, 3.1.6, 3.1.6, 4.1.2
- [179] Doriana Ruffino. Some implications of knightian uncertainty for finance and regulation. *FEDS Notes*, April 2014. doi: 10.17016/2380-7172.0013. URL <https://www.federalreserve.gov/econresdata/notes/feds-notes/2014/some-implications-of-knightian-uncertainty-for-finance-and-regulation-20140410.html>. 3.1.1
- [180] Donald G. Saari. *Geometry of Voting*, volume 3 of *Studies in Economic Theory*. Springer Berlin Heidelberg, 1 edition, 2012. ISBN 978-3-642-48644-9. doi: 10.1007/978-3-642-48644-9. 3.1.5
- [181] Martin Saveski, Steven Jecmen, Nihar Shah, and Johan Ugander. Counterfactual evaluation of peer-review assignment policies. *Advances in Neural Information Processing Systems*, 36:58765–58786, 2023. 1, 3.2.2
- [182] Sara Schroter, Nick Black, Stephen Evans, James Carpenter, Fiona Godlee, and Richard Smith. Effects of training on quality of peer review: randomised controlled trial. *Bmj*, 328(7441):673, 2004. 2
- [183] Sara Schroter, Wim EJ Weber, Elizabeth Loder, Jack Wilkinson, and Jamie J Kirkham. Evaluation of editors’ abilities to predict the citation potential of research manuscripts submitted to the bmj: a cohort study. *Bmj*, 379, 2022. 1.1, 3.1.1
- [184] Science Foundation Ireland. Innovate for ireland programme. <https://www.sfi.ie/funding/funding-calls/innovate-for-ireland/>, August 2024. Accessed: 2025-05-07. 1.1, 3, 3.1.2, 1
- [185] Nihar B Shah. An overview of challenges, experiments, and computational solutions in peer review (extended version). <https://www.cs.cmu.edu/~nihars/preprints/SurveyPeerReview.pdf> Shorter version published in the *Communications of the ACM*, 2022. 4.1.2
- [186] Nihar B Shah. Challenges, experiments, and computational solutions in peer review. *Communications of the ACM*, 65(6):76–87, 2022. 1, 2.1.1, 2.1.5
- [187] Nihar B Shah. An overview of challenges, experiments, and computational solutions in peer review. <http://bit.ly/PeerReviewOverview> (Abridged version published in the *Communications of the ACM*), June 2022. 1.1, 2.2.2, 3.1.1
- [188] Nihar B. Shah, Behzad Tabibian, Krikamol Muandet, Isabelle Guyon, and Ulrike von Luxburg. Design and analysis of the NIPS 2016 review process. *Journal of Machine Learning Research*, 19(49):1–34, 2018. URL <http://jmlr.org/papers/v19/17-511.html>. 2.1.4, 2.1.4, 2.1.4
- [189] Nihar B Shah, Melisa Bok, Xukun Liu, and Andrew McCallum. Identity theft in ai con-

- ference peer review. *Communications of the ACM*, 68(12):32–34, 2025. 3.2.2
- [190] Vitaly Shmatikov and Ming-Hsiu Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In Dieter Gollmann, Jan Meier, and Andrei Sabelfeld, editors, *Computer Security – ESORICS 2006*, pages 18–33, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-44605-7. 4.1.2, 4.1.5
- [191] SIGMOD 2025 Program Committee. SIGMOD 2025 call for research papers. https://2025.sigmod.org/calls_papers_sigmod_research.shtml, 2025. Accessed May 5, 2026. 3.2.2, 3.2.7
- [192] Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. A long way to go: Investigating length correlations in RLHF. *arXiv preprint arXiv:2310.03716*, 2023. 2.1.2, 2.1.5
- [193] Siddarth Srinivasan and Jamie Morgenstern. Auctions and prediction markets for scientific peer review. *arXiv preprint arXiv:2109.00923*, 2021. 1
- [194] Thomas Steinke and Jonathan Ullman. Tight lower bounds for differentially private selection. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 552–563. IEEE Computer Society, 2017. doi: 10.1109/FOCS.2017.57. <https://arxiv.org/abs/1704.03024>. 3.2.2, 3.2.5
- [195] Ivan Stelmakh, Nihar B Shah, Aarti Singh, and Hal Daumé III. A novice-reviewer experiment to address scarcity of qualified reviewers in large conferences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4785–4793, 2021. 2, 2.1.2
- [196] Cass R Sunstein. Knightian uncertainty. *Available at SSRN 4662711*, 2023. 1.1, 3.1.1, 3.1.2
- [197] Cecilia Superchi, José Antonio González, Ivan Solà, Erik Cobo, Darko Hren, and Isabelle Boutron. Tools used to assess the quality of peer review reports: a methodological systematic review. *BMC medical research methodology*, 19:1–14, 2019. 2.1.3
- [198] Swiss National Science Foundation. Drawing lots as a tie-breaker, March 2021. URL <https://www.snf.ch/en/JyifP2I9SUo8CPxI/news/news-210331-drawing-lots-as-a-tie-breaker>. Accessed 2025-05-14. 3.1.1
- [199] Sijun Tan, Jibang Wu, Xiaohui Bei, and Haifeng Xu. Least square calibration for peer reviews. *Advances in Neural Information Processing Systems*, 34:27069–27080, 2021. 3.1.6, 3.1.6
- [200] The British Academy. Partial randomisation trial extended after diversity of applicants and award holders increases. <https://www.thebritishacademy.ac.uk/news/partial-randomisation-trial-extended-small-research-grants/>, April 2025. Accessed: 2025-05-07. 3, 3.1.2, 1, 3.2.6
- [201] Yves Tillé. Remarks on some misconceptions about unequal probability sampling without replacement. *Computer Science Review*, 47:100533, 2023. 3.1.4
- [202] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):5233, 2019. 4.2.7
- [203] Giulia Trentacosti. Using lotteries to allocate research funding: Perspectives from switzer-

- land, September 2021. URL <https://www.rug.nl/library/open-access/blog/using-lotteries-to-allocate-research-funding-perspectives-from-switzerland?lang=en>. University of Groningen Library Open Science Blog. 3, 3.1.2, 3.1.4
- [204] Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nikolai Zeldovich. Stadium: A distributed metadata-private messaging system. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, page 423–440, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350853. doi: 10.1145/3132747.3132783. URL <https://doi.org/10.1145/3132747.3132783>. 4.1.2
- [205] Keith Tyser, Jason Lee, Avi Shporer, Madeleine Udell, Dov Te'eni, and Iddo Drori. Open-reviewer: Mitigating challenges in LLM reviewing. *submitted to ICLR 2024 on OpenReview.net*, 2024. 2.2.1
- [206] Alexander Ugarov. Peer prediction for peer review: designing a marketplace for ideas. *arXiv preprint arXiv:2303.16855*, 2023. 1
- [207] UIDAI. Bio-challenge. <https://biochallenge.uidai.gov.in/>, 2025. Accessed on July 3, 2025. 4.2.1
- [208] UK Metascience Unit. A year in metascience (2025). Technical report, Department for Science, Innovation and Technology, June 30 2025. URL <https://www.gov.uk/government/publications/a-year-in-metascience-2025>. Published 30 June 2025. 3.2.1
- [209] USENIX Security 2025 Program Committee. Usenix security 2025 conference format. <https://github.com/USENIX-Security-2025/conference-format>, April 2024. Version 1.1. 3, 3.1.2, 1, 3.1.6, 3.2.2, 3.2.7
- [210] Jelle van den Hooff, David Lazar, Matei Zaharia, and Nikolai Zeldovich. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the 25th Symposium on Operating Systems Principles*, SOSP '15, page 137–152, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450338349. doi: 10.1145/2815400.2815417. URL <https://doi.org/10.1145/2815400.2815417>. 4.1.2
- [211] Susan Van Rooyen, Nick Black, and Fiona Godlee. Development of the review quality instrument (rqi) for assessing peer reviews of manuscripts. *Journal of clinical epidemiology*, 52(7):625–629, 1999. 2.1.3
- [212] Susan Van Rooyen, Fiona Godlee, Stephen Evans, Nick Black, and Richard Smith. Effect of open peer review on quality of reviews and on reviewers' recommendations: a randomised trial. *Bmj*, 318(7175):23–27, 1999. 2
- [213] Susan Van Rooyen, Tony Delamothe, and Stephen JW Evans. Effect on peer review of telling reviewers that their signed reviews might be posted on the web: randomised controlled trial. *Bmj*, 341, 2010. 2
- [214] Jan P Vandenbroucke, Erik von Elm, Douglas G Altman, Peter C Gøtzsche, Cynthia D Mulrow, Stuart J Pocock, Charles Poole, James J Schlesselman, Matthias Egger, and Strobe Initiative. Strengthening the reporting of observational studies in epidemiology

- (strobe): explanation and elaboration. *Annals of internal medicine*, 147(8):W-163, 2007. 2.2.2
- [215] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10-48550, 2017. 4.2.7
- [216] T. N. Vijaykumar. Potential organized fraud in on-going asplos reviews, Nov 2020. URL <https://medium.com/@tnvijayk/potential-organized-fraud-in-on-going-asplos-reviews-874ce14a3ebe>. 4.1.2
- [217] T. N. Vijaykumar. Potential organized fraud in ACM/IEEE computer architecture conferences. <https://medium.com/@tnvijayk/potential-organized-fraud-in-acm-ieee-computer-architecture-conferences-ccd61169370d>, 2020. 2.1.5
- [218] Volkswagen Foundation. Partially randomized procedure – lottery and peer review. <https://www.volkswagenstiftung.de/en/funding/peer-review/partially-randomized-procedure-lottery-and-peer-review>, May 2025. Accessed: 2025-05-07. 1.1, 3, 3.1.2, 1, 3.2.6
- [219] Elizabeth Walsh, Maeve Rooney, Louis Appleby, and Greg Wilkinson. Open peer review: A randomised controlled trial. *The British Journal of Psychiatry*, 176(1):47-51, 2000. doi: 10.1192/bjp.176.1.47. 2
- [220] Jingyan Wang and Nihar B Shah. Your 2 is my 1, your 3 is my 9: Handling arbitrary miscalibrations in ratings. In *AAMAS*, 2019. 2.1.4, 1, 4.1.2
- [221] Jingyan Wang, Ivan Stelmakh, Yuting Wei, and Nihar B Shah. Debiasing evaluations that are biased by evaluations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10120-10128, 2021. 2.1.2
- [222] Weiran Wang and Canyi Lu. Projection onto the capped simplex. *arXiv preprint arXiv:1503.01002*, 2015. 3.2.4, 3.2.4
- [223] Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375-389, 2010. doi: 10.1198/jasa.2009.tm08651. 4.1.5
- [224] Ellen J Weber, Patricia P Katz, Joseph F Waeckerle, and Michael L Callahan. Author perception of peer review: impact of review quality and acceptance on satisfaction. *JAMA*, 287(21):2790-2793, 2002. 2.1.2, 2.1.4
- [225] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I. Weidele, Claudio Bellei, Tom Robinson, and Charles E. Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics, 2019. 4.2.5, 4.2.7
- [226] Aidan S Weitzner, Matthew Davis, Andrew H Han, Olivia O Liu, Anuj B Patel, Brian D Sites, and Steven P Cohen. How predictive is peer review for gauging impact? the association between reviewer rating scores, publication status, and article impact measured by citations in a pain subspecialty journal. *Regional Anesthesia & Pain Medicine*, 2024. 1.1, 3.1.1
- [227] John Wieting, Kevin Gimpel, Graham Neubig, and Taylor Berg-Kirkpatrick. Simple and

- effective paraphrastic similarity from parallel translations. In *ACL*, pages 4602–4608, Florence, Italy, July 2019. 2.2.1
- [228] Wikipedia Terms of Service. Sockpuppetry - alternative accounts. Wikipedia TOS, 2022. Accessed on April 28,2022. 4.1.1
- [229] Gerhard J Woeginger. On the approximability of average completion time scheduling under precedence constraints. *Discrete Applied Mathematics*, 131(1):237–252, 2003. 3, 3.1.4
- [230] James G Wrightson, Paul Blazey, D Moher, KM Khan, and CL Ardern. Gpt for rcts? using ai to determine adherence to reporting guidelines. *medRxiv*, pages 2023–12, 2023. 2.2.2
- [231] Ruihan Wu, Chuan Guo, Felix Wu, Rahul Kidambi, Laurens van der Maaten, and Kilian Weinberger. Making paper reviewing robust to bid manipulation attacks. In *ICML*, 2021. 4.1.2
- [232] Ruihan Wu, Chuan Guo, Felix Wu, Rahul Kidambi, Laurens van der Maaten, and Kilian Q. Weinberger. Making paper reviewing robust to bid manipulation attacks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 11240–11250, Virtual Event, 2021. PMLR. URL <http://proceedings.mlr.press/v139/wu21b.html>. 4.2.7
- [233] Yuanzhang Xiao, Florian Dörfler, and Mihaela van der Schaar. Rating and matching in peer review systems. In *Allerton conference*, 2014. 1, 2.1.5
- [234] Yuanzhang Xiao, Florian Dörfler, and Mihaela Van Der Schaar. Incentive design in peer review: Rating and repeated endogenous matching. *IEEE Transactions on Network Science and Engineering*, 2018. 1, 2.1.5
- [235] Yichong Xu, Han Zhao, Xiaofei Shi, and Nihar Shah. On strategyproof conference review. In *IJCAI*, 2019. 2.1.5
- [236] Minji Yoon, Yue Wu, John Palowitch, Bryan Perozzi, and Russ Salakhutdinov. Graph generative model for benchmarking graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning, ICML’23*, , Honolulu, Hawaii, USA,, 2023. JMLR.org. 4.2.6
- [237] Haoyang Yu and Jerome P Reiter. Differentially private verification of regression predictions from synthetic data. *Trans. Data Priv.*, 11(3):279–297, 2018. 4.2.4
- [238] Quan Yuan, Zhikun Zhang, Linkang Du, Min Chen, Peng Cheng, and Mingyang Sun. PrivGraph: Differentially private graph data publication by exploiting community information. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 3241–3258, Anaheim, CA, August 2023. USENIX Association. ISBN 978-1-939133-37-3. URL <https://www.usenix.org/conference/usenixsecurity23/presentation/yuan-quan>. 4.2.6
- [239] Kiarash Zahirnia, Yaochen Hu, Mark Coates, and Oliver Schulte. Neural graph generation from graph statistics. *Advances in Neural Information Processing Systems*, 36, 2024. 4.2.6